aws

Entwicklerhandbuch

Amazon Textract

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Textract: Entwicklerhandbuch

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Die Handelsmarken und Handelsaufmachung von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, auf eine Art und Weise, dass Kunden irrgeführt werden könnten oder Amazon schlecht gemacht oder diskreditiert werden könnte. Alle anderen Handelsmarken, die nicht Eigentum von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise zu Amazon gehören oder nicht, mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

Was ist Amazon Textract? 1
Erstmalige Amazon Textract Textract-Benutzer 2
Verwenden von Amazon Textract mit einemAWSSDK 3
So funktioniert es 4
Erkennen von Text 5
Analysieren von Dokumenten 5
Analysieren von Rechnungen und Belegen 7
Analysieren von Ausweisdokumenten 11
Eingabe-Dokumente 12
Amazon Textract Antwortobjekte 14
Antwortobjekte für Texterkennung und Dokumentanalyse 14
Antwortobjekte auf Rechnung und Wareneingang 35
Antwortobjekte der Identitä 38
Artikelspeicherort auf einer Dokumentseite 40
Bounding Box 41
Polygon 44
Erste Schritte 46
Schritt 1: Einrichten eines Kontos 46
Anmelden bei AWS 46
Erstellen eines IAM-Benutzers 47
Nächster Schritt 48
Schritt 2: Einrichten derAWS CLIundAWS-SDKs 48
Nächster Schritt 51
Schritt 3: Erste Schritte mit derAWS CLIundAWSSDK API 51
Formatieren der AWS CLI-Beispiele 51
Dokumente mit synchronen Operationen verarbeiten 52
Aufrufen von Amazon Textract synchrone Operationen 53
Anfrage 53
Antwort 55
Erkennen von Text-Dokument 126
Analysieren von Text-Dokumenten 139
Analysieren von Rechnungs- und Belegen 151
Analysieren von ID-Dokumenten 164
Dokumente mit asynchronen Operationen verarbeiten 170

Asynchrone Operationen aufrufen ... 171
Starten der Texterkennung ... 172
Abrufen des Erledigungsstatus einer Amazon Textract Textract-Analyseanforderung ... 174
Ergebnisse der Amazon Textract Textract-Texterkennung abrufen ... 176
Konfigurieren von asynchronen Operationen ... 185
Amazon Textract Zugriff auf Ihr Amazon SNS SNS-Thema gewähren ... 187
Erkennen oder Analysieren von Text in einem mehrseitigen Dokument ... 188
Durchführen von asynchronen Operationen ... 189
Amazon Textract Ergebnisbenachrichtigung ... 215
Umgang mit gedrosselten Anrufen und gelöschten ... 217
Bewährte Methoden für Amazon Textract ... 223
Bereitstellen eines optimalen Eingabedokuments ... 223
Verwenden von Zuverlässigkeitswert ... 224
Erwägen Sie die Verwendung von ... 224
Tutorials ... 225
Voraussetzungen ... 225
Extrahieren von Schlüssel-Wert-Paaren aus einem Formulardokument ... 226
Exportieren von Tabellen in eine CSV-Datei ... 229
Erstellen einesAWS LambdaFunktion ... 239
So rufen Sie den DetectDocumentText-Vorgang von einer Lambda-Funktion aus auf: ... 239
Weitere Codebeispiele ... 242
Codebeispiele ... 244
Aktionen ... 244
Analysieren eines Dokuments ... 245
Text in einem Dokument erkennen ... 248
Abrufen von Daten zu einem Dokumentanalyse-Auftrag ... 251
Starten Sie die asynchrone Analyse eines Dokuments ... 253
Starten der asynchronen Texterkennung ... 256
Serviceübergreifende Beispiele ... 258
Erstellen Sie eine Amazon-Textract-Explorer-Anwendung ... 258
Erkennt Entitys in Text, der aus einem Bild extrahiert ... 260
Amazon A2I und Amazon Textract ... 262
Kernkonzepte von Amazon A2I ... 262
Aktivierungsbedingungen der Prüfung durch Menschen ... 262
Arbeitsablauf für menschliche Überprüfung (Flow-Definition) ... 264
Menschliche Schleifen ... 266

Verwenden von Amazon A2I ... 266
Erstellen eines Workflows für die Prüfung durch Menschen ... 267
Analysieren des Dokuments ... 273
Überwachen von Human L ... 275
Ausgabedaten und Worker-Metriken anzeigen ... 276
Sicherheit ... 279
Datenschutz ... 279
Verschlüsselung in Amazon Textract ... 280
Richtlinie für den Datenverkehr zwischen Netzwerken ... 281
Identity and Access Management ... 282
Zielgruppe ... 282
Authentifizierung mit Identitäten ... 283
Verwalten des Zugriffs mit Richtlinien ... 286
Funktionsweise von Amazon Textract mit IAM ... 289
Beispiele für identitätsbasierte Richtlinien ... 293
Fehlerbehebung ... 297
Protokollieren und überwachen ... 300
Überwachung ... 300
CloudWatch-Metriken für Amazon Textract ... 305
Protokollieren von Amazon-Textract-API-Aufrufen mitAWS CloudTrail ... 307
Amazon Textract Informationen in CloudTrail ... 307
Grundlagen zu Amazon Textract Textract-Protokolldateieinträgen ... 309
Compliance-Validierung ... 311
Ausfallsicherheit ... 312
Sicherheit der Infrastruktur ... 313
Konfigurations- und Schwachstellenanalyse ... 313
VPC-Endpunkte (AWS PrivateLink) ... 313
Überlegungen zu Amazon Textract VPC-Endpunkten ... 314
Erstellen eines Schnittstellen-VPC-Endpunkts für Amazon Textract ... 314
Erstellen einer VPC-Endpunktrichtlinie für Amazon Textract ... 314
API-Referenz ... 316
Aktionen ... 316
AnalyzeDocument ... 317
AnalyzeExpense ... 324
AnalyzeID ... 331
DetectDocumentText ... 336

GetDocumentAnalysis ... 341
GetDocumentTextDetection ... 348
GetExpenseAnalysis ... 355
StartDocumentAnalysis ... 363
StartDocumentTextDetection ... 370
StartExpenseAnalysis ... 376
Datentypen ... 382
AnalyzeIDDetections ... 383
Block ... 385
BoundingBox ... 390
Document ... 392
DocumentLocation ... 394
DocumentMetadata ... 395
ExpenseDetection ... 396
ExpenseDocument ... 398
ExpenseField ... 400
ExpenseType ... 402
Geometry ... 403
HumanLoopActivationOutput ... 404
HumanLoopConfig ... 406
HumanLoopDataAttributes ... 408
IdentityDocument ... 409
IdentityDocumentField ... 410
LineItemFields ... 411
LineItemGroup ... 412
NormalizedValue ... 413
NotificationChannel ... 414
OutputConfig ... 416
Point ... 418
Relationship ... 419
S3Object ... 421
Warning ... 423
Einschränkungen ... 424
Amazon Textract ... 424
Dokumentverlauf ... 426
AWS-Glossar ... 428

..... cdxxix

Was ist Amazon Textract?

Mit Amazon Textract ist es einfach, Texterkennungs- und -analysefunktionen für Dokumente zu Ihren Anwendungen hinzuzufügen. Mit Amazon Textract Textract-Kunden können:

- Erkennen Sie getippten und handgeschriebenen Text in einer Vielzahl von Dokumenten, einschließlich Finanzberichten, Krankenakten und Steuerformularen.
- Extrahieren Sie Text, Formulare und Tabellen aus Dokumenten mit strukturierten Daten mithilfe der Amazon Textract Document Analysis API.
- Verarbeiten Sie Rechnungen und Belege mit der AnalyzeExpense-API.
- Verarbeiten Sie Ausweisdokumente wie Führerscheine und Pässe, die von der US-Regierung ausgestellt wurden, unter Verwendung der AnalyeID-API.

Amazon Textract basiert auf derselben bewährten, hochgradig skalierbaren, Deep-Learning-Technologie, die von Amazon's Computer Vision Wissenschaftlern entwickelt wurde, damit täglich Milliarden Bilder und Videos analysiert werden können. Um es nutzen zu können, sind keine Machine Learning-Kompetenz erforderlich. Amazon Textract enthält einfache, einfach zu bedienende APIs, die Bilddateien und PDF-Dateien analysieren können. Amazon Textract lernt ständig von neuen Daten, und Amazon fügt dem Service ständig neue Funktionen hinzu.

Die folgenden Situationen sind gängige Anwendungsfälle für die Verwendung von Amazon Textract:

- Erstellen eines intelligenten Suchindex— Mit Amazon Textract können Sie Textbibliotheken erstellen, die in Bild- und PDF-Dateien erkannt werden.
- Verwenden einer intelligenten Textextraktion für die natürliche Sprachausgabe (NLP)— Amazon Textract gibt Ihnen die Kontrolle darüber, wie Text als Eingabe für NLP-Anwendungen gruppiert wird. Es kann Text als Wörter und Zeilen extrahieren. Es gruppiert auch Text nach Tabellenzellen, wenn die Amazon Textract Textract-Dokumenttabellenanalyse aktiviert ist.
- Beschleunigung der Erfassung und Normalisierung von Daten aus verschiedenen Quellen— Amazon Textract ermöglicht die Extraktion von Text- und Tabellendaten aus einer Vielzahl von Dokumenten wie Finanzdokumenten, Forschungsberichten und medizinischen Notizen. Mit Amazon Textract Analyze Document APIs können Sie einfach und schnell unstrukturierte und strukturierte Daten aus Ihren Dokumenten extrahieren.
- Automatisieren der Datenerfassung aus Formularen— Amazon Textract ermöglicht es, strukturierte Daten aus Formularen zu extrahieren. Mit Amazon Textract Analysis APIs können Sie

Extraktionsfunktionen in bestehende Geschäftsworkflows integrieren, sodass Benutzerdaten, die über Formulare übermittelt werden, in ein brauchbares Format extrahiert werden können.

Einige Vorteile der Verwendung von Amazon Textract sind:

- Integration der Dokumenttexterkennung in Ihre Apps— Amazon Textract entfernt die Komplexität des Aufbaus von Texterkennungskapazitäten in Ihre Anwendungen, da eine leistungsstarke und akkurate Analyse in einer übersichtlichen API bereitgestellt wird. Sie benötigen keine Fachkenntnisse auf den Bereichen Computer Vision oder Deep Learning, um mithilfe von Amazon Textract Dokumenttext zu erkennen. Mit Amazon Textract Text-APIs können Sie einfach Texterkennungs in jede mobile, vernetzte oder Web-Geräteanwendung integrieren.
- Skalierbare Dokumentenanalyse— Mit Amazon Textract können Sie Daten schnell aus Millionen von Dokumenten analysieren und extrahieren, was die Entscheidungsfindung beschleunigen kann.
- Niedrige Kosten— Bei Amazon Textract zahlen Sie nur für die Dokumente, aus denen eine Sprachausgabe analysiert wird. Es fallen keine Mindestgebühren oder Vorausleistungen an. Mit unserem gestaffelten Preissystem können Sie kostenfrei einsteigen und mit zunehmendem Wachstum dank unserem gestaffelten Preissystem von mehr sparen.

Bei synchroner Verarbeitung kann Amazon Textract einseitige Dokumente für Anwendungen analysieren, bei denen die Latenz kritisch ist. Amazon Textract bietet auch asynchrone Vorgänge, um den Support auf mehrseitige Dokumente auszudehnen.

Erstmalige Amazon Textract Textract-Benutzer

Wenn Sie Amazon Textract zum ersten Mal verwenden, empfehlen wir, sich die folgenden Abschnitte nacheinander durchzulesen:

1. Funktionsweise von Amazon Textract— In diesem Abschnitt werden die Amazon Textract Textract-Komponenten für eine durchgehende Erfahrung vorgestellt.
2. Erste Schritte mit Amazon Textract— In diesem Abschnitt erstellen Sie Ihr Konto und testen die Amazon Textract Textract-API.

Verwenden von Amazon Textract mit einemAWSSDK

AWS Software Development Kits (SDKs) sind für viele gängige Programmiersprachen erhältlich. Jedes SDK bietet eine API, Codebeispiele und Dokumentation, die es Entwicklern erleichtern, Anwendungen in ihrer bevorzugten Sprache zu erstellen.

SDK-Dokumentation	Codebeispiele
AWS SDK für C++	AWS SDK für C++-Codebeispiele
AWS SDK für Go	AWS SDK für Go-Codebeispiele
AWS SDK für Java	AWS SDK für Java-Codebeispiele
AWS SDK für JavaScript	AWS SDK für JavaScript-Codebeispiele
AWS SDK für .NET	AWS SDK für .NET-Codebeispiele
AWS SDK für PHP	AWS SDK für PHP-Codebeispiele
AWS SDK für Python (Boto3)	AWS SDK für Python (Boto3)-Codebeispiele
AWS SDK für Ruby	AWS SDK für Ruby-Codebeispiele

Beispiel für die Verfügbarkeit

Sie können nicht finden, was Sie brauchen? Fordern Sie ein Codebeispiel an, indem Sie unten den Link Provide feedback (Feedback geben) auswählen.

Funktionsweise von Amazon Textract

Mit Amazon Textract können Sie Text in ein- oder mehrseitigen Eingabedokumenten erkennen und analysieren (siehe Eingabe-Dokumente) enthalten.

Amazon Textract bietet Vorgänge für die folgenden Aktionen.

- Nur Text wird erkannt. Weitere Informationen finden Sie unter Erkennen von Text aus.
- Erkennen und Analysieren von Beziehungen zwischen Text. Weitere Informationen finden Sie unter Analysieren von Dokumenten aus.
- Erkennen und Analysieren von Text in Rechnungen und Quittungen. Weitere Informationen finden Sie unter Analysieren von Rechnungen und Belegen aus.
- Erkennen und Analysieren von Text in Regierungsausweisdokumenten. Weitere Informationen finden Sie unter Analysieren von Ausweisdokumenten aus.

Amazon Textract bietet synchrone Vorgänge für die Verarbeitung kleiner, einseitiger Dokumente und nahezu Echtzeitantworten. Weitere Informationen finden Sie unter Dokumente mit synchronen Operationen verarbeiten . Amazon Textract bietet auch asynchrone Vorgänge, mit denen Sie größere, mehrseitige Dokumente verarbeiten können. Asynchrone Antworten sind nicht in Echtzeit. Weitere Informationen finden Sie unter Dokumente mit asynchronen Operationen verarbeiten .

Wenn ein Amazon Textract Textract-Vorgang ein Dokument verarbeitet, werden die Ergebnisse in einem Array von the section called “Block”-Objekte oder ein Array von the section called “ExpenseDocument” Objekte. Beide Objekte enthalten Informationen, die über Elemente erkannt werden, einschließlich ihrer Position im Dokument und ihrer Beziehung zu anderen Elementen im Dokument. Weitere Informationen finden Sie unter Amazon Textract Antwortobjekte . Für Beispiele, die zeigen, wie Sie verwenden `Block`objekte, siehe Tutorials aus.

Themen

- Erkennen von Text
- Analysieren von Dokumenten
- Analysieren von Rechnungen und Belegen
- Analysieren von Ausweisdokumenten
- Eingabe-Dokumente
- Amazon Textract Antwortobjekte

- Artikelspeicherort auf einer Dokumentseite

Erkennen von Text

Amazon Textract bietet synchrone und asynchrone Vorgänge, die nur den in einem Dokument erkannten Text zurückgeben. Für beide Operationsgruppen werden die folgenden Informationen in mehreren Fällen zurückgegebenthe section called “Block”Objekte.

- Die Zeilen und Wörter des erkannten Textes
- Die Beziehungen zwischen den Zeilen und Wörtern des erkannten Textes
- Die Seite, auf der der erkannte Text angezeigt wird
- Die Position der Textzeilen und Wörter auf der Dokumentseite

Weitere Informationen finden Sie unter the section called “Zeilen und Wörter des Textes” .

Um Text synchron zu erkennen, verwenden Sie dieDetectDocumentTextAPI-Betrieb, und übergeben Sie eine Dokumentdatei als Eingabe. Der gesamte Ergebnissatz wird von der Operation zurückgegeben. Weitere Informationen sowie ein Beispiel finden Sie unter Dokumente mit synchronen Operationen verarbeiten.

> **Note**
>
> Der Betrieb der Amazon Rekognition API`DetectText`ist anders als`DetectDocumentText`aus. Du benutzt`DetectText`um Text in Live-Szenen wie Poster oder Verkehrszeichen zu erkennen.

Um Text asynchron zu erkennen, verwenden SieStartDocumentTextDetectionum mit der Verarbeitung einer Eingabedokumentdatei zu beginnen. Rufen Sie an, um die Ergebnisse zu erhaltenGetDocumentTextDetectionaus. Die Ergebnisse werden in einer oder mehreren Antworten von`GetDocumentTextDetection`aus. Weitere Informationen sowie ein Beispiel finden Sie unter Dokumente mit asynchronen Operationen verarbeiten.

Analysieren von Dokumenten

Amazon Textract analysiert Dokumente und Formulare auf Beziehungen zwischen erkannten Texten. Amazon Textract Textract-Analysevorgänge geben 3 Kategorien der Dokumentextraktion zurück -

Text, Formulare und Tabellen. Die Analyse von Rechnungen und Belegen wird durch einen anderen Prozess abgewickelt, weitere Informationen finden Sie unterAnalysieren von Rechnungen und Belegenaus.

Textextraktion

Der Rohtext, der aus einem Dokument extrahiert wurde. Weitere Informationen finden Sie unterTextzeilen und Wörteraus.

Extraktion von Formularen

Formulardaten sind mit Textelementen verknüpft, die aus einem Dokument extrahiert wurden. Amazon Textract stellt Formulardaten als Schlüssel-Wert-Paare dar. Im folgenden Beispiel ist eine der von Amazon Textract erkannten TextzeilenName: Jane Doeaus. Amazon Textract identifiziert auch einen Schlüssel (Name:) und ein Wert (Jane Doe) enthalten. Weitere Informationen finden Sie unterFormulardaten (Schlüssel-Wert-Paare)aus.

Name: Jane Doe

Adresse: 123 Any Street, Anytown, USA

Geburtsdatum: 12-26-1980

Schlüssel-Wert-Paare werden auch verwendet, um Kontrollkästchen oder Optionsfelder (Optionsfelder) darzustellen, die aus Formularen extrahiert werden.

Männlich:☑

Weitere Informationen finden Sie unterAuswahl-Elementeaus.

Extraktion von Tabellen

Amazon Textract kann Tabellen, Tabellenzellen und die Elemente in Tabellenzellen extrahieren und kann so programmiert sein, dass die Ergebnisse in einer JSON-, .csv- oder einer TXT-Datei zurückgegeben werden.

Name	Adresse
Ana Carolina	123 Jede Stadt

Weitere Informationen finden Sie unter Tabellen. Selektionselemente können auch aus Tabellen extrahiert werden. Weitere Informationen finden Sie unterAuswahl-Elementeaus.

Für analysierte Artikel gibt Amazon Textract Folgendes in mehrerenthe section called “Block”Objekte:

- Die Zeilen und Wörter des erkannten Textes
- Der Inhalt der erkannten Elemente
- Die Beziehung zwischen erkannten Elementen
- Die Seite, auf der das Element erkannt wurde
- Die Position des Elements auf der Dokumentseite

Sie können synchrone oder asynchrone Operationen verwenden, um Text in einem Dokument zu analysieren. Um Text synchron zu analysieren, verwenden Sie dieAnalyzeDocument-Operation, und übergeben Sie ein Dokument als Eingabe.`AnalyzeDocument`gibt den gesamten Ergebnissatz zurück. Weitere Informationen finden Sie unter Analysieren von Dokumenttext mit Amazon Textract .

Um Text asynchron zu erkennen, verwenden SieStartDocumentAnalysisum mit der Verarbeitung zu beginnen. Rufen Sie an, um die Ergebnisse zu erhaltenGetDocumentAnalysisaus. Die Ergebnisse werden in einer oder mehreren Antworten von`GetDocumentAnalysis`aus. Weitere Informationen sowie ein Beispiel finden Sie unter Erkennen oder Analysieren von Text in einem mehrseitigen Dokument.

Um anzugeben, welche Art von Analyse durchgeführt werden soll, können Sie die`FeatureTypes`listet Eingabeparameter auf. Fügen Sie der Liste TABLES hinzu, um Informationen über die im Eingabedokument erkannten Tabellen zurückzugeben, z. B. Tabellenzellen, Zelltext und Auswahlelemente in Zellen. Fügen Sie FORMS hinzu, um Wortbeziehungen wie Schlüssel-Wert-Paare und Auswahlelemente zurückzugeben. Um beide Analysetypen durchzuführen, fügen Sie sowohl TABLES als auch FORMS hinzu`FeatureTypes`aus.

Alle Zeilen und Wörter, die im Dokument erkannt werden, sind in der Antwort enthalten (einschließlich Text, der nicht mit dem Wert von`FeatureTypes`) enthalten.

Analysieren von Rechnungen und Belegen

Amazon Textract extrahiert relevante Daten wie Kontaktinformationen, gekaufte Artikel und den Namen des Lieferanten aus fast jeder Rechnung oder Quittung, ohne dass Vorlagen oder Konfigurationen erforderlich sind. Rechnungen und Belege verwenden häufig verschiedene Layouts, was es schwierig und zeitaufwändig macht, Daten in großem Maßstab manuell zu extrahieren. Amazon Textract verwendet ML, um den Kontext von Rechnungen und Belegen zu verstehen, und extrahiert automatisch Daten wie Rechnungs- oder Empfangsdatum,

Rechnungs- oder Belegnummer, Artikelpreise, Gesamtbetrag und Zahlungsbedingungen, um Ihren Geschäftsanforderungen gerecht zu werden.

Amazon Textract identifiziert auch Anbieternamen, die für Ihre Workflows entscheidend sind, aber möglicherweise nicht explizit gekennzeichnet sind. Amazon Textract kann beispielsweise den Händlernamen auf einer Quittung finden, auch wenn er nur in einem Logo oben auf der Seite ohne explizite Schlüssel-Wert-Paarkombination angegeben ist. Amazon Textract macht es Ihnen auch leicht, Eingaben aus verschiedenen Belegen und Rechnungen zu konsolidieren, die unterschiedliche Wörter für dasselbe Konzept verwenden. Amazon Textract ordnet beispielsweise Beziehungen zwischen Feldnamen in verschiedenen Dokumenten wie Kundennummer, Kundennummer und Konto-ID ab und gibt Standardtaxonomie als`INVOICE_RECEIPT_ID`aus. In diesem Fall repräsentiert Amazon Textract Daten konsistent über verschiedene Dokumenttypen hinweg. Felder, die nicht mit der Standardtaxonomie übereinstimmen, werden kategorisiert als`OTHER`aus.

Nachfolgend ist eine Liste der Standardfelder, die AnalyzeExpense derzeit unterstützt:

- Anbieter-Name:`VENDOR_NAME`
- Gesamt:`TOTAL`
- Adresse des Empfängers:`RECEIVER_ADDRESS`
- Rechnung/Zahlungsdatum:`INVOICE_RECEIPT_DATE`
- Rechnung/Belegnummer:`INVOICE_RECEIPT_ID`
- Zahlungsbedingungen:`PAYMENT_TERMS`
- Zwischensumme:`SUBTOTAL`
- Fälligkeitsdatum:`DUE_DATE`
- Steuer:`TAX`
- Rechnungssteuerzahler-ID (SSN/ITIN oder EIN):`TAX_PAYER_ID`
- Elementname:`ITEM_NAME`
- Preis des Artikels:`PRICE`
- Artikelmenge:`QUANTITY`

Die AnalyzeExpense-API gibt die folgenden Elemente für eine bestimmte Dokumentseite zurück:

- Die Anzahl der Belege oder Rechnungen innerhalb einer Seite, die als`ExpenseIndex`
- Der standardisierte Name für einzelne Felder dargestellt als`Type`

- Der tatsächliche Name des Feldes, wie es im Dokument angezeigt wird, dargestellt als`LabelDetection`
- Der Wert des entsprechenden Feldes, dargestellt als`ValueDetection`
- Die Anzahl der Seiten innerhalb des eingereichten Dokuments, dargestellt als`Pages`
- Die Seitenzahl, unter der das Feld, der Wert oder die Einzelposten erkannt wurden, dargestellt als`PageNumber`
- Die Geometrie, die den Begrenzungsrahmen und die Koordinatenposition des einzelnen Feldes, Werts oder der Einzelposten auf der Seite enthält, dargestellt als`Geometry`
- Der Konfidenzwert, der mit jedem im Dokument erkannten Daten verknüpft ist, dargestellt als`Confidence`
- Die gesamte Reihe der gekauften Einzelposten, dargestellt als`EXPENSE_ROW`

Das Folgende ist ein Teil der API-Ausgabe für einen von AnalyzeExpense verarbeiteten Beleg, der die Summe anzeigt: 55,64$ in dem als Standardfeld extrahierten Dokument`TOTAL`, tatsächlicher Text auf dem Dokument als „Gesamt“, Konfidenzwert von „97.1“, Seitenzahl „1“, Der Gesamtwert als „55,64$“ und der Begrenzungsrahmen- und Polygonkoordinaten:


```
{
    "Type": {
        "Text": "TOTAL",
        "Confidence": 99.94717407226562
    },
    "LabelDetection": {
        "Text": "Total:",
        "Geometry": {
            "BoundingBox": {
                "Width": 0.09809663146734238,
                "Height": 0.0234375,
                "Left": 0.36822840571403503,
                "Top": 0.8017578125
            },
            "Polygon": [
                {
                    "X": 0.36822840571403503,
                    "Y": 0.8017578125
                },
                {
                    "X": 0.466325044631958,
                    "Y": 0.8017578125
```

```
                },
                {
                    "X": 0.466325044631958,
                    "Y": 0.8251953125
                },
                {
                    "X": 0.36822840571403503,
                    "Y": 0.8251953125
                }
            ]
        },
        "Confidence": 97.10792541503906
    },
        "ValueDetection": {
            "Text": "$55.64",
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.10395314544439316,
                    "Height": 0.0244140625,
                    "Left": 0.66837477684021,
                    "Top": 0.802734375
                },
                "Polygon": [
                    {
                        "X": 0.66837477684021,
                        "Y": 0.802734375
                    },
                    {
                        "X": 0.7723279595375061,
                        "Y": 0.802734375
                    },
                    {
                        "X": 0.7723279595375061,
                        "Y": 0.8271484375
                    },
                    {
                        "X": 0.66837477684021,
                        "Y": 0.8271484375
                    }
                ]
            },
        "Confidence": 99.85165405273438
    },
    "PageNumber": 1
```

```
}
```

Sie können synchrone Vorgänge verwenden, um eine Rechnung oder einen Beleg zu analysieren. Um diese Dokumente zu analysieren, verwenden Sie den AnalyzeExpense-Vorgang und übergeben eine Quittung oder Rechnung an sie.`AnalyzeExpense`gibt den gesamten Ergebnissatz zurück. Weitere Informationen finden Sie unter Rechnungen und Belege mit Amazon Textract analysieren .

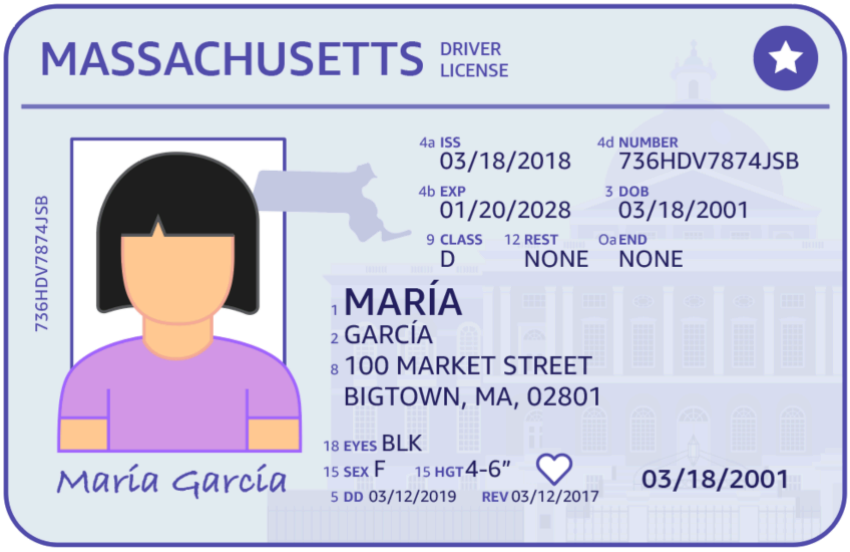
Um Rechnungen und Belege asynchron zu analysieren, verwenden SieStartExpenseAnalysisum mit der Verarbeitung einer Eingabedokumentdatei zu beginnen. Rufen Sie an, um die Ergebnisse zu erhaltenGetExpenseAnalysisaus. Die Ergebnisse für einen bestimmten Aufruf anStartExpenseAnalysiswerden von zurückgegeben`GetExpenseAnalysis`aus. Weitere Informationen sowie ein Beispiel finden Sie unter Dokumente mit asynchronen Operationen verarbeiten.

Analysieren von Ausweisdokumenten

Amazon Textract kann relevante Informationen aus Pässen, Führerscheinen und anderen Identitätsdokumenten extrahieren, die von der US-Regierung mit der AnalyeID-API ausgestellt wurden. Mit Analyze ID können Unternehmen schnell und genau Informationen aus IDs wie US-Führerscheinen, staatlichen IDs und Reisepässen extrahieren, die unterschiedliche Vorlagen oder Formate haben. Die AnalyeID-API gibt zwei Kategorien von Datentypen zurück:

- Schlüssel-Wert-Paare, die für ID verfügbar sind, wie Geburtsdatum, Ausstellungsdatum, ID #, Klasse und Einschränkungen.
- Implizierte Felder im Dokument, denen möglicherweise keine expliziten Schlüssel zugeordnet sind, wie Name, Adresse und Ausgestellt von.

Schlüsselnamen sind innerhalb der Antwort standardisiert. Wenn Ihr Führerschein beispielsweise LIC # (Lizenznummer) und Reisepass Nein angibt, gibt die Analyse-ID-Antwort den standardisierten Schlüssel zusammen mit dem Rohschlüssel (z. B. LIC#) als „Document ID“ zurück. Diese Standardisierung ermöglicht es Kunden, Informationen auf einfache Weise über viele IDs hinweg zu kombinieren, die unterschiedliche Begriffe für dasselbe Konzept verwenden.



Analyze ID gibt Informationen in den Strukturen zurück, die aufgerufen werden`IdentityDocumentFields`aus. Das sind`JSON`-Strukturen, die zwei Informationen enthalten: den normalisierten Typ und den mit dem Typ verknüpften Wert. Diese beiden haben auch einen Konfidenzwert. Weitere Informationen finden Sie unter Antwortobjekte der Identitä .

Sie können synchrone Vorgänge verwenden, um einen Führerschein oder Reisepass zu analysieren. Um diese Dokumente zu analysieren, verwenden Sie die AnalyeID-Operation und übergeben ein Ausweisdokument an sie.`AnalyzeID`gibt den gesamten Ergebnissatz zurück. Weitere Informationen finden Sie unter Analysieren der Identitätsdokumentation mit Amazon Textract .

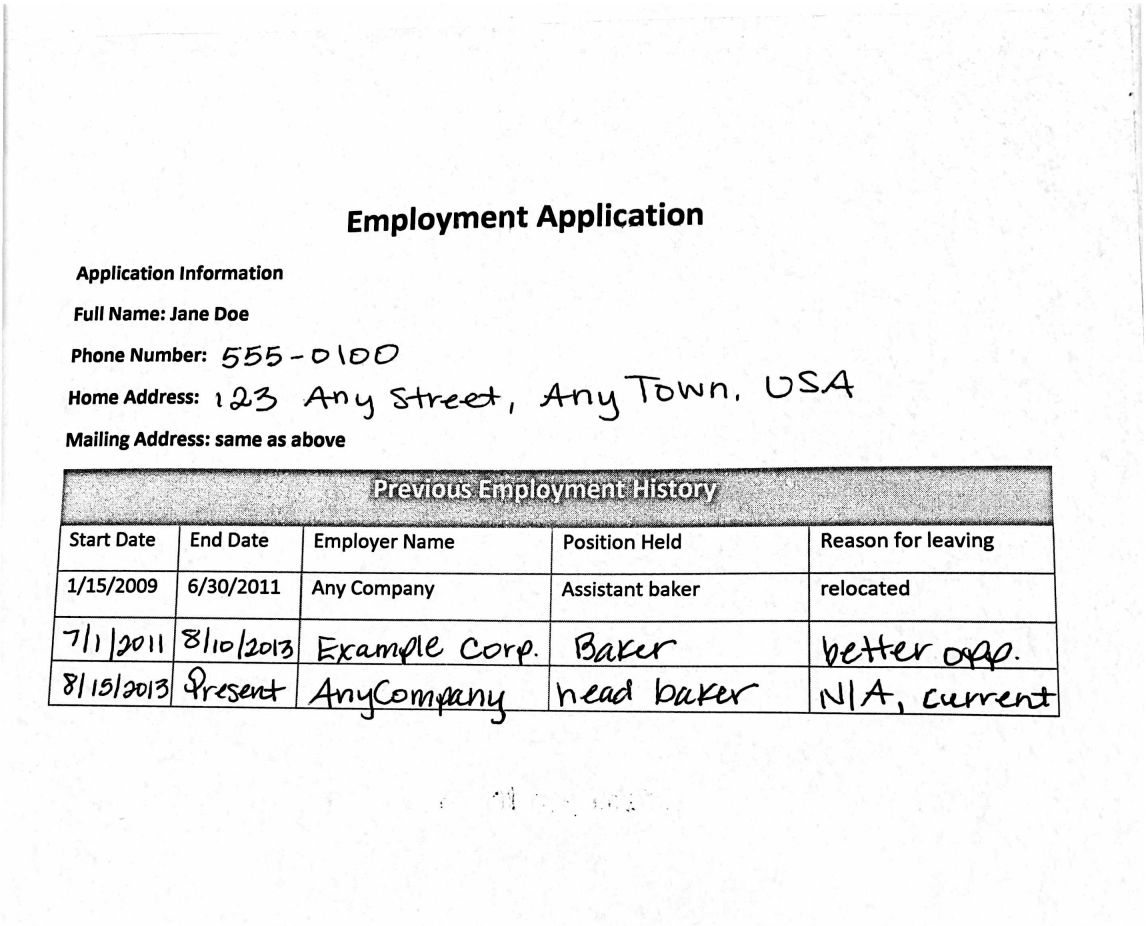
> **Note**
>
> Einige Ausweisdokumente, wie Führerscheine, haben zwei Seiten. Sie können die Vorder- und Rückbilder von Treiberlizenzen als separate Images innerhalb derselben Analyse-ID-API-Anforderung übergeben.

Eingabe-Dokumente

Eine geeignete Eingabe für eine Amazon Textract Textract-Operation ist ein einzelnes oder mehrseitiges Dokument. Einige Beispiele sind ein Rechtsdokument, ein Formular, ein Ausweis oder ein Brief. Ein Formular ist ein Dokument mit Fragen oder Aufforderungen für einen Benutzer, Antworten zu geben. Einige Beispiele sind ein Anmeldeformular für Patienten, ein Steuerformular oder ein Versicherungsanspruchsformular.

Ein Dokument kann im JPEG-, PNG-, PDF- oder TIFF-Format vorliegen. Mit Dateien im PDF- und TIFF-Format können Sie mehrseitige Dokumente verarbeiten. Weitere Informationen darüber, wie Amazon Textract Dokumente darstellt als`Block`objekte, siehe Antwortobjekte für Texterkennung und Dokumentanalyse aus.

Nachfolgend ist ein Beispiel für ein akzeptables Eingabedokument.



Employment Application

Application Information

Full Name: Jane Doe

Phone Number: 555-0100

Home Address: 123 Any Street, Any Town, USA

Mailing Address: same as above

Previous Employment History

Start Date	End Date	Employer Name	Position Held	Reason for leaving
1/15/2009	6/30/2011	Any Company	Assistant baker	relocated
7/1/2011	8/10/2013	Example Corp.	Baker	better opp.
8/15/2013	Present	AnyCompany	head baker	N/A, current

Weitere Informationen zu den Limits von Dokumenten finden Sie unter Hard Limits in Amazon Textract aus.

Für synchrone Operationen von Amazon Textract können Sie Eingabedokumente verwenden, die in einem Amazon S3 S3-Bucket gespeichert sind, oder Sie können base64-codierte Bildbytes übergeben. Weitere Informationen finden Sie unter Aufrufen von Amazon Textract synchrone Operationen . Für asynchrone Vorgänge müssen Sie Eingabedokumente in einem Amazon S3 S3-Bucket angeben. Weitere Informationen finden Sie unter Asynchrone Operationen von Amazon Textract aufrufen .

Amazon Textract Antwortobjekte

Amazon Textract Textract-Operationen geben je nach ausgeführtem Vorgang verschiedene Objekttypen zurück. Zum Erkennen von Text und zum Analysieren eines generischen Dokuments gibt der Vorgang ein Block-Objekt zurück. Für die Analyse einer Rechnung oder eines Belegs gibt der Vorgang ein ExpenseDocuments -Objekt zurück. Für die Analyse der Identitätsdokumentation gibt der Vorgang ein IdentityDocumentFields -Objekt zurück. Weitere Informationen zu diesen Antwortobjekten finden Sie in folgenden Abschnitten:

Themen

- Antwortobjekte für Texterkennung und Dokumentanalyse
- Antwortobjekte auf Rechnung und Wareneingang
- Antwortobjekte der Identitä

Antwortobjekte für Texterkennung und Dokumentanalyse

Wenn Amazon Textract ein Dokument verarbeitet, erstellt es eine Liste vonBlockObjekte für den erkannten oder analysierten Text. Jeder Block enthält Informationen über ein erkanntes Element, in dem er sich befindet, und das Vertrauen, das Amazon Textract in die Genauigkeit der Verarbeitung hat.

Ein Dokument besteht aus den folgenden Arten von`Block`Objekte.

- Seiten
- Textzeilen und Wörter
- Formulardaten (Schlüssel-Wert-Paare)
- Tabellen und Zellen
- Auswahl-Elemente

Der Inhalt eines Blocks hängt von der Operation ab, die Sie aufrufen. Wenn Sie eine der Texterkennungsoperationen aufrufen, werden die Seiten, Zeilen und Wörter des erkannten Textes zurückgegeben. Weitere Informationen finden Sie unter Erkennen von Text . Wenn Sie einen der Dokumentanalysevorgänge aufrufen, werden Informationen über erkannte Seiten, Schlüssel-Wert-Paare, Tabellen, Auswahlelemente und Text zurückgegeben. Weitere Informationen finden Sie unter Analysieren von Dokumenten .

EtwasBlockObjektfelder sind beiden Verarbeitungsarten gemeinsam. Beispielsweise hat jeder Block einen eindeutigen Bezeichner.

Für Beispiele, die zeigen, wie Sie verwendenBlockobjekte, sieheTutorialsaus.

Dokument-Layout

Amazon Textract gibt eine Darstellung eines Dokuments als Liste verschiedener Arten vonBlockObjekte, die in einer Eltern-zu-Kind-Beziehung oder einem Schlüssel-Wert-Paar verknüpft sind. Metadaten, die die Anzahl der Seiten in einem Dokument angeben, werden ebenfalls zurückgegeben. Nachfolgend ist der JSON für ein typischesBlockObjekt des TypsPAGEaus.


```
{
    "Blocks": [
        {
            "Geometry": {
                "BoundingBox": {
                    "Width": 1.0,
                    "Top": 0.0,
                    "Left": 0.0,
                    "Height": 1.0
                },
                "Polygon": [
                    {
                        "Y": 0.0,
                        "X": 0.0
                    },
                    {
                        "Y": 0.0,
                        "X": 1.0
                    },
                    {
                        "Y": 1.0,
                        "X": 1.0
                    },
                    {
                        "Y": 1.0,
                        "X": 0.0
                    }
                ]
            },
            "Relationships": [
                {
```

```
                    "Type": "CHILD",
                    "Ids": [
                        "2602b0a6-20e3-4e6e-9e46-3be57fd0844b",
                        "82aedd57-187f-43dd-9eb1-4f312ca30042",
                        "52be1777-53f7-42f6-a7cf-6d09bdc15a30",
                        "7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c"
                    ]
                }
            ],
            "BlockType": "PAGE",
            "Id": "8136b2dc-37c1-4300-a9da-6ed8b276ea97"
        }.....

    ],
    "DocumentMetadata": {
        "Pages": 1
    }
}
```

Ein Dokument wird aus einem oder mehrerenPAGEblockiert. Jede Seite enthält eine Liste von untergeordneten Blöcken für die auf der Seite erkannten Hauptelemente, wie Textzeilen und Tabellen. Weitere Informationen finden Sie unter Seiten .

Sie können den Typ eines`Block`Objekt durch Inspizieren des`BlockType`field.

EIN`Block`object enthält eine Liste von verwandten`Block`Objekte im`Relationships`Feld, das ist ein Array vonRelationshipObjekte. EIN`Relationships`array ist entweder vom Typ CHILD oder vom Typ VALUE. Ein Array vom Typ CHILD wird verwendet, um die Elemente aufzulisten, die untergeordnete Elemente des aktuellen Blocks sind. Beispiel: Wenn der aktuelle Block beispielsweise den Typ LINE hat,`Relationships`enthält eine Liste von IDs für die WORD-Blöcke, aus denen die Textzeile besteht. Um Schlüssel-Wert-Paare zu enthalten, wird ein Array vom Typ VALUE verwendet. Sie können den Typ der Beziehung bestimmen, indem Sie die`Type`field des`Relationship`-Objekt.

Untergeordnete Blöcke haben keine Informationen über ihre übergeordneten Blockobjekte.

Für Beispiele, die zeigen`Block`Informationen finden Sie unterDokumente mit synchronen Operationen verarbeitenaus.

Wahrscheinlichkeit

Bei Amazon Textract-Operationen gibt die prozentuale Sicherheit zurück, die Amazon Textract in die Genauigkeit des erkannten Artikels hat. Um das Vertrauen zu erlangen, benutze die`Confidence`field

desBlock-Objekt. Ein höherer Wert weist auf ein höheres Vertrauen hin. Je nach Szenario benötigen Erkennungen mit geringem Vertrauen möglicherweise eine visuelle Bestätigung durch einen Menschen.

Geometry

Amazon Textract Textract-Vorgänge geben mit Ausnahme der Identitätsanalyse Standortinformationen über den Standort erkannter Artikel auf einer Dokumentseite zurück. Um den Standort zu erhalten, verwenden Sie denGeometryfield desBlock-Objekt. Weitere Informationen finden Sie unterArtikelspeicherort auf einer Dokumentseite

Seiten

Ein Dokument besteht aus einer oder mehreren Seiten. EINthe section called “Block”Objekt des TypsPAGEexistiert für jede Seite des Dokuments. EINPAGEblock-Objekt enthält eine Liste der untergeordneten IDs für die Textzeilen, Schlüssel-Wert-Paare und Tabellen, die auf der Dokumentseite erkannt werden.

Der JSON für einenPAGEDer Block sieht in etwa so aus.

```
{

    "Geometry": ....
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "2602b0a6-20e3-4e6e-9e46-3be57fd0844b", // Line - Hello, world.
                "82aedd57-187f-43dd-9eb1-4f312ca30042", // Line - How are you?
                "52be1777-53f7-42f6-a7cf-6d09bdc15a30",
                "7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c"
            ]
        }
    ],
    "BlockType": "PAGE",
    "Id": "8136b2dc-37c1-4300-a9da-6ed8b276ea97"  // Page identifier
},
```

Wenn Sie asynchrone Vorgänge mit einem mehrseitigen Dokument im PDF-Format verwenden, können Sie die Seite ermitteln, auf der sich ein Block befindet, indem Sie diePagefield desBlock-

Objekt. Ein gescanntes Bild (ein Bild im JPEG-, PNG-, PDF- oder TIFF-Format) wird als einseitiges Dokument angesehen, auch wenn sich mehr als eine Dokumentseite auf dem Bild befindet. Asynchrone Operationen geben immer ein`Page`Wert 1 für gescannte Bilder.

Die Gesamtzahl der Seiten wird im`Pages`field von`DocumentMetadata`aus.`DocumentMetadata`wird mit jeder Liste von`Block`Objekte, die von einem Amazon Textract Textract-Vorgang zurückgegeben wurden.

Zeilen und Wörter des Textes

Entdeckter Text, der von Amazon Textract Textract-Vorgängen zurückgegeben wird, wird in einer Liste vonthe section called “Block”Objekte. Diese Objekte stellen Textzeilen oder Textwörter dar, die auf einer Dokumentseite erkannt werden. Der folgende Text zeigt zwei Textzeilen, die aus mehreren Wörtern bestehen.

Dies ist Text.

In zwei separaten Zeilen.

Entdeckter Text wird im`Text`field eines`Block`-Objekt. Die`BlockType`bestimmt, ob der Text eine Textzeile (LINE) oder ein Wort (WORD) ist. EINWORTEs handelt sich um ein oder mehrere lateinische ISO-Basiszeichen, die nicht durch Leerzeichen getrennt sind. EINLINIEist eine Reihe von tabulatorgetrennten und zusammenhängenden Wörtern.

Darüber hinaus wird Amazon Textract feststellen, ob ein Textabschnitt handgeschrieben oder gedruckt wurde`TextTypes`field. Diese werden als HANDSCHRIFT bzw. GEDRUCKT zurückgegeben.

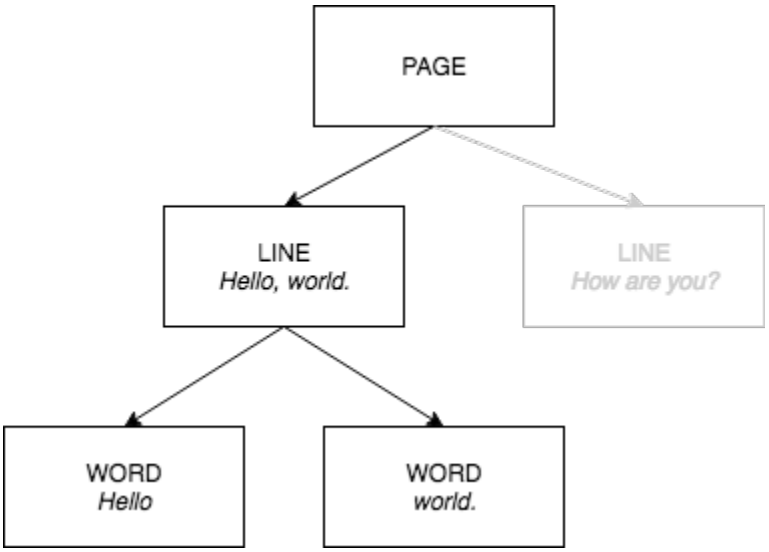
Der andere`Block`Eigenschaften sind allen Blocktypen gemeinsam, wie ID, Konfidenz und Geometrieinformationen. Weitere Informationen finden Sie unter the section called “Antwortobjekte für Texterkennung und Dokumentanalyse” .

Um nur Zeilen und Wörter zu erkennen, können Sie`DetectDocumentText`oder`StartDocumentTextDetection`aus. Weitere Informationen finden Sie unter Erkennen von Text . Um den erkannten Text (Zeilen und Wörter) und Informationen darüber zu erhalten, wie er sich auf andere Teile des Dokuments bezieht, z. B.`AnalyzeDocument`oder`StartDocumentAnalysis`aus. Weitere Informationen finden Sie unter Analysieren von Dokumenten .

`PAGE`,`LINE`, und`WORD`Blöcke sind in einer Eltern-zu-Kind-Beziehung miteinander verwandt. EIN`PAGE`block ist das Elternteil für alle`LINE`blockiert Objekte auf einer Dokumentseite. Da eine LINIE

ein oder mehrere Wörter haben kann,`Relationships`array für einen LINE-Block speichert die IDs für untergeordnete WORD-Blöcke, aus denen die Textzeile besteht.

Das folgende Diagramm zeigt, wie die LinieHallo world.im TextHallo world. Wie geht's dir?wird vertreten durch`Block`Objekte.



Nachfolgend ist die JSON-Ausgabe von`DetectDocumentText`wenn der SatzHallo world. Wie geht's dir?wird erkannt. Das erste Beispiel ist der JSON für die Dokumentseite. Beachten Sie, wie die CHILD-IDs es Ihnen ermöglichen, durch das Dokument zu navigieren.

```
{
    "Geometry": {...},
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "d7fbd604-d609-4d69-857d-247a3f591238", // Line - Hello, world.
                "b6c19a93-6493-4d8e-958f-853c8f7ca055" //  Line - How are you?
            ]
        }
    ],
    "BlockType": "PAGE",
    "Id": "56ec1d77-171f-4881-9852-2b5b7e761608"
},
```

Das Folgende ist der JSON für die LINE-Blöcke, aus denen die Zeile „Hello, World“ besteht:

```
{
```

```
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "7f97e2ca-063e-47a8-981c-8beee31afc01", // Word - Hello,
                "4b990aa0-af96-4369-b90f-dbe02538ed21"  // Word - world.
            ]
        }
    ],
    "Confidence": 99.63229370117188,
    "Geometry": {...},
    "Text": "Hello, world.",
    "BlockType": "LINE",
    "Id": "d7fbd604-d609-4d69-857d-247a3f591238"
},
```

Im Folgenden ist der JSON für den WORD-Block für das WortHallo,:

```
{
    "Geometry": {...},
    "Text": "Hello,",
    "TextType": "PRINTED",
    "BlockType": "WORD",
    "Confidence": 99.74746704101562,
    "Id": "7f97e2ca-063e-47a8-981c-8beee31afc01"
},
```

Der letzte JSON ist der WORD-Block für das Wortwelt.:

```
{
    "Geometry": {...},
    "Text": "world.",
    "TextType": "PRINTED",
    "BlockType": "WORD",
    "Confidence": 99.5171127319336,
    "Id": "4b990aa0-af96-4369-b90f-dbe02538ed21"
},
```

Formulardaten (Schlüssel-Wert-Paare)

Amazon Textract kann Formulardaten aus Dokumenten als Schlüssel-Wert-Paare extrahieren. Beispielsweise kann Amazon Textract im folgenden Text einen Schlüssel (Name:) und ein Wert (Ana Carolina) enthalten.

Name: Ana Carolina

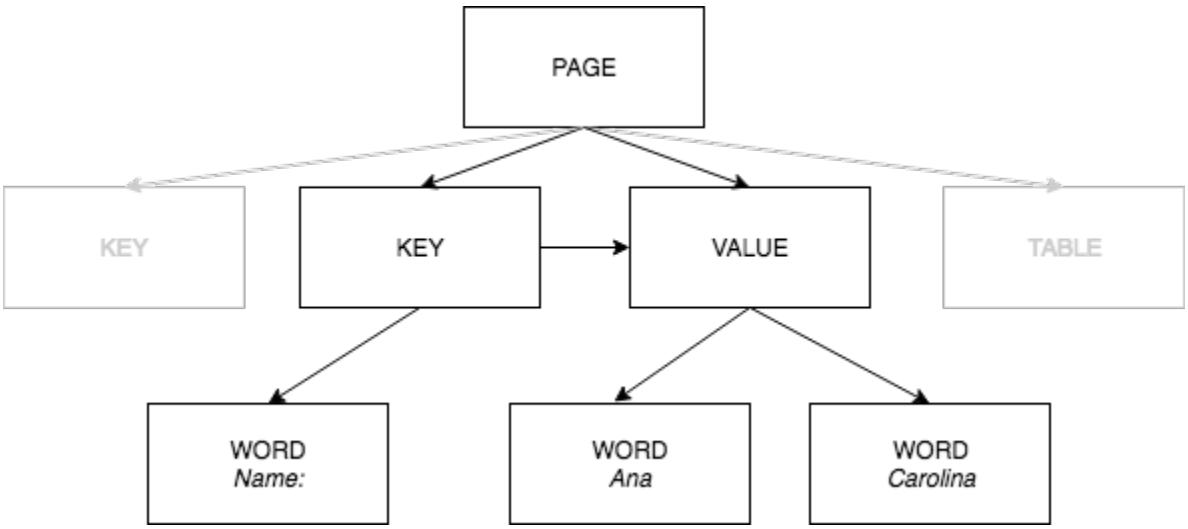
Erkannte Schlüssel-Wert-Paare werden alsBlockObjekte in den Antworten vonAnalyzeDocumentundGetDocumentAnalysisaus. Sie können das`FeatureTypes`Input-Parameter zum Abrufen von Informationen über Schlüssel-Wert-Paare, Tabellen oder beides. Verwenden Sie nur für Schlüssel-Wert-Paare den Wert`FORMS`aus. Ein Beispiel finden Sie unter Extrahieren von Schlüssel-Wert-Paaren aus einem Formulardokument. Für allgemeine Informationen, wie ein Dokument dargestellt wird`Block`objekte, sieheAntwortobjekte für Texterkennung und Dokumentanalyseaus.

Blockobjekte mit dem Typ KEY_VALUE_SET sind die Container für KEY- oder VALUE Block-Objekte, die Informationen zu verknüpften Textelementen speichern, die in einem Dokument erkannt wurden. Sie können das`EntityType`-Attribut, um festzustellen, ob ein Block ein KEY oder ein WERT ist.

- EINSCHLÜSSELDas -Objekt enthält Informationen über den Schlüssel für verknüpften Text. Beispiel,Name:aus. Ein KEY-Block hat zwei Beziehungslisten. Eine Beziehung vom Typ VALUE ist eine Liste, die die ID des VALUE-Blocks enthält, der mit dem Schlüssel verknüpft ist. Eine Beziehung vom Typ CHILD ist eine Liste von IDs für die WORD-Blöcke, aus denen der Text des Schlüssels besteht.
- EINWERTDas -Objekt enthält Informationen über den Text, der einem Schlüssel zugeordnet ist. Für das obige Beispiel gilt:Ana Carolinaist der -Wert für den SchlüsselName:aus. Ein VALUE-Block hat eine Beziehung zu einer Liste von CHILD-Blöcken, die WORD-Blöcke identifizieren. Jeder WORD-Block enthält eines der Wörter, aus denen der Text des Wertes besteht. EIN`VALUE`Das -Objekt kann auch Informationen über ausgewählte Elemente enthalten. Weitere Informationen finden Sie unter Auswahl-Elemente .

Jede Instanz eines KEY_VALUE_SET`Block`object ist untergeordnet der PAGE`Block`Objekt, das der aktuellen Seite entspricht.

Das folgende Diagramm zeigt, wie das Schlüssel-Wert-PaarName: Ana Carolinawird vertreten durch`Block`Objekte.



Die folgenden Beispiele zeigen, wie das Schlüssel-Wert-PaarName: Ana Carolinawird durch JSON vertreten.

Der PAGE Block hat CHILD-Blöcke vom TypKEY_VALUE_SETfür jeden im Dokument erkannten KEY- und VALUE-Block.

```
{
    "Geometry": ....
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "2602b0a6-20e3-4e6e-9e46-3be57fd0844b",
                "82aedd57-187f-43dd-9eb1-4f312ca30042",
                "52be1777-53f7-42f6-a7cf-6d09bdc15a30", // Key - Name:
                "7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c"  // Value - Ana Caroline
            ]
        }
    ],
    "BlockType": "PAGE",
    "Id": "8136b2dc-37c1-4300-a9da-6ed8b276ea97"  // Page identifier
},
```

Der folgende JSON zeigt, dass der KEY-Block (52be1777-53f7-42f6-a7cf-6d09bdc15a30) eine Beziehung zum VALUE-Block (7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c) hat. Es hat auch einen CHILD-Block für den WORD-Block (c734fca6-c4c4-415c-b6c1-30f7510b72ee), der den Text für den Schlüssel (Name:) enthalten.

```
{
    "Relationships": [
        {
            "Type": "VALUE",
            "Ids": [
                "7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c"  // Value identifier
            ]
        },
        {
            "Type": "CHILD",
            "Ids": [
                "c734fca6-c4c4-415c-b6c1-30f7510b72ee"  // Name:
            ]
        }
    ],
    "Confidence": 51.55965805053711,
    "Geometry": ....,
    "BlockType": "KEY_VALUE_SET",
    "EntityTypes": [
        "KEY"
    ],
    "Id": "52be1777-53f7-42f6-a7cf-6d09bdc15a30"  //Key identifier
},
```

Der folgende JSON zeigt, dass VALUE Block 7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c eine CHILD-Liste von IDs für die WORD-Blöcke enthält, die den Text des Wertes ausmachen (AnaundCarolina) enthalten.

```
{
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "db553509-64ef-4ecf-ad3c-bea62cc1cd8a", // Ana
                "e5d7646c-eaa2-413a-95ad-f4ae19f53ef3"  // Carolina
            ]
        }
    ],
    "Confidence": 51.55965805053711,
    "Geometry": ....,
    "BlockType": "KEY_VALUE_SET",
    "EntityTypes": [
```

```
        "VALUE"
    ],
    "Id": "7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c" // Value identifier
}
```

Der folgende JSON zeigt die`Block`Objekte für die WörterName:,Ana, undCarolinaaus.

```
{
    "Geometry": {...},
    "Text": "Name:",
    "TextType": "PRINTED".
    "BlockType": "WORD",
    "Confidence": 99.56285858154297,
    "Id": "c734fca6-c4c4-415c-b6c1-30f7510b72ee"
},
 {
    "Geometry": {...},
    "Text": "Ana",
    "TextType": "PRINTED",
    "BlockType": "WORD",
    "Confidence": 99.52057647705078,
    "Id": "db553509-64ef-4ecf-ad3c-bea62cc1cd8a"
},
{
    "Geometry": {...},
    "Text": "Carolina",
    "TextType": "PRINTED",
    "BlockType": "WORD",
    "Confidence": 99.84207916259766,
    "Id": "e5d7646c-eaa2-413a-95ad-f4ae19f53ef3"
},
```

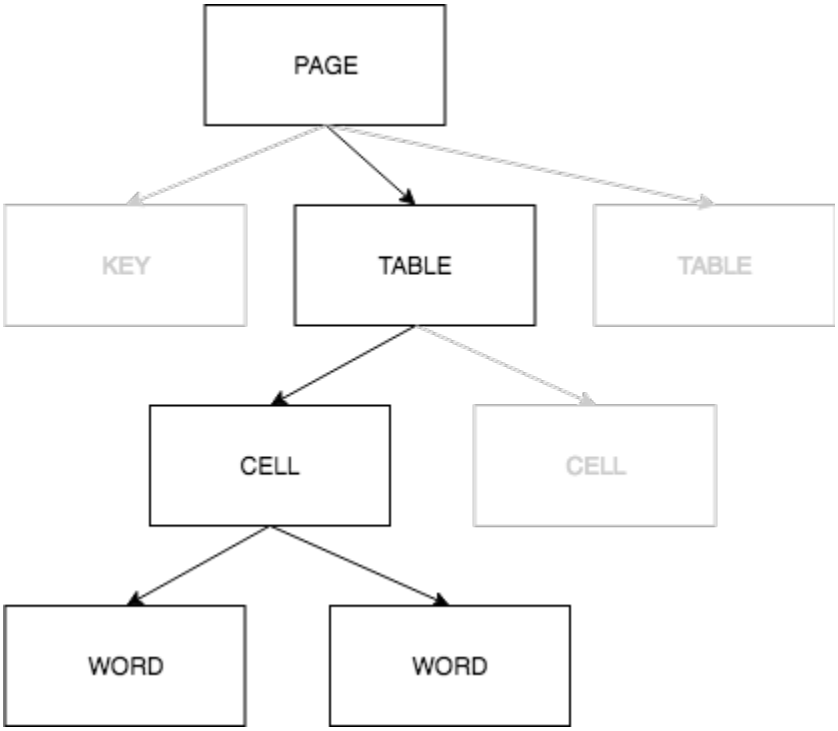
Tabellen

Amazon Textract kann Tabellen und die Zellen in einer Tabelle extrahieren. Wenn beispielsweise die folgende Tabelle in einem Formular erkannt wird, erkennt Amazon Textract eine Tabelle mit vier Zellen.

Name	Adresse
Ana Carolina	123 Any Town

Erkannte Tabellen werden zurückgegeben alsBlockObjekte in den Antworten vonAnalyzeDocumentundGetDocumentAnalysisaus. Sie können das`FeatureTypes`Input-Parameter zum Abrufen von Informationen über Schlüssel-Wert-Paare, Tabellen oder beides. Verwenden Sie nur für Tabellen den Wert`TABLES`aus. Ein Beispiel finden Sie unter Exportieren von Tabellen in eine CSV-Datei. Für allgemeine Informationen, wie ein Dokument dargestellt wird`Block`objekte, siehe Antwortobjekte für Texterkennung und Dokumentanalyseaus.

Das folgende Diagramm zeigt, wie eine einzelne Zelle in einer Tabelle durch`Block`Objekte.



Eine Zelle enthält`WORD`Blöcke für erkannte Wörter und`SELECTION_ELEMENT`Blöcke für Selektionselemente wie Kontrollkästchen.

Das Folgende ist ein teilweiser JSON für die vorhergehende Tabelle, die vier Zellen enthält.

Das PAGE Block -Objekt verfügt über eine Liste von CHILD-Block-IDs für den TABLE-Block und jede erkannte Textzeile.

```
{
    "Geometry": {...},
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "f2a4ad7b-f21d-4966-b548-c859b84f66a4",   // Line - Name
```

```
                "4dce3516-ffeb-45e0-92a2-60770e9cb744",   // Line  - Address
                "ee506578-768f-4696-8f4b-e4917e429f50",   // Line - Ana Carolina
                "33fc7223-411b-4399-8a90-ccd3c5a2c196",   // Line  - 123 Any Town
                "3f9665be-379d-4ae7-be44-d02f32b049c2"    // Table
            ]
        }
    ],
    "BlockType": "PAGE",
    "Id": "78c3ce84-ae70-418e-add7-27058418adf6"
},
```

Der TABLE-Block enthält eine Liste von untergeordneten IDs für die Zellen in der Tabelle. Ein TABLE-Block enthält auch Geometrieinformationen für die Tabellenposition im Dokument. Der folgende JSON zeigt, dass die Tabelle vier Zellen enthält, die im`Ids`Array.

```
{
    "Geometry": {...},
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "505e9581-0d1c-42fb-a214-6ff736822e8c",
                "6fca44d4-d3d3-46ab-b22f-7fca1fbaaf02",
                "9778bd78-f3fe-4ae1-9b78-e6d29b89e5e9",
                "55404b05-ae12-4159-9003-92b7c129532e"
            ]
        }
    ],
    "BlockType": "TABLE",
    "Confidence": 92.5705337524414,
    "Id": "3f9665be-379d-4ae7-be44-d02f32b049c2"
},
```

Der Blocktyp für die Tabellenzellen ist CELL. Die`Block`-Objekt für jede Zelle enthält Informationen über die Zellenposition im Vergleich zu anderen Zellen in der Tabelle. Es enthält auch Geometrieinformationen für die Position der Zelle im Dokument. Für das obige Beispiel gilt:`505e9581-0d1c-42fb-a214-6ff736822e8c`ist die untergeordnete ID für die Zelle, die das Wort enthältNameaus. Das folgende Beispiel sind die Informationen für die Zelle.

```
{
    "Geometry": {...},
    "Relationships": [
```

```
        {
            "Type": "CHILD",
            "Ids": [
                "e9108c8e-0167-4482-989e-8b6cd3c3653e"
            ]
        }
    ],
    "Confidence": 100.0,
    "RowSpan": 1,
    "RowIndex": 1,
    "ColumnIndex": 1,
    "ColumnSpan": 1,
    "BlockType": "CELL",
    "Id": "505e9581-0d1c-42fb-a214-6ff736822e8c"
},
```

Jede Zelle hat eine Position in einer Tabelle, wobei die erste Zelle 1,1 ist. Im vorherigen Beispiel wird die Zelle mit dem WertNamebefindet sich in Zeile 1, Spalte 1. Die Zelle mit dem Wert123 Any Townbefindet sich in Zeile 2, Spalte 2. Ein Zellblockobjekt enthält diese Informationen im`RowIndex`und`ColumnIndex`unterscheiden sich nicht. Die untergeordnete Liste enthält die IDs für die WORD-Block-Objekte, die den Text enthalten, der sich innerhalb der Zelle befindet. Die Wörter in der Liste befinden sich in der Reihenfolge, in der sie erkannt werden, von oben links in der Zelle bis unten rechts in der Zelle. Im vorhergehenden Beispiel hat die Zelle eine untergeordnete ID mit dem Wert e9108c8e-0167-4482-989e-8b6cd3c3653e. Die folgende Ausgabe ist für den WORD-Block mit dem ID-Wert von e9108c8e-0167-4482-989e-8b6cd3c3653e:

```
"Geometry": {...},
"Text": "Name",
"TextType": "Printed",
"BlockType": "WORD",
"Confidence": 99.81139373779297,
"Id": "e9108c8e-0167-4482-989e-8b6cd3c3653e"
},
```

Auswahl-Elemente

Amazon Textract kann Auswahlelemente wie Optionsfelder (Optionsfelder) und Kontrollkästchen auf einer Dokumentseite erkennen. Selektionselemente können in Formular-Daten und in Tische aus. Wenn beispielsweise die folgende Tabelle in einem Formular erkannt wird, erkennt Amazon Textract die Kontrollkästchen in den Tabellenzellen.

	Stimme zu	Neutral	Stimme nicht zu
Guter Service	☑	☐	☐
Benutzerfreundlich	☐	☑	☐
Fairer Preis	☑	☐	☐

Erkannte Auswahlelemente werden als zurückgegebenBlockObjekte in den Antworten vonAnalyzeDocumentundGetDocumentAnalysisaus.

> **Note**
>
> Sie können dasFeatureTypesInput-Parameter zum Abrufen von Informationen über Schlüssel-Wert-Paare, Tabellen oder beides. Wenn Sie beispielsweise nach Tabellen filtern, enthält die Antwort die Auswahlelemente, die in Tabellen erkannt werden. Selektionselemente, die in Schlüssel-Wert-Paaren erkannt werden, sind nicht in der Antwort enthalten.

Informationen über ein Selektionselement sind in einemBlockObjekt des TypsSELECTION_ELEMENTaus. Um den Status eines wählbaren Elements zu ermitteln, verwenden Sie dieSelectionStatusfield desSELECTION_ELEMENTblockieren. Der Status kann entweder seinAUSGEWÄHLToderNICHT_SELECTEDaus. Beispielsweise würde der Wert vonSelectionStatusfür das vorherige Bild istAUSGEWÄHLTaus.

EINSELECTION_ELEMENT Block-Objekt ist entweder einem Schlüssel-Wert-Paar oder einer Tabellenzelle zugeordnet. EINSELECTION_ELEMENT Block-Objekt enthält Informationen zum Begrenzungsrahmen für ein Auswahlelement imGeometryfield. EINSELECTION_ELEMENT Blockobject ist kein Kind einesPAGE Block-Objekt.

Formulardaten (Schlüssel-Wert-Paare)

Ein Schlüssel-Wert-Paar wird verwendet, um ein Selektionselement darzustellen, das in einem Formular erkannt wird. DieKEYblock enthält den Text für das Auswahlelement. DieVALUEblock enthält den SELECTION_ELEMENT-Block. Das folgende Diagramm zeigt, wie Selektionselemente dargestellt werdenthe section called “Block”Objekte.

Weitere Informationen zu Schlüssel-Wert-Paaren finden Sie unterFormulardaten (Schlüssel-Wert-Paare)aus.

Das folgende JSON-Snippet zeigt den Schlüssel für ein Schlüssel-Wert-Paar, das ein Auswahlelement (male ☑) enthalten. Die untergeordnete ID (Id bd14cfd5-9005-498b-a7f3-45ceb171f0ff) ist die ID des WORD-Blocks, der den Text für das Auswahlelement enthält (männlich) enthalten. Die Wert-ID (Id 24aaac7f-fcce-49c7-a4f0-3688b05586d4) ist die ID desVALUEBlock, der das enthältSELECTION_ELEMENTBlock-Objekt.

```
{
    "Relationships": [
        {
            "Type": "VALUE",
            "Ids": [
                "24aaac7f-fcce-49c7-a4f0-3688b05586d4"  // Value containing Selection
 Element
            ]
        },
        {
            "Type": "CHILD",
            "Ids": [
                "bd14cfd5-9005-498b-a7f3-45ceb171f0ff"  // WORD - male
            ]
        }
    ],
    "Confidence": 94.15619659423828,
    "Geometry": {
        "BoundingBox": {
            "Width": 0.022914813831448555,
            "Top": 0.08072036504745483,
            "Left": 0.18966935575008392,
            "Height": 0.014860388822853565
        },
        "Polygon": [
            {
                "Y": 0.08072036504745483,
                "X": 0.18966935575008392
            },
            {
                "Y": 0.08072036504745483,
                "X": 0.21258416771888733
            },
            {
```

```
                "Y": 0.09558075666427612,
                "X": 0.21258416771888733
            },
            {
                "Y": 0.09558075666427612,
                "X": 0.18966935575008392
            }
        ]
    },
    "BlockType": "KEY_VALUE_SET",
    "EntityTypes": [
        "KEY"
    ],
    "Id": "a118dc43-d5f7-49a2-a20a-5f876d9ffd79"
}
```

Das folgende JSON-Snippet ist der WORD-Block für das WortMännlichaus. Der WORD-Block hat auch einen übergeordneten LINE-Block.

```
{
    "Geometry": {
        "BoundingBox": {
            "Width": 0.022464623674750328,
            "Top": 0.07842985540628433,
            "Left": 0.18863198161125183,
            "Height": 0.01617223583161831
        },
        "Polygon": [
            {
                "Y": 0.07842985540628433,
                "X": 0.18863198161125183
            },
            {
                "Y": 0.07842985540628433,
                "X": 0.2110965996980667
            },
            {
                "Y": 0.09460209310054779,
                "X": 0.2110965996980667
            },
            {
                "Y": 0.09460209310054779,
                "X": 0.18863198161125183
```

```
            }
        ]
    },
    "Text": "Male",
    "BlockType": "WORD",
    "Confidence": 54.06439208984375,
    "Id": "bd14cfd5-9005-498b-a7f3-45ceb171f0ff"
},
```

Der VALUE-Block hat ein Kind (Id f2f5e8cd-e73a-4e99-a095-053acd3b6bfb), das der SELECTION_ELEMENT-Block ist.

```
{
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "f2f5e8cd-e73a-4e99-a095-053acd3b6bfb"  // Selection element
            ]
        }
    ],
    "Confidence": 94.15619659423828,
    "Geometry": {
        "BoundingBox": {
            "Width": 0.017281491309404373,
            "Top": 0.07643391191959381,
            "Left": 0.2271782010793686,
            "Height": 0.026274094358086586
        },
        "Polygon": [
            {
                "Y": 0.07643391191959381,
                "X": 0.2271782010793686
            },
            {
                "Y": 0.07643391191959381,
                "X": 0.24445968866348267
            },
            {
                "Y": 0.10270800441503525,
                "X": 0.24445968866348267
            },
            {
```

```
                "Y": 0.10270800441503525,
                "X": 0.2271782010793686
            }
        ]
    },
    "BlockType": "KEY_VALUE_SET",
    "EntityTypes": [
        "VALUE"
    ],
    "Id": "24aaac7f-fcce-49c7-a4f0-3688b05586d4"
},
}
```

Der folgende JSON ist der SELECTION_ELEMENT-Block. Der Wert von`SelectionStatus`Gibt an, dass das Kontrollkästchen aktiviert ist.

```
{
    "Geometry": {
        "BoundingBox": {
            "Width": 0.020316146314144135,
            "Top": 0.07575977593660355,
            "Left": 0.22590067982673645,
            "Height": 0.027631107717752457
        },
        "Polygon": [
            {
                "Y": 0.07575977593660355,
                "X": 0.22590067982673645
            },
            {
                "Y": 0.07575977593660355,
                "X": 0.2462168186903
            },
            {
                "Y": 0.1033908873796463,
                "X": 0.2462168186903
            },
            {
                "Y": 0.1033908873796463,
                "X": 0.22590067982673645
            }
        ]
    },
```

```
    "BlockType": "SELECTION_ELEMENT",
    "SelectionStatus": "SELECTED",
    "Confidence": 74.14942932128906,
    "Id": "f2f5e8cd-e73a-4e99-a095-053acd3b6bfb"
}
```

Tabellen-Zellen

Amazon Textract kann Selektionselemente in einer Tabellenzelle erkennen. Die Zellen in der folgenden Tabelle haben beispielsweise Kontrollkästchen.

	Stimme zu	Neutral	Stimme nicht zu
Guter Service	☑	☐	☐
Benutzerfreundlich	☐	☑	☐
Fairer Preis	☑	☐	☐

EINCELLblock kann untergeordnete Elemente enthaltenSELECTION_ELEMENTObjekte für Selektionselemente sowie untergeordnete ElementeWORDBlöcke für erkannten Text.

Weitere Informationen zu Tabellen finden Sie unterTabellenaus.

Der TABLEBlock-Objekt für die vorherige Tabelle sieht ähnlich aus.

```
{
    "Geometry": {.....},
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "652c09eb-8945-473d-b1be-fa03ac055928",
                "37efc5cc-946d-42cd-aa04-e68e5ed4741d",
                "4a44940a-435a-4c5c-8a6a-7fea341fa295",
                "2de20014-9a3b-4e26-b453-0de755144b1a",
                "8ed78aeb-5c9a-4980-b669-9e08b28671d2",
                "1f8e1c68-2c97-47b2-847c-a19619c02ca9",
                "9927e1d1-6018-4960-ac17-aadb0a94f4d9",
                "68f0ed8b-a887-42a5-b618-f68b494a6034",
```

```
                "fcba16e0-6bd7-4ea5-b86e-36e8330b68ea",
                "2250357c-ae34-4ed9-86da-45dac5a5e903",
                "c63ad40d-5a14-4646-a8df-2d4304213dbc",   // Cell
                "2b8417dc-e65f-4fcd-aa0f-61a23f1e8cb0",
                "26c62932-72f0-4dc2-9893-1ae27829c060",
                "27f291cc-abf4-4c23-aa24-676abe99cb1e",
                "7e5ce028-1bcd-4d9f-ad42-15ac181c5b47",
                "bf32e3d2-efa2-4fc1-b09b-ab9cc52ff734"
            ]
        }
    ],
    "BlockType": "TABLE",
    "Confidence": 99.99993896484375,
    "Id": "f66eac36-2e74-406e-8032-14d1c14e0b86"
}
```

Die ZELLEBL0CKObjekt (Id c63ad40d-5a14-4646-a8df-2d4304213dbc) für die Zelle, die das Kontrollkästchen enthältGuter ServiceSieht wie folgt aus. Es beinhaltet ein Kind`Block`(Id = 26d122fd-c5f4-4b53-92c4-0ae92730ee1e) das ist der`SELECTION_ELEMENT` `Block`-Objekt für das Kontrollkästchen.

```
{
    "Geometry": {.....},
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "26d122fd-c5f4-4b53-92c4-0ae92730ee1e"  // Selection Element
            ]
        }
    ],
    "Confidence": 79.741689682006836,
    "RowSpan": 1,
    "RowIndex": 3,
    "ColumnIndex": 3,
    "ColumnSpan": 1,
    "BlockType": "CELL",
    "Id": "c63ad40d-5a14-4646-a8df-2d4304213dbc"
}
```

Das SELECTION_ELEMENT`Block`-Objekt für das Kontrollkästchen lautet wie folgt. Der Wert von`SelectionStatus`Gibt an, dass das Kontrollkästchen aktiviert ist.

```
{
    "Geometry": {.......},
    "BlockType": "SELECTION_ELEMENT",
    "SelectionStatus": "SELECTED",
    "Confidence": 88.79517364501953,
    "Id": "26d122fd-c5f4-4b53-92c4-0ae92730ee1e"
}
```

Antwortobjekte auf Rechnung und Wareneingang

Wenn Sie eine Rechnung oder eine Quittung an die AnalyzeExpense-API senden, wird eine Reihe von ExpenseDocuments -Objekten zurückgegeben. Jedes ExpenseDocument ist weiter unterteilt in`LineItemGroups`und`SummaryFields`aus. Die meisten Rechnungen und Belege enthalten Informationen wie den Kreditorennamen, die Belegnummer, das Wareneingangsdatum oder den Gesamtbetrag. AnalyzeExpense gibt diese Informationen zurück unter`SummaryFields`aus. Quittungen und Rechnungen enthalten auch Details zu den gekauften Artikeln. Die AnalyzeExpense-API gibt diese Informationen unter`LineItemGroups`aus. Die`ExpenseIndex`identifiziert eindeutig die Kosten und assoziiert die entsprechenden`SummaryFields`und`LineItemGroups`in diesen Kosten festgestellt.

Die detaillierteste Datenebene in der AnalyzeExpense-Antwort besteht aus`Type`,`ValueDetection`, und`LabelDetection`(Optional). Die einzelnen Entitäten sind:

- Typ: Bezieht sich darauf, welche Art von Informationen auf hohem Niveau erkannt werden.
- LabelDetection: Bezieht sich auf die Bezeichnung eines zugehörigen Wertes im Text des Dokuments.`LabelDetection`ist optional und wird nur zurückgegeben, wenn das Etikett geschrieben wurde.
- valueDetection: Bezieht sich auf den Wert des zurückgegebenen Labels oder Typs.

Die AnalyzeExpense-API erkennt auch`ITEM`,`QUANTITY`, und`PRICE`innerhalb von Einzelposten als normalisierte Felder. Wenn sich auf dem Belegbild einen anderen Text in einem Einzelposten befindet, wie z. B. SKU oder detaillierte Beschreibung, wird dieser als`EXPENSE_ROW`wie im folgenden Beispiel gezeigt:

```
            {
                                "Type": {
                                    "Text": "EXPENSE_ROW",
```

```
                "Confidence": 99.95216369628906
            },
            "ValueDetection": {
                "Text": "Banana 5 $2.5",
                "Geometry": {
                  …
                },
                "Confidence": 98.11214447021484
            }
```

Das obige Beispiel zeigt, wie die AnalyzeExpense-API die gesamte Zeile auf einer Quittung zurückgibt, die Informationen zu Einzelposten über 5 Bananen enthält, die für 2,5 USD verkauft werden.

Typ

Nachfolgend ist ein Beispiel für den Standard- oder Normtyp des Schlüssel-Wert-Paares:

```
{
     "PageNumber": 1,
     "Type": {
         "Text": "VENDOR_NAME",
         "Confidence": 70.0
     },
     "ValueDetection": {
         "Geometry": { ... },
         "Text": "AMAZON",
         "Confidence": 87.89806365966797
     }
 }
```

Auf der Quittung wurde „Händlername“ nicht explizit aufgeführt. Die API „Kosten analysieren“ erkannte das Dokument jedoch als Quittung und kategorisierte den Wert „AMAZON“ als Typ`VENDOR_NAME`aus.

LabelDetection

Es folgt ein Beispiel für Text, wie er auf einer Kundendokumentseite angezeigt wird:

```
            {
                 "PageNumber": 1,
                 "Type": {
                     "Text": "OTHER",
                     "Confidence": 70.0
                 },
                 "LabelDetection": {
                     "Geometry": { ... },
                     "Text": "CASHIER",
                     "Confidence": 88.19171142578125
                 },
                 "ValueDetection": {
                     "Geometry": { ... },
                     "Text": "Mina",
                     "Confidence": 87.89806365966797
                 }
             }
```

Das Beispieldokument enthielt „KASSIERER Mina“. Die API „Kosten analysieren“ hat den Ist-Wert extrahiert und unter`LabelDetection`aus. Bei implizierten Werten wie „Händlername“, bei denen der „Schlüssel“ nicht explizit in der Quittung angezeigt wird,`LabelDetection`wird nicht in das AnalyzeExpense-Element aufgenommen. In solchen Fällen wird die AnalyzeExpense-API nicht zurückgegeben`LabelDetection`aus.

valueDetection

Im Folgenden ist ein Beispiel, der den „Wert“ des Schlüssel-Wert-Paares zeigt.

```
            {
                 "PageNumber": 1,
                 "Type": {
                     "Text": "OTHER",
                     "Confidence": 70.0
                 },
                 "LabelDetection": {
                     "Geometry": { ... },
                     "Text": "CASHIER",
                     "Confidence": 88.19171142578125
                 },
```

```
                    "ValueDetection": {
                        "Geometry": { ... },
                        "Text": "Mina",
                        "Confidence": 87.89806365966797
                    }
                }
```

Im Beispiel enthielt das Dokument „KASSIERER Mina“. Die AnalyzeExpense-API erkannte den Kassiererwert als Mina und gab ihn unter`ValueDetection`aus.

Antwortobjekte der Identitä

Wenn Sie ein Ausweisdokument an die AnalyeID-API einreichen, wird eine Reihe von`IdentityDocumentField`Objekte. Jedes dieser Objekte enthält`Type`, und`Value`aus.`Type`zeichnet das normalisierte Feld auf, das Amazon Textract erkennt, und`Value`zeichnet den Text auf, der mit dem normalisierten Feld verknüpft ist.

Nachfolgend finden Sie ein Beispiel für`IdentityDocumentField`, der Kürze halber verkürzt.

```
{
    "DocumentMetadata": {
        "Pages": 1
    },
    "IdentityDocumentFields": [
        {
            "Type": {
                "Text": "first name"
            },
            "ValueDetection": {
                "Text": "jennifer",
                "Confidence": 99.99908447265625
            }
        },
        {
            "Type": {
                "Text": "last name"
            },
            "ValueDetection": {
                "Text": "sample",
                "Confidence": 99.99758911132812
            }
```

```
        },
```

Dies sind zwei Beispiele für IdentityDocumentFields, die von einer längeren Antwort abgeschnitten wurden. Es besteht eine Trennung zwischen dem erkannten Typ und dem Wert für diesen Typ. Hier ist es der Vor- und Nachname. Diese Struktur wiederholt sich mit allen enthaltenen Informationen. Wenn ein Typ nicht als normalisiertes Feld erkannt wird, wird er als „anderer“ aufgeführt.

Im Folgenden finden Sie eine Liste der normalisierten Felder für Führerscheine:

- Vorname
- Nachname
- zweiter Vorname
- Suffix
- Stadt in Adresse
- Postleitzahl in Adresse
- Status in Adresse
- Bezirk
- Nummer des Dokuments
- Ablaufdatum
- Geburtsdatum
- Name des Staates
- Datum der Ausgabe
- class
- Einschränkungen
- Befürwortungen
- ID-Typ
- Veteran
- address

Es folgt eine Liste der normalisierten Felder für US-Pässe:

- Vorname
- Nachname

- zweiter Vorname
- Nummer des Dokuments
- Ablaufdatum
- Geburtsdatum
- Geburtsort
- Datum der Ausgabe
- ID-Typ

Artikelspeicherort auf einer Dokumentseite

Amazon Textract Textract-Operationen geben den Speicherort und die Geometrie der Elemente zurück, die auf einer Dokumentseite gefunden wurden.DetectDocumentTextundGetDocumentTextDetectiongeben Sie die Position und Geometrie für Linien und Wörter zurück, währendAnalyzeDocumentundGetDocumentAnalysisGibt die Position und Geometrie von Schlüssel-Wert-Paaren, Tabellen, Zellen und Selektionselementen zurück.

Um zu ermitteln, wo sich ein Element auf einer Dokumentseite befindet, verwenden Sie den Begrenzungsrahmen (Geometry) Informationen, die von der Amazon Textract Textract-Operation in einemBlock-Objekt. Die`Geometry`Das -Objekt enthält zwei Arten von Standorten und geometrischen Informationen für erkannte Elemente:

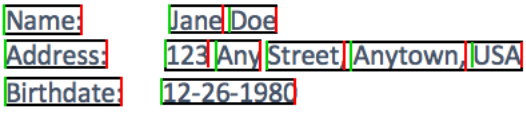
- Eine Achse ausgerichtetBoundingBoxDas -Objekt, das die linke obere Koordinate sowie die Breite und Höhe des Elements enthält.
- Ein Polygonobjekt, das den Umriss des Elements beschreibt, angegeben als Array vonPointObjekte, die enthaltenX(horizontale Achse) undY(vertikale Achse) Dokumentseitenkoordinaten jedes Punktes.

Der JSON für einen`Block`Das -Objekt sieht folgendermaßen aus. Beachten Sie das`BoundingBox`und`Polygon`unterscheiden sich nicht.

```
{
    "Geometry": {
        "BoundingBox": {
            "Width": 0.053907789289951324,
            "Top": 0.08913730084896088,
            "Left": 0.11085548996925354,
```

```
            "Height": 0.013171200640499592
        },
        "Polygon": [
            {
                "Y": 0.08985357731580734,
                "X": 0.11085548996925354
            },
            {
                "Y": 0.08913730084896088,
                "X": 0.16447919607162476
            },
            {
                "Y": 0.10159222036600113,
                "X": 0.16476328670978546
            },
            {
                "Y": 0.10230850428342819,
                "X": 0.11113958805799484
            }
        ]
    },
    "Text": "Name:",
    "TextType": "PRINTED",
    "BlockType": "WORD",
    "Confidence": 99.56285858154297,
    "Id": "c734fca6-c4c4-415c-b6c1-30f7510b72ee"
},
```


Sie können Geometrieinformationen verwenden, um Begrenzungsrahmen um erkannte Elemente zu zeichnen. Für ein Beispiel, das`BoundingBox`und`Polygon`Informationen zum Zeichnen von Feldern um Linien und vertikale Linien am Anfang und Ende jedes Wortes finden Sie unterErkennen von Dokumenttext mit Amazon Textractaus. Die Beispielausgabe ähnelt der folgenden.



Bounding Box

Ein Begrenzungsrahmen (`BoundingBox`) hat folgende Eigenschaften:

- Höhe — Die Höhe des Begrenzungsrahmens als Verhältnis der Gesamtseitenhöhe des Dokuments.

- Links — Die X-Koordinate des oberen linken Punktes des Begrenzungsrahmens als Verhältnis der Gesamtseitenbreite des Dokuments.
- Oben — Die Y-Koordinate des oberen linken Punktes des Begrenzungsrahmens als Verhältnis der Gesamtseitenhöhe des Dokuments.
- Breite — Die Breite des Begrenzungsrahmens als Verhältnis der Gesamtseitenbreite des Dokuments.

Jede BoundingBox-Eigenschaft hat einen Wert zwischen 0 und 1. Der Wert ist ein Verhältnis der gesamten Bildbreite (gilt für`Left`und`Width`) oder Höhe (gilt für`Height`und`Top`) enthalten. Beispiel: Bei einem Eingangsbild mit 700 x 200 Pixeln und einer linken oberen Koordinate des Begrenzungsrahmens (350,50) Pixeln gibt die API eine`Left`Wert von 0,5 (350/700) und a`Top`Wert von 0,25 (50/200).

Das folgende Diagramm zeigt den Bereich einer Dokumentseite, die jede BoundingBox-Eigenschaft abdeckt.

Um den Begrenzungsrahmen mit der richtigen Position und Größe anzuzeigen, müssen Sie die BoundingBox-Werte mit der Breite oder Höhe der Dokumentseite (abhängig vom gewünschten Wert) multiplizieren, um die Pixelwerte zu erhalten. Sie verwenden die Pixelwerte, um den Begrenzungsrahmen anzuzeigen. Ein Beispiel ist die Verwendung einer Dokumentseite mit 608 Pixel Breite x 588 Pixel Höhe und die folgenden Begrenzungsrahmenwerte für analysierten Text:

```
BoundingBox.Left: 0.3922065
BoundingBox.Top: 0.15567766
BoundingBox.Width: 0.284666
BoundingBox.Height: 0.2930403
```

Die Position des Text-Begrenzungsrahmens in Pixeln wird wie folgt berechnet:

```
Left coordinate = BoundingBox.Left (0.3922065) * document page width (608)
= 238

Top coordinate = BoundingBox.Top (0.15567766) * document page height (588)
= 91

Bounding box width = BoundingBox.Width (0.284666) * document page width
(608) = 173
```

`Bounding box height = BoundingBox.Height (0.2930403) * document page height (588) = 172`

Mit diesen Werten können Sie einen Begrenzungsrahmen um den analysierten Text herum anzeigen. Die folgenden Beispiele für Java und Python zeigen, wie ein Begrenzungsrahmen angezeigt wird.

Java

```
   public void ShowBoundingBox(int imageHeight, int imageWidth, BoundingBox box,
Graphics2D g2d) {

        float left = imageWidth * box.getLeft();
        float top = imageHeight * box.getTop();

        // Display bounding box.
        g2d.setColor(new Color(0, 212, 0));
        g2d.drawRect(Math.round(left / scale), Math.round(top / scale),
                Math.round((imageWidth * box.getWidth()) / scale),
Math.round((imageHeight * box.getHeight())) / scale);

   }
```

Python

In diesem Python-Beispiel wird das`response`den von zurückgegebenenDetectDocumentTextAPI-Operation.

```
def process_text_detection(response):

    # Get the text blocks
    blocks = response['Blocks']
    width, height = image.size
    draw = ImageDraw.Draw(image)
    print('Detected Document Text')

    # Create image showing bounding box/polygon the detected lines/text
    for block in blocks:

        draw = ImageDraw.Draw(image)

        if block['BlockType'] == "LINE":
            box=block['Geometry']['BoundingBox']
```

```
            left = width * box['Left']
            top = height * box['Top']
            draw.rectangle([left,top, left + (width * box['Width']), top +(height *
box['Height'])],outline='black')

    # Display the image
    image.show()

    return len(blocks)
```

Polygon

Das Polygon wurde zurückgegeben von`AnalyzeDocument`ist eine Reihe vonPointObjekte. EACH`Point`hat eine X- und Y-Koordinate für einen bestimmten Speicherort auf der Dokumentseite. Wie die BoundingBox-Koordinaten werden die Polygonkoordinaten auf die Breite und Höhe des Dokuments normalisiert und liegen zwischen 0 und 1.

Sie können Punkte im Polygon-Array verwenden, um einen feinkörnigen Begrenzungsrahmen um ein`Block`-Objekt. Sie berechnen die Position jedes Polygonpunkts auf der Dokumentseite mit der gleichen Technik, die für`BoundingBoxes`aus. Multiplizieren Sie die X-Koordinate mit der Seitenbreite des Dokuments und multiplizieren Sie die Y-Koordinate mit der Seitenhöhe des Dokuments.

Das folgende Beispiel zeigt, wie die vertikalen Linien eines Polygons angezeigt werden.

```
   public void ShowPolygonVerticals(int imageHeight, int imageWidth, List <Point>
points, Graphics2D g2d) {

       g2d.setColor(new Color(0, 212, 0));
       Object[] parry = points.toArray();
       g2d.setStroke(new BasicStroke(2));

       g2d.drawLine(Math.round(((Point) parry[0]).getX() * imageWidth),
               Math.round(((Point) parry[0]).getY() * imageHeight),
Math.round(((Point) parry[3]).getX() * imageWidth),
               Math.round(((Point) parry[3]).getY() * imageHeight));

       g2d.setColor(new Color(255, 0, 0));
       g2d.drawLine(Math.round(((Point) parry[1]).getX() * imageWidth),
               Math.round(((Point) parry[1]).getY() * imageHeight),
Math.round(((Point) parry[2]).getX() * imageWidth),
```

```
                Math.round(((Point) parry[2]).getY() * imageHeight));

    }
```

Erste Schritte mit Amazon Textract

Dieser Abschnitt enthält alle Themen für die ersten Schritte mit Amazon Textract. Falls Sie Amazon Textract zum ersten Mal benutzen, empfehlen wir, sich zuerst mit den Konzepten und der Terminologie in vertraut zu machen.Funktionsweise von Amazon Textractaus.

Sie können die API mit der Demonstration in der Amazon Textract Textract-Konsole ausprobieren. Weitere Informationen finden Sie unterhttps://console.aws.amazon.com/textract/aus.

Themen

- Schritt 1: Einrichten eines AWS-Kontos und Erstellen eines IAM-Benutzers
- Schritt 2: Einrichten derAWS CLIundAWS-SDKs
- Schritt 3: Erste Schritte mit derAWS CLIundAWSSDK API

Schritt 1: Einrichten eines AWS-Kontos und Erstellen eines IAM-Benutzers

Bevor Sie Amazon Textract zum ersten Mal verwenden können, führen Sie die folgenden Schritte aus:

1. Anmelden bei AWS.
2. Erstellen eines IAM-Benutzers.

Anmelden bei AWS

Wenn Sie sich für Amazon Web Services (AWS) anmelden, wird Ihr AWS-Konto automatisch für alle in AWS veröffentlichten Services registriert. Berechnet werden Ihnen aber nur die Services, die Sie nutzen.

Mit Amazon Textract zahlen Sie nur für die Ressourcen, die Sie wirklich nutzen. Weitere Informationen zu Amazon Textract Textract-Nutzungsgebühren finden Sie unterAmazon Textract — Preiseaus. Wenn Sie ein neuer AWS-Kunde sind, können Sie kostenlos mit Amazon Textract beginnen. Weitere Informationen finden Sie unter AWS Free Usage Tier (kostenloses Nutzungskontingent für AWS).

Wenn Sie bereits ein AWS-Konto haben, wechseln Sie zur nächsten Aufgabe. Wenn Sie noch kein AWS-Konto haben, führen Sie die folgenden Schritte zum Erstellen eines Kontos aus.

So erstellen Sie ein AWS-Konto

1. Öffnen Sie https://portal.aws.amazon.com/billing/signup.
2. Folgen Sie den Online-Anweisungen.

   Bei der Anmeldung müssen Sie auch einen Telefonanruf entgegennehmen und einen Verifizierungscode über die Telefontasten eingeben.

Notieren Sie Ihre AWS-Konto-ID. Sie benötigen sie im nächsten Schritt.

Erstellen eines IAM-Benutzers

Services in AWS, wie z. B. Amazon Textract, erfordern beim Zugriff die Eingabe von Anmeldeinformationen. Der Service kann feststellen, ob Sie über die Berechtigung für den Zugriff auf die Ressourcen im Besitz dieses Service verfügen. Für die Konsole müssen Sie Ihr Passwort eingeben. Sie können für Ihr AWS-Konto Zugriffsschlüssel erstellen, um auf die AWS CLI oder die API zuzugreifen. Wir raten Ihnen jedoch davon ab, mittels der Anmeldeinformationen für Ihr AWS-Konto auf AWS zuzugreifen. Stattdessen empfehlen wir Folgendes:

- Verwenden vonAWS Identity and Access Management(IAM) um einen IAM-Benutzer zu erstellen.
- Fügen Sie den Benutzer zu einer IAM-Gruppe mit Administratorberechtigungen hinzu.

Danach können Sie mithilfe einer speziellen URL und der Anmeldeinformationen des IAM-Benutzers auf AWS zugreifen.

Wenn Sie sich zwar bei AWS angemeldet, aber für sich selbst keinen IAM-Benutzer erstellt haben, können Sie mithilfe der IAM-Konsole einen Benutzer erstellen. Befolgen Sie die Schritte zum Einrichten des IAM-Benutzers in Ihrem Konto.

Erstellen eines IAM-Benutzers und Anmelden in der Konsole

1. Erstellen Sie einen IAM-Benutzer mit Administratorberechtigungen in Ihrem AWS-Konto. Detaillierte Anweisungen finden Sie unterErstellen Ihrer ersten Administratorgruppe für IAM-Benutzer und Ihrer ersten AdministratorgruppeimIAM User Guideaus.

2. Melden Sie sich als IAM-Benutzer mit einer speziellen URL in der AWS-Managementkonsole an. Weitere Informationen finden Sie unterWie sich Benutzer bei Ihrem Konto anmeldenimIAM User Guideaus.

> **Note**
>
> Ein IAM-Benutzer mit Administratorberechtigungen besitzt uneingeschränkten Zugriff auf dieAWSDienste in Ihrem Konto. Bei den Codebeispielen in diesem Handbuch wird davon ausgegangen, dass ein Benutzer mit der`AmazonTextractFullAccess`-Berechtigungen`AmazonS3ReadOnlyAccess`ist erforderlich für Beispiele, die auf Dokumente zugreifen, die in einem Amazon S3 S3-Bucket gespeichert sind. Je nach Ihren Sicherheitsanforderungen können Sie eine IAM-Gruppe verwenden, die auf diese Berechtigungen beschränkt ist. Weitere Informationen finden Sie unterIAM-Gruppen erstellenaus.

Weitere Informationen zu IAM finden Sie unter:

- AWS Identity and Access Management (IAM)
- Erste Schritte
- IAM Benutzerhandbuch

Nächster Schritt

Schritt 2: Einrichten derAWS CLIundAWS-SDKs

Schritt 2: Einrichten derAWS CLIundAWS-SDKs

Die folgenden Schritte veranschaulichen, wie Sie die AWS Command Line Interface (AWS CLI)- und AWS-SDKs installieren, die die Beispiele in dieser Dokumentation verwenden.

Es gibt eine Reihe verschiedener Möglichkeiten zum Authentifizieren von AWS-SDK-Aufrufen. In den Beispielen in diesem Handbuch wird davon ausgegangen, dass Sie ein Standardprofil für Anmeldeinformationen zum Aufrufen von AWS CLI-Befehlen und AWS-SDK-API-Operationen verwenden. Ihre Standardanmeldeinformationen funktionieren über Dienste hinweg. Wenn Sie also

Ihre Anmeldeinformationen bereits konfiguriert haben, müssen Sie dies nicht erneut tun. Wenn Sie jedoch einen weiteren Satz von Anmeldeinformationen für diesen Dienst erstellen möchten, können Sie ein Namenprofil erstellen. Weitere Informationen zum Erstellen von Profilen finden Sie untersiehe Benannte Profile.

Für eine Liste der verfügbarenAWSRegionen, siehe-Regionen und -EndpunkteimAmazon Web Services Services-Allgemeine Referenzaus.

So richten Sie die AWS CLI- und AWS-SDKs ein

1. Laden Sie die AWS CLI und die AWS-SDKs herunter, die Sie verwenden möchten, und installieren Sie sie. Dieser Leitfaden enthält Beispiele für dieAWS CLI, Java und Python. Weitere Informationen zu anderen AWS-SDKs finden Sie unter Tools für Amazon Web Services.

   - AWS CLI
   - AWS SDK für Java
   - AWS SDK für Python (Boto3)

2. Erstellen Sie einen Zugriffsschlüssel für den Benutzer, den Sie unter erstellt habenErstellen eines IAM-Benutzersaus.

   a. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
   b. Klicken Sie im Navigationsbereich auf Users.
   c. Wählen Sie den Namen des Benutzers aus, den Sie unter erstellt habenErstellen eines IAM-Benutzersaus.
   d. Wechseln Sie zur Registerkarte Security credentials (Sicherheitsanmeldeinformationen).
   e. Wählen Sie Create access key (Zugriffsschlüssel erstellen) aus. Wählen Sie dann Download .csv file (CSV-Datei herunterladen), um die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel in einer CSV-Datei auf dem Computer zu speichern. Speichern Sie die Datei an einem sicheren Ort. Sie haben keinen Zugriff auf den geheimen Zugriffsschlüssel mehr, nachdem das Dialogfeld geschlossen wird. Nachdem Sie die CSV-Datei heruntergeladen haben, klicken Sie aufSchließenaus.

3. Legen Sie Anmeldeinformationen in der AWS-Anmeldeinformationsprofil-Datei auf Ihrem lokalen System fest. Diese befindet sich an folgendem Speicherort:

   - `~/.aws/credentials`(Linux, macOS oder Unix).
   - `C:\Users\USERNAME\.aws\credentials`(Windows).

Die `.aws`-Ordner existiert vor Ihrer ersten Erstkonfiguration Ihrer AWS-Instance nicht. Wenn Sie Ihre Anmeldeinformationen zum ersten Mal mit der CLI konfigurieren, wird dieser Ordner erstellt. Weitere Informationen zu AWS-Anmeldeinformationen finden Sie unterKonfigurations- und Anmeldeinformationsdateieinstellungen

Diese Datei sollte Zeilen im folgenden Format enthalten:

```
[default]
aws_access_key_id = your_access_key_id
aws_secret_access_key = your_secret_access_key
```

Ersetzen Sie Ihre Zugriffsschlüssel-ID und Ihren geheimen Zugriffsschlüssel füryour_access_key_idundyour_secret_access_keyaus.

4. Legen Sie die AWS-Standardregion in AWS fest`config`Datei auf Ihrem lokalen System befindet sich an folgendem Speicherort:

   - `~/.aws/config`(Linux, macOS oder Unix).
   - `C:\Users\USERNAME\.aws\config`(Windows).

   Die `.aws`-Ordner existiert vor Ihrer ersten Erstkonfiguration Ihrer AWS-Instance nicht. Wenn Sie Ihre Anmeldeinformationen zum ersten Mal mit der CLI konfigurieren, wird dieser Ordner erstellt. Weitere Informationen zu AWS-Anmeldeinformationen finden Sie unterKonfigurations- und Anmeldeinformationsdateieinstellungen

   Diese Datei sollte die folgenden Zeilen enthalten:

   ```
   [default]
   region = your_aws_region
   ```

   Ersetzen Sie die AWS-Region, die Sie wünschen (z. B. „us-west-2") fürIhre_aws_regionaus.

   > **Note**
   >
   > Wenn Sie keine Region auswählen, wird standardmäßig „us-east-1" verwendet.

Nächster Schritt

Schritt 3: Erste Schritte mit derAWS CLIundAWSSDK API

Schritt 3: Erste Schritte mit derAWS CLIundAWSSDK API

Nachdem Sie dieAWS CLIundAWSBei SDKs, die Sie verwenden möchten, können Sie Anwendungen erstellen, die Amazon Textract verwenden. In den folgenden Themen werden erste Schritte mit Amazon Textract beschrieben.

- Analysieren von Dokumenttext mit Amazon Textract

Formatieren der AWS CLI-Beispiele

Die AWS CLI-Beispiele in diesem Handbuch sind für das Linux-Betriebssystem formatiert. Um die Beispiele mit Microsoft Windows anzuwenden, müssen Sie die JSON-Formatierung des `--document`-Parameters ändern und die Zeilenumbrüche von Backslashes (\) zu Einfügezeichen (^) ändern. Weitere Informationen zur JSON-Formatierung finden Sie unter Angeben von Parameterwerten für die AWS-Befehlszeilenschnittstelle.

Dokumente mit synchronen Operationen verarbeiten

Amazon Textract kann Text in einseitigen Dokumenten erkennen und analysieren, die als Bilder im JPEG-, PNG-, PDF- und TIFF-Format bereitgestellt werden. Die Vorgänge sind synchron und geben Ergebnisse nahezu in Echtzeit zurück. Weitere Informationen zu Dokumenten finden Sie unter Antwortobjekte für Texterkennung und Dokumentanalyse.

In diesem Abschnitt wird beschrieben, wie Sie Amazon Textract verwenden können, um Text in einem einseitigen Dokument synchron zu erkennen und zu analysieren. Informationen zum Erkennen und Analysieren von Text in mehrseitigen Dokumenten oder um JPEG- und PNG-Dokumente asynchron zu erkennen, finden Sie unterDokumente mit asynchronen Operationen verarbeitenaus.

Sie können Amazon Textract synchrone Operationen für folgende Zwecke verwenden:

- Texterkennung — Sie können Zeilen und Wörter in einem einseitigen Dokumentbild erkennen, indem Sie dieDetectDocumentTextverwenden. Weitere Informationen finden Sie unter Erkennen von Text .
- Textanalyse — Sie können Beziehungen zwischen erkanntem Text in einem einseitigen Dokument identifizieren, indem Sie dieAnalyzeDocumentverwenden. Weitere Informationen finden Sie unter Analysieren von Dokumenten .
- Rechnungs- und Wareneingangsanalyse — Mithilfe des AnalyzeExpense-Vorgangs können Sie finanzielle Beziehungen zwischen erkanntem Text auf einer einseitigen Rechnung oder Quittung identifizieren. Weitere Informationen finden Sie unterAnalysieren von Rechnungen und Belegen
- Identitätsdokumentanalyse — Sie können von der US-Regierung ausgestellte Ausweisdokumente analysieren und Informationen zusammen mit gängigen Arten von Informationen in Ausweisdokumenten extrahieren. Weitere Informationen finden Sie unterAnalysieren von Ausweisdokumentenaus.

Themen

- Aufrufen von Amazon Textract synchrone Operationen
- Erkennen von Dokumenttext mit Amazon Textract
- Analysieren von Dokumenttext mit Amazon Textract
- Rechnungen und Belege mit Amazon Textract analysieren
- Analysieren der Identitätsdokumentation mit Amazon Textract

Aufrufen von Amazon Textract synchrone Operationen

Amazon Textract Textract-Vorgänge verarbeiten Dokumentbilder, die in einem lokalen Dateisystem gespeichert sind, oder Dokumentbilder, die in einem Amazon S3 S3-Bucket gespeichert sind. Sie geben an, wo sich das Eingabedokument befindet, indem Sie dieDocumentEingabeparameter Das Dokumentbild kann entweder im PNG-, JPEG-, PDF- oder TIFF-Format vorliegen. Ergebnisse für synchrone Operationen werden sofort zurückgegeben und nicht zum Abrufen gespeichert.

Ein vollständiges Beispiel finden Sie unterErkennen von Dokumenttext mit Amazon Textractaus.

Anfrage

Im Folgenden wird beschrieben, wie Anfragen in Amazon Textract funktionieren.

Dokumente, die als Bildbytes übergeben wurden

Sie können ein Dokumentbild an einen Amazon Textract Textract-Vorgang übergeben, indem Sie das Image als base64-kodiertes Byte-Array übergeben. Ein Beispiel ist ein Dokumentabbild, das aus einem lokalen Dateisystem geladen wird. Wenn Sie ein verwenden, muss Ihr Code möglicherweise keine Dokumentdatei-Bytes codieren, wenn Sie ein verwenden.AWSSDK zum Aufrufen von Amazon Textract Textract-API-Operationen.

Die Bildbytes sind im`Bytes`field des`Document`Eingabeparameter Das folgende Beispiel zeigt die Eingabe-JSON für einen Amazon Textract Textract-Vorgang, der die Bildbytes im`Bytes`Eingabeparameter

```
{
    "Document": {
        "Bytes": "/9j/4AAQSk....."
    }
}
```

> **Note**
>
> Bei Verwendung derAWS CLIkönnen Sie keine Bildbytes an Amazon Textract Textract-Vorgänge übergeben. Stattdessen müssen Sie ein Bild referenzieren, das in einem Amazon S3 S3-Bucket gespeichert ist.

Der folgende Java-Code zeigt, wie Sie ein Bild aus einem lokalen Dateisystem laden und einen Amazon Textract Textract-Vorgang aufrufen.

```
String document="input.png";

ByteBuffer imageBytes;
try (InputStream inputStream = new FileInputStream(new File(document))) {
    imageBytes = ByteBuffer.wrap(IOUtils.toByteArray(inputStream));
}
AmazonTextract client = AmazonTextractClientBuilder.defaultClient();

DetectDocumentTextRequest request = new DetectDocumentTextRequest()
        .withDocument(new Document()
                .withBytes(imageBytes));


DetectDocumentTextResult result = client.detectDocumentText(request);
```

In einem Amazon S3 S3-Bucket gespeicherte Dokumente

Amazon Textract kann Dokumentbilder analysieren, die in einem Amazon S3 S3-Bucket gespeichert sind. Geben Sie den Bucket and Dateinamen mithilfe derS3Objectfield desDocumentEingabeparameter Das folgende Beispiel zeigt die Eingabe-JSON für einen Amazon Textract Textract-Vorgang, der ein Dokument verarbeitet, das in einem Amazon S3 Bucket gespeichert ist.

```
{
    "Document": {
        "S3Object": {
            "Bucket": "bucket",
            "Name": "input.png"
        }
    }
}
```

Das folgende Beispiel zeigt, wie Sie einen Amazon Textract Textract-Vorgang mithilfe eines Bildes aufrufen, das in einem Amazon S3 Bucket gespeichert ist.

```
String document="input.png";
String bucket="bucket";
```

```
AmazonTextract client = AmazonTextractClientBuilder.defaultClient();

DetectDocumentTextRequest request = new DetectDocumentTextRequest()
        .withDocument(new Document()
                .withS3Object(new S3Object()
                        .withName(document)
                        .withBucket(bucket)));

DetectDocumentTextResult result = client.detectDocumentText(request);
```

Antwort

Das folgende Beispiel zeigt die JSON-Antwort von einem Aufruf an`DetectDocumentText`aus. Weitere Informationen finden Sie unter Erkennen von Text .

```
{
{
  "DocumentMetadata": {
    "Pages": 1
  },
  "Blocks": [
    {
      "BlockType": "PAGE",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.9995205998420715,
          "Height": 1.0,
          "Left": 0.0,
          "Top": 0.0
        },
        "Polygon": [
          {
            "X": 0.0,
            "Y": 0.0
          },
          {
            "X": 0.9995205998420715,
            "Y": 2.297314024515845E-16
          },
          {
            "X": 0.9995205998420715,
            "Y": 1.0
          },
```

```
          {
            "X": 0.0,
            "Y": 1.0
          }
        ]
      },
      "Id": "ca4b9171-7109-4adb-a811-e09bbe4834dd",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "26085884-d005-4144-b4c2-4d83dc50739b",
            "ee9d01bc-d91c-401d-8c0a-eec76f5f7862",
            "404bb3d3-d7ab-4008-a195-5dec87a08664",
            "8ae1b4ba-67c1-4486-bd20-54f461886ce9",
            "47aab5ab-be2c-4c73-97c7-d0a45454e843",
            "dd06bb49-6a56-4ea7-beec-a2aa09835c3c",
            "8837153d-81b8-4031-a49f-83a3d81803c2",
            "5dae3b74-9e95-4b62-99b7-93b88fe70648",
            "4508da80-64d8-42a8-8846-cfafe6eab10c",
            "e87be7a9-5519-42e1-b18e-ae10e2d3ed13",
            "f04bb223-d075-41c3-b328-7354611c826b",
            "a234f0e8-67de-46f4-a7c7-0bbe8d5159ce",
            "61b20e27-ff8a-450a-a8b1-bc0259f82fd6",
            "445f4fdd-c77b-4a7b-a2fc-6ca07cfe9ed7",
            "359f3870-7183-43f5-b638-970f5cefe4d5",
            "b9deea0a-244c-4d54-b774-cf03fbaaa8b1",
            "e2a43881-f620-44f2-b067-500ce7dc8d4d",
            "41756974-64ef-432d-b4b2-34702505975a",
            "93d96d32-8b4a-4a98-9578-8b4df4f227a6",
            "bc907357-63d6-43c0-ab87-80d7e76d377e",
            "2d727ca7-3acb-4bb9-a564-5885c90e9325",
            "f32a5989-cbfb-41e6-b0fc-ce1c77c014bd",
            "e0ba06d0-dbb6-4962-8047-8cac3adfe45a",
            "b6ed204d-ae01-4b75-bb91-c85d4147a37e",
            "ac4b9ee0-c9b2-4239-a741-5753e5282033",
            "ebc18885-48d7-45b8-90e3-d172b4357802",
            "babf6360-789e-49c1-9c78-0784acc14a0c"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
```

```
      "Confidence": 99.93761444091797,
      "Text": "Employment Application",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.3391372561454773,
          "Height": 0.06906412541866302,
          "Left": 0.29548385739326477,
          "Top": 0.027493247762322426
        },
        "Polygon": [
          {
            "X": 0.29548385739326477,
            "Y": 0.027493247762322426
          },
          {
            "X": 0.6346210837364197,
            "Y": 0.027493247762322426
          },
          {
            "X": 0.6346210837364197,
            "Y": 0.0965573713183403
          },
          {
            "X": 0.29548385739326477,
            "Y": 0.0965573713183403
          }
        ]
      },
      "Id": "26085884-d005-4144-b4c2-4d83dc50739b",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "ed48dacc-d089-498f-8e93-1cee1e5f39f3",
            "ac7370f3-cbb7-4cd9-a8f9-bdcb2252caaf"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.91246795654297,
      "Text": "Application Information",
      "Geometry": {
```

```
        "BoundingBox": {
          "Width": 0.19878505170345306,
          "Height": 0.03754019737243652,
          "Left": 0.03988289833068848,
          "Top": 0.14050349593162537
        },
        "Polygon": [
          {
            "X": 0.03988289833068848,
            "Y": 0.14050349593162537
          },
          {
            "X": 0.23866795003414154,
            "Y": 0.14050349593162537
          },
          {
            "X": 0.23866795003414154,
            "Y": 0.1780436933040619
          },
          {
            "X": 0.03988289833068848,
            "Y": 0.1780436933040619
          }
        ]
      },
      "Id": "ee9d01bc-d91c-401d-8c0a-eec76f5f7862",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "efe3fc6d-becb-4520-80ee-49a329386aee",
            "c2260852-6cfd-4a71-9fc6-62b2f9b02355"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.88693237304688,
      "Text": "Full Name: Jane Doe",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.16733919084072113,
          "Height": 0.031106337904930115,
```

```
          "Left": 0.03899926319718361,
          "Top": 0.21361036598682404
        },
        "Polygon": [
          {
            "X": 0.03899926319718361,
            "Y": 0.21361036598682404
          },
          {
            "X": 0.20633845031261444,
            "Y": 0.21361036598682404
          },
          {
            "X": 0.20633845031261444,
            "Y": 0.24471670389175415
          },
          {
            "X": 0.03899926319718361,
            "Y": 0.24471670389175415
          }
        ]
      },
      "Id": "404bb3d3-d7ab-4008-a195-5dec87a08664",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "e94eb587-9545-4215-b0fc-8e8cb1172958",
            "090aeba5-8428-4b7a-a54b-7a95a774120e",
            "64ff0abb-736b-4a6b-aa8d-ad2c0086ae1d",
            "565ffc30-89d6-4295-b8c6-d22b4ed76584"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.9206314086914,
      "Text": "Phone Number: 555-0100",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.3115004599094391,
          "Height": 0.047169625759124756,
          "Left": 0.03604753687977791,
```

```
          "Top": 0.2812676727771759
        },
        "Polygon": [
          {
            "X": 0.03604753687977791,
            "Y": 0.2812676727771759
          },
          {
            "X": 0.3475480079650879,
            "Y": 0.2812676727771759
          },
          {
            "X": 0.3475480079650879,
            "Y": 0.32843729853630066
          },
          {
            "X": 0.03604753687977791,
            "Y": 0.32843729853630066
          }
        ]
      },
      "Id": "8ae1b4ba-67c1-4486-bd20-54f461886ce9",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "d782f847-225b-4a1b-b52d-f252f8221b1f",
            "fa69c5cd-c80d-4fac-81df-569edae8d259",
            "d4bbc0f1-ae02-41cf-a26f-8a1e899968cc"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.48902893066406,
      "Text": "Home Address: 123 Any Street, Any Town. USA",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.7431139945983887,
          "Height": 0.09577702730894089,
          "Left": 0.03359385207295418,
          "Top": 0.3258342146873474
        },
```

```
        "Polygon": [
          {
            "X": 0.03359385207295418,
            "Y": 0.3258342146873474
          },
          {
            "X": 0.7767078280448914,
            "Y": 0.3258342146873474
          },
          {
            "X": 0.7767078280448914,
            "Y": 0.4216112196445465
          },
          {
            "X": 0.03359385207295418,
            "Y": 0.4216112196445465
          }
        ]
      },
      "Id": "47aab5ab-be2c-4c73-97c7-d0a45454e843",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "acfbed90-4a00-42c6-8a90-d0a0756eea36",
            "046c8a40-bb0e-4718-9c71-954d3630e1dd",
            "82b838bc-4591-4287-8dea-60c94a4925e4",
            "5cdcde7a-f5a6-4231-a941-b6396e42e7ba",
            "beafd497-185f-487e-b070-db4df5803e94",
            "ef1b77fb-8ba6-41fe-ba53-dce039af22ed",
            "7b555310-e7f8-4cd2-bb3d-dcec37f3d90e",
            "b479c24d-448d-40ef-9ed5-36a6ef08e5c7"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.89382934570312,
      "Text": "Mailing Address: same as above",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.26575741171836853,
          "Height": 0.039571404457092285,
```

```
          "Left": 0.03068041242659092,
          "Top": 0.43351811170578003
        },
        "Polygon": [
          {
            "X": 0.03068041242659092,
            "Y": 0.43351811170578003
          },
          {
            "X": 0.2964377999305725,
            "Y": 0.43351811170578003
          },
          {
            "X": 0.2964377999305725,
            "Y": 0.4730895161628723
          },
          {
            "X": 0.03068041242659092,
            "Y": 0.4730895161628723
          }
        ]
      },
      "Id": "dd06bb49-6a56-4ea7-beec-a2aa09835c3c",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "d7261cdc-6ac5-4711-903c-4598fe94952d",
            "287f80c3-6db2-4dd7-90ec-5f017c80aa31",
            "ce31c3ad-b51e-4068-be64-5fc9794bc1bc",
            "e96eb92c-6774-4d6f-8f4a-68a7618d4c66",
            "88b85c05-427a-4d4f-8cc4-3667234e8364"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 94.67343139648438,
      "Text": "Previous Employment History",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.3309842050075531,
          "Height": 0.051920413970947266,
```

```
          "Left": 0.3194798231124878,
          "Top": 0.5172380208969116
        },
        "Polygon": [
          {
            "X": 0.3194798231124878,
            "Y": 0.5172380208969116
          },
          {
            "X": 0.6504639983177185,
            "Y": 0.5172380208969116
          },
          {
            "X": 0.6504639983177185,
            "Y": 0.5691584348678589
          },
          {
            "X": 0.3194798231124878,
            "Y": 0.5691584348678589
          }
        ]
      },
      "Id": "8837153d-81b8-4031-a49f-83a3d81803c2",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "8b324501-bf38-4ce9-9777-6514b7ade760",
            "b0cea99a-5045-464d-ac8a-a63ab0470995",
            "b92a6ee5-ca59-44dc-9c47-534c133b11e7"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.66949462890625,
      "Text": "Start Date",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08310240507125854,
          "Height": 0.030944595113396645,
          "Left": 0.034429505467414856,
          "Top": 0.6123942136764526
```

```
        },
        "Polygon": [
          {
            "X": 0.034429505467414856,
            "Y": 0.6123942136764526
          },
          {
            "X": 0.1175319030880928,
            "Y": 0.6123942136764526
          },
          {
            "X": 0.1175319030880928,
            "Y": 0.6433387994766235
          },
          {
            "X": 0.034429505467414856,
            "Y": 0.6433387994766235
          }
        ]
      },
      "Id": "5dae3b74-9e95-4b62-99b7-93b88fe70648",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "ffe8b8e0-df59-4ac5-9aba-6b54b7c51b45",
            "91e582cd-9871-4e9c-93cc-848baa426338"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.86717224121094,
      "Text": "End Date",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.07581500709056854,
          "Height": 0.03223184868693352,
          "Left": 0.14846202731132507,
          "Top": 0.6120467782020569
        },
        "Polygon": [
          {
```

```
            "X": 0.14846202731132507,
            "Y": 0.6120467782020569
          },
          {
            "X": 0.22427703440189362,
            "Y": 0.6120467782020569
          },
          {
            "X": 0.22427703440189362,
            "Y": 0.6442786455154419
          },
          {
            "X": 0.14846202731132507,
            "Y": 0.6442786455154419
          }
        ]
      },
      "Id": "4508da80-64d8-42a8-8846-cfafe6eab10c",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "7c97b56b-699f-49b0-93f4-98e6d90b107c",
            "7af04e27-0c15-447e-a569-b30edb99a133"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.9539794921875,
      "Text": "Employer Name",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.1347292959690094,
          "Height": 0.0392492413520813,
          "Left": 0.2647075653076172,
          "Top": 0.6140711903572083
        },
        "Polygon": [
          {
            "X": 0.2647075653076172,
            "Y": 0.6140711903572083
          },
```

```
          {
            "X": 0.3994368314743042,
            "Y": 0.6140711903572083
          },
          {
            "X": 0.3994368314743042,
            "Y": 0.6533204317092896
          },
          {
            "X": 0.2647075653076172,
            "Y": 0.6533204317092896
          }
        ]
      },
      "Id": "e87be7a9-5519-42e1-b18e-ae10e2d3ed13",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "a9bfeb55-75cd-47cd-b953-728e602a3564",
            "9f0f9c06-d02c-4b07-bb39-7ade70be2c1b"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.35584259033203,
      "Text": "Position Held",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.11393272876739502,
          "Height": 0.03415105864405632,
          "Left": 0.49973347783088684,
          "Top": 0.614840030670166
        },
        "Polygon": [
          {
            "X": 0.49973347783088684,
            "Y": 0.614840030670166
          },
          {
            "X": 0.6136661767959595,
            "Y": 0.614840030670166
```

```
          },
          {
            "X": 0.6136661767959595,
            "Y": 0.6489911079406738
          },
          {
            "X": 0.49973347783088684,
            "Y": 0.6489911079406738
          }
        ]
      },
      "Id": "f04bb223-d075-41c3-b328-7354611c826b",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "6d5edf02-845c-40e0-9514-e56d0d652ae0",
            "3297ab59-b237-45fb-ae60-a108f0c95ac2"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.9817886352539,
      "Text": "Reason for leaving",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.16511960327625275,
          "Height": 0.04062700271606445,
          "Left": 0.7430596351623535,
          "Top": 0.6116235852241516
        },
        "Polygon": [
          {
            "X": 0.7430596351623535,
            "Y": 0.6116235852241516
          },
          {
            "X": 0.9081792235374451,
            "Y": 0.6116235852241516
          },
          {
            "X": 0.9081792235374451,
```

```
            "Y": 0.6522505879402161
          },
          {
            "X": 0.7430596351623535,
            "Y": 0.6522505879402161
          }
        ]
      },
      "Id": "a234f0e8-67de-46f4-a7c7-0bbe8d5159ce",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "f4b8cf26-d2da-4a76-8345-69562de3cc11",
            "386d4a63-1194-4c0e-a18d-4d074a0b1f93",
            "a8622541-1896-4d54-8d10-7da2c800ec5c"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.77413177490234,
      "Text": "1/15/2009",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08799663186073303,
          "Height": 0.03832906484603882,
          "Left": 0.03175082430243492,
          "Top": 0.691371738910675
        },
        "Polygon": [
          {
            "X": 0.03175082430243492,
            "Y": 0.691371738910675
          },
          {
            "X": 0.11974745243787766,
            "Y": 0.691371738910675
          },
          {
            "X": 0.11974745243787766,
            "Y": 0.7297008037567139
          },
```

```
          {
            "X": 0.03175082430243492,
            "Y": 0.7297008037567139
          }
        ]
      },
      "Id": "61b20e27-ff8a-450a-a8b1-bc0259f82fd6",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "da7a6482-0964-49a4-bc7d-56942ff3b4e1"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.72286224365234,
      "Text": "6/30/2011",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08843101561069489,
          "Height": 0.03991425037384033,
          "Left": 0.14642837643623352,
          "Top": 0.6919752955436707
        },
        "Polygon": [
          {
            "X": 0.14642837643623352,
            "Y": 0.6919752955436707
          },
          {
            "X": 0.2348593920469284,
            "Y": 0.6919752955436707
          },
          {
            "X": 0.2348593920469284,
            "Y": 0.731889545917511
          },
          {
            "X": 0.14642837643623352,
            "Y": 0.731889545917511
          }
```

```
        ]
      },
      "Id": "445f4fdd-c77b-4a7b-a2fc-6ca07cfe9ed7",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "5a8da66a-ecce-4ee9-a765-a46d6cdc6cde"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.86936950683594,
      "Text": "Any Company",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.11800950765609741,
          "Height": 0.03943679481744766,
          "Left": 0.2626699209213257,
          "Top": 0.6972727179527283
        },
        "Polygon": [
          {
            "X": 0.2626699209213257,
            "Y": 0.6972727179527283
          },
          {
            "X": 0.3806794285774231,
            "Y": 0.6972727179527283
          },
          {
            "X": 0.3806794285774231,
            "Y": 0.736709475517273
          },
          {
            "X": 0.2626699209213257,
            "Y": 0.736709475517273
          }
        ]
      },
      "Id": "359f3870-7183-43f5-b638-970f5cefe4d5",
      "Relationships": [
```

```
        {
          "Type": "CHILD",
          "Ids": [
            "77749c2b-aa7f-450e-8dd2-62bcaf253ba2",
            "713bad19-158d-4e3e-b01f-f5707ddb04e5"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.582275390625,
      "Text": "Assistant baker",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.13280922174453735,
          "Height": 0.032666124403476715,
          "Left": 0.49814170598983765,
          "Top": 0.699238657951355
        },
        "Polygon": [
          {
            "X": 0.49814170598983765,
            "Y": 0.699238657951355
          },
          {
            "X": 0.630950927734375,
            "Y": 0.699238657951355
          },
          {
            "X": 0.630950927734375,
            "Y": 0.7319048047065735
          },
          {
            "X": 0.49814170598983765,
            "Y": 0.7319048047065735
          }
        ]
      },
      "Id": "b9deea0a-244c-4d54-b774-cf03fbaaa8b1",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
```

```
            "989944f9-f684-4714-87d8-9ad9a321d65c",
            "ae82e2aa-1601-4e0c-8340-1db7ad0c9a31"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.96180725097656,
      "Text": "relocated",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08668994903564453,
          "Height": 0.033302485942840576,
          "Left": 0.7426905632019043,
          "Top": 0.6974037289619446
        },
        "Polygon": [
          {
            "X": 0.7426905632019043,
            "Y": 0.6974037289619446
          },
          {
            "X": 0.8293805122375488,
            "Y": 0.6974037289619446
          },
          {
            "X": 0.8293805122375488,
            "Y": 0.7307062149047852
          },
          {
            "X": 0.7426905632019043,
            "Y": 0.7307062149047852
          }
        ]
      },
      "Id": "e2a43881-f620-44f2-b067-500ce7dc8d4d",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "a9cf9a8c-fdaa-413e-9346-5a28a98aebdb"
          ]
        }
```

```
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.98190307617188,
      "Text": "7/1/2011",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09747002273797989,
          "Height": 0.07067441940307617,
          "Left": 0.028500309213995934,
          "Top": 0.7745237946510315
        },
        "Polygon": [
          {
            "X": 0.028500309213995934,
            "Y": 0.7745237946510315
          },
          {
            "X": 0.12597033381462097,
            "Y": 0.7745237946510315
          },
          {
            "X": 0.12597033381462097,
            "Y": 0.8451982140541077
          },
          {
            "X": 0.028500309213995934,
            "Y": 0.8451982140541077
          }
        ]
      },
      "Id": "41756974-64ef-432d-b4b2-34702505975a",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "0f711065-1872-442a-ba6d-8fababaa452a"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
```

```
      "Confidence": 99.98418426513672,
      "Text": "8/10/2013",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.10664612054824829,
          "Height": 0.06439518928527832,
          "Left": 0.14159755408763885,
          "Top": 0.7791688442230225
        },
        "Polygon": [
          {
            "X": 0.14159755408763885,
            "Y": 0.7791688442230225
          },
          {
            "X": 0.24824367463588715,
            "Y": 0.7791688442230225
          },
          {
            "X": 0.24824367463588715,
            "Y": 0.8435640335083008
          },
          {
            "X": 0.14159755408763885,
            "Y": 0.8435640335083008
          }
        ]
      },
      "Id": "93d96d32-8b4a-4a98-9578-8b4df4f227a6",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "a92d8eef-db28-45ba-801a-5da0f589d277"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.98075866699219,
      "Text": "Example Corp.",
      "Geometry": {
        "BoundingBox": {
```

```
          "Width": 0.2114926278591156,
          "Height": 0.058415766805410385,
          "Left": 0.26764172315597534,
          "Top": 0.794414758682251
        },
        "Polygon": [
          {
            "X": 0.26764172315597534,
            "Y": 0.794414758682251
          },
          {
            "X": 0.47913435101509094,
            "Y": 0.794414758682251
          },
          {
            "X": 0.47913435101509094,
            "Y": 0.8528305292129517
          },
          {
            "X": 0.26764172315597534,
            "Y": 0.8528305292129517
          }
        ]
      },
      "Id": "bc907357-63d6-43c0-ab87-80d7e76d377e",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "d6962efb-34ab-4ffb-9f2f-5f263e813558",
            "1876c8ea-d3e8-4c39-870e-47512b3b5080"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.91166687011719,
      "Text": "Baker",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09931200742721558,
          "Height": 0.06008726358413696,
          "Left": 0.5098910331726074,
```

```
          "Top": 0.787897527217865
        },
        "Polygon": [
          {
            "X": 0.5098910331726074,
            "Y": 0.787897527217865
          },
          {
            "X": 0.609203040599823,
            "Y": 0.787897527217865
          },
          {
            "X": 0.609203040599823,
            "Y": 0.847984790802002
          },
          {
            "X": 0.5098910331726074,
            "Y": 0.847984790802002
          }
        ]
      },
      "Id": "2d727ca7-3acb-4bb9-a564-5885c90e9325",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "00adeaef-ed57-44eb-b8a9-503575236d62"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.93852233886719,
      "Text": "better opp.",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.18919607996940613,
          "Height": 0.06994765996932983,
          "Left": 0.7428008317947388,
          "Top": 0.7928366661071777
        },
        "Polygon": [
          {
```

```
            "X": 0.7428008317947388,
            "Y": 0.7928366661071777
          },
          {
            "X": 0.9319968819618225,
            "Y": 0.7928366661071777
          },
          {
            "X": 0.9319968819618225,
            "Y": 0.8627843260765076
          },
          {
            "X": 0.7428008317947388,
            "Y": 0.8627843260765076
          }
        ]
      },
      "Id": "f32a5989-cbfb-41e6-b0fc-ce1c77c014bd",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "c0fc9a58-7a4b-4f69-bafd-2cff32be2665",
            "bf6dc8ee-2fb3-4b6c-aee4-31e96912a2d8"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.92573547363281,
      "Text": "8/15/2013",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.10257463902235031,
          "Height": 0.05412459373474121,
          "Left": 0.027909137308597565,
          "Top": 0.8608770370483398
        },
        "Polygon": [
          {
            "X": 0.027909137308597565,
            "Y": 0.8608770370483398
          },
```

```
          {
            "X": 0.13048377633094788,
            "Y": 0.8608770370483398
          },
          {
            "X": 0.13048377633094788,
            "Y": 0.915001630783081
          },
          {
            "X": 0.027909137308597565,
            "Y": 0.915001630783081
          }
        ]
      },
      "Id": "e0ba06d0-dbb6-4962-8047-8cac3adfe45a",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "5384f860-f857-4a94-9438-9dfa20eed1c6"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.99625396728516,
      "Text": "Present",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09982697665691376,
          "Height": 0.06888341903686523,
          "Left": 0.1420602649450302,
          "Top": 0.8511748909950256
        },
        "Polygon": [
          {
            "X": 0.1420602649450302,
            "Y": 0.8511748909950256
          },
          {
            "X": 0.24188724160194397,
            "Y": 0.8511748909950256
          },
```

```
          {
            "X": 0.24188724160194397,
            "Y": 0.9200583100318909
          },
          {
            "X": 0.1420602649450302,
            "Y": 0.9200583100318909
          }
        ]
      },
      "Id": "b6ed204d-ae01-4b75-bb91-c85d4147a37e",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "0bb96ed6-b2e6-4da4-90b3-b85561bbd89d"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.9826431274414,
      "Text": "AnyCompany",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.18611276149749756,
          "Height": 0.08581399917602539,
          "Left": 0.2615866959095001,
          "Top": 0.869536280632019
        },
        "Polygon": [
          {
            "X": 0.2615866959095001,
            "Y": 0.869536280632019
          },
          {
            "X": 0.4476994574069977,
            "Y": 0.869536280632019
          },
          {
            "X": 0.4476994574069977,
            "Y": 0.9553502798080444
          },
```

```
          {
            "X": 0.2615866959095001,
            "Y": 0.9553502798080444
          }
        ]
      },
      "Id": "ac4b9ee0-c9b2-4239-a741-5753e5282033",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "25343360-d906-440a-88b7-92eb89e95949"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.99549102783203,
      "Text": "head baker",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.1937451809644699,
          "Height": 0.056156039237976074,
          "Left": 0.49359121918678284,
          "Top": 0.8702592849731445
        },
        "Polygon": [
          {
            "X": 0.49359121918678284,
            "Y": 0.8702592849731445
          },
          {
            "X": 0.6873363852500916,
            "Y": 0.8702592849731445
          },
          {
            "X": 0.6873363852500916,
            "Y": 0.9264153242111206
          },
          {
            "X": 0.49359121918678284,
            "Y": 0.9264153242111206
          }
```

```
        ]
      },
      "Id": "ebc18885-48d7-45b8-90e3-d172b4357802",
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "0ef3c194-8322-4575-94f1-82819ee57e3a",
            "d296acd9-3e9a-4985-95f8-f863614f2c46"
          ]
        }
      ]
    },
    {
      "BlockType": "LINE",
      "Confidence": 99.98360443115234,
      "Text": "N/A, current",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.22544169425964355,
          "Height": 0.06588292121887207,
          "Left": 0.7411766648292542,
          "Top": 0.8722732067108154
        },
        "Polygon": [
          {
            "X": 0.7411766648292542,
            "Y": 0.8722732067108154
          },
          {
            "X": 0.9666183590888977,
            "Y": 0.8722732067108154
          },
          {
            "X": 0.9666183590888977,
            "Y": 0.9381561279296875
          },
          {
            "X": 0.7411766648292542,
            "Y": 0.9381561279296875
          }
        ]
      },
      "Id": "babf6360-789e-49c1-9c78-0784acc14a0c",
```

```
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "195cfb5b-ae06-4203-8520-4e4b0a73b5ce",
            "549ef3f9-3a13-4b77-bc25-fb2e0996839a"
          ]
        }
      ]
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.94815826416016,
      "Text": "Employment",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.17462396621704102,
          "Height": 0.06266549974679947,
          "Left": 0.29548385739326477,
          "Top": 0.03389188274741173
        },
        "Polygon": [
          {
            "X": 0.29548385739326477,
            "Y": 0.03389188274741173
          },
          {
            "X": 0.4701078236103058,
            "Y": 0.03389188274741173
          },
          {
            "X": 0.4701078236103058,
            "Y": 0.0965573862195015
          },
          {
            "X": 0.29548385739326477,
            "Y": 0.0965573862195015
          }
        ]
      },
      "Id": "ed48dacc-d089-498f-8e93-1cee1e5f39f3"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 99.92706298828125,
      "Text": "Application",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.15933875739574432,
          "Height": 0.062391020357608795,
          "Left": 0.47528234124183655,
          "Top": 0.027493247762322426
        },
        "Polygon": [
          {
            "X": 0.47528234124183655,
            "Y": 0.027493247762322426
          },
          {
            "X": 0.6346211433410645,
            "Y": 0.027493247762322426
          },
          {
            "X": 0.6346211433410645,
            "Y": 0.08988427370786667
          },
          {
            "X": 0.47528234124183655,
            "Y": 0.08988427370786667
          }
        ]
      },
      "Id": "ac7370f3-cbb7-4cd9-a8f9-bdcb2252caaf"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.9821548461914,
      "Text": "Application",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09610454738140106,
          "Height": 0.03656719997525215,
          "Left": 0.03988289833068848,
          "Top": 0.14147649705410004
        },
```

```
        "Polygon": [
          {
            "X": 0.03988289833068848,
            "Y": 0.14147649705410004
          },
          {
            "X": 0.13598744571208954,
            "Y": 0.14147649705410004
          },
          {
            "X": 0.13598744571208954,
            "Y": 0.1780436933040619
          },
          {
            "X": 0.03988289833068848,
            "Y": 0.1780436933040619
          }
        ]
      },
      "Id": "efe3fc6d-becb-4520-80ee-49a329386aee"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.84278106689453,
      "Text": "Information",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.10029315203428268,
          "Height": 0.03209415823221207,
          "Left": 0.13837480545043945,
          "Top": 0.14050349593162537
        },
        "Polygon": [
          {
            "X": 0.13837480545043945,
            "Y": 0.14050349593162537
          },
          {
            "X": 0.23866795003414154,
            "Y": 0.14050349593162537
          },
          {
            "X": 0.23866795003414154,
```

```
            "Y": 0.17259766161441803
          },
          {
            "X": 0.13837480545043945,
            "Y": 0.17259766161441803
          }
        ]
      },
      "Id": "c2260852-6cfd-4a71-9fc6-62b2f9b02355"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.83993530273438,
      "Text": "Full",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.03039788082242012,
          "Height": 0.031106330454349518,
          "Left": 0.03899926319718361,
          "Top": 0.21361036598682404
        },
        "Polygon": [
          {
            "X": 0.03899926319718361,
            "Y": 0.21361036598682404
          },
          {
            "X": 0.06939714401960373,
            "Y": 0.21361036598682404
          },
          {
            "X": 0.06939714401960373,
            "Y": 0.24471670389175415
          },
          {
            "X": 0.03899926319718361,
            "Y": 0.24471670389175415
          }
        ]
      },
      "Id": "e94eb587-9545-4215-b0fc-8e8cb1172958"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 99.93611907958984,
      "Text": "Name:",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.05555811896920204,
          "Height": 0.030184319242835045,
          "Left": 0.07123806327581406,
          "Top": 0.2137702852487564
        },
        "Polygon": [
          {
            "X": 0.07123806327581406,
            "Y": 0.2137702852487564
          },
          {
            "X": 0.1267961859703064,
            "Y": 0.2137702852487564
          },
          {
            "X": 0.1267961859703064,
            "Y": 0.2439546138048172
          },
          {
            "X": 0.07123806327581406,
            "Y": 0.2439546138048172
          }
        ]
      },
      "Id": "090aeba5-8428-4b7a-a54b-7a95a774120e"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.91043853759766,
      "Text": "Jane",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.03905024006962776,
          "Height": 0.02941947989165783,
          "Left": 0.12933772802352905,
          "Top": 0.214289128780365
        },
```

```
        "Polygon": [
          {
            "X": 0.12933772802352905,
            "Y": 0.214289128780365
          },
          {
            "X": 0.16838796436786652,
            "Y": 0.214289128780365
          },
          {
            "X": 0.16838796436786652,
            "Y": 0.24370861053466797
          },
          {
            "X": 0.12933772802352905,
            "Y": 0.24370861053466797
          }
        ]
      },
      "Id": "64ff0abb-736b-4a6b-aa8d-ad2c0086ae1d"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.86123657226562,
      "Text": "Doe",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.035229459404945374,
          "Height": 0.030427640303969383,
          "Left": 0.17110899090766907,
          "Top": 0.21377210319042206
        },
        "Polygon": [
          {
            "X": 0.17110899090766907,
            "Y": 0.21377210319042206
          },
          {
            "X": 0.20633845031261444,
            "Y": 0.21377210319042206
          },
          {
            "X": 0.20633845031261444,
```

```
            "Y": 0.244199737906456
          },
          {
            "X": 0.17110899090766907,
            "Y": 0.244199737906456
          }
        ]
      },
      "Id": "565ffc30-89d6-4295-b8c6-d22b4ed76584"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.92633056640625,
      "Text": "Phone",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.052783288061618805,
          "Height": 0.03104414977133274,
          "Left": 0.03604753687977791,
          "Top": 0.28701552748680115
        },
        "Polygon": [
          {
            "X": 0.03604753687977791,
            "Y": 0.28701552748680115
          },
          {
            "X": 0.08883082121610641,
            "Y": 0.28701552748680115
          },
          {
            "X": 0.08883082121610641,
            "Y": 0.31805968284606934
          },
          {
            "X": 0.03604753687977791,
            "Y": 0.31805968284606934
          }
        ]
      },
      "Id": "d782f847-225b-4a1b-b52d-f252f8221b1f"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 99.86275482177734,
      "Text": "Number:",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.07424934208393097,
          "Height": 0.030300479382276535,
          "Left": 0.0915418416261673,
          "Top": 0.28639692068099976
        },
        "Polygon": [
          {
            "X": 0.0915418416261673,
            "Y": 0.28639692068099976
          },
          {
            "X": 0.16579118371009827,
            "Y": 0.28639692068099976
          },
          {
            "X": 0.16579118371009827,
            "Y": 0.3166973888874054
          },
          {
            "X": 0.0915418416261673,
            "Y": 0.3166973888874054
          }
        ]
      },
      "Id": "fa69c5cd-c80d-4fac-81df-569edae8d259"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.97282409667969,
      "Text": "555-0100",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.17021971940994263,
          "Height": 0.047169629484415054,
          "Left": 0.17732827365398407,
          "Top": 0.2812676727771759
        },
```

```
        "Polygon": [
          {
            "X": 0.17732827365398407,
            "Y": 0.2812676727771759
          },
          {
            "X": 0.3475480079650879,
            "Y": 0.2812676727771759
          },
          {
            "X": 0.3475480079650879,
            "Y": 0.32843729853630066
          },
          {
            "X": 0.17732827365398407,
            "Y": 0.32843729853630066
          }
        ]
      },
      "Id": "d4bbc0f1-ae02-41cf-a26f-8a1e899968cc"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.66238403320312,
      "Text": "Home",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.049357783049345016,
          "Height": 0.03134990110993385,
          "Left": 0.03359385207295418,
          "Top": 0.36172014474868774
        },
        "Polygon": [
          {
            "X": 0.03359385207295418,
            "Y": 0.36172014474868774
          },
          {
            "X": 0.0829516351222992,
            "Y": 0.36172014474868774
          },
          {
            "X": 0.0829516351222992,
```

```
            "Y": 0.3930700421333313
          },
          {
            "X": 0.03359385207295418,
            "Y": 0.3930700421333313
          }
        ]
      },
      "Id": "acfbed90-4a00-42c6-8a90-d0a0756eea36"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.6871109008789,
      "Text": "Address:",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.07411003112792969,
          "Height": 0.0314042791724205,
          "Left": 0.08516156673431396,
          "Top": 0.3600046932697296
        },
        "Polygon": [
          {
            "X": 0.08516156673431396,
            "Y": 0.3600046932697296
          },
          {
            "X": 0.15927159786224365,
            "Y": 0.3600046932697296
          },
          {
            "X": 0.15927159786224365,
            "Y": 0.3914089798927307
          },
          {
            "X": 0.08516156673431396,
            "Y": 0.3914089798927307
          }
        ]
      },
      "Id": "046c8a40-bb0e-4718-9c71-954d3630e1dd"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 99.93781280517578,
      "Text": "123",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.05761868134140968,
          "Height": 0.05008566007018089,
          "Left": 0.1750781387090683,
          "Top": 0.35484206676483154
        },
        "Polygon": [
          {
            "X": 0.1750781387090683,
            "Y": 0.35484206676483154
          },
          {
            "X": 0.23269681632518768,
            "Y": 0.35484206676483154
          },
          {
            "X": 0.23269681632518768,
            "Y": 0.40492773056030273
          },
          {
            "X": 0.1750781387090683,
            "Y": 0.40492773056030273
          }
        ]
      },
      "Id": "82b838bc-4591-4287-8dea-60c94a4925e4"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.96530151367188,
      "Text": "Any",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.06814215332269669,
          "Height": 0.06354366987943649,
          "Left": 0.2550157308578491,
          "Top": 0.35471394658088684
        },
```

```
        "Polygon": [
          {
            "X": 0.2550157308578491,
            "Y": 0.35471394658088684
          },
          {
            "X": 0.3231579065322876,
            "Y": 0.35471394658088684
          },
          {
            "X": 0.3231579065322876,
            "Y": 0.41825762391090393
          },
          {
            "X": 0.2550157308578491,
            "Y": 0.41825762391090393
          }
        ]
      },
      "Id": "5cdcde7a-f5a6-4231-a941-b6396e42e7ba"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.87527465820312,
      "Text": "Street,",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.12156613171100616,
          "Height": 0.05449587106704712,
          "Left": 0.3357025980949402,
          "Top": 0.3550415635108948
        },
        "Polygon": [
          {
            "X": 0.3357025980949402,
            "Y": 0.3550415635108948
          },
          {
            "X": 0.45726871490478516,
            "Y": 0.3550415635108948
          },
          {
            "X": 0.45726871490478516,
```

```
            "Y": 0.4095374345779419
          },
          {
            "X": 0.3357025980949402,
            "Y": 0.4095374345779419
          }
        ]
      },
      "Id": "beafd497-185f-487e-b070-db4df5803e94"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.99514770507812,
      "Text": "Any",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.07748188823461533,
          "Height": 0.07339789718389511,
          "Left": 0.47723668813705444,
          "Top": 0.3482133150100708
        },
        "Polygon": [
          {
            "X": 0.47723668813705444,
            "Y": 0.3482133150100708
          },
          {
            "X": 0.554718554019928,
            "Y": 0.3482133150100708
          },
          {
            "X": 0.554718554019928,
            "Y": 0.4216112196445465
          },
          {
            "X": 0.47723668813705444,
            "Y": 0.4216112196445465
          }
        ]
      },
      "Id": "ef1b77fb-8ba6-41fe-ba53-dce039af22ed"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 96.80656433105469,
      "Text": "Town.",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.11213835328817368,
          "Height": 0.057233039289712906,
          "Left": 0.5563329458236694,
          "Top": 0.3331930637359619
        },
        "Polygon": [
          {
            "X": 0.5563329458236694,
            "Y": 0.3331930637359619
          },
          {
            "X": 0.6684713363647461,
            "Y": 0.3331930637359619
          },
          {
            "X": 0.6684713363647461,
            "Y": 0.3904260993003845
          },
          {
            "X": 0.5563329458236694,
            "Y": 0.3904260993003845
          }
        ]
      },
      "Id": "7b555310-e7f8-4cd2-bb3d-dcec37f3d90e"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.98260498046875,
      "Text": "USA",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08771833777427673,
          "Height": 0.05706935003399849,
          "Left": 0.6889894604682922,
          "Top": 0.3258342146873474
        },
```

```
        "Polygon": [
          {
            "X": 0.6889894604682922,
            "Y": 0.3258342146873474
          },
          {
            "X": 0.7767078280448914,
            "Y": 0.3258342146873474
          },
          {
            "X": 0.7767078280448914,
            "Y": 0.3829035460948944
          },
          {
            "X": 0.6889894604682922,
            "Y": 0.3829035460948944
          }
        ]
      },
      "Id": "b479c24d-448d-40ef-9ed5-36a6ef08e5c7"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.9583969116211,
      "Text": "Mailing",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.06291338801383972,
          "Height": 0.03957144916057587,
          "Left": 0.03068041242659092,
          "Top": 0.43351811170578003
        },
        "Polygon": [
          {
            "X": 0.03068041242659092,
            "Y": 0.43351811170578003
          },
          {
            "X": 0.09359379857778549,
            "Y": 0.43351811170578003
          },
          {
            "X": 0.09359379857778549,
```

```
            "Y": 0.4730895459651947
          },
          {
            "X": 0.03068041242659092,
            "Y": 0.4730895459651947
          }
        ]
      },
      "Id": "d7261cdc-6ac5-4711-903c-4598fe94952d"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.87476348876953,
      "Text": "Address:",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.07364854216575623,
          "Height": 0.03147412836551666,
          "Left": 0.0954652726650238,
          "Top": 0.43450701236724854
        },
        "Polygon": [
          {
            "X": 0.0954652726650238,
            "Y": 0.43450701236724854
          },
          {
            "X": 0.16911381483078003,
            "Y": 0.43450701236724854
          },
          {
            "X": 0.16911381483078003,
            "Y": 0.465981125831604
          },
          {
            "X": 0.0954652726650238,
            "Y": 0.465981125831604
          }
        ]
      },
      "Id": "287f80c3-6db2-4dd7-90ec-5f017c80aa31"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 99.94071960449219,
      "Text": "same",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.04640670120716095,
          "Height": 0.026415130123496056,
          "Left": 0.17156922817230225,
          "Top": 0.44010937213897705
        },
        "Polygon": [
          {
            "X": 0.17156922817230225,
            "Y": 0.44010937213897705
          },
          {
            "X": 0.2179759293794632,
            "Y": 0.44010937213897705
          },
          {
            "X": 0.2179759293794632,
            "Y": 0.46652451157569885
          },
          {
            "X": 0.17156922817230225,
            "Y": 0.46652451157569885
          }
        ]
      },
      "Id": "ce31c3ad-b51e-4068-be64-5fc9794bc1bc"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.76510620117188,
      "Text": "as",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.02041218988597393,
          "Height": 0.025104399770498276,
          "Left": 0.2207803726196289,
          "Top": 0.44124215841293335
        },
```

```
        "Polygon": [
          {
            "X": 0.2207803726196289,
            "Y": 0.44124215841293335
          },
          {
            "X": 0.24119256436824799,
            "Y": 0.44124215841293335
          },
          {
            "X": 0.24119256436824799,
            "Y": 0.4663465619087219
          },
          {
            "X": 0.2207803726196289,
            "Y": 0.4663465619087219
          }
        ]
      },
      "Id": "e96eb92c-6774-4d6f-8f4a-68a7618d4c66"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.9301528930664,
      "Text": "above",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.05268359184265137,
          "Height": 0.03216424956917763,
          "Left": 0.24375422298908234,
          "Top": 0.4354657828807831
        },
        "Polygon": [
          {
            "X": 0.24375422298908234,
            "Y": 0.4354657828807831
          },
          {
            "X": 0.2964377999305725,
            "Y": 0.4354657828807831
          },
          {
            "X": 0.2964377999305725,
```

```
            "Y": 0.4676300287246704
          },
          {
            "X": 0.24375422298908234,
            "Y": 0.4676300287246704
          }
        ]
      },
      "Id": "88b85c05-427a-4d4f-8cc4-3667234e8364"
    },
    {
      "BlockType": "WORD",
      "Confidence": 85.3905029296875,
      "Text": "Previous",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09860499948263168,
          "Height": 0.04000622034072876,
          "Left": 0.3194798231124878,
          "Top": 0.5194430351257324
        },
        "Polygon": [
          {
            "X": 0.3194798231124878,
            "Y": 0.5194430351257324
          },
          {
            "X": 0.4180848002433777,
            "Y": 0.5194430351257324
          },
          {
            "X": 0.4180848002433777,
            "Y": 0.5594492554664612
          },
          {
            "X": 0.3194798231124878,
            "Y": 0.5594492554664612
          }
        ]
      },
      "Id": "8b324501-bf38-4ce9-9777-6514b7ade760"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 99.14524841308594,
      "Text": "Employment",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.14039960503578186,
          "Height": 0.04645847901701927,
          "Left": 0.4214291274547577,
          "Top": 0.5219109654426575
        },
        "Polygon": [
          {
            "X": 0.4214291274547577,
            "Y": 0.5219109654426575
          },
          {
            "X": 0.5618287324905396,
            "Y": 0.5219109654426575
          },
          {
            "X": 0.5618287324905396,
            "Y": 0.568369448184967
          },
          {
            "X": 0.4214291274547577,
            "Y": 0.568369448184967
          }
        ]
      },
      "Id": "b0cea99a-5045-464d-ac8a-a63ab0470995"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.48454284667969,
      "Text": "History",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08361124992370605,
          "Height": 0.05192042887210846,
          "Left": 0.5668527483940125,
          "Top": 0.5172380208969116
        },
```

```
        "Polygon": [
          {
            "X": 0.5668527483940125,
            "Y": 0.5172380208969116
          },
          {
            "X": 0.6504639983177185,
            "Y": 0.5172380208969116
          },
          {
            "X": 0.6504639983177185,
            "Y": 0.5691584348678589
          },
          {
            "X": 0.5668527483940125,
            "Y": 0.5691584348678589
          }
        ]
      },
      "Id": "b92a6ee5-ca59-44dc-9c47-534c133b11e7"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.78699493408203,
      "Text": "Start",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.041341401636600494,
          "Height": 0.030926469713449478,
          "Left": 0.034429505467414856,
          "Top": 0.6124123334884644
        },
        "Polygon": [
          {
            "X": 0.034429505467414856,
            "Y": 0.6124123334884644
          },
          {
            "X": 0.07577090710401535,
            "Y": 0.6124123334884644
          },
          {
            "X": 0.07577090710401535,
```

```
            "Y": 0.6433387994766235
          },
          {
            "X": 0.034429505467414856,
            "Y": 0.6433387994766235
          }
        ]
      },
      "Id": "ffe8b8e0-df59-4ac5-9aba-6b54b7c51b45"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.55198669433594,
      "Text": "Date",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.03923053666949272,
          "Height": 0.03072454035282135,
          "Left": 0.07830137014389038,
          "Top": 0.6123942136764526
        },
        "Polygon": [
          {
            "X": 0.07830137014389038,
            "Y": 0.6123942136764526
          },
          {
            "X": 0.1175319105386734,
            "Y": 0.6123942136764526
          },
          {
            "X": 0.1175319105386734,
            "Y": 0.6431187391281128
          },
          {
            "X": 0.07830137014389038,
            "Y": 0.6431187391281128
          }
        ]
      },
      "Id": "91e582cd-9871-4e9c-93cc-848baa426338"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 99.8897705078125,
      "Text": "End",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.03212086856365204,
          "Height": 0.03193363919854164,
          "Left": 0.14846202731132507,
          "Top": 0.6120467782020569
        },
        "Polygon": [
          {
            "X": 0.14846202731132507,
            "Y": 0.6120467782020569
          },
          {
            "X": 0.1805828958749771,
            "Y": 0.6120467782020569
          },
          {
            "X": 0.1805828958749771,
            "Y": 0.6439804434776306
          },
          {
            "X": 0.14846202731132507,
            "Y": 0.6439804434776306
          }
        ]
      },
      "Id": "7c97b56b-699f-49b0-93f4-98e6d90b107c"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.8445816040039,
      "Text": "Date",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.03987143933773041,
          "Height": 0.03142518177628517,
          "Left": 0.1844055950641632,
          "Top": 0.612853467464447
        },
```

```
        "Polygon": [
          {
            "X": 0.1844055950641632,
            "Y": 0.612853467464447
          },
          {
            "X": 0.22427703440189362,
            "Y": 0.612853467464447
          },
          {
            "X": 0.22427703440189362,
            "Y": 0.6442786455154419
          },
          {
            "X": 0.1844055950641632,
            "Y": 0.6442786455154419
          }
        ]
      },
      "Id": "7af04e27-0c15-447e-a569-b30edb99a133"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.9652328491211,
      "Text": "Employer",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08150768280029297,
          "Height": 0.0392492301762104,
          "Left": 0.2647075653076172,
          "Top": 0.6140711903572083
        },
        "Polygon": [
          {
            "X": 0.2647075653076172,
            "Y": 0.6140711903572083
          },
          {
            "X": 0.34621524810791016,
            "Y": 0.6140711903572083
          },
          {
            "X": 0.34621524810791016,
```

```
            "Y": 0.6533204317092896
          },
          {
            "X": 0.2647075653076172,
            "Y": 0.6533204317092896
          }
        ]
      },
      "Id": "a9bfeb55-75cd-47cd-b953-728e602a3564"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.94273376464844,
      "Text": "Name",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.05018233880400658,
          "Height": 0.03248906135559082,
          "Left": 0.34925445914268494,
          "Top": 0.6162016987800598
        },
        "Polygon": [
          {
            "X": 0.34925445914268494,
            "Y": 0.6162016987800598
          },
          {
            "X": 0.3994368016719818,
            "Y": 0.6162016987800598
          },
          {
            "X": 0.3994368016719818,
            "Y": 0.6486907601356506
          },
          {
            "X": 0.34925445914268494,
            "Y": 0.6486907601356506
          }
        ]
      },
      "Id": "9f0f9c06-d02c-4b07-bb39-7ade70be2c1b"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 98.85071563720703,
      "Text": "Position",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.07007700204849243,
          "Height": 0.03255689889192581,
          "Left": 0.49973347783088684,
          "Top": 0.6164342164993286
        },
        "Polygon": [
          {
            "X": 0.49973347783088684,
            "Y": 0.6164342164993286
          },
          {
            "X": 0.5698104500770569,
            "Y": 0.6164342164993286
          },
          {
            "X": 0.5698104500770569,
            "Y": 0.6489911079406738
          },
          {
            "X": 0.49973347783088684,
            "Y": 0.6489911079406738
          }
        ]
      },
      "Id": "6d5edf02-845c-40e0-9514-e56d0d652ae0"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.86096954345703,
      "Text": "Held",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.04017873853445053,
          "Height": 0.03292537108063698,
          "Left": 0.5734874606132507,
          "Top": 0.614840030670166
        },
```

```
        "Polygon": [
          {
            "X": 0.5734874606132507,
            "Y": 0.614840030670166
          },
          {
            "X": 0.6136662364006042,
            "Y": 0.614840030670166
          },
          {
            "X": 0.6136662364006042,
            "Y": 0.6477653980255127
          },
          {
            "X": 0.5734874606132507,
            "Y": 0.6477653980255127
          }
        ]
      },
      "Id": "3297ab59-b237-45fb-ae60-a108f0c95ac2"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.97740936279297,
      "Text": "Reason",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.06497219949960709,
          "Height": 0.03248770162463188,
          "Left": 0.7430596351623535,
          "Top": 0.6136704087257385
        },
        "Polygon": [
          {
            "X": 0.7430596351623535,
            "Y": 0.6136704087257385
          },
          {
            "X": 0.8080317974090576,
            "Y": 0.6136704087257385
          },
          {
            "X": 0.8080317974090576,
```

```
            "Y": 0.6461580991744995
          },
          {
            "X": 0.7430596351623535,
            "Y": 0.6461580991744995
          }
        ]
      },
      "Id": "f4b8cf26-d2da-4a76-8345-69562de3cc11"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.98371887207031,
      "Text": "for",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.029645200818777084,
          "Height": 0.03462234139442444,
          "Left": 0.8108851909637451,
          "Top": 0.6117717623710632
        },
        "Polygon": [
          {
            "X": 0.8108851909637451,
            "Y": 0.6117717623710632
          },
          {
            "X": 0.8405303955078125,
            "Y": 0.6117717623710632
          },
          {
            "X": 0.8405303955078125,
            "Y": 0.6463940739631653
          },
          {
            "X": 0.8108851909637451,
            "Y": 0.6463940739631653
          }
        ]
      },
      "Id": "386d4a63-1194-4c0e-a18d-4d074a0b1f93"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 99.98424530029297,
      "Text": "leaving",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.06517849862575531,
          "Height": 0.040626998990774155,
          "Left": 0.8430007100105286,
          "Top": 0.6116235852241516
        },
        "Polygon": [
          {
            "X": 0.8430007100105286,
            "Y": 0.6116235852241516
          },
          {
            "X": 0.9081792235374451,
            "Y": 0.6116235852241516
          },
          {
            "X": 0.9081792235374451,
            "Y": 0.6522505879402161
          },
          {
            "X": 0.8430007100105286,
            "Y": 0.6522505879402161
          }
        ]
      },
      "Id": "a8622541-1896-4d54-8d10-7da2c800ec5c"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.77413177490234,
      "Text": "1/15/2009",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08799663186073303,
          "Height": 0.03832906112074852,
          "Left": 0.03175082430243492,
          "Top": 0.691371738910675
        },
```

```
        "Polygon": [
          {
            "X": 0.03175082430243492,
            "Y": 0.691371738910675
          },
          {
            "X": 0.11974745243787766,
            "Y": 0.691371738910675
          },
          {
            "X": 0.11974745243787766,
            "Y": 0.7297008037567139
          },
          {
            "X": 0.03175082430243492,
            "Y": 0.7297008037567139
          }
        ]
      },
      "Id": "da7a6482-0964-49a4-bc7d-56942ff3b4e1"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.72286224365234,
      "Text": "6/30/2011",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08843102306127548,
          "Height": 0.03991425037384033,
          "Left": 0.14642837643623352,
          "Top": 0.6919752955436707
        },
        "Polygon": [
          {
            "X": 0.14642837643623352,
            "Y": 0.6919752955436707
          },
          {
            "X": 0.2348593920469284,
            "Y": 0.6919752955436707
          },
          {
            "X": 0.2348593920469284,
```

```
            "Y": 0.731889545917511
          },
          {
            "X": 0.14642837643623352,
            "Y": 0.731889545917511
          }
        ]
      },
      "Id": "5a8da66a-ecce-4ee9-a765-a46d6cdc6cde"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.92295837402344,
      "Text": "Any",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.034067559987306595,
          "Height": 0.037968240678310394,
          "Left": 0.2626699209213257,
          "Top": 0.6972727179527283
        },
        "Polygon": [
          {
            "X": 0.2626699209213257,
            "Y": 0.6972727179527283
          },
          {
            "X": 0.2967374622821808,
            "Y": 0.6972727179527283
          },
          {
            "X": 0.2967374622821808,
            "Y": 0.7352409362792969
          },
          {
            "X": 0.2626699209213257,
            "Y": 0.7352409362792969
          }
        ]
      },
      "Id": "77749c2b-aa7f-450e-8dd2-62bcaf253ba2"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 99.81578063964844,
      "Text": "Company",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08160992711782455,
          "Height": 0.03890080004930496,
          "Left": 0.29906952381134033,
          "Top": 0.6978086829185486
        },
        "Polygon": [
          {
            "X": 0.29906952381134033,
            "Y": 0.6978086829185486
          },
          {
            "X": 0.3806794583797455,
            "Y": 0.6978086829185486
          },
          {
            "X": 0.3806794583797455,
            "Y": 0.736709475517273
          },
          {
            "X": 0.29906952381134033,
            "Y": 0.736709475517273
          }
        ]
      },
      "Id": "713bad19-158d-4e3e-b01f-f5707ddb04e5"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.37964630126953,
      "Text": "Assistant",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.0789310410618782,
          "Height": 0.03139699995517731,
          "Left": 0.49814170598983765,
          "Top": 0.7005078196525574
        },
```

```
        "Polygon": [
          {
            "X": 0.49814170598983765,
            "Y": 0.7005078196525574
          },
          {
            "X": 0.5770727396011353,
            "Y": 0.7005078196525574
          },
          {
            "X": 0.5770727396011353,
            "Y": 0.7319048047065735
          },
          {
            "X": 0.49814170598983765,
            "Y": 0.7319048047065735
          }
        ]
      },
      "Id": "989944f9-f684-4714-87d8-9ad9a321d65c"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.784912109375,
      "Text": "baker",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.050264399498701096,
          "Height": 0.03237773850560188,
          "Left": 0.5806865096092224,
          "Top": 0.699238657951355
        },
        "Polygon": [
          {
            "X": 0.5806865096092224,
            "Y": 0.699238657951355
          },
          {
            "X": 0.630950927734375,
            "Y": 0.699238657951355
          },
          {
            "X": 0.630950927734375,
```

```
            "Y": 0.7316163778305054
          },
          {
            "X": 0.5806865096092224,
            "Y": 0.7316163778305054
          }
        ]
      },
      "Id": "ae82e2aa-1601-4e0c-8340-1db7ad0c9a31"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.96180725097656,
      "Text": "relocated",
      "TextType": "PRINTED",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08668994158506393,
          "Height": 0.03330250084400177,
          "Left": 0.7426905632019043,
          "Top": 0.6974037289619446
        },
        "Polygon": [
          {
            "X": 0.7426905632019043,
            "Y": 0.6974037289619446
          },
          {
            "X": 0.8293805122375488,
            "Y": 0.6974037289619446
          },
          {
            "X": 0.8293805122375488,
            "Y": 0.7307062149047852
          },
          {
            "X": 0.7426905632019043,
            "Y": 0.7307062149047852
          }
        ]
      },
      "Id": "a9cf9a8c-fdaa-413e-9346-5a28a98aebdb"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 99.98190307617188,
      "Text": "7/1/2011",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09747002273797989,
          "Height": 0.07067439705133438,
          "Left": 0.028500309213995934,
          "Top": 0.7745237946510315
        },
        "Polygon": [
          {
            "X": 0.028500309213995934,
            "Y": 0.7745237946510315
          },
          {
            "X": 0.12597033381462097,
            "Y": 0.7745237946510315
          },
          {
            "X": 0.12597033381462097,
            "Y": 0.8451982140541077
          },
          {
            "X": 0.028500309213995934,
            "Y": 0.8451982140541077
          }
        ]
      },
      "Id": "0f711065-1872-442a-ba6d-8fababaa452a"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.98418426513672,
      "Text": "8/10/2013",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.10664612054824829,
          "Height": 0.06439515948295593,
          "Left": 0.14159755408763885,
          "Top": 0.7791688442230225
        },
```

```
        "Polygon": [
          {
            "X": 0.14159755408763885,
            "Y": 0.7791688442230225
          },
          {
            "X": 0.24824367463588715,
            "Y": 0.7791688442230225
          },
          {
            "X": 0.24824367463588715,
            "Y": 0.843563973903656
          },
          {
            "X": 0.14159755408763885,
            "Y": 0.843563973903656
          }
        ]
      },
      "Id": "a92d8eef-db28-45ba-801a-5da0f589d277"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.97722625732422,
      "Text": "Example",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.12127546221017838,
          "Height": 0.05682983994483948,
          "Left": 0.26764172315597534,
          "Top": 0.794414758682251
        },
        "Polygon": [
          {
            "X": 0.26764172315597534,
            "Y": 0.794414758682251
          },
          {
            "X": 0.3889172077178955,
            "Y": 0.794414758682251
          },
          {
            "X": 0.3889172077178955,
```

```
            "Y": 0.8512446284294128
          },
          {
            "X": 0.26764172315597534,
            "Y": 0.8512446284294128
          }
        ]
      },
      "Id": "d6962efb-34ab-4ffb-9f2f-5f263e813558"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.98429870605469,
      "Text": "Corp.",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.07650306820869446,
          "Height": 0.05481306090950966,
          "Left": 0.4026312530040741,
          "Top": 0.7980174422264099
        },
        "Polygon": [
          {
            "X": 0.4026312530040741,
            "Y": 0.7980174422264099
          },
          {
            "X": 0.47913432121276855,
            "Y": 0.7980174422264099
          },
          {
            "X": 0.47913432121276855,
            "Y": 0.8528305292129517
          },
          {
            "X": 0.4026312530040741,
            "Y": 0.8528305292129517
          }
        ]
      },
      "Id": "1876c8ea-d3e8-4c39-870e-47512b3b5080"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 99.91166687011719,
      "Text": "Baker",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09931197017431259,
          "Height": 0.06008723005652428,
          "Left": 0.5098910331726074,
          "Top": 0.787897527217865
        },
        "Polygon": [
          {
            "X": 0.5098910331726074,
            "Y": 0.787897527217865
          },
          {
            "X": 0.609203040599823,
            "Y": 0.787897527217865
          },
          {
            "X": 0.609203040599823,
            "Y": 0.8479847311973572
          },
          {
            "X": 0.5098910331726074,
            "Y": 0.8479847311973572
          }
        ]
      },
      "Id": "00adeaef-ed57-44eb-b8a9-503575236d62"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.98870849609375,
      "Text": "better",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.10782185196876526,
          "Height": 0.06207133084535599,
          "Left": 0.7428008317947388,
          "Top": 0.7928366661071777
        },
```

```
        "Polygon": [
          {
            "X": 0.7428008317947388,
            "Y": 0.7928366661071777
          },
          {
            "X": 0.8506226539611816,
            "Y": 0.7928366661071777
          },
          {
            "X": 0.8506226539611816,
            "Y": 0.8549079895019531
          },
          {
            "X": 0.7428008317947388,
            "Y": 0.8549079895019531
          }
        ]
      },
      "Id": "c0fc9a58-7a4b-4f69-bafd-2cff32be2665"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.8883285522461,
      "Text": "opp.",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.07421936094760895,
          "Height": 0.058906231075525284,
          "Left": 0.8577775359153748,
          "Top": 0.8038780689239502
        },
        "Polygon": [
          {
            "X": 0.8577775359153748,
            "Y": 0.8038780689239502
          },
          {
            "X": 0.9319969415664673,
            "Y": 0.8038780689239502
          },
          {
            "X": 0.9319969415664673,
```

```
            "Y": 0.8627843260765076
          },
          {
            "X": 0.8577775359153748,
            "Y": 0.8627843260765076
          }
        ]
      },
      "Id": "bf6dc8ee-2fb3-4b6c-aee4-31e96912a2d8"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.92573547363281,
      "Text": "8/15/2013",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.10257463902235031,
          "Height": 0.05412459000945091,
          "Left": 0.027909137308597565,
          "Top": 0.8608770370483398
        },
        "Polygon": [
          {
            "X": 0.027909137308597565,
            "Y": 0.8608770370483398
          },
          {
            "X": 0.13048377633094788,
            "Y": 0.8608770370483398
          },
          {
            "X": 0.13048377633094788,
            "Y": 0.915001630783081
          },
          {
            "X": 0.027909137308597565,
            "Y": 0.915001630783081
          }
        ]
      },
      "Id": "5384f860-f857-4a94-9438-9dfa20eed1c6"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 99.99625396728516,
      "Text": "Present",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.09982697665691376,
          "Height": 0.06888339668512344,
          "Left": 0.1420602649450302,
          "Top": 0.8511748909950256
        },
        "Polygon": [
          {
            "X": 0.1420602649450302,
            "Y": 0.8511748909950256
          },
          {
            "X": 0.24188724160194397,
            "Y": 0.8511748909950256
          },
          {
            "X": 0.24188724160194397,
            "Y": 0.9200583100318909
          },
          {
            "X": 0.1420602649450302,
            "Y": 0.9200583100318909
          }
        ]
      },
      "Id": "0bb96ed6-b2e6-4da4-90b3-b85561bbd89d"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.9826431274414,
      "Text": "AnyCompany",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.18611273169517517,
          "Height": 0.08581399917602539,
          "Left": 0.2615866959095001,
          "Top": 0.869536280632019
        },
```

```
        "Polygon": [
          {
            "X": 0.2615866959095001,
            "Y": 0.869536280632019
          },
          {
            "X": 0.4476994276046753,
            "Y": 0.869536280632019
          },
          {
            "X": 0.4476994276046753,
            "Y": 0.9553502798080444
          },
          {
            "X": 0.2615866959095001,
            "Y": 0.9553502798080444
          }
        ]
      },
      "Id": "25343360-d906-440a-88b7-92eb89e95949"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.99523162841797,
      "Text": "head",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.07429949939250946,
          "Height": 0.05485520139336586,
          "Left": 0.49359121918678284,
          "Top": 0.8714361190795898
        },
        "Polygon": [
          {
            "X": 0.49359121918678284,
            "Y": 0.8714361190795898
          },
          {
            "X": 0.5678907036781311,
            "Y": 0.8714361190795898
          },
          {
            "X": 0.5678907036781311,
```

```
            "Y": 0.926291286945343
          },
          {
            "X": 0.49359121918678284,
            "Y": 0.926291286945343
          }
        ]
      },
      "Id": "0ef3c194-8322-4575-94f1-82819ee57e3a"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.99574279785156,
      "Text": "baker",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.1019822508096695,
          "Height": 0.05615599825978279,
          "Left": 0.585354208946228,
          "Top": 0.8702592849731445
        },
        "Polygon": [
          {
            "X": 0.585354208946228,
            "Y": 0.8702592849731445
          },
          {
            "X": 0.6873364448547363,
            "Y": 0.8702592849731445
          },
          {
            "X": 0.6873364448547363,
            "Y": 0.9264153242111206
          },
          {
            "X": 0.585354208946228,
            "Y": 0.9264153242111206
          }
        ]
      },
      "Id": "d296acd9-3e9a-4985-95f8-f863614f2c46"
    },
    {
```

```
      "BlockType": "WORD",
      "Confidence": 99.9880599975586,
      "Text": "N/A,",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.08230073750019073,
          "Height": 0.06588289886713028,
          "Left": 0.7411766648292542,
          "Top": 0.8722732067108154
        },
        "Polygon": [
          {
            "X": 0.7411766648292542,
            "Y": 0.8722732067108154
          },
          {
            "X": 0.8234773874282837,
            "Y": 0.8722732067108154
          },
          {
            "X": 0.8234773874282837,
            "Y": 0.9381561279296875
          },
          {
            "X": 0.7411766648292542,
            "Y": 0.9381561279296875
          }
        ]
      },
      "Id": "195cfb5b-ae06-4203-8520-4e4b0a73b5ce"
    },
    {
      "BlockType": "WORD",
      "Confidence": 99.97914123535156,
      "Text": "current",
      "TextType": "HANDWRITING",
      "Geometry": {
        "BoundingBox": {
          "Width": 0.12791454792022705,
          "Height": 0.04768490046262741,
          "Left": 0.8387037515640259,
          "Top": 0.8843405842781067
        },
```

```
        "Polygon": [
          {
            "X": 0.8387037515640259,
            "Y": 0.8843405842781067
          },
          {
            "X": 0.9666182994842529,
            "Y": 0.8843405842781067
          },
          {
            "X": 0.9666182994842529,
            "Y": 0.9320254921913147
          },
          {
            "X": 0.8387037515640259,
            "Y": 0.9320254921913147
          }
        ]
      },
      "Id": "549ef3f9-3a13-4b77-bc25-fb2e0996839a"
    }
  ],
  "DetectDocumentTextModelVersion": "1.0",
  "ResponseMetadata": {
    "RequestId": "337129e6-3af7-4014-842b-f6484e82cbf6",
    "HTTPStatusCode": 200,
    "HTTPHeaders": {
      "x-amzn-requestid": "337129e6-3af7-4014-842b-f6484e82cbf6",
      "content-type": "application/x-amz-json-1.1",
      "content-length": "45675",
      "date": "Mon, 09 Nov 2020 23:54:38 GMT"
    },
    "RetryAttempts": 0
  }
}
}
```


Erkennen von Dokumenttext mit Amazon Textract

Um Text in einem Dokument zu erkennen, verwenden Sie dieDetectDocumentText-Operation, und übergeben Sie eine Dokumentdatei als Eingabe.`DetectDocumentText`gibt eine JSON-Struktur zurück, die Zeilen und Wörter des erkannten Textes, die Position des Textes im Dokument und die

Beziehungen zwischen erkanntem Text enthält. Weitere Informationen finden Sie unter Erkennen von Text .

Sie können ein Eingabedokument als Bild-Byte-Array (base64-verschlüsselte Bild-Bytes) oder als Amazon S3 -Objekt zur Verfügung stellen. Bei dieser Vorgehensweise laden Sie eine Bilddatei in Ihren S3-Bucket hoch und geben den Dateinamen an.

So erkennen Sie Text in einem Dokument (API)

1. Wenn Sie dies noch nicht getan haben:

   a. Erstellen oder aktualisieren Sie einen IAM-Benutzer mit`AmazonTextractFullAccess`und`AmazonS3ReadOnlyAccess`Berechtigungen Weitere Informationen finden Sie unter Schritt 1: Einrichten eines AWS-Kontos und Erstellen eines IAM-Benutzers .

   b. Installieren und konfigurieren Sie die AWS CLI- und AWS-SDKs. Weitere Informationen finden Sie unter Schritt 2: Einrichten derAWS CLIundAWS-SDKs .

2. Laden Sie ein Dokument in Ihren S3-Bucket hoch.

   Detaillierte Anweisungen finden Sie unterHochladen von Objekten in Amazon S3imAmazon Simple Storage Service — Benutzerhandbuchaus.

3. Verwenden Sie die folgenden Beispiele zum Aufrufen der `DetectDocumentText`-Operation.

   Java

   Der folgende Beispielcode zeigt das Dokument und die Felder um Zeilen mit erkanntem Text an.

   In der Funktion`main`, ersetzen Sie die Werte von`bucket`und`document`Mit den Namen des Amazon S3 S3-Bucket und des Dokuments, die Sie in Schritt 2 verwendet haben.

```
//Calls DetectDocumentText.
//Loads document from S3 bucket. Displays the document and bounding boxes around
 detected lines/words of text.
package com.amazonaws.samples;

import java.awt.*;
import java.awt.image.BufferedImage;
import java.util.List;
import javax.imageio.ImageIO;
```

```
import javax.swing.*;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.S3ObjectInputStream;
import com.amazonaws.client.builder.AwsClientBuilder.EndpointConfiguration;
import com.amazonaws.services.textract.AmazonTextract;
import com.amazonaws.services.textract.AmazonTextractClientBuilder;
import com.amazonaws.services.textract.model.Block;
import com.amazonaws.services.textract.model.BoundingBox;
import com.amazonaws.services.textract.model.DetectDocumentTextRequest;
import com.amazonaws.services.textract.model.DetectDocumentTextResult;
import com.amazonaws.services.textract.model.Document;
import com.amazonaws.services.textract.model.S3Object;
import com.amazonaws.services.textract.model.Point;
import com.amazonaws.services.textract.model.Relationship;

public class DocumentText extends JPanel {

    private static final long serialVersionUID = 1L;

    BufferedImage image;
    DetectDocumentTextResult result;

    public DocumentText(DetectDocumentTextResult documentResult, BufferedImage
 bufImage) throws Exception {
        super();

        result = documentResult; // Results of text detection.
        image = bufImage; // The image containing the document.

    }

    // Draws the image and text bounding box.
    public void paintComponent(Graphics g) {

        int height = image.getHeight(this);
        int width = image.getWidth(this);

        Graphics2D g2d = (Graphics2D) g; // Create a Java2D version of g.

        // Draw the image.
        g2d.drawImage(image, 0, 0, image.getWidth(this) , image.getHeight(this),
 this);
```

```
        // Iterate through blocks and display polygons around lines of detected
text.
        List<Block> blocks = result.getBlocks();
        for (Block block : blocks) {
            DisplayBlockInfo(block);
            if ((block.getBlockType()).equals("LINE")) {
                ShowPolygon(height, width, block.getGeometry().getPolygon(),
g2d);
                /*
                  ShowBoundingBox(height, width,
block.getGeometry().getBoundingBox(), g2d);
                 */
            } else { // its a word, so just show vertical lines.
                ShowPolygonVerticals(height, width,
block.getGeometry().getPolygon(), g2d);
            }
        }
    }

    // Show bounding box at supplied location.
    private void ShowBoundingBox(int imageHeight, int imageWidth, BoundingBox
box, Graphics2D g2d) {

        float left = imageWidth * box.getLeft();
        float top = imageHeight * box.getTop();

        // Display bounding box.
        g2d.setColor(new Color(0, 212, 0));
        g2d.drawRect(Math.round(left), Math.round(top),
                Math.round(imageWidth * box.getWidth()), Math.round(imageHeight
* box.getHeight()));

    }

    // Shows polygon at supplied location
    private void ShowPolygon(int imageHeight, int imageWidth, List<Point>
points, Graphics2D g2d) {

        g2d.setColor(new Color(0, 0, 0));
        Polygon polygon = new Polygon();

        // Construct polygon and display
        for (Point point : points) {
            polygon.addPoint((Math.round(point.getX() * imageWidth)),
```

```
                    Math.round(point.getY() * imageHeight));
        }
        g2d.drawPolygon(polygon);
    }

    // Draws only the vertical lines in the supplied polygon.
    private void ShowPolygonVerticals(int imageHeight, int imageWidth,
List<Point> points, Graphics2D g2d) {

        g2d.setColor(new Color(0, 212, 0));
        Object[] parry = points.toArray();
        g2d.setStroke(new BasicStroke(2));

        g2d.drawLine(Math.round(((Point) parry[0]).getX() * imageWidth),
                Math.round(((Point) parry[0]).getY() * imageHeight),
Math.round(((Point) parry[3]).getX() * imageWidth),
                Math.round(((Point) parry[3]).getY() * imageHeight));

        g2d.setColor(new Color(255, 0, 0));
        g2d.drawLine(Math.round(((Point) parry[1]).getX() * imageWidth),
                Math.round(((Point) parry[1]).getY() * imageHeight),
Math.round(((Point) parry[2]).getX() * imageWidth),
                Math.round(((Point) parry[2]).getY() * imageHeight));

    }
    //Displays information from a block returned by text detection and text
analysis
    private void DisplayBlockInfo(Block block) {
        System.out.println("Block Id : " + block.getId());
        if (block.getText()!=null)
            System.out.println("    Detected text: " + block.getText());
        System.out.println("    Type: " + block.getBlockType());

        if (block.getBlockType().equals("PAGE") !=true) {
            System.out.println("    Confidence: " +
block.getConfidence().toString());
        }
        if(block.getBlockType().equals("CELL"))
        {
            System.out.println("    Cell information:");
            System.out.println("        Column: " + block.getColumnIndex());
            System.out.println("        Row: " + block.getRowIndex());
            System.out.println("        Column span: " + block.getColumnSpan());
            System.out.println("        Row span: " + block.getRowSpan());
```

```
        }

        System.out.println("    Relationships");
        List<Relationship> relationships=block.getRelationships();
        if(relationships!=null) {
            for (Relationship relationship : relationships) {
                System.out.println("        Type: " + relationship.getType());
                System.out.println("        IDs: " +
relationship.getIds().toString());
            }
        } else {
            System.out.println("        No related Blocks");
        }

        System.out.println("    Geometry");
        System.out.println("        Bounding Box: " +
block.getGeometry().getBoundingBox().toString());
        System.out.println("        Polygon: " +
block.getGeometry().getPolygon().toString());

        List<String> entityTypes = block.getEntityTypes();

        System.out.println("    Entity Types");
        if(entityTypes!=null) {
            for (String entityType : entityTypes) {
                System.out.println("        Entity Type: " + entityType);
            }
        } else {
            System.out.println("        No entity type");
        }
        if(block.getPage()!=null)
            System.out.println("    Page: " + block.getPage());
        System.out.println();
    }

    public static void main(String arg[]) throws Exception {

        // The S3 bucket and document
        String document = "";
        String bucket = "";


        AmazonS3 s3client = AmazonS3ClientBuilder.standard()
```

```
                .withEndpointConfiguration(
                        new EndpointConfiguration("https://
s3.amazonaws.com","us-east-1"))
                .build();


        // Get the document from S3
        com.amazonaws.services.s3.model.S3Object s3object =
 s3client.getObject(bucket, document);
        S3ObjectInputStream inputStream = s3object.getObjectContent();
        BufferedImage image = ImageIO.read(inputStream);

        // Call DetectDocumentText
        EndpointConfiguration endpoint = new EndpointConfiguration(
                "https://textract.us-east-1.amazonaws.com", "us-east-1");
        AmazonTextract client = AmazonTextractClientBuilder.standard()
                .withEndpointConfiguration(endpoint).build();


        DetectDocumentTextRequest request = new DetectDocumentTextRequest()
            .withDocument(new Document().withS3Object(new
 S3Object().withName(document).withBucket(bucket)));

        DetectDocumentTextResult result = client.detectDocumentText(request);

        // Create frame and panel.
        JFrame frame = new JFrame("RotateImage");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        DocumentText panel = new DocumentText(result, image);
        panel.setPreferredSize(new Dimension(image.getWidth() ,
 image.getHeight() ));
        frame.setContentPane(panel);
        frame.pack();
        frame.setVisible(true);

    }
}
```

AWS CLI

Dieser AWS CLI-Befehl zeigt die JSON-Ausgabe für die `detect-document-text`-CLI-Operation an.

Ersetzen Sie die Werte von`Bucket`und`Name`Mit den Namen des Amazon S3 S3-Bucket und des Dokuments, die Sie in Schritt 2 verwendet haben.

```
aws textract detect-document-text \
 --document '{"S3Object":{"Bucket":"bucket","Name":"document"}}'
```

Python

Der folgende Beispielcode zeigt das Dokument und die Felder um erkannte Textzeilen an.

In der Funktion`main`, ersetzen Sie die Werte von`bucket`und`document`Mit den Namen des Amazon S3 S3-Bucket und des Dokuments, die Sie in Schritt 2 verwendet haben.

```
#Detects text in a document stored in an S3 bucket. Display polygon box around
 text and angled text
import boto3
import io
from io import BytesIO
import sys

import psutil
import time

import math
from PIL import Image, ImageDraw, ImageFont


# Displays information about a block returned by text detection and text
 analysis
def DisplayBlockInformation(block):
    print('Id: {}'.format(block['Id']))
    if 'Text' in block:
        print('    Detected: ' + block['Text'])
    print('    Type: ' + block['BlockType'])

    if 'Confidence' in block:
        print('    Confidence: ' + "{:.2f}".format(block['Confidence']) + "%")

    if block['BlockType'] == 'CELL':
        print("    Cell information")
        print("        Column: " + str(block['ColumnIndex']))
        print("        Row: " + str(block['RowIndex']))
```

```
            print("        ColumnSpan: " + str(block['ColumnSpan']))
            print("        RowSpan: " + str(block['RowSpan']))

        if 'Relationships' in block:
            print('    Relationships: {}'.format(block['Relationships']))
        print('    Geometry: ')
        print('        Bounding Box: {}'.format(block['Geometry']['BoundingBox']))
        print('        Polygon: {}'.format(block['Geometry']['Polygon']))

        if block['BlockType'] == "KEY_VALUE_SET":
            print ('    Entity Type: ' + block['EntityTypes'][0])
        if 'Page' in block:
            print('Page: ' + block['Page'])
        print()

def process_text_detection(bucket, document):


    #Get the document from S3
    s3_connection = boto3.resource('s3')

    s3_object = s3_connection.Object(bucket,document)
    s3_response = s3_object.get()

    stream = io.BytesIO(s3_response['Body'].read())
    image=Image.open(stream)


    # Detect text in the document

    client = boto3.client('textract')
    #process using image bytes
    #image_binary = stream.getvalue()
    #response = client.detect_document_text(Document={'Bytes': image_binary})

    #process using S3 object
    response = client.detect_document_text(
        Document={'S3Object': {'Bucket': bucket, 'Name': document}})

    #Get the text blocks
    blocks=response['Blocks']
    width, height =image.size
    draw = ImageDraw.Draw(image)
    print ('Detected Document Text')
```

```
    # Create image showing bounding box/polygon the detected lines/text
    for block in blocks:
            print('Type: ' + block['BlockType'])
            if block['BlockType'] != 'PAGE':
                print('Detected: ' + block['Text'])
                print('Confidence: ' + "{:.2f}".format(block['Confidence']) +
"%")

            print('Id: {}'.format(block['Id']))
            if 'Relationships' in block:
                print('Relationships: {}'.format(block['Relationships']))
            print('Bounding Box: {}'.format(block['Geometry']['BoundingBox']))
            print('Polygon: {}'.format(block['Geometry']['Polygon']))
            print()
            draw=ImageDraw.Draw(image)
            # Draw WORD - Green -  start of word, red - end of word
            if block['BlockType'] == "WORD":
                draw.line([(width * block['Geometry']['Polygon'][0]['X'],
                height * block['Geometry']['Polygon'][0]['Y']),
                (width * block['Geometry']['Polygon'][3]['X'],
                height * block['Geometry']['Polygon'][3]['Y'])],fill='green',
                width=2)

                draw.line([(width * block['Geometry']['Polygon'][1]['X'],
                height * block['Geometry']['Polygon'][1]['Y']),
                (width * block['Geometry']['Polygon'][2]['X'],
                height * block['Geometry']['Polygon'][2]['Y'])],
                fill='red',
                width=2)


            # Draw box around entire LINE
            if block['BlockType'] == "LINE":
                points=[]

                for polygon in block['Geometry']['Polygon']:
                    points.append((width * polygon['X'], height * polygon['Y']))

                draw.polygon((points), outline='black')

                # Uncomment to draw bounding box
                #box=block['Geometry']['BoundingBox']
                #left = width * box['Left']
```

```
                #top = height * box['Top']
                #draw.rectangle([left,top, left + (width * box['Width']), top
 +(height * box['Height'])],outline='black')


    # Display the image
    image.show()
    # display image for 10 seconds


    return len(blocks)

def main():

    bucket = ''
    document = ''
    block_count=process_text_detection(bucket,document)
    print("Blocks detected: " + str(block_count))

if __name__ == "__main__":
    main()
```

Node.js

Der folgende Beispielcode von Node.js zeigt das Dokument und die Felder um erkannte Textzeilen an und gibt ein Bild der Ergebnisse in das Verzeichnis aus, aus dem Sie den Code ausführen. Es nutzt die`image-size`und`images`-Pakete.

In der Funktion`main`, ersetzen Sie die Werte von`bucket`und`document`Mit den Namen des Amazon S3 S3-Bucket und des Dokuments, die Sie in Schritt 2 verwendet haben. Ersetzen Sie den Wert von`regionConfig`mit dem Namen der Region, in der sich Ihr Konto befindet.

```
async function main(){

// Import AWS
const AWS = require("aws-sdk")
// Use Image-Size to get
const sizeOf = require('image-size');
// Image tool to draw buffers
const images = require("images");
```

```
// Create a canvas and get the context
const { createCanvas } = require('canvas')
const canvas = createCanvas(200, 200)
const ctx = canvas.getContext('2d')

// Set variables
const bucket = 'bucket-name' // the s3 bucket name
const photo  = 'image-name' // the name of file
const regionConfig = 'region'

// Set region if needed
AWS.config.update({region:regionConfig});

// Connect to Textract
const client = new AWS.Textract();
// Connect to S3 to display image
const s3 = new AWS.S3();

// Define paramaters
const params = {
  Document: {
    S3Object: {
      Bucket: bucket,
      Name: photo
    },
  },
}

// Function to display image
async function getImage(){
  const imageData =  s3.getObject(
    {
        Bucket: bucket,
        Key: photo
      }

  ).promise();
  return imageData;
}

// get image
var imageData = await getImage()

// Get the height, width of the image
```

```
const dimensions = sizeOf(imageData.Body)
const width = dimensions.width
const height = dimensions.height
console.log(imageData.Body)
console.log(width, height)

canvas.width = width;
canvas.height = height;

try{
  // Call API and log response
  const res = await client.detectDocumentText(params).promise();
  var image = images(imageData.Body).size(width, height)
  //console.log the type of block, text, text type, and confidence
  res.Blocks.forEach(block => {
    console.log(`Block Type: ${block.BlockType}`),
    console.log(`Text: ${block.Text}`)
    console.log(`TextType: ${block.TextType}`)
    console.log(`Confidence: ${block.Confidence}`)

    // Draw box around detected text using polygons
    ctx.strokeStyle = 'rgba(0,0,0,0.5)';
    ctx.beginPath();
    block.Geometry.Polygon.forEach(({X, Y}) =>
    ctx.lineTo(width * X - 10, height * Y - 10)
    );
    ctx.closePath();
    ctx.stroke();
    console.log("-----")
  })

  // render image
  var buffer = canvas.toBuffer("image/png");
  image.draw(images(buffer), 10, 10)
  image.save("output-image.jpg");

} catch (err){
console.error(err);}

}

main()
```

4. Führen Sie das Beispiel aus. Die Python- und Java-Beispiele zeigen das Dokumentbild an. Eine schwarze Box umgibt jede Zeile mit erkanntem Text. Eine grüne vertikale Linie ist der Beginn eines erkannten Wortes. Eine rote vertikale Linie ist das Ende eines erkannten Wortes. DieAWS CLIBeispiel zeigt nur die JSON-Ausgabe für`DetectDocumentText`verwenden.

Analysieren von Dokumenttext mit Amazon Textract

Um Text in einem Dokument zu analysieren, verwenden Sie dieAnalyzeDocument-Operation, und übergeben Sie eine Dokumentdatei als Eingabe.`AnalyzeDocument`gibt eine JSON-Struktur zurück, die den analysierten Text enthält. Weitere Informationen finden Sie unter Analysieren von Dokumenten .

Sie können ein Eingabedokument als Bild-Byte-Array (base64-verschlüsselte Bild-Bytes) oder als Amazon S3 -Objekt zur Verfügung stellen. Bei dieser Vorgehensweise laden Sie eine Bilddatei in Ihren S3-Bucket hoch und geben den Dateinamen an.

So analysieren Sie Text in einem Dokument (API)

1. Wenn Sie dies noch nicht getan haben:

   a. Erstellen oder aktualisieren Sie einen IAM-Benutzer mit`AmazonTextractFullAccess`und`AmazonS3ReadOnlyAccess`Berechtigungen Weitere Informationen finden Sie unter Schritt 1: Einrichten eines AWS-Kontos und Erstellen eines IAM-Benutzers .

   b. Installieren und konfigurieren Sie die AWS CLI- und AWS-SDKs. Weitere Informationen finden Sie unter Schritt 2: Einrichten derAWS CLIundAWS-SDKs .

2. Laden Sie ein Bild auf Ihren S3-Bucket hoch, auf dem ein Dokument abgebildet sind.

   Detaillierte Anweisungen finden Sie unterHochladen von Objekten in Amazon S3imAmazon Simple Storage Service — Benutzerhandbuchaus.

3. Verwenden Sie die folgenden Beispiele zum Aufrufen der `AnalyzeDocument`-Operation.

   Java

   Der folgende Beispielcode zeigt das Dokument und die Felder um erkannte Elemente an.

   In der Funktion`main`, ersetzen Sie die Werte von`bucket`und`document`Mit den Namen des Amazon S3 S3-Bucket und des Dokumentbilds, das Sie in Schritt 2 verwendet haben.

```
//Loads document from S3 bucket. Displays the document and polygon around
 detected lines of text.
package com.amazonaws.samples;

import java.awt.*;
import java.awt.image.BufferedImage;
import java.util.List;
import javax.imageio.ImageIO;
import javax.swing.*;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.S3ObjectInputStream;
import com.amazonaws.services.textract.AmazonTextract;
import com.amazonaws.services.textract.AmazonTextractClientBuilder;
import com.amazonaws.services.textract.model.AnalyzeDocumentRequest;
import com.amazonaws.services.textract.model.AnalyzeDocumentResult;
import com.amazonaws.services.textract.model.Block;
import com.amazonaws.services.textract.model.BoundingBox;
import com.amazonaws.services.textract.model.Document;
import com.amazonaws.services.textract.model.S3Object;
import com.amazonaws.services.textract.model.Point;
import com.amazonaws.services.textract.model.Relationship;
import com.amazonaws.client.builder.AwsClientBuilder.EndpointConfiguration;

public class AnalyzeDocument extends JPanel {

    private static final long serialVersionUID = 1L;

    BufferedImage image;

    AnalyzeDocumentResult result;

    public AnalyzeDocument(AnalyzeDocumentResult documentResult, BufferedImage
 bufImage) throws Exception {
        super();

        result = documentResult; // Results of text detection.
        image = bufImage; // The image containing the document.

    }

    // Draws the image and text bounding box.
```

```
   public void paintComponent(Graphics g) {

       int height = image.getHeight(this);
       int width = image.getWidth(this);

       Graphics2D g2d = (Graphics2D) g; // Create a Java2D version of g.

       // Draw the image.
       g2d.drawImage(image, 0, 0, image.getWidth(this), image.getHeight(this),
this);

       // Iterate through blocks and display bounding boxes around everything.

       List<Block> blocks = result.getBlocks();
       for (Block block : blocks) {
           DisplayBlockInfo(block);
           switch(block.getBlockType()) {

           case "KEY_VALUE_SET":
               if (block.getEntityTypes().contains("KEY")){
                   ShowBoundingBox(height, width,
block.getGeometry().getBoundingBox(), g2d, new Color(255,0,0));
               }
               else {  //VALUE
                   ShowBoundingBox(height, width,
block.getGeometry().getBoundingBox(), g2d, new Color(0,255,0));
               }
               break;
           case "TABLE":
               ShowBoundingBox(height, width,
block.getGeometry().getBoundingBox(), g2d, new Color(0,0,255));
               break;
           case "CELL":
               ShowBoundingBox(height, width,
block.getGeometry().getBoundingBox(), g2d, new Color(255,255,0));
               break;
           case "SELECTION_ELEMENT":
               if (block.getSelectionStatus().equals("SELECTED"))
                   ShowSelectedElement(height, width,
block.getGeometry().getBoundingBox(), g2d, new Color(0,0,255));
               break;
           default:
               //PAGE, LINE & WORD
```

```
                //ShowBoundingBox(height, width,
block.getGeometry().getBoundingBox(), g2d, new Color(200,200,0));
            }
        }

         // uncomment to show polygon around all blocks
         //ShowPolygon(height,width,block.getGeometry().getPolygon(),g2d);


    }

    // Show bounding box at supplied location.
    private void ShowBoundingBox(int imageHeight, int imageWidth, BoundingBox
box, Graphics2D g2d, Color color) {

        float left = imageWidth * box.getLeft();
        float top = imageHeight * box.getTop();

        // Display bounding box.
        g2d.setColor(color);
        g2d.drawRect(Math.round(left), Math.round(top),
                Math.round(imageWidth * box.getWidth()), Math.round(imageHeight
* box.getHeight()));

    }
    private void ShowSelectedElement(int imageHeight, int imageWidth,
BoundingBox box, Graphics2D g2d, Color color) {

        float left = imageWidth * box.getLeft();
        float top = imageHeight * box.getTop();

        // Display bounding box.
        g2d.setColor(color);
        g2d.fillRect(Math.round(left), Math.round(top),
                Math.round(imageWidth * box.getWidth()), Math.round(imageHeight
* box.getHeight()));

    }

    // Shows polygon at supplied location
    private void ShowPolygon(int imageHeight, int imageWidth, List<Point>
points, Graphics2D g2d) {

        g2d.setColor(new Color(0, 0, 0));
```

```
        Polygon polygon = new Polygon();

        // Construct polygon and display
        for (Point point : points) {
            polygon.addPoint((Math.round(point.getX() * imageWidth)),
                    Math.round(point.getY() * imageHeight));
        }
        g2d.drawPolygon(polygon);
    }
    //Displays information from a block returned by text detection and text
analysis
    private void DisplayBlockInfo(Block block) {
        System.out.println("Block Id : " + block.getId());
        if (block.getText()!=null)
            System.out.println("    Detected text: " + block.getText());
        System.out.println("    Type: " + block.getBlockType());

        if (block.getBlockType().equals("PAGE") !=true) {
            System.out.println("    Confidence: " +
block.getConfidence().toString());
        }
        if(block.getBlockType().equals("CELL"))
        {
            System.out.println("    Cell information:");
            System.out.println("        Column: " + block.getColumnIndex());
            System.out.println("        Row: " + block.getRowIndex());
            System.out.println("        Column span: " + block.getColumnSpan());
            System.out.println("        Row span: " + block.getRowSpan());

        }

        System.out.println("    Relationships");
        List<Relationship> relationships=block.getRelationships();
        if(relationships!=null) {
            for (Relationship relationship : relationships) {
                System.out.println("        Type: " + relationship.getType());
                System.out.println("        IDs: " +
relationship.getIds().toString());
            }
        } else {
            System.out.println("        No related Blocks");
        }

        System.out.println("    Geometry");
```

```
        System.out.println("        Bounding Box: " +
 block.getGeometry().getBoundingBox().toString());
        System.out.println("        Polygon: " +
 block.getGeometry().getPolygon().toString());

        List<String> entityTypes = block.getEntityTypes();

        System.out.println("    Entity Types");
        if(entityTypes!=null) {
            for (String entityType : entityTypes) {
                System.out.println("        Entity Type: " + entityType);
            }
        } else {
            System.out.println("        No entity type");
        }

        if(block.getBlockType().equals("SELECTION_ELEMENT")) {
            System.out.print("    Selection element detected: ");
            if (block.getSelectionStatus().equals("SELECTED")){
                System.out.println("Selected");
            }else {
                System.out.println(" Not selected");
            }
        }

        if(block.getPage()!=null)
            System.out.println("    Page: " + block.getPage());
        System.out.println();
    }

    public static void main(String arg[]) throws Exception {

        // The S3 bucket and document
        String document = "";
        String bucket = "";

        AmazonS3 s3client = AmazonS3ClientBuilder.standard()
                .withEndpointConfiguration(
                        new EndpointConfiguration("https://
s3.amazonaws.com","us-east-1"))
                .build();


        // Get the document from S3
```

```
        com.amazonaws.services.s3.model.S3Object s3object =
 s3client.getObject(bucket, document);
        S3ObjectInputStream inputStream = s3object.getObjectContent();
        BufferedImage image = ImageIO.read(inputStream);

        // Call AnalyzeDocument
        EndpointConfiguration endpoint = new EndpointConfiguration(
                "https://textract.us-east-1.amazonaws.com", "us-east-1");
        AmazonTextract client = AmazonTextractClientBuilder.standard()
                .withEndpointConfiguration(endpoint).build();


        AnalyzeDocumentRequest request = new AnalyzeDocumentRequest()
                .withFeatureTypes("TABLES","FORMS")
                 .withDocument(new Document().
                        withS3Object(new
 S3Object().withName(document).withBucket(bucket)));


        AnalyzeDocumentResult result = client.analyzeDocument(request);

        // Create frame and panel.
        JFrame frame = new JFrame("RotateImage");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        AnalyzeDocument panel = new AnalyzeDocument(result, image);
        panel.setPreferredSize(new Dimension(image.getWidth(),
 image.getHeight()));
        frame.setContentPane(panel);
        frame.pack();
        frame.setVisible(true);

    }
}
```

AWS CLI

Dieser AWS CLI-Befehl zeigt die JSON-Ausgabe für die `detect-document-text`-CLI-Operation an.

Ersetzen Sie die Werte von`Bucket`und`Name`Mit den Namen des Amazon S3 S3-Bucket und des Dokuments, die Sie in Schritt 2 verwendet haben.

```
aws textract analyze-document \
```

```
 --document '{"S3Object":{"Bucket":"bucket","Name":"document"}}' \
 --feature-types '["TABLES","FORMS"]'
```

Python

Der folgende Beispielcode zeigt das Dokument und die Felder um erkannte Elemente an.

In der Funktion`main`, ersetzen Sie die Werte von`bucket`und`document`Mit den Namen des Amazon S3 S3-Bucket und des Dokuments, die Sie in Schritt 2 verwendet haben.

```
#Analyzes text in a document stored in an S3 bucket. Display polygon box around
 text and angled text
import boto3
import io
from io import BytesIO
import sys

import math
from PIL import Image, ImageDraw, ImageFont

def ShowBoundingBox(draw,box,width,height,boxColor):

    left = width * box['Left']
    top = height * box['Top']
    draw.rectangle([left,top, left + (width * box['Width']), top +(height *
 box['Height'])],outline=boxColor)

def ShowSelectedElement(draw,box,width,height,boxColor):

    left = width * box['Left']
    top = height * box['Top']
    draw.rectangle([left,top, left + (width * box['Width']), top +(height *
 box['Height'])],fill=boxColor)

# Displays information about a block returned by text detection and text
 analysis
def DisplayBlockInformation(block):
    print('Id: {}'.format(block['Id']))
    if 'Text' in block:
        print('    Detected: ' + block['Text'])
    print('    Type: ' + block['BlockType'])
```

```
    if 'Confidence' in block:
        print('    Confidence: ' + "{:.2f}".format(block['Confidence']) + "%")

    if block['BlockType'] == 'CELL':
        print("    Cell information")
        print("        Column:" + str(block['ColumnIndex']))
        print("        Row:" + str(block['RowIndex']))
        print("        Column Span:" + str(block['ColumnSpan']))
        print("        RowSpan:" + str(block['ColumnSpan']))

    if 'Relationships' in block:
        print('    Relationships: {}'.format(block['Relationships']))
    print('    Geometry: ')
    print('        Bounding Box: {}'.format(block['Geometry']['BoundingBox']))
    print('        Polygon: {}'.format(block['Geometry']['Polygon']))

    if block['BlockType'] == "KEY_VALUE_SET":
        print ('    Entity Type: ' + block['EntityTypes'][0])

    if block['BlockType'] == 'SELECTION_ELEMENT':
        print('    Selection element detected: ', end='')

        if block['SelectionStatus'] =='SELECTED':
            print('Selected')
        else:
            print('Not selected')

    if 'Page' in block:
        print('Page: ' + block['Page'])
    print()

def process_text_analysis(bucket, document):

    #Get the document from S3
    s3_connection = boto3.resource('s3')

    s3_object = s3_connection.Object(bucket,document)
    s3_response = s3_object.get()

    stream = io.BytesIO(s3_response['Body'].read())
    image=Image.open(stream)

    # Analyze the document
    client = boto3.client('textract')
```

```
    image_binary = stream.getvalue()
    response = client.analyze_document(Document={'Bytes': image_binary},
        FeatureTypes=["TABLES", "FORMS"])

    ### Alternatively, process using S3 object ###
    #response = client.analyze_document(
    #    Document={'S3Object': {'Bucket': bucket, 'Name': document}},
    #    FeatureTypes=["TABLES", "FORMS"])

    ### To use a local file ###
    # with open("pathToFile", 'rb') as img_file:
        ### To display image using PIL ###
    #    image = Image.open()
        ### Read bytes ###
    #    img_bytes = img_file.read()
    #    response = client.analyze_document(Document={'Bytes': img_bytes},
 FeatureTypes=["TABLES", "FORMS"])


    #Get the text blocks
    blocks=response['Blocks']
    width, height =image.size
    draw = ImageDraw.Draw(image)
    print ('Detected Document Text')

    # Create image showing bounding box/polygon the detected lines/text
    for block in blocks:

        DisplayBlockInformation(block)

        draw=ImageDraw.Draw(image)
        if block['BlockType'] == "KEY_VALUE_SET":
            if block['EntityTypes'][0] == "KEY":
                ShowBoundingBox(draw, block['Geometry']
['BoundingBox'],width,height,'red')
            else:
                ShowBoundingBox(draw, block['Geometry']
['BoundingBox'],width,height,'green')

        if block['BlockType'] == 'TABLE':
            ShowBoundingBox(draw, block['Geometry']['BoundingBox'],width,height,
 'blue')
```

```
        if block['BlockType'] == 'CELL':
            ShowBoundingBox(draw, block['Geometry']['BoundingBox'],width,height,
 'yellow')
        if block['BlockType'] == 'SELECTION_ELEMENT':
            if block['SelectionStatus'] =='SELECTED':
                ShowSelectedElement(draw, block['Geometry']
['BoundingBox'],width,height, 'blue')

            #uncomment to draw polygon for all Blocks
            #points=[]
            #for polygon in block['Geometry']['Polygon']:
            #    points.append((width * polygon['X'], height * polygon['Y']))
            #draw.polygon((points), outline='blue')

    # Display the image
    image.show()
    return len(blocks)


def main():

    bucket = ''
    document = ''
    block_count=process_text_analysis(bucket,document)
    print("Blocks detected: " + str(block_count))

if __name__ == "__main__":
    main()
```

Node.js

Der folgende Beispielcode zeigt das Dokument und die Felder um erkannte Elemente an.

Ersetzen Sie im folgenden Code die Werte von`bucket`und`photo`Mit den Namen des Amazon S3 S3-Bucket und des Dokuments, die Sie in Schritt 2 verwendet haben. Ersetzen Sie den Wert von`region`mit der Region, die Ihrem -Konto zugeordnet ist.

```
// Import required AWS SDK clients and commands for Node.js
import { AnalyzeDocumentCommand } from  "@aws-sdk/client-textract";
import  { TextractClient } from "@aws-sdk/client-textract";

// Set the AWS Region.
const REGION = "region"; //e.g. "us-east-1"
```

```
// Create SNS service object.
const textractClient = new TextractClient({ region: REGION });

const bucket = 'buckets'
const photo = 'photo'

// Set params
const params = {
    Document: {
      S3Object: {
        Bucket: bucket,
        Name: photo
      },
    },
    FeatureTypes: ['TABLES', 'FORMS'],
  }

const displayBlockInfo = async (response) => {
    try {
        response.Blocks.forEach(block => {
            console.log(`ID: ${block.Id}`)
            console.log(`Block Type: ${block.BlockType}`)
            if ("Text" in block && block.Text !== undefined){
                console.log(`Text: ${block.Text}`)
            }
            else{}
            if ("Confidence" in block && block.Confidence !== undefined){
                console.log(`Confidence: ${block.Confidence}`)
            }
            else{}
            if (block.BlockType == 'CELL'){
                console.log("Cell info:")
                console.log(`   Column Index - ${block.ColumnIndex}`)
                console.log(`   Row - ${block.RowIndex}`)
                console.log(`   Column Span - ${block.ColumnSpan}`)
                console.log(`   Row Span - ${block.RowSpan}`)
            }
            if ("Relationships" in block && block.Relationships !== undefined){
                console.log(block.Relationships)
                console.log("Geometry:")
                console.log(`   Bounding Box -
 ${JSON.stringify(block.Geometry.BoundingBox)}`)
                console.log(`   Polygon -
 ${JSON.stringify(block.Geometry.Polygon)}`)
```

```
            }
            console.log("-----")
        });
      } catch (err) {
        console.log("Error", err);
      }
}

const analyze_document_text = async () => {
    try {
        const analyzeDoc = new AnalyzeDocumentCommand(params);
        const response = await textractClient.send(analyzeDoc);
        //console.log(response)
        displayBlockInfo(response)
        return response; // For unit tests.
      } catch (err) {
        console.log("Error", err);
      }
}

analyze_document_text()
```

4. Führen Sie das Beispiel aus. Die Python- und Java-Beispiele zeigen das Dokumentbild mit den folgenden farbigen Begrenzungsrahmen an:

   - Rot — KEY Block-Objekte
   - Grün — VALUE Block-Objekte
   - Blau — TABLE Blockobjekte
   - Gelb — CELL Block-Objekte

   Ausgewählte Auswahlelemente sind mit Blau gefüllt.

   DieAWS CLIBeispiel zeigt nur die JSON-Ausgabe für`AnalyzeDocument`verwenden.

Rechnungen und Belege mit Amazon Textract analysieren

Um Rechnungs- und Belege zu analysieren, verwenden Sie die AnalyzeExpense-API und übergeben eine Dokumentdatei als Eingabe.`AnalyzeExpense`ist eine synchrone Operation, die eine JSON-Struktur zurückgibt, die den analysierten Text enthält. Weitere Informationen finden Sie unter Analysieren von Rechnungen und Belegen .

Um Rechnungen und Belege asynchron zu analysieren, verwenden Sie`StartExpenseAnalysis`um mit der Verarbeitung einer Eingabedokumentdatei zu beginnen und zu verwenden`GetExpenseAnalysis`um die Ergebnisse zu erhalten.

Sie können ein Eingabedokument als Bild-Byte-Array (base64-verschlüsselte Bild-Bytes) oder als Amazon S3 -Objekt zur Verfügung stellen. Bei dieser Vorgehensweise laden Sie eine Bilddatei in Ihren S3-Bucket hoch und geben den Dateinamen an.

So analysieren Sie eine Rechnung oder Quittung (API)

1. Wenn Sie dies noch nicht getan haben:

   a. Erstellen oder aktualisieren Sie einen IAM-Benutzer mit`AmazonTextractFullAccess`und`AmazonS3ReadOnlyAccess`Berechtigungen Weitere Informationen finden Sie unter Schritt 1: Einrichten eines AWS-Kontos und Erstellen eines IAM-Benutzers .

   b. Installieren und konfigurieren Sie die AWS CLI- und AWS-SDKs. Weitere Informationen finden Sie unter Schritt 2: Einrichten derAWS CLIundAWS-SDKs .

2. Laden Sie ein Bild auf Ihren S3-Bucket hoch, auf dem ein Dokument abgebildet sind.

   Detaillierte Anweisungen finden Sie unterHochladen von Objekten in Amazon S3imAmazon Simple Storage Service — Benutzerhandbuchaus.

3. Verwenden Sie die folgenden Beispiele zum Aufrufen der`AnalyzeExpense`-Operation.

   CLI

```
  aws textract analyze-expense --document '{"S3Object": {"Bucket": "bucket
 name","Name":
"object name"}}'
```

   Python

```
import boto3
import io
from PIL import Image, ImageDraw

def draw_bounding_box(key, val, width, height, draw):
    # If a key is Geometry, draw the bounding box info in it
```

```
    if "Geometry" in key:
        # Draw bounding box information
        box = val["BoundingBox"]
        left = width * box['Left']
        top = height * box['Top']
        draw.rectangle([left, top, left + (width * box['Width']), top + (height
 * box['Height'])],
                       outline='black')

# Takes a field as an argument and prints out the detected labels and values
def print_labels_and_values(field):
    # Only if labels are detected and returned
    if "LabelDetection" in field:
        print("Summary Label Detection - Confidence: {}".format(
            str(field.get("LabelDetection")["Confidence"])) + ", "
              + "Summary Values: {}".format(str(field.get("LabelDetection")
["Text"])))
        print(field.get("LabelDetection")["Geometry"])
    else:
        print("Label Detection - No labels returned.")
    if "ValueDetection" in field:
        print("Summary Value Detection - Confidence: {}".format(
            str(field.get("ValueDetection")["Confidence"])) + ", "
              + "Summary Values: {}".format(str(field.get("ValueDetection")
["Text"])))
        print(field.get("ValueDetection")["Geometry"])
    else:
        print("Value Detection - No values returned")

def process_text_detection(bucket, document):
    # Get the document from S3
    s3_connection = boto3.resource('s3')
    s3_object = s3_connection.Object(bucket, document)
    s3_response = s3_object.get()

    # opening binary stream using an in-memory bytes buffer
    stream = io.BytesIO(s3_response['Body'].read())

    # loading stream into image
    image = Image.open(stream)

    # Detect text in the document
    client = boto3.client('textract', region_name="us-east-1")
```

```
    # process using S3 object
    response = client.analyze_expense(
        Document={'S3Object': {'Bucket': bucket, 'Name': document}})

    # Set width and height to display image and draw bounding boxes
    # Create drawing object
    width, height = image.size
    draw = ImageDraw.Draw(image)

    for expense_doc in response["ExpenseDocuments"]:
        for line_item_group in expense_doc["LineItemGroups"]:
            for line_items in line_item_group["LineItems"]:
                for expense_fields in line_items["LineItemExpenseFields"]:
                    print_labels_and_values(expense_fields)
                    print()

        print("Summary:")
        for summary_field in expense_doc["SummaryFields"]:
            print_labels_and_values(summary_field)
            print()

        #For draw bounding boxes
        for line_item_group in expense_doc["LineItemGroups"]:
            for line_items in line_item_group["LineItems"]:
                for expense_fields in line_items["LineItemExpenseFields"]:
                    for key, val in expense_fields["ValueDetection"].items():
                        if "Geometry" in key:
                            draw_bounding_box(key, val, width, height, draw)

        for label in expense_doc["SummaryFields"]:
            if "LabelDetection" in label:
                for key, val in label["LabelDetection"].items():
                    draw_bounding_box(key, val, width, height, draw)

    # Display the image
    image.show()

def main():
    bucket = 'Bucket-Name'
    document = 'Document-Name'
    process_text_detection(bucket, document)

if __name__ == "__main__":
    main()
```

Java

```
package com.amazonaws.samples;

import java.awt.*;
import java.awt.image.BufferedImage;
import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.util.List;
import java.util.concurrent.CompletableFuture;
import javax.imageio.ImageIO;
import javax.swing.*;
import software.amazon.awssdk.auth.credentials.AwsBasicCredentials;
import software.amazon.awssdk.auth.credentials.StaticCredentialsProvider;
import software.amazon.awssdk.core.ResponseBytes;
import software.amazon.awssdk.core.async.AsyncResponseTransformer;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.*;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import software.amazon.awssdk.services.textract.TextractClient;
import software.amazon.awssdk.services.textract.model.AnalyzeExpenseRequest;
import software.amazon.awssdk.services.textract.model.AnalyzeExpenseResponse;
import software.amazon.awssdk.services.textract.model.BoundingBox;
import software.amazon.awssdk.services.textract.model.Document;
import software.amazon.awssdk.services.textract.model.ExpenseDocument;
import software.amazon.awssdk.services.textract.model.ExpenseField;
import software.amazon.awssdk.services.textract.model.LineItemFields;
import software.amazon.awssdk.services.textract.model.LineItemGroup;
import software.amazon.awssdk.services.textract.model.S3Object;
import software.amazon.awssdk.services.textract.model.Point;

/**
 *
 * Demo code to parse Textract AnalyzeExpense API
 *
 */
public class TextractAnalyzeExpenseSample extends JPanel {
```

```
private static final long serialVersionUID = 1L;

BufferedImage image;
static AnalyzeExpenseResponse result;

public TextractAnalyzeExpenseSample(AnalyzeExpenseResponse documentResult,
BufferedImage bufImage) throws Exception {
 super();

 result = documentResult; // Results of analyzeexpense summaryfields and
lineitemgroups detection.
 image = bufImage; // The image containing the document.

}

// Draws the image and text bounding box.
public void paintComponent(Graphics g) {

 Graphics2D g2d = (Graphics2D) g; // Create a Java2D version of g.

 // Draw the image.
 g2d.drawImage(image, 0, 0, image.getWidth(this), image.getHeight(this), this);

 // Iterate through summaryfields and lineitemgroups and display boundedboxes
around lines of detected label and value.
 List<ExpenseDocument> expenseDocuments = result.expenseDocuments();
 for (ExpenseDocument expenseDocument : expenseDocuments) {

  if (expenseDocument.hasSummaryFields()) {
   DisplayAnalyzeExpenseSummaryInfo(expenseDocument);
   List<ExpenseField> summaryfields = expenseDocument.summaryFields();
   for (ExpenseField summaryfield : summaryfields) {

    if (summaryfield.valueDetection() != null) {
     ShowBoundingBox(image.getHeight(this), image.getWidth(this),
       summaryfield.valueDetection().geometry().boundingBox(), g2d, new
Color(0, 0, 0));
    }

    if (summaryfield.labelDetection() != null) {

     ShowBoundingBox(image.getHeight(this), image.getWidth(this),
       summaryfield.labelDetection().geometry().boundingBox(), g2d, new
Color(0, 0, 0));
```

```
    }

   }

  }

  if (expenseDocument.hasLineItemGroups()) {
   DisplayAnalyzeExpenseLineItemGroupsInfo(expenseDocument);

   List<LineItemGroup> lineitemgroups = expenseDocument.lineItemGroups();

   for (LineItemGroup lineitemgroup : lineitemgroups) {

    if (lineitemgroup.hasLineItems()) {

     List<LineItemFields> lineItems = lineitemgroup.lineItems();
     for (LineItemFields lineitemfield : lineItems) {

      if (lineitemfield.hasLineItemExpenseFields()) {

       List<ExpenseField> expensefields =
lineitemfield.lineItemExpenseFields();
       for (ExpenseField expensefield : expensefields) {

        if (expensefield.valueDetection() != null) {
         ShowBoundingBox(image.getHeight(this), image.getWidth(this),
           expensefield.valueDetection().geometry().boundingBox(), g2d,
           new Color(0, 0, 0));
        }

        if (expensefield.labelDetection() != null) {
         ShowBoundingBox(image.getHeight(this), image.getWidth(this),
           expensefield.labelDetection().geometry().boundingBox(), g2d,
           new Color(0, 0, 0));
        }

       }
      }

     }

    }
```

```
    }

   }
  }

}

// Show bounding box at supplied location.
private void ShowBoundingBox(float imageHeight, float imageWidth, BoundingBox
box, Graphics2D g2d, Color color) {

 float left = imageWidth * box.left();
 float top = imageHeight * box.top();

 // Display bounding box.
 g2d.setColor(color);
 g2d.drawRect(Math.round(left), Math.round(top), Math.round(imageWidth *
box.width()),
    Math.round(imageHeight * box.height()));

}

private void ShowSelectedElement(float imageHeight, float imageWidth,
BoundingBox box, Graphics2D g2d,
  Color color) {

 float left = (float) imageWidth * (float) box.left();
 float top = (float) imageHeight * (float) box.top();
 System.out.println(left);
 System.out.println(top);

 // Display bounding box.
 g2d.setColor(color);
 g2d.fillRect(Math.round(left), Math.round(top), Math.round(imageWidth *
box.width()),
    Math.round(imageHeight * box.height()));

}

// Shows polygon at supplied location
private void ShowPolygon(int imageHeight, int imageWidth, List<Point> points,
Graphics2D g2d) {

 g2d.setColor(new Color(0, 0, 0));
```

```
 Polygon polygon = new Polygon();

 // Construct polygon and display
 for (Point point : points) {
  polygon.addPoint((Math.round(point.x() * imageWidth)), Math.round(point.y() *
imageHeight));
 }
 g2d.drawPolygon(polygon);
}

private void DisplayAnalyzeExpenseSummaryInfo(ExpenseDocument expensedocument)
{
 System.out.println(" ExpenseId : " + expensedocument.expenseIndex());
 System.out.println("    Expense Summary information:");
 if (expensedocument.hasSummaryFields()) {

  List<ExpenseField> summaryfields = expensedocument.summaryFields();

  for (ExpenseField summaryfield : summaryfields) {

   System.out.println("    Page: " + summaryfield.pageNumber());
   if (summaryfield.type() != null) {

    System.out.println("    Expense Summary Field Type:" +
summaryfield.type().text());

   }
   if (summaryfield.labelDetection() != null) {

    System.out.println("    Expense Summary Field Label:" +
summaryfield.labelDetection().text());
    System.out.println("    Geometry");
    System.out.println("        Bounding Box: "
      + summaryfield.labelDetection().geometry().boundingBox().toString());
    System.out.println(
      "        Polygon: " +
summaryfield.labelDetection().geometry().polygon().toString());

   }
   if (summaryfield.valueDetection() != null) {
    System.out.println("    Expense Summary Field Value:" +
summaryfield.valueDetection().text());
    System.out.println("    Geometry");
    System.out.println("        Bounding Box: "
```

```
      + summaryfield.valueDetection().geometry().boundingBox().toString());
    System.out.println(
      "        Polygon: " +
summaryfield.valueDetection().geometry().polygon().toString());

   }

  }

 }

}

private void DisplayAnalyzeExpenseLineItemGroupsInfo(ExpenseDocument
expensedocument) {

 System.out.println(" ExpenseId : " + expensedocument.expenseIndex());
 System.out.println("    Expense LineItemGroups information:");

 if (expensedocument.hasLineItemGroups()) {

  List<LineItemGroup> lineitemgroups = expensedocument.lineItemGroups();

  for (LineItemGroup lineitemgroup : lineitemgroups) {

   System.out.println("    Expense LineItemGroupsIndexID :" +
lineitemgroup.lineItemGroupIndex());

   if (lineitemgroup.hasLineItems()) {

    List<LineItemFields> lineItems = lineitemgroup.lineItems();

    for (LineItemFields lineitemfield : lineItems) {

     if (lineitemfield.hasLineItemExpenseFields()) {

      List<ExpenseField> expensefields = lineitemfield.lineItemExpenseFields();
      for (ExpenseField expensefield : expensefields) {

       if (expensefield.type() != null) {
        System.out.println("    Expense LineItem Field Type:" +
expensefield.type().text());

       }
```

```
       if (expensefield.valueDetection() != null) {
        System.out.println(
          "    Expense Summary Field Value:" +
expensefield.valueDetection().text());
        System.out.println("    Geometry");
        System.out.println("        Bounding Box: "
          + expensefield.valueDetection().geometry().boundingBox().toString());
        System.out.println("        Polygon: "
          + expensefield.valueDetection().geometry().polygon().toString());

       }

       if (expensefield.labelDetection() != null) {
        System.out.println(
          "    Expense LineItem Field Label:" +
expensefield.labelDetection().text());
        System.out.println("    Geometry");
        System.out.println("        Bounding Box: "
          + expensefield.labelDetection().geometry().boundingBox().toString());
        System.out.println("        Polygon: "
          + expensefield.labelDetection().geometry().polygon().toString());
       }

      }
     }

    }

   }
  }
 }

}

public static void main(String arg[]) throws Exception {

 // Creates a default async client with credentials and AWS Region loaded from
 // the
 // environment


 S3AsyncClient client =
S3AsyncClient.builder().region(Region.US_EAST_1).build();
```

```
  System.out.println("Creating the S3 Client");

  // Start the call to Amazon S3, not blocking to wait for the result
  CompletableFuture<ResponseBytes<GetObjectResponse>> responseFuture =
 client.getObject(
    GetObjectRequest.builder().bucket("textractanalyzeexpense").key("input/
sample-receipt.jpg").build(),
    AsyncResponseTransformer.toBytes());

  System.out.println("Successfully read the object");

  // When future is complete (either successfully or in error), handle the
  // response
  CompletableFuture<ResponseBytes<GetObjectResponse>> operationCompleteFuture =
 responseFuture
    .whenComplete((getObjectResponse, exception) -> {
     if (getObjectResponse != null) {
      // At this point, the file my-file.out has been created with the data
      // from S3; let's just print the object version
      // Convert this into Async call and remove the below block from here and
 put it
      // outside


      TextractClient textractclient =
 TextractClient.builder().region(Region.US_EAST_1).build();

      AnalyzeExpenseRequest request = AnalyzeExpenseRequest.builder()
        .document(
          Document.builder().s3Object(S3Object.builder().name("YOURObjectName")
            .bucket("YOURBucket").build()).build())
        .build();

      AnalyzeExpenseResponse result = textractclient.analyzeExpense(request);

      System.out.print(result.toString());

      ByteArrayInputStream bais = new
 ByteArrayInputStream(getObjectResponse.asByteArray());
      try {
       BufferedImage image = ImageIO.read(bais);
       System.out.println("Successfully read the image");
       JFrame frame = new JFrame("Expense Image");
```

```
       frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
       TextractAnalyzeExpense panel = new TextractAnalyzeExpense(result, image);
       panel.setPreferredSize(new Dimension(image.getWidth(),
 image.getHeight()));
       frame.setContentPane(panel);
       frame.pack();
       frame.setVisible(true);
      } catch (IOException e) {
       throw new RuntimeException(e);
      } catch (Exception e) {
       // TODO Auto-generated catch block
       e.printStackTrace();
      }
     } else {
      // Handle the error
      exception.printStackTrace();
     }
    });

  // We could do other work while waiting for the AWS call to complete in
  // the background, but we'll just wait for "whenComplete" to finish instead
  operationCompleteFuture.join();

 }
}
```

Node.Js

```
                    // Import required AWS SDK clients and commands for Node.js
import { AnalyzeExpenseCommand } from  "@aws-sdk/client-textract";
import  { TextractClient } from "@aws-sdk/client-textract";

// Set the AWS Region.
const REGION = "region"; //e.g. "us-east-1"
// Create SNS service object.
const textractClient = new TextractClient({ region: REGION });

const bucket = 'bucket'
const photo = 'photo'
```

```
// Set params
const params = {
    Document: {
      S3Object: {
        Bucket: bucket,
        Name: photo
      },
    },
  }

const process_text_detection = async () => {
    try {
        const aExpense = new AnalyzeExpenseCommand(params);
        const response = await textractClient.send(aExpense);
        //console.log(response)
        response.ExpenseDocuments.forEach(doc => {
            doc.LineItemGroups.forEach(items => {
                items.LineItems.forEach(fields => {
                    fields.LineItemExpenseFields.forEach(expenseFields =>{
                        console.log(expenseFields)
                    })
                }
                )}
                )
            }
        )
        return response; // For unit tests.
      } catch (err) {
        console.log("Error", err);
      }
}

process_text_detection()
```

4. Dies bietet Ihnen die JSON-Ausgabe für die`AnalyzeExpense`verwenden.

Analysieren der Identitätsdokumentation mit Amazon Textract

Um Ausweisdokumente zu analysieren, verwenden Sie die AnalyeID-API und übergeben eine Dokumentdatei als Eingabe.`AnalyzeID`gibt eine JSON-Struktur zurück, die den analysierten Text enthält. Weitere Informationen finden Sie unter Analysieren von Ausweisdokumenten .

Sie können ein Eingabedokument als Bild-Byte-Array (base64-verschlüsselte Bild-Bytes) oder als Amazon S3 -Objekt zur Verfügung stellen. Bei dieser Vorgehensweise laden Sie eine Bilddatei in Ihren S3-Bucket hoch und geben den Dateinamen an.

So analysieren Sie ein Ausweisdokument (API)

1. Wenn Sie dies noch nicht getan haben:

   a. Erstellen oder aktualisieren Sie einen IAM-Benutzer mit`AmazonTextractFullAccess`und`AmazonS3ReadOnlyAccess`Berechtigungen Weitere Informationen finden Sie unter Schritt 1: Einrichten eines AWS-Kontos und Erstellen eines IAM-Benutzers .

   b. Installieren und konfigurieren Sie die AWS CLI- und AWS-SDKs. Weitere Informationen finden Sie unter Schritt 2: Einrichten derAWS CLIundAWS-SDKs .

2. Laden Sie ein Bild auf Ihren S3-Bucket hoch, auf dem ein Dokument abgebildet sind.

   Detaillierte Anweisungen finden Sie unterHochladen von Objekten in Amazon S3imAmazon Simple Storage Service — Benutzerhandbuchaus.

3. Verwenden Sie die folgenden Beispiele zum Aufrufen der `AnalyzeID`-Operation.

   CLI

   Im folgenden Beispiel wird eine Eingabedatei aus einem S3-Bucket aufgenommen und führt den`AnalyzeID`Operation drauf. Ersetzen Sie im folgenden Code den Wert von*`Eimer`*Mit dem Namen Ihres S3-Buckets, dem Wert von*`Ordner`*Mit dem Namen der Datei in Ihrem -Bucket und dem Wert von*`Region`*mit dem Namen`region`verknüpft mit Ihrem -Konto.

   ```
   aws textract analyze-id --document-pages '[{"S3Object":
   {"Bucket":"bucket","Name":"name"}}]' --region region
   ```

   Sie können die API auch mit der Vorder- und Rückseite eines Führerscheins aufrufen, indem Sie der Eingabe ein weiteres S3-Objekt hinzufügen.

```
aws textract analyze-id --document-pages '[{"S3Object":
{"Bucket":"bucket","Name":"name front"}}, {"S3Object":
{"Bucket":"bucket","Name":"name back"}}]' --region us-east-1
```

Wenn Sie auf einem Windows-Gerät auf die CLI zugreifen, verwenden Sie doppelte Anführungszeichen anstelle von einfachen Anführungszeichen und entgehen Sie den inneren doppelten Anführungszeichen durch umgekehrten Schrägstrich (d. h.\), um eventuell auftretende Parserfehler zu beheben. Ein Beispiel finden Sie unten:

```
aws textract analyze-id --document-pages "[{\"S3Object\":{\"Bucket\":\"bucket\",
\"Name\":\"name\"}}]" --region region
```

Python

Im folgenden Beispiel wird eine Eingabedatei aus einem S3-Bucket aufgenommen und führt den`AnalyzeID`-Operation darauf, gibt die erkannten Schlüssel-Wert-Paare zurück. Ersetzen Sie im folgenden Code den Wert von*Bucketname*Mit dem Namen Ihres S3-Buckets, dem Wert von*file_name*Mit dem Namen der Datei in Ihrem -Bucket und dem Wert von*Region*mit dem Namen`region`verknüpft mit Ihrem -Konto.

```
import boto3

bucket_name = "bucket-name"
file_name = "file-name"
region = "region-name"

def analyze_id(region, bucket_name, file_name):

    textract_client = boto3.client('textract', region_name=region)
    response = textract_client.analyze_id(DocumentPages=[{"S3Object":
{"Bucket":bucket_name,"Name":file_name}}])

    for doc_fields in response['IdentityDocuments']:
        for id_field in doc_fields['IdentityDocumentFields']:
            for key, val in id_field.items():
                if "Type" in str(key):
                    print("Type: " + str(val['Text']))
            for key, val in id_field.items():
                if "ValueDetection" in str(key):
                    print("Value Detection: " + str(val['Text']))
            print()
```

```
analyze_id(region, bucket_name, file_name)
```

Java

Im folgenden Beispiel wird eine Eingabedatei aus einem S3-Bucket aufgenommen und führt den`AnalyzeID`-Operation darauf, die erkannten Daten zurückgeben. Ersetzen Sie in der Funktion main die Werte von`s3bucket`und`sourceDoc`Mit den Namen des Amazon S3 S3-Bucket und des Dokumentbilds, das Sie in Schritt 2 verwendet haben.

```
/*
   Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
   SPDX-License-Identifier: Apache-2.0
*/

package com.amazonaws.samples;

import com.amazonaws.regions.Regions;
import com.amazonaws.services.textract.AmazonTextractClient;
import com.amazonaws.services.textract.AmazonTextractClientBuilder;
import com.amazonaws.services.textract.model.*;
import java.util.ArrayList;
import java.util.List;

public class AnalyzeIdentityDocument {

    public static void main(String[] args) {

        final String USAGE = "\n" +
                "Usage:\n" +
                "    <s3bucket><sourceDoc> \n\n" +
                "Where:\n" +
                "    s3bucket - the Amazon S3 bucket where the document is
 located. \n" +
                "    sourceDoc - the name of the document. \n";

        if (args.length != 1) {
            System.out.println(USAGE);
            System.exit(1);
        }

        String s3bucket = "bucket-name"; //args[0];
        String sourceDoc = "sourcedoc-name";  //args[1];
```

```
       AmazonTextractClient textractClient = (AmazonTextractClient)
AmazonTextractClientBuilder.standard()
               .withRegion(Regions.US_EAST_1)
               .build();

       getDocDetails(textractClient, s3bucket, sourceDoc);
   }

   public static void getDocDetails(AmazonTextractClient textractClient, String
s3bucket, String sourceDoc ) {

      try {

           S3Object s3 = new S3Object();
           s3.setBucket(s3bucket);
           s3.setName(sourceDoc);

           com.amazonaws.services.textract.model.Document myDoc = new
com.amazonaws.services.textract.model.Document();
           myDoc.setS3Object(s3);

           List<Document> list1 = new ArrayList();
           list1.add(myDoc);

           AnalyzeIDRequest idRequest = new AnalyzeIDRequest();
           idRequest.setDocumentPages(list1);

           AnalyzeIDResult result = textractClient.analyzeID(idRequest);
           List<IdentityDocument> docs =  result.getIdentityDocuments();
           for (IdentityDocument doc: docs) {

               List<IdentityDocumentField>idFields =
doc.getIdentityDocumentFields();
               for (IdentityDocumentField field: idFields) {
                   System.out.println("Field type is "+
field.getType().getText());
                   System.out.println("Field value is "+
field.getValueDetection().getText());
               }
           }

      } catch (Exception e) {
           e.printStackTrace();
      }
```

```
    }
}
```

4. Dies bietet Ihnen die JSON-Ausgabe für die`AnalyzeID`verwenden.

Dokumente mit asynchronen Operationen verarbeiten

Amazon Textract kann Text in mehrseitigen Dokumenten im PDF- oder TIFF-Format erkennen und analysieren. Dies schließt Rechnungen und Belege ein. Die Verarbeitung von mehrseitigen Dokumenten ist ein asynchroner Vorgang. Die asynchrone Verarbeitung von Dokumenten ist nützlich für die Verarbeitung großer, mehrseitiger Dokumente. Beispielsweise dauert die Verarbeitung einer PDF-Datei mit über 1.000 Seiten eine Weile. Durch die asynchrone Verarbeitung der PDF-Datei kann Ihre Anwendung andere Aufgaben ausführen, während sie auf den Abschluss des Prozesses wartet.

In diesem Abschnitt wird beschrieben, wie Sie Amazon Textract verwenden können, um Text in einem mehrseitigen oder einseitigen Dokument asynchron zu erkennen und zu analysieren. Mehrseitige Dokumente müssen im PDF- oder TIFF-Format vorliegen. Einzelseitige Dokumente, die mit asynchronen Operationen verarbeitet werden, können im JPEG-, PNG-, TIFF- oder PDF-Format vorliegen.

Sie können asynchrone Operationen von Amazon Textract für die folgenden Zwecke verwenden:

- Texterkennung — Sie können Zeilen und Wörter in einem mehrseitigen Dokument erkennen. Die asynchronen Operationen sindStartDocumentTextDetectionundGetDocumentTextDetectionaus. Weitere Informationen finden Sie unter Erkennen von Text .
- Textanalyse — Sie können Beziehungen zwischen erkanntem Text in einem mehrseitigen Dokument identifizieren. Die asynchronen Operationen sindStartDocumentAnalysisundGetDocumentAnalysisaus. Weitere Informationen finden Sie unter Analysieren von Dokumenten .
- Kostenanalyse — Sie können Datenbeziehungen auf mehrseitigen Rechnungen und Belegen identifizieren. Amazon Textract behandelt jede Rechnung oder eine Belegseite eines mehrseitigen Dokuments als Einzelbeleg oder Rechnung. Es behält den Kontext nicht von einer Seite zur anderen eines mehrseitigen Dokuments bei. Die asynchronen Operationen sindStartExpenseAnalysisundGetExpenseAnalysisaus. Weitere Informationen finden Sie unter Analysieren von Rechnungen und Belegen .

Themen

- Asynchrone Operationen von Amazon Textract aufrufen
- Konfigurieren von Amazon Textract für asynchrone Vorgänge
- Erkennen oder Analysieren von Text in einem mehrseitigen Dokument
- Amazon Textract Ergebnisbenachrichtigung

Asynchrone Operationen von Amazon Textract aufrufen

Amazon Textract bietet eine asynchrone API, mit der Sie mehrseitige Dokumente im PDF- oder TIFF-Format verarbeiten können. Sie können auch asynchrone Operationen verwenden, um einseitige Dokumente im JPEG-, PNG-, TIFF- oder PDF-Format zu verarbeiten.

Die Informationen in diesem Thema verwenden Texterkennungsvorgänge, um zu zeigen, wie asynchrone Operationen von Amazon Textract verwendet werden. Derselbe Ansatz funktioniert mit den Textanalyseoperationen vonthe section called “StartDocumentAnalysis”undthe section called “GetDocumentAnalysis”aus. Es funktioniert auch gleich mitthe section called “StartExpenseAnalysis”undthe section called “GetExpenseAnalysis”aus.

Ein Beispiel finden Sie unter Erkennen oder Analysieren von Text in einem mehrseitigen Dokument.

Amazon Textract verarbeitet asynchron ein Dokument, das in einem Amazon S3 S3-Bucket gespeichert ist. Sie beginnen mit der Verarbeitung, indem Sie ein`StartBetrieb`, wieStartDocumentTextDetectionaus. Der Erledigungsstatus der Anfrage wird in einem Amazon Simple Notification Service (Amazon SNS) -Thema veröffentlicht. Um den Erledigungsstatus aus dem Amazon-SNS-Thema zu erhalten, können Sie eine Amazon Simple Queue Service (Amazon SQS) -Warteschlange oder eineAWS LambdaFunktion. Sobald Sie den Erledigungsstatus bekommen haben, rufen Sie eine `Get`-Operation auf, wie z. B. GetDocumentTextDetection, um die Ergebnisse der Anfrage zu erhalten.

Die Ergebnisse asynchroner Anrufe werden standardmäßig verschlüsselt und 7 Tage lang in einem Bucket im Besitz von Amazon Textract gespeichert, es sei denn, Sie geben einen Amazon S3 S3-Bucket mit einem Vorgang an`OutputConfig`Argument.

Die folgende Tabelle zeigt die entsprechenden Start- und Abgabevorgänge für die verschiedenen von Amazon Textract unterstützten asynchronen Verarbeitungstypen:

API-Operationen für Asynchrone Operationen von Amazon Textract starten/abrufen

Art der Verarbeitung	API starten	Erhalten einer API
Texterkennung	StartDocumentTextDetection	GetDocumentTextDetection
Textanalyse	StartDocumentAnalysis	GetDocumentAnalysis
Kostenanalyse	StarteExpenseAnalysis	getExpenseAnalysis

Für ein Beispiel, das verwendetAWS LambdaFunktionen finden Sie unterDokumentenverarbeitung im großen Maßstab mit Amazon Textractaus.

Im folgenden Diagramm wird der Vorgang zur Erkennung von Dokumenttext in einem Dokumentbild, das in einem Amazon-S3-Bucket gespeichert ist, zeigt. Im Diagramm erhält eine Amazon SQS SQS-Warteschlange den Erledigungsstatus aus dem Amazon SNS SNS-Thema.

Der im vorhergehenden Diagramm angezeigte Prozess ist für die Analyse von Text und Rechnungen/Belegen identisch. Sie beginnen mit der Analyse von Text, indem Sie anrufenthe section called “StartDocumentAnalysis”und analysieren Sie Rechnungen/Quittungen, indem Sie anrufenthe section called “StartExpenseAnalysis”Sie erhalten die Ergebnisse, indem Sie anrufenthe section called “GetDocumentAnalysis”oderthe section called “GetExpenseAnalysis”beziehungsweise.

Starten der Texterkennung

Sie starten eine Amazon Textract--Texterkennungsanforderung, indem Sie aufrufenStartDocumentTextDetectionaus. Im Folgenden sehen Sie ein Beispiel für eine JSON-Anforderung, die von `StartDocumentTextDetection` übergeben wird.

```
{
    "DocumentLocation": {
        "S3Object": {
            "Bucket": "bucket",
            "Name": "image.pdf"
        }
    },
    "ClientRequestToken": "DocumentDetectionToken",
    "NotificationChannel": {
        "SNSTopicArn": "arn:aws:sns:us-east-1:nnnnnnnnnn:topic",
        "RoleArn": "arn:aws:iam::nnnnnnnnnn:role/roleTopic"
    },
    "JobTag": "Receipt"
}
```

Der EingabeparameterDocumentLocationliefert den Dateinamen des -Dokuments und den Amazon S3 S3-Bucket, aus dem sie abgerufen werden kann.NotificationChannelenthält den Amazon-Ressourcennamen (ARN) des Amazon-SNS-Themas, das Amazon Textract benachrichtigt, wenn die Anfrage zur Texterkennung beendet ist. Das Amazon SNS SNS-Thema muss sich in derselben AWS-Region befinden wie der Amazon Textract Textract-Endpunkt,

den Sie aufrufen.`NotificationChannel`enthält auch den ARN für eine Rolle, die es Amazon Textract erlaubt, das Amazon SNS SNS-Thema zu veröffentlichen. Sie erteilen Amazon Textract Textract-Veröffentlichungsberechtigungen für Ihre Amazon SNS SNS-Themen, indem Sie eine IAM-Servicerolle erstellen. Weitere Informationen finden Sie unter Konfigurieren von Amazon Textract für asynchrone Vorgänge .

Sie können auch einen optionalen Eingabeparameter angeben,`JobTag`, der es Ihnen ermöglicht, den Auftrag oder Gruppen von Aufträgen im Erledigungsstatus zu identifizieren, der im Amazon-SNS-Thema veröffentlicht ist. Sie können beispielsweise die Verwendung von`JobTag`um die Art des zu bearbeitenden Dokuments zu ermitteln, z. B. ein Steuerformular oder eine Quittung.

Um ein versehentliches doppeltes Ausführen von Analyseaufträgen zu vermeiden, können Sie optional ein idempotentes Token, `ClientRequestToken`, bereitstellen. Wenn Sie einen Wert angeben für`ClientRequestToken`, der`Start`-Operation gibt das Gleiche zurück`JobId`für mehrere identische Anrufe an die`Start`Betrieb, wie`StartDocumentTextDetection`aus. Ein Token `ClientRequestToken` hat eine Lebensdauer von 7 Tagen. Nach 7 Tagen können Sie es wiederverwenden. Wenn Sie das Token während der Token-Lebensdauer wiederverwenden, geschieht folgendes:

- Wenn Sie das Token mit der gleichen `Start`-Operation und den gleichen Eingabeparametern wiederverwenden, wird dieselbe `JobId` zurückgegeben. Der Auftrag wird nicht erneut ausgeführt und Amazon Textract sendet keinen Erledigungsstatus an das registrierte Amazon SNS SNS-Thema.
- Wenn Sie das Token mit der gleichen `Start`-Operation und einer geringfügigen Änderung der Eingabeparameter wiederverwenden, wird eine `idempotentparametermismatchexception`-Ausnahme (HTTP-Statuscode: 400) ausgelöst.
- Wenn Sie das Token mit einer anderen `Start`-Operation wiederverwenden, ist die Operation erfolgreich.

Ein weiterer optionaler Parameter ist verfügbar`OutputConfig`, mit dem Sie einstellen können, wo Ihre Ausgabe platziert wird. Standardmäßig speichert Amazon Textract die Ergebnisse intern und kann nur von den API-Vorgängen abrufen aufgerufen werden. mit`OutputConfig`aktiviert, können Sie den Namen des Buckets festlegen, an den die Ausgabe gesendet wird, und das Dateipräfix der Ergebnisse, in dem Sie Ihre Ergebnisse herunterladen können. Darüber hinaus können Sie die`KMSKeyID`Parameter an einen vom Kunden verwalteten Schlüssel, um Ihre Ausgabe zu verschlüsseln. Ohne diesen Parametersatz verschlüsselt Amazon Textract serverseitig mit demVon AWS verwalteter Schlüsselfür Amazon S3

> **Note**
>
> Bevor Sie diesen Parameter verwenden, stellen Sie sicher, dass Sie die PutObject-Berechtigung für den Ausgabe-Bucket haben Stellen Sie außerdem sicher, dass Sie über die Berechtigungen Decrypt, ReEncrypt, GenerateDataKey und DescribeKey verfügen für dieAWS KMSSchlüssel, wenn Sie sich entscheiden, ihn zu verwenden.

Die Antwort auf die `StartDocumentTextDetection`-Operation ist eine Auftrags-ID (`JobId`). Verwenden von`JobId`Um Anfragen zu verfolgen und die Analyseergebnisse zu erhalten, nachdem Amazon Textract den Status der Fertigstellung im Amazon-SNS-Thema veröffentlicht hat. Im Folgenden wird ein Beispiel gezeigt:

```
{"JobId":"270c1cc5e1d0ea2fbc59d97cb69a72a5495da75851976b14a1784ca90fc180e3"}
```

Wenn Sie gleichzeitig zu viele Jobs starten, rufen Sie an`StartDocumentTextDetection`RAISE ein`LimitExceededException`-Ausnahme (HTTP-Statuscode: 400), bis die Anzahl der gleichzeitig ausgeführten Aufträge unter dem Amazon Textract Texttract-Service-Limit liegt.

Wenn Sie feststellen, dass LimitExceededException-Ausnahmen bei Teilaktivitäten ausgelöst werden, empfiehlt sich für die Verwaltung eingehender Anforderungen die Verwendung einer Amazon SQS SQS-Warteschlange. KontaktAWSSupport, wenn Sie feststellen, dass Ihre durchschnittliche Anzahl gleichzeitiger Anforderungen nicht von einer Amazon SQS SQS-Warteschlange verwaltet werden kann und Sie weiterhin erhalten`LimitExceededException`Ausnahmen.

Abrufen des Erledigungsstatus einer Amazon Textract Textract-Analyseanforderung

Amazon Textract sendet eine Benachrichtigung über den Abschluss der Analyse an das registrierte Amazon SNS SNS-Thema. Die Benachrichtigung enthält die Auftrags-ID und den Erledigungsstatus der Operation in einer JSON-Zeichenfolge. Eine erfolgreiche Anfrage zur Texterkennung hat eine`SUCCEEDED`Status. Das folgende Ergebnis zeigt beispielsweise die erfolgreiche Verarbeitung eines Texterkennungsauftrags.

```
{
    "JobId": "642492aea78a86a40665555dc375ee97bc963f342b29cd05030f19bd8fd1bc5f",
    "Status": "SUCCEEDED",
```

```
    "API": "StartDocumentTextDetection",
    "JobTag": "Receipt",
    "Timestamp": 1543599965969,
    "DocumentLocation": {
        "S3ObjectName": "document",
        "S3Bucket": "bucket"
    }
}
```

Weitere Informationen finden Sie unter Amazon Textract Ergebnisbenachrichtigung .

Verwenden Sie eine der folgenden Optionen, um die Statusinformationen zu erhalten, die im Amazon-SNS-Thema von Amazon SNS-Thema von Amazon Textract veröffentlicht werden:

- AWS Lambda— Sie können eine abonnierenAWS Lambda-Funktion, die Sie in ein Amazon SNS SNS-Thema schreiben. Die Funktion wird aufgerufen, wenn Amazon Textract das Amazon SNS SNS-Thema darüber informiert, dass die Anfrage abgeschlossen ist. Verwenden Sie eine Lambda-Funktion, wenn Sie serverseitigen Code zur Verarbeitung der Ergebnisse einer Texterkennungsanforderung benötigen. Sie können beispielsweise serverseitigen Code verwenden, um das Bild mit Anmerkungen zu versehen oder einen Bericht über den erkannten Text zu erstellen, bevor die Informationen an eine Client-Anwendung zurückgegeben werden.
- Amazon SQS- Sie können eine Amazon SQS SQS-Warteschlange für ein Amazon SNS SNS-Thema abonnieren. Sie fragen dann die Amazon SQS SQS-Warteschlange ab, um den von Amazon Textract veröffentlichten Erledigungsstatus abzurufen, wenn eine Anfrage zur Texterkennung abgeschlossen ist. Weitere Informationen finden Sie unter Erkennen oder Analysieren von Text in einem mehrseitigen Dokument . Verwenden Sie eine Amazon SQS SQS-Warteschlange, wenn Sie Amazon Textract Textract-Operationen nur von einer Client-Anwendung aus aufrufen möchten.

> **Important**
>
> Wir raten davon ab, den Status der Auftragserfüllung durch wiederholtes Aufrufen des Amazon Textract`Get`verwenden. Dies liegt daran, dass Amazon Textract die`Get`-Betrieb, wenn zu viele Anfragen gestellt werden. Wenn Sie mehrere Dokumente gleichzeitig verarbeiten, ist es einfacher und effizienter, eine SQS-Warteschlange für die Erledigungsbenachrichtigung zu überwachen, als Amazon Textract nach dem Status jedes Auftrags einzeln abzufragen.

Ergebnisse der Amazon Textract Textract-Texterkennung abrufen

Um die Ergebnisse einer Texterkennungsanforderung zu erhalten, stellen Sie zunächst sicher, dass der Erledigungsstatus, der aus dem Amazon-SNS-Thema abgerufen wird, lautetSUCCEEDEDaus. Rufen Sie dann `GetDocumentTextDetection` auf, wodurch der Wert `JobId` übergeben wird, den `StartDocumentTextDetection` zurückgibt. Die JSON-Ausgabe sieht folgendermaßen oder ähnlich aus:

```
{
    "JobId": "270c1cc5e1d0ea2fbc59d97cb69a72a5495da75851976b14a1784ca90fc180e3",
    "MaxResults": 10,
    "SortBy": "TIMESTAMP"
}
```

`JobId`ist der Bezeichner für den Texterkennungsvorgang. Da die Texterkennung große Datenmengen erzeugen kann, verwenden Sie`MaxResults`um die maximale Anzahl der Ergebnisse anzugeben, die in einem einzigen`Get`verwenden. Der Standardwert für`MaxResults`ist 1.000. Wenn Sie einen Wert größer als 1.000 angeben, werden nur 1.000 Ergebnisse zurückgegeben. Wenn die Operation nicht alle Ergebnisse zurückgibt, wird ein Paginierungs-Token für die nächste Seite zurückgegeben. Um die nächste Ergebnisseite abzurufen, geben Sie das Token in der`NextToken`-Parameter.

> **Note**
>
> Amazon Textract bewahrt die Ergebnisse asynchroner Operationen für 7 Tage auf. Nach dieser Zeit können Sie die Ergebnisse nicht mehr abrufen.

Die`GetDocumentTextDetection`Die Operation Response JSON ähnelt dem folgenden Beispiel. Die Gesamtzahl der erkannten Seiten wird in zurückgegeben`DocumentMetadata`aus. Der erkannte Text wird im`Blocks`Array. Weitere Informationen zu`Block`objekte, siehe[Antwortobjekte für Texterkennung und Dokumentanalyse](#)aus.

```
{
    "DocumentMetadata": {
        "Pages": 1
    },
    "JobStatus": "SUCCEEDED",
    "Blocks": [
```

```
        {
            "BlockType": "PAGE",
            "Geometry": {
                "BoundingBox": {
                    "Width": 1.0,
                    "Height": 1.0,
                    "Left": 0.0,
                    "Top": 0.0
                },
                "Polygon": [
                    {
                        "X": 0.0,
                        "Y": 0.0
                    },
                    {
                        "X": 1.0,
                        "Y": 0.0
                    },
                    {
                        "X": 1.0,
                        "Y": 1.0
                    },
                    {
                        "X": 0.0,
                        "Y": 1.0
                    }
                ]
            },
            "Id": "64533157-c47e-401a-930e-7ca1bb3ac3fa",
            "Relationships": [
                {
                    "Type": "CHILD",
                    "Ids": [
                        "4297834d-dcb1-413b-8908-3b96866ebbb5",
                        "1d85ba24-2877-4d09-b8b2-393833d769e9",
                        "193e9c47-fd87-475a-ba09-3fda210d8784",
                        "bd8aeb62-961b-4b47-b78a-e4ed9eeecd0f"
                    ]
                }
            ],
            "Page": 1
        },
        {
            "BlockType": "LINE",
```

```
            "Confidence": 53.301639556884766,
            "Text": "ellooworio",
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.9999999403953552,
                    "Height": 0.5365243554115295,
                    "Left": 0.0,
                    "Top": 0.46347561478614807
                },
                "Polygon": [
                    {
                        "X": 0.0,
                        "Y": 0.46347561478614807
                    },
                    {
                        "X": 0.9999999403953552,
                        "Y": 0.46347561478614807
                    },
                    {
                        "X": 0.9999999403953552,
                        "Y": 1.0
                    },
                    {
                        "X": 0.0,
                        "Y": 1.0
                    }
                ]
            },
            "Id": "4297834d-dcb1-413b-8908-3b96866ebbb5",
            "Relationships": [
                {
                    "Type": "CHILD",
                    "Ids": [
                        "170c3eb9-5155-4bec-8c44-173bba537e70"
                    ]
                }
            ],
            "Page": 1
        },
        {
            "BlockType": "LINE",
            "Confidence": 89.15632629394531,
            "Text": "He llo,",
            "Geometry": {
```

```
                "BoundingBox": {
                    "Width": 0.33642634749412537,
                    "Height": 0.49159330129623413,
                    "Left": 0.13885067403316498,
                    "Top": 0.17169663310050964
                },
                "Polygon": [
                    {
                        "X": 0.13885067403316498,
                        "Y": 0.17169663310050964
                    },
                    {
                        "X": 0.47527703642845154,
                        "Y": 0.17169663310050964
                    },
                    {
                        "X": 0.47527703642845154,
                        "Y": 0.6632899641990662
                    },
                    {
                        "X": 0.13885067403316498,
                        "Y": 0.6632899641990662
                    }
                ]
            },
            "Id": "1d85ba24-2877-4d09-b8b2-393833d769e9",
            "Relationships": [
                {
                    "Type": "CHILD",
                    "Ids": [
                        "516ae823-3bab-4f9a-9d74-ad7150d128ab",
                        "6bcf4ea8-bbe8-4686-91be-b98dd63bc6a6"
                    ]
                }
            ],
            "Page": 1
        },
        {
            "BlockType": "LINE",
            "Confidence": 82.44834899902344,
            "Text": "worlo",
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.33182239532470703,
```

```
                "Height": 0.3766750991344452,
                "Left": 0.5091826915740967,
                "Top": 0.23131252825260162
            },
            "Polygon": [
                {
                    "X": 0.5091826915740967,
                    "Y": 0.23131252825260162
                },
                {
                    "X": 0.8410050868988037,
                    "Y": 0.23131252825260162
                },
                {
                    "X": 0.8410050868988037,
                    "Y": 0.607987642288208
                },
                {
                    "X": 0.5091826915740967,
                    "Y": 0.607987642288208
                }
            ]
        },
        "Id": "193e9c47-fd87-475a-ba09-3fda210d8784",
        "Relationships": [
            {
                "Type": "CHILD",
                "Ids": [
                    "ed135c3b-35dd-4085-8f00-26aedab0125f"
                ]
            }
        ],
        "Page": 1
    },
    {
        "BlockType": "LINE",
        "Confidence": 88.50325775146484,
        "Text": "world",
        "Geometry": {
            "BoundingBox": {
                "Width": 0.35004907846450806,
                "Height": 0.19635874032974243,
                "Left": 0.527581512928009,
                "Top": 0.30100569128990173
```

```
                },
                "Polygon": [
                    {
                        "X": 0.527581512928009,
                        "Y": 0.30100569128990173
                    },
                    {
                        "X": 0.8776305913925171,
                        "Y": 0.30100569128990173
                    },
                    {
                        "X": 0.8776305913925171,
                        "Y": 0.49736443161964417
                    },
                    {
                        "X": 0.527581512928009,
                        "Y": 0.49736443161964417
                    }
                ]
            },
            "Id": "bd8aeb62-961b-4b47-b78a-e4ed9eeecd0f",
            "Relationships": [
                {
                    "Type": "CHILD",
                    "Ids": [
                        "9e28834d-798e-4a62-8862-a837dfd895a6"
                    ]
                }
            ],
            "Page": 1
        },
        {
            "BlockType": "WORD",
            "Confidence": 53.301639556884766,
            "Text": "ellooworio",
            "Geometry": {
                "BoundingBox": {
                    "Width": 1.0,
                    "Height": 0.5365243554115295,
                    "Left": 0.0,
                    "Top": 0.46347561478614807
                },
                "Polygon": [
                    {
```

```
                    "X": 0.0,
                    "Y": 0.46347561478614807
                },
                {
                    "X": 1.0,
                    "Y": 0.46347561478614807
                },
                {
                    "X": 1.0,
                    "Y": 1.0
                },
                {
                    "X": 0.0,
                    "Y": 1.0
                }
            ]
        },
        "Id": "170c3eb9-5155-4bec-8c44-173bba537e70",
        "Page": 1
    },
    {
        "BlockType": "WORD",
        "Confidence": 88.46246337890625,
        "Text": "He",
        "Geometry": {
            "BoundingBox": {
                "Width": 0.15350718796253204,
                "Height": 0.29955607652664185,
                "Left": 0.13885067403316498,
                "Top": 0.21856294572353363
            },
            "Polygon": [
                {
                    "X": 0.13885067403316498,
                    "Y": 0.21856294572353363
                },
                {
                    "X": 0.292357861995697,
                    "Y": 0.21856294572353363
                },
                {
                    "X": 0.292357861995697,
                    "Y": 0.5181190371513367
                },
```

```
                    {
                        "X": 0.13885067403316498,
                        "Y": 0.5181190371513367
                    }
                ]
            },
            "Id": "516ae823-3bab-4f9a-9d74-ad7150d128ab",
            "Page": 1
        },
        {
            "BlockType": "WORD",
            "Confidence": 89.8501968383789,
            "Text": "llo,",
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.17724157869815826,
                    "Height": 0.49159327149391174,
                    "Left": 0.2980354428291321,
                    "Top": 0.17169663310050964
                },
                "Polygon": [
                    {
                        "X": 0.2980354428291321,
                        "Y": 0.17169663310050964
                    },
                    {
                        "X": 0.47527703642845154,
                        "Y": 0.17169663310050964
                    },
                    {
                        "X": 0.47527703642845154,
                        "Y": 0.6632899045944214
                    },
                    {
                        "X": 0.2980354428291321,
                        "Y": 0.6632899045944214
                    }
                ]
            },
            "Id": "6bcf4ea8-bbe8-4686-91be-b98dd63bc6a6",
            "Page": 1
        },
        {
            "BlockType": "WORD",
```

```
            "Confidence": 82.44834899902344,
            "Text": "worlo",
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.33182239532470703,
                    "Height": 0.3766750991344452,
                    "Left": 0.5091826915740967,
                    "Top": 0.23131252825260162
                },
                "Polygon": [
                    {
                        "X": 0.5091826915740967,
                        "Y": 0.23131252825260162
                    },
                    {
                        "X": 0.8410050868988037,
                        "Y": 0.23131252825260162
                    },
                    {
                        "X": 0.8410050868988037,
                        "Y": 0.607987642288208
                    },
                    {
                        "X": 0.5091826915740967,
                        "Y": 0.607987642288208
                    }
                ]
            },
            "Id": "ed135c3b-35dd-4085-8f00-26aedab0125f",
            "Page": 1
        },
        {
            "BlockType": "WORD",
            "Confidence": 88.50325775146484,
            "Text": "world",
            "Geometry": {
                "BoundingBox": {
                    "Width": 0.35004907846450806,
                    "Height": 0.19635874032974243,
                    "Left": 0.527581512928009,
                    "Top": 0.30100569128990173
                },
                "Polygon": [
                    {
```

```
                        "X": 0.527581512928009,
                        "Y": 0.30100569128990173
                    },
                    {
                        "X": 0.8776305913925171,
                        "Y": 0.30100569128990173
                    },
                    {
                        "X": 0.8776305913925171,
                        "Y": 0.49736443161964417
                    },
                    {
                        "X": 0.527581512928009,
                        "Y": 0.49736443161964417
                    }
                ]
            },
            "Id": "9e28834d-798e-4a62-8862-a837dfd895a6",
            "Page": 1
        }
    ]
}
```


Konfigurieren von Amazon Textract für asynchrone Vorgänge

Die folgenden Verfahren zeigen Ihnen, wie Sie Amazon Textract für die Verwendung mit einem Amazon Simple Notification Service (Amazon SNS) -Thema und einer Amazon-Simple-Queue-Service-Warteschlange (Amazon SQS) konfigurieren.

> **Note**
>
> Wenn Sie diese Anweisungen verwenden, um das Erkennen oder Analysieren von Text in einem mehrseitigen Dokument Beispiel müssen Sie die Schritte 3 — 6 nicht ausführen. Das Beispiel enthält Code zum Erstellen und Konfigurieren des Amazon SNS SNS-Themas und die Amazon SQS SQS-Warteschlange.

So konfigurieren Sie Amazon Textract

1. Einrichten eines AWS-Konto für den Zugriff auf Amazon Textract. Weitere Informationen finden Sie unter Schritt 1: Einrichten eines AWS-Kontos und Erstellen eines IAM-Benutzers .

Stellen Sie sicher, dass der Benutzer mindestens über die folgenden Berechtigungen verfügt:

- AmazonTextractFullAccess
- AmazonS3ReadOnlyAccess
- AmazonSNSFullAccess
- AmazonSQSFullAccess

2. Installieren und konfigurieren Sie das erforderliche AWS-SDK. Weitere Informationen finden Sie unter Schritt 2: Einrichten derAWS CLIundAWS-SDKs .
3. Erstellen Sie ein Amazon SNS-Thema.aus. Stellen Sie dem Themennamen voranAmazonTextractaus. Notieren Sie den ARN (Amazon-Ressourcenname). Stellen Sie sicher, dass sich das -Thema in derselben -Region befindet wie dasAWSEndpunkt, den Sie mit Ihrem AWS-Konto verwenden.
4. Erstellen einer Amazon SQS SQS-Standardwarteschlangeindem Sie dieAmazon SQS SQS-Konsoleaus. Notieren Sie den ARN der Warteschlange.
5. Abonnieren Sie die Warteschlange zum Thema, das Sie in Schritt 3 erstellt haben.
6. Erteilen Sie die Berechtigung für das Amazon SNS SNS-Thema zum Senden von Nachrichten an die Amazon SQS SQS-Warteschlangeaus.
7. Erstellen Sie eine IAM-Servicerolle, um Amazon Textract Zugriff auf Ihre Amazon SNS SNS-Themen zu gewähren. Notieren Sie sich den Amazon-Ressourcennamen (ARN) der Service-Rolle. Weitere Informationen finden Sie unter Amazon Textract Zugriff auf Ihr Amazon SNS SNS-Thema gewähren .
8. Fügen Sie die folgende Inlinerichtlinie hinzuan den IAM-Anwender, den Sie in Schritt 1 erstellt haben.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "MySid",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "Service role ARN from step 7"
        }
    ]
}
```

Geben Sie der Inlinerichtlinie einen Namen.

9. Jetzt können Sie die Beispiele in ausführenErkennen oder Analysieren von Text in einem mehrseitigen Dokumentaus.

Amazon Textract Zugriff auf Ihr Amazon SNS SNS-Thema gewähren

Amazon Textract benötigt die Berechtigung, eine Nachricht an Ihr Amazon SNS SNS-Thema zu senden, wenn ein asynchroner Vorgang abgeschlossen ist. Sie verwenden eine IAM-Servicerolle, um Amazon Textract Zugriff auf das Amazon SNS SNS-Thema zu gewähren.

Wenn Sie das Amazon SNS SNS-Thema erstellen, müssen Sie dem Themennamen vorangestellt werden**`AmazonTextract`**—beispielsweise**`AmazonTextractMyTopicName`**aus.

1. Melden Sie sich bei der IAM-Konsole an (https://console.aws.amazon.com/iam).
2. Wählen Sie im Navigationsbereich Roles aus.
3. Wählen Sie Create role (Rolle erstellen) aus.
4. Wählen Sie unter Select type of trusted entity (Typ der vertrauenswürdigen Entität wählen) die Option AWS service (Service) aus.
5. FürWählen Sie den Service aus, der diese Rolle verwendet, wählenTextractaus.
6. Wählen Sie Weiter. Berechtigungen.
7. Stellen Sie sicher, dass dieAmazonTextractServiceRoleDie Richtlinie wurde in die Liste der angehängten Richtlinien aufgenommen. Um die Richtlinie in der Liste anzuzeigen, geben Sie einen Teil des Richtliniennamens in derFiltern von Richtlinienaus.
8. Wählen Sie Weiter. Tags.
9. Sie müssen keine Tags hinzufügen, wählen Sie alsoWeiter: Prüfen.
10. Geben Sie im Bereich Review (Prüfen) für Role name (Rollenname) einen Namen für die Rolle ein (z. B. `TextractRole`). In :Rollenbeschreibungaktualisieren Sie die Beschreibung für die Rolle und wählen Sie dannErstellen einer -Rolleaus.
11. Wählen Sie die neue Rolle aus, um die Detailseite der Rolle zu öffnen.
12. Kopieren Sie unter Summary (Übersicht)den Role ARN (Rollen-ARN)-Wert und speichern Sie ihn.
13. Wählen Sie Trust Relationships (Vertrauensbeziehungen) aus.
14. Klicken Sie aufBearbeiten von Vertrauensstellungenund stellen Sie sicher, dass die Vertrauensrichtlinie wie folgt aussieht.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "textract.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

15. Wählen Sie Update Trust Policy.

Erkennen oder Analysieren von Text in einem mehrseitigen Dokument

Dieses Verfahren zeigt Ihnen, wie Sie Text in einem mehrseitigen Dokument mithilfe von Amazon Textract Textract-Erkennungsvorgängen, einem in einem Amazon S3 S3-Bucket gespeicherten Dokument, einem Amazon SNS SNS-Thema und einer Amazon SQS SQS-Warteschlange erkennen oder analysieren. Die Verarbeitung von mehrseitigen Dokumenten ist ein asynchroner Vorgang. Weitere Informationen finden Sie unter Asynchrone Operationen von Amazon Textract aufrufen .

Sie können die Art der Verarbeitung auswählen, die der Code ausführen soll: Texterkennung, Textanalyse oder Kostenanalyse.

Die Verarbeitungsergebnisse werden in einem Array vonthe section called “Block”-Objekte, die sich je nach Art der Verarbeitung unterscheiden, die Sie verwenden.

Um Text in mehrseitigen Dokumenten zu erkennen oder zu analysieren, gehen Sie wie folgt vor:

1. Erstellen Sie das Amazon SNS SNS-Thema und die Amazon SQS SQS-Warteschlange.
2. Abonnieren Sie die Warteschlange das -Thema.
3. Erteilen Sie dem -Thema die Berechtigung zum Senden von Nachrichten an die Warteschlange.
4. Beginnen Sie mit der Verarbeitung des Dokuments Verwenden Sie die entsprechende Operation für die von Ihnen gewählte Analysetyp:

- StartDocumentTextDetectionfür Aufgaben zur Texterkennung.
- StartDocumentAnalysisfür Textanalyseaufgaben.
- StartExpenseAnalysisfür Aufgabenanalyseaufgaben.

5. Holen Sie sich den Status der Erledigung aus der Amazon SQS SQS-Warteschlange. Der Beispielcode verfolgt die Job-ID (`JobId`) das wird von der`Start`verwenden. Sie liefert nur die Ergebnisse für übereinstimmende Auftragskennungen, die aus dem Erledigungsstatus gelesen werden. Dies ist wichtig, wenn andere Anwendungen die gleiche Warteschlange und das gleiche Thema verwenden. Der Einfachheit halber werden in diesem Beispiel Aufträge gelöscht, die nicht übereinstimmen. Ziehen Sie in Betracht, die gelöschten Aufträge zur weiteren Untersuchung zu einer Amazon SQS SQS-Warteschlange für unzustellbare Nachrichten hinzufügen.
6. Rufen Sie die Verarbeitungsergebnisse ab und zeigen Sie sie an, indem Sie die entsprechende Operation für den ausgewählten Analysetyp aufrufen:
   - GetDocumentTextDetectionfür Aufgaben zur Texterkennung.
   - GetDocumentAnalysisfür Textanalyseaufgaben.
   - GetExpenseAnalysisfür Aufgabenanalyseaufgaben.
7. Löschen Sie das Amazon SNS SNS-Thema und die Amazon SQS SQS-Warteschlange.

Durchführen von asynchronen Operationen

Der Beispielcode für dieses Verfahren wird in Java, Python und demAWS CLIaus. Bevor Sie beginnen, installieren Sie das entsprechendeAWSSDK. Weitere Informationen finden Sie unter Schritt 2: Einrichten derAWS CLIundAWS-SDKs .

So erkennen oder analysieren Sie Text in einem mehrseitigen Dokument

1. Konfigurieren Sie den Benutzerzugriff auf Amazon Textract und konfigurieren Sie Amazon Textract Textract-Zugriff auf Amazon SNS. Weitere Informationen finden Sie unter Konfigurieren von Amazon Textract für asynchrone Vorgänge . Um dieses Verfahren abzuschließen, benötigen Sie ein mehrseitiges Dokument im PDF-Format. Überspringen Sie die Schritte 3 — 6, da der Beispielcode das Amazon SNS SNS-Thema und die Amazon SQS SQS-Warteschlange erstellt und konfiguriert. Wenn kompletIm Beispiel der CLI müssen Sie keine SQS-Warteschlange einrichten.
2. Laden Sie eine mehrseitige Dokumentdatei im PDF- oder TIFF-Format in Ihren Amazon S3 S3-Bucket hoch. (Einzelseitige Dokumente im JPEG-, PNG-, TIFF- oder PDF-Format können ebenfalls verarbeitet werden).

Detaillierte Anweisungen finden Sie unterHochladen von Objekten in Amazon S3imAmazon Simple Storage Service — Benutzerhandbuchaus.

3. Verwenden Sie FolgendesAWS SDK für Java, SDK for Python (Boto3) oderAWS CLICode, um entweder Text zu erkennen oder Text in einem mehrseitigen Dokument zu analysieren. In der`main`Funktion:

   - Ersetzen Sie den Wert von`roleArn`In dem IAM-Rollen-ARN, in dem Sie gespeichert habenAmazon Textract Zugriff auf Ihr Amazon SNS SNS-Thema gewährenaus.
   - Ersetzen Sie die Werte von`bucket`und`document`Mit dem Bucket-Namen und dem Namen der Dokumentdatei, die Sie in Schritt 2 angegeben haben.
   - Ersetzen Sie den Wert des`type`Eingabeparameter des`ProcessDocument-`Funktion mit der Art der Verarbeitung, die Sie ausführen möchten. Verwenden von`ProcessType.DETECTIO`Num Text zu erkennen. Verwenden von`ProcessType.ANALYSIS`um Text zu analysieren.
   - Ersetzen Sie für das Python-Beispiel den Wert von`region_name`mit der Region, in der Ihr Kunde tätig ist.

   Für denAWS CLIBeispiel, führen Sie folgende Schritte aus:

   - Beim AnrufenStartDocumentTextDetection, ersetzen Sie den Wert von`bucket-name`Ersetzen Sie den Namen Ihres S3-Buckets und ersetzen Sie`file-name`Mit dem Namen der Datei, die Sie in Schritt 2 angegeben haben. Geben Sie die Region Ihres Buckets an, indem Sie ersetzen`region-name`mit dem Namen Ihrer Region. Beachten Sie, dass das CLI-Beispiel SQS nicht verwendet.
   - Beim AnrufenGetDocumentTextDetectionersetzen`job-id-number`mit dem`job-id`zurückgegeben vonStartDocumentTextDetectionaus. Geben Sie die Region Ihres Buckets an, indem Sie ersetzen`region-name`mit dem Namen Ihrer Region.

   Java

```
package com.amazonaws.samples;

import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
```

```
import com.amazonaws.auth.policy.Condition;
import com.amazonaws.auth.policy.Policy;
import com.amazonaws.auth.policy.Principal;
import com.amazonaws.auth.policy.Resource;
import com.amazonaws.auth.policy.Statement;
import com.amazonaws.auth.policy.Statement.Effect;
import com.amazonaws.auth.policy.actions.SQSActions;
import com.amazonaws.services.sns.AmazonSNS;
import com.amazonaws.services.sns.AmazonSNSClientBuilder;
import com.amazonaws.services.sns.model.CreateTopicRequest;
import com.amazonaws.services.sns.model.CreateTopicResult;
import com.amazonaws.services.sqs.AmazonSQS;
import com.amazonaws.services.sqs.AmazonSQSClientBuilder;
import com.amazonaws.services.sqs.model.CreateQueueRequest;
import com.amazonaws.services.sqs.model.Message;
import com.amazonaws.services.sqs.model.QueueAttributeName;
import com.amazonaws.services.sqs.model.SetQueueAttributesRequest;
import com.amazonaws.services.textract.AmazonTextract;
import com.amazonaws.services.textract.AmazonTextractClientBuilder;
import com.amazonaws.services.textract.model.Block;
import com.amazonaws.services.textract.model.DocumentLocation;
import com.amazonaws.services.textract.model.DocumentMetadata;
import com.amazonaws.services.textract.model.GetDocumentAnalysisRequest;
import com.amazonaws.services.textract.model.GetDocumentAnalysisResult;
import com.amazonaws.services.textract.model.GetDocumentTextDetectionRequest;
import com.amazonaws.services.textract.model.GetDocumentTextDetectionResult;
import com.amazonaws.services.textract.model.NotificationChannel;
import com.amazonaws.services.textract.model.Relationship;
import com.amazonaws.services.textract.model.S3Object;
import com.amazonaws.services.textract.model.StartDocumentAnalysisRequest;
import com.amazonaws.services.textract.model.StartDocumentAnalysisResult;
import com.amazonaws.services.textract.model.StartDocumentTextDetectionRequest;
import com.amazonaws.services.textract.model.StartDocumentTextDetectionResult;
import com.fasterxml.jackson.databind.JsonNode;
import com.fasterxml.jackson.databind.ObjectMapper;;
public class DocumentProcessor {

    private static String sqsQueueName=null;
    private static String snsTopicName=null;
    private static String snsTopicArn = null;
    private static String roleArn= null;
    private static String sqsQueueUrl = null;
    private static String sqsQueueArn = null;
```

```
    private static String startJobId = null;
    private static String bucket = null;
    private static String document = null;
    private static AmazonSQS sqs=null;
    private static AmazonSNS sns=null;
    private static AmazonTextract textract = null;

    public enum ProcessType {
        DETECTION,ANALYSIS
    }

    public static void main(String[] args) throws Exception {

        String document = "document";
        String bucket = "bucket";
        String roleArn="role";

        sns = AmazonSNSClientBuilder.defaultClient();
        sqs= AmazonSQSClientBuilder.defaultClient();
        textract=AmazonTextractClientBuilder.defaultClient();

        CreateTopicandQueue();
        ProcessDocument(bucket,document,roleArn,ProcessType.DETECTION);
        DeleteTopicandQueue();
        System.out.println("Done!");


    }
    // Creates an SNS topic and SQS queue. The queue is subscribed to the
topic.
    static void CreateTopicandQueue()
    {
        //create a new SNS topic
        snsTopicName="AmazonTextractTopic" +
Long.toString(System.currentTimeMillis());
        CreateTopicRequest createTopicRequest = new
CreateTopicRequest(snsTopicName);
        CreateTopicResult createTopicResult =
sns.createTopic(createTopicRequest);
        snsTopicArn=createTopicResult.getTopicArn();

        //Create a new SQS Queue
        sqsQueueName="AmazonTextractQueue" +
Long.toString(System.currentTimeMillis());
```

```
        final CreateQueueRequest createQueueRequest = new
CreateQueueRequest(sqsQueueName);
        sqsQueueUrl = sqs.createQueue(createQueueRequest).getQueueUrl();
        sqsQueueArn = sqs.getQueueAttributes(sqsQueueUrl,
Arrays.asList("QueueArn")).getAttributes().get("QueueArn");

        //Subscribe SQS queue to SNS topic
        String sqsSubscriptionArn = sns.subscribe(snsTopicArn, "sqs",
sqsQueueArn).getSubscriptionArn();

        // Authorize queue
          Policy policy = new Policy().withStatements(
                  new Statement(Effect.Allow)
                  .withPrincipals(Principal.AllUsers)
                  .withActions(SQSActions.SendMessage)
                  .withResources(new Resource(sqsQueueArn))
                  .withConditions(new
Condition().withType("ArnEquals").withConditionKey("aws:SourceArn").withValues(snsTopic
                  );


          Map queueAttributes = new HashMap();
          queueAttributes.put(QueueAttributeName.Policy.toString(),
policy.toJson());
          sqs.setQueueAttributes(new SetQueueAttributesRequest(sqsQueueUrl,
queueAttributes));


         System.out.println("Topic arn: " + snsTopicArn);
         System.out.println("Queue arn: " + sqsQueueArn);
         System.out.println("Queue url: " + sqsQueueUrl);
         System.out.println("Queue sub arn: " + sqsSubscriptionArn );
     }
    static void DeleteTopicandQueue()
    {
        if (sqs !=null) {
            sqs.deleteQueue(sqsQueueUrl);
            System.out.println("SQS queue deleted");
        }

        if (sns!=null) {
            sns.deleteTopic(snsTopicArn);
            System.out.println("SNS topic deleted");
        }
```

```
   }

   //Starts the processing of the input document.
   static void ProcessDocument(String inBucket, String inDocument, String
inRoleArn, ProcessType type) throws Exception
   {
       bucket=inBucket;
       document=inDocument;
       roleArn=inRoleArn;

       switch(type)
       {
           case DETECTION:
               StartDocumentTextDetection(bucket, document);
               System.out.println("Processing type: Detection");
               break;
           case ANALYSIS:
               StartDocumentAnalysis(bucket,document);
               System.out.println("Processing type: Analysis");
               break;
           default:
               System.out.println("Invalid processing type. Choose Detection or
Analysis");
               throw new Exception("Invalid processing type");

       }

       System.out.println("Waiting for job: " + startJobId);
       //Poll queue for messages
       List<Message> messages=null;
       int dotLine=0;
       boolean jobFound=false;

       //loop until the job status is published. Ignore other messages in
queue.
       do{
           messages = sqs.receiveMessage(sqsQueueUrl).getMessages();
           if (dotLine++<40){
               System.out.print(".");
           }else{
               System.out.println();
               dotLine=0;
           }
```

```
            if (!messages.isEmpty()) {
                //Loop through messages received.
                for (Message message: messages) {
                    String notification = message.getBody();

                    // Get status and job id from notification.
                    ObjectMapper mapper = new ObjectMapper();
                    JsonNode jsonMessageTree = mapper.readTree(notification);
                    JsonNode messageBodyText = jsonMessageTree.get("Message");
                    ObjectMapper operationResultMapper = new ObjectMapper();
                    JsonNode jsonResultTree =
operationResultMapper.readTree(messageBodyText.textValue());
                    JsonNode operationJobId = jsonResultTree.get("JobId");
                    JsonNode operationStatus = jsonResultTree.get("Status");
                    System.out.println("Job found was " + operationJobId);
                    // Found job. Get the results and display.
                    if(operationJobId.asText().equals(startJobId)){
                        jobFound=true;
                        System.out.println("Job id: " + operationJobId );
                        System.out.println("Status : " +
operationStatus.toString());
                        if (operationStatus.asText().equals("SUCCEEDED")){
                            switch(type)
                            {
                                case DETECTION:
                                    GetDocumentTextDetectionResults();
                                    break;
                                case ANALYSIS:
                                    GetDocumentAnalysisResults();
                                    break;
                                default:
                                    System.out.println("Invalid processing type.
Choose Detection or Analysis");
                                    throw new Exception("Invalid processing
type");

                            }
                        }
                        else{
                            System.out.println("Document analysis failed");
                        }


sqs.deleteMessage(sqsQueueUrl,message.getReceiptHandle());
```

```
                    }

                    else{
                        System.out.println("Job received was not job " +
startJobId);
                        //Delete unknown message. Consider moving message to
dead letter queue

sqs.deleteMessage(sqsQueueUrl,message.getReceiptHandle());
                    }
                }
            }
            else {
                Thread.sleep(5000);
            }
        } while (!jobFound);

        System.out.println("Finished processing document");
    }

    private static void StartDocumentTextDetection(String bucket, String
document) throws Exception{

        //Create notification channel
        NotificationChannel channel= new NotificationChannel()
                .withSNSTopicArn(snsTopicArn)
                .withRoleArn(roleArn);

        StartDocumentTextDetectionRequest req = new
StartDocumentTextDetectionRequest()
                .withDocumentLocation(new DocumentLocation()
                    .withS3Object(new S3Object()
                        .withBucket(bucket)
                        .withName(document)))
                .withJobTag("DetectingText")
                .withNotificationChannel(channel);

        StartDocumentTextDetectionResult startDocumentTextDetectionResult =
textract.startDocumentTextDetection(req);
        startJobId=startDocumentTextDetectionResult.getJobId();
    }

 //Gets the results of processing started by StartDocumentTextDetection
    private static void GetDocumentTextDetectionResults() throws Exception{
```

```
        int maxResults=1000;
        String paginationToken=null;
        GetDocumentTextDetectionResult response=null;
        Boolean finished=false;

        while (finished==false)
        {
            GetDocumentTextDetectionRequest documentTextDetectionRequest= new
GetDocumentTextDetectionRequest()
                    .withJobId(startJobId)
                    .withMaxResults(maxResults)
                    .withNextToken(paginationToken);
            response =
textract.getDocumentTextDetection(documentTextDetectionRequest);
            DocumentMetadata documentMetaData=response.getDocumentMetadata();

            System.out.println("Pages: " +
documentMetaData.getPages().toString());

            //Show blocks information
            List<Block> blocks= response.getBlocks();
            for (Block block : blocks) {
                DisplayBlockInfo(block);
            }
            paginationToken=response.getNextToken();
            if (paginationToken==null)
                finished=true;

        }

    }

    private static void StartDocumentAnalysis(String bucket, String document)
throws Exception{
        //Create notification channel
        NotificationChannel channel= new NotificationChannel()
                .withSNSTopicArn(snsTopicArn)
                .withRoleArn(roleArn);

        StartDocumentAnalysisRequest req = new StartDocumentAnalysisRequest()
                .withFeatureTypes("TABLES","FORMS")
                .withDocumentLocation(new DocumentLocation()
                    .withS3Object(new S3Object()
                        .withBucket(bucket)
```

```
                        .withName(document)))
                .withJobTag("AnalyzingText")
                .withNotificationChannel(channel);

        StartDocumentAnalysisResult startDocumentAnalysisResult =
textract.startDocumentAnalysis(req);
        startJobId=startDocumentAnalysisResult.getJobId();
    }
    //Gets the results of processing started by StartDocumentAnalysis
    private static void GetDocumentAnalysisResults() throws Exception{

        int maxResults=1000;
        String paginationToken=null;
        GetDocumentAnalysisResult response=null;
        Boolean finished=false;

        //loops until pagination token is null
        while (finished==false)
        {
            GetDocumentAnalysisRequest documentAnalysisRequest= new
GetDocumentAnalysisRequest()
                    .withJobId(startJobId)
                    .withMaxResults(maxResults)
                    .withNextToken(paginationToken);

            response = textract.getDocumentAnalysis(documentAnalysisRequest);

            DocumentMetadata documentMetaData=response.getDocumentMetadata();

            System.out.println("Pages: " +
documentMetaData.getPages().toString());

            //Show blocks, confidence and detection times
            List<Block> blocks= response.getBlocks();

            for (Block block : blocks) {
                DisplayBlockInfo(block);
            }
            paginationToken=response.getNextToken();
            if (paginationToken==null)
                finished=true;
        }

    }
```

```
   //Displays Block information for text detection and text analysis
   private static void DisplayBlockInfo(Block block) {
       System.out.println("Block Id : " + block.getId());
       if (block.getText()!=null)
           System.out.println("\tDetected text: " + block.getText());
       System.out.println("\tType: " + block.getBlockType());

       if (block.getBlockType().equals("PAGE") !=true) {
           System.out.println("\tConfidence: " +
block.getConfidence().toString());
       }
       if(block.getBlockType().equals("CELL"))
       {
           System.out.println("\tCell information:");
           System.out.println("\t\tColumn: " + block.getColumnIndex());
           System.out.println("\t\tRow: " + block.getRowIndex());
           System.out.println("\t\tColumn span: " + block.getColumnSpan());
           System.out.println("\t\tRow span: " + block.getRowSpan());

       }

       System.out.println("\tRelationships");
       List<Relationship> relationships=block.getRelationships();
       if(relationships!=null) {
           for (Relationship relationship : relationships) {
               System.out.println("\t\tType: " + relationship.getType());
               System.out.println("\t\tIDs: " +
relationship.getIds().toString());
           }
       } else {
           System.out.println("\t\tNo related Blocks");
       }

       System.out.println("\tGeometry");
       System.out.println("\t\tBounding Box: " +
block.getGeometry().getBoundingBox().toString());
       System.out.println("\t\tPolygon: " +
block.getGeometry().getPolygon().toString());

       List<String> entityTypes = block.getEntityTypes();

       System.out.println("\tEntity Types");
       if(entityTypes!=null) {
           for (String entityType : entityTypes) {
```

```
                System.out.println("\t\tEntity Type: " + entityType);
            }
        } else {
            System.out.println("\t\tNo entity type");
        }

        if(block.getBlockType().equals("SELECTION_ELEMENT")) {
            System.out.print("    Selection element detected: ");
            if (block.getSelectionStatus().equals("SELECTED")){
                System.out.println("Selected");
            }else {
                System.out.println(" Not selected");
            }
        }
        if(block.getPage()!=null)
            System.out.println("\tPage: " + block.getPage());
        System.out.println();
    }
}
```

AWS CLI

DieserAWS CLIstartet die asynchrone Erkennung von Text in einem angegebenen Dokument. Sie gibt zurück.`job-id`das kann verwendet werden, um die Ergebnisse des Nachweises neu zu erstellen.

```
aws textract start-document-text-detection --document-location
"{\"S3Object\":{\"Bucket\":\"bucket-name\",\"Name\":\"file-name\"}}" --
region region-name
```

DieserAWS CLIgibt die Ergebnisse für einen asynchronen Amazon Textract Textract-Vorgang zurück, wenn sie mit einem`job-id`aus.

```
aws textract get-document-text-detection --region region-name --job-id job-id-
number
```

Wenn Sie auf einem Windows-Gerät auf die CLI zugreifen, verwenden Sie doppelte Anführungszeichen anstelle von einfachen Anführungszeichen und entgehen Sie den inneren doppelten Anführungszeichen durch umgekehrten Schrägstrich (d. h.\), um eventuell auftretende Parserfehler zu beheben. Ein Beispiel finden Sie nachfolgend.

```
aws textract start-document-text-detection --document-location "{\"S3Object\":
{\"Bucket\":\"bucket\",\"Name\":\"document\"}}" --region region-name
```

Python

```
import boto3
import json
import sys
import time


class ProcessType:
    DETECTION = 1
    ANALYSIS = 2


class DocumentProcessor:
    jobId = ''
    region_name = ''

    roleArn = ''
    bucket = ''
    document = ''

    sqsQueueUrl = ''
    snsTopicArn = ''
    processType = ''

    def __init__(self, role, bucket, document, region):
        self.roleArn = role
        self.bucket = bucket
        self.document = document
        self.region_name = region

        self.textract = boto3.client('textract', region_name=self.region_name)
        self.sqs = boto3.client('sqs')
        self.sns = boto3.client('sns')

    def ProcessDocument(self, type):
        jobFound = False

        self.processType = type
```

```
        validType = False

        # Determine which type of processing to perform
        if self.processType == ProcessType.DETECTION:
            response = self.textract.start_document_text_detection(
                DocumentLocation={'S3Object': {'Bucket': self.bucket, 'Name':
self.document}},
                NotificationChannel={'RoleArn': self.roleArn, 'SNSTopicArn':
self.snsTopicArn})
            print('Processing type: Detection')
            validType = True

        if self.processType == ProcessType.ANALYSIS:
            response = self.textract.start_document_analysis(
                DocumentLocation={'S3Object': {'Bucket': self.bucket, 'Name':
self.document}},
                FeatureTypes=["TABLES", "FORMS"],
                NotificationChannel={'RoleArn': self.roleArn, 'SNSTopicArn':
self.snsTopicArn})
            print('Processing type: Analysis')
            validType = True

        if validType == False:
            print("Invalid processing type. Choose Detection or Analysis.")
            return

        print('Start Job Id: ' + response['JobId'])
        dotLine = 0
        while jobFound == False:
            sqsResponse = self.sqs.receive_message(QueueUrl=self.sqsQueueUrl,
MessageAttributeNames=['ALL'],
                                                   MaxNumberOfMessages=10)

            if sqsResponse:

                if 'Messages' not in sqsResponse:
                    if dotLine < 40:
                        print('.', end='')
                        dotLine = dotLine + 1
                    else:
                        print()
                        dotLine = 0
                    sys.stdout.flush()
                    time.sleep(5)
```

```
                    continue

                for message in sqsResponse['Messages']:
                    notification = json.loads(message['Body'])
                    textMessage = json.loads(notification['Message'])
                    print(textMessage['JobId'])
                    print(textMessage['Status'])
                    if str(textMessage['JobId']) == response['JobId']:
                        print('Matching Job Found:' + textMessage['JobId'])
                        jobFound = True
                        self.GetResults(textMessage['JobId'])
                        self.sqs.delete_message(QueueUrl=self.sqsQueueUrl,

 ReceiptHandle=message['ReceiptHandle'])
                    else:
                        print("Job didn't match:" +
                              str(textMessage['JobId']) + ' : ' +
 str(response['JobId']))
                    # Delete the unknown message. Consider sending to dead
 letter queue
                    self.sqs.delete_message(QueueUrl=self.sqsQueueUrl,

 ReceiptHandle=message['ReceiptHandle'])

        print('Done!')

    def CreateTopicandQueue(self):

        millis = str(int(round(time.time() * 1000)))

        # Create SNS topic
        snsTopicName = "AmazonTextractTopic" + millis

        topicResponse = self.sns.create_topic(Name=snsTopicName)
        self.snsTopicArn = topicResponse['TopicArn']

        # create SQS queue
        sqsQueueName = "AmazonTextractQueue" + millis
        self.sqs.create_queue(QueueName=sqsQueueName)
        self.sqsQueueUrl = self.sqs.get_queue_url(QueueName=sqsQueueName)
['QueueUrl']

        attribs = self.sqs.get_queue_attributes(QueueUrl=self.sqsQueueUrl,
```

```
                                                    AttributeNames=['QueueArn'])
['Attributes']

        sqsQueueArn = attribs['QueueArn']

        # Subscribe SQS queue to SNS topic
        self.sns.subscribe(
            TopicArn=self.snsTopicArn,
            Protocol='sqs',
            Endpoint=sqsQueueArn)

        # Authorize SNS to write SQS queue
        policy = """{{
  "Version":"2012-10-17",
  "Statement":[
    {{
      "Sid":"MyPolicy",
      "Effect":"Allow",
      "Principal" : {{"AWS" : "*"}},
      "Action":"SQS:SendMessage",
      "Resource": "{}",
      "Condition":{{
        "ArnEquals":{{
          "aws:SourceArn": "{}"
        }}
      }}
    }}
  ]
}}""".format(sqsQueueArn, self.snsTopicArn)

        response = self.sqs.set_queue_attributes(
            QueueUrl=self.sqsQueueUrl,
            Attributes={
                'Policy': policy
            })

    def DeleteTopicandQueue(self):
        self.sqs.delete_queue(QueueUrl=self.sqsQueueUrl)
        self.sns.delete_topic(TopicArn=self.snsTopicArn)

    # Display information about a block
    def DisplayBlockInfo(self, block):

        print("Block Id: " + block['Id'])
```

```
        print("Type: " + block['BlockType'])
        if 'EntityTypes' in block:
            print('EntityTypes: {}'.format(block['EntityTypes']))

        if 'Text' in block:
            print("Text: " + block['Text'])

        if block['BlockType'] != 'PAGE':
            print("Confidence: " + "{:.2f}".format(block['Confidence']) + "%")

        print('Page: {}'.format(block['Page']))

        if block['BlockType'] == 'CELL':
            print('Cell Information')
            print('\tColumn: {} '.format(block['ColumnIndex']))
            print('\tRow: {}'.format(block['RowIndex']))
            print('\tColumn span: {} '.format(block['ColumnSpan']))
            print('\tRow span: {}'.format(block['RowSpan']))

            if 'Relationships' in block:
                print('\tRelationships: {}'.format(block['Relationships']))

        print('Geometry')
        print('\tBounding Box: {}'.format(block['Geometry']['BoundingBox']))
        print('\tPolygon: {}'.format(block['Geometry']['Polygon']))

        if block['BlockType'] == 'SELECTION_ELEMENT':
            print('    Selection element detected: ', end='')
            if block['SelectionStatus'] == 'SELECTED':
                print('Selected')
            else:
                print('Not selected')

    def GetResults(self, jobId):
        maxResults = 1000
        paginationToken = None
        finished = False

        while finished == False:

            response = None

            if self.processType == ProcessType.ANALYSIS:
                if paginationToken == None:
```

```
                    response = self.textract.get_document_analysis(JobId=jobId,

MaxResults=maxResults)
                else:
                    response = self.textract.get_document_analysis(JobId=jobId,

MaxResults=maxResults,

NextToken=paginationToken)

            if self.processType == ProcessType.DETECTION:
                if paginationToken == None:
                    response =
self.textract.get_document_text_detection(JobId=jobId,

MaxResults=maxResults)
                else:
                    response =
self.textract.get_document_text_detection(JobId=jobId,

MaxResults=maxResults,

NextToken=paginationToken)

            blocks = response['Blocks']
            print('Detected Document Text')
            print('Pages: {}'.format(response['DocumentMetadata']['Pages']))

            # Display block information
            for block in blocks:
                self.DisplayBlockInfo(block)
                print()
                print()

            if 'NextToken' in response:
                paginationToken = response['NextToken']
            else:
                finished = True

    def GetResultsDocumentAnalysis(self, jobId):
        maxResults = 1000
        paginationToken = None
        finished = False
```

```
        while finished == False:

            response = None
            if paginationToken == None:
                response = self.textract.get_document_analysis(JobId=jobId,

 MaxResults=maxResults)
            else:
                response = self.textract.get_document_analysis(JobId=jobId,

 MaxResults=maxResults,

 NextToken=paginationToken)

                # Get the text blocks
            blocks = response['Blocks']
            print('Analyzed Document Text')
            print('Pages: {}'.format(response['DocumentMetadata']['Pages']))
            # Display block information
            for block in blocks:
                self.DisplayBlockInfo(block)
                print()
                print()

                if 'NextToken' in response:
                    paginationToken = response['NextToken']
                else:
                    finished = True


def main():
    roleArn = ''
    bucket = ''
    document = ''
    region_name = ''

    analyzer = DocumentProcessor(roleArn, bucket, document, region_name)
    analyzer.CreateTopicandQueue()
    analyzer.ProcessDocument(ProcessType.DETECTION)
    analyzer.DeleteTopicandQueue()


if __name__ == "__main__":
```

```
    main()
```

Node.JS

Ersetzen Sie in diesem Beispiel den Wert von`roleArn`In dem IAM-Rollen-ARN, in dem Sie gespeichert habenAmazon Textract Zugriff auf Ihr Amazon SNS SNS-Thema gewährenaus. Ersetzen Sie die Werte von`bucket`und`document`Mit dem Bucket-Namen und dem Namen der Dokumentdatei, die Sie in Schritt 2 oben angegeben haben. Ersetzen Sie den Wert von`processType`mit der Art der Verarbeitung, die Sie für das Eingabedokument verwenden möchten. Ersetzen Sie abschließend den Wert von`REGION`mit der Region, in der Ihr Kunde tätig ist.

```
 // snippet-start:[sqs.JavaScript.queues.createQueueV3]
// Import required AWS SDK clients and commands for Node.js
import { CreateQueueCommand, GetQueueAttributesCommand, GetQueueUrlCommand,
    SetQueueAttributesCommand, DeleteQueueCommand, ReceiveMessageCommand,
 DeleteMessageCommand } from  "@aws-sdk/client-sqs";
  import {CreateTopicCommand, SubscribeCommand, DeleteTopicCommand } from "@aws-
sdk/client-sns";
  import  { SQSClient } from "@aws-sdk/client-sqs";
  import  { SNSClient } from "@aws-sdk/client-sns";
  import  { TextractClient, StartDocumentTextDetectionCommand,
 StartDocumentAnalysisCommand, GetDocumentAnalysisCommand,
 GetDocumentTextDetectionCommand, DocumentMetadata } from "@aws-sdk/client-
textract";
  import { stdout } from "process";

  // Set the AWS Region.
  const REGION = "us-east-1"; //e.g. "us-east-1"
  // Create SNS service object.
  const sqsClient = new SQSClient({ region: REGION });
  const snsClient = new SNSClient({ region: REGION });
  const textractClient = new TextractClient({ region: REGION });

  // Set bucket and video variables
  const bucket = "bucket-name";

  const documentName = "document-name";
  const roleArn = "role-arn"
  const processType = "DETECTION"
  var startJobId = ""
```

```
 var ts = Date.now();
 const snsTopicName = "AmazonTextractExample" + ts;
 const snsTopicParams = {Name: snsTopicName}
 const sqsQueueName = "AmazonTextractQueue-" + ts;

 // Set the parameters
 const sqsParams = {
   QueueName: sqsQueueName, //SQS_QUEUE_URL
   Attributes: {
     DelaySeconds: "60", // Number of seconds delay.
     MessageRetentionPeriod: "86400", // Number of seconds delay.
   },
 };

 // Process a document based on operation type
 const processDocumment = async (type, bucket, videoName, roleArn, sqsQueueUrl,
snsTopicArn) =>
   {
   try
   {
       // Set job found and success status to false initially
     var jobFound = false
     var succeeded = false
     var dotLine = 0
     var processType = type
     var validType = false

     if (processType == "DETECTION"){
       var response = await textractClient.send(new
StartDocumentTextDetectionCommand({DocumentLocation:{S3Object:{Bucket:bucket,
Name:videoName}},
         NotificationChannel:{RoleArn: roleArn, SNSTopicArn: snsTopicArn}}))
       console.log("Processing type: Detection")
       validType = true
     }

     if (processType == "ANALYSIS"){
       var response = await textractClient.send(new
StartDocumentAnalysisCommand({DocumentLocation:{S3Object:{Bucket:bucket,
Name:videoName}},
         NotificationChannel:{RoleArn: roleArn, SNSTopicArn: snsTopicArn}}))
       console.log("Processing type: Analysis")
       validType = true
     }
```

```
     if (validType == false){
         console.log("Invalid processing type. Choose Detection or Analysis.")
         return
     }
   // while not found, continue to poll for response
   console.log(`Start Job ID: ${response.JobId}`)
   while (jobFound == false){
     var sqsReceivedResponse = await sqsClient.send(new
ReceiveMessageCommand({QueueUrl:sqsQueueUrl,
       MaxNumberOfMessages:'ALL', MaxNumberOfMessages:10}));
     if (sqsReceivedResponse){
       var responseString = JSON.stringify(sqsReceivedResponse)
       if (!responseString.includes('Body')){
         if (dotLine < 40) {
           console.log('.')
           dotLine = dotLine + 1
         }else {
           console.log('')
           dotLine = 0
         };
         stdout.write('', () => {
           console.log('');
         });
         await new Promise(resolve => setTimeout(resolve, 5000));
         continue
       }
     }

       // Once job found, log Job ID and return true if status is succeeded
       for (var message of sqsReceivedResponse.Messages){
           console.log("Retrieved messages:")
           var notification = JSON.parse(message.Body)
           var rekMessage = JSON.parse(notification.Message)
           var messageJobId = rekMessage.JobId
           if (String(rekMessage.JobId).includes(String(startJobId))){
               console.log('Matching job found:')
               console.log(rekMessage.JobId)
               jobFound = true
               // GET RESULTS FUNCTION HERE
               var operationResults = await GetResults(processType,
rekMessage.JobId)
               //GET RESULTS FUMCTION HERE
               console.log(rekMessage.Status)
```

```
            if (String(rekMessage.Status).includes(String("SUCCEEDED"))){
                succeeded = true
                console.log("Job processing succeeded.")
                var sqsDeleteMessage = await sqsClient.send(new
DeleteMessageCommand({QueueUrl:sqsQueueUrl,
ReceiptHandle:message.ReceiptHandle}));
            }
            }else{
            console.log("Provided Job ID did not match returned ID.")
            var sqsDeleteMessage = await sqsClient.send(new
DeleteMessageCommand({QueueUrl:sqsQueueUrl,
ReceiptHandle:message.ReceiptHandle}));
            }
       }

   console.log("Done!")
   }
   }catch (err) {
       console.log("Error", err);
     }
 }

 // Create the SNS topic and SQS Queue
 const createTopicandQueue = async () => {
   try {
     // Create SNS topic
     const topicResponse = await snsClient.send(new
CreateTopicCommand(snsTopicParams));
     const topicArn = topicResponse.TopicArn
     console.log("Success", topicResponse);
     // Create SQS Queue
     const sqsResponse = await sqsClient.send(new
CreateQueueCommand(sqsParams));
     console.log("Success", sqsResponse);
     const sqsQueueCommand = await sqsClient.send(new
GetQueueUrlCommand({QueueName: sqsQueueName}))
     const sqsQueueUrl = sqsQueueCommand.QueueUrl
     const attribsResponse = await sqsClient.send(new
GetQueueAttributesCommand({QueueUrl: sqsQueueUrl, AttributeNames:
['QueueArn']}))
     const attribs = attribsResponse.Attributes
     console.log(attribs)
     const queueArn = attribs.QueueArn
     // subscribe SQS queue to SNS topic
```

```
     const subscribed = await snsClient.send(new SubscribeCommand({TopicArn:
topicArn, Protocol:'sqs', Endpoint: queueArn}))
     const policy = {
       Version: "2012-10-17",
       Statement: [
         {
           Sid: "MyPolicy",
           Effect: "Allow",
           Principal: {AWS: "*"},
           Action: "SQS:SendMessage",
           Resource: queueArn,
           Condition: {
             ArnEquals: {
               'aws:SourceArn': topicArn
             }
           }
         }
       ]
     };

     const response = sqsClient.send(new SetQueueAttributesCommand({QueueUrl:
sqsQueueUrl, Attributes: {Policy: JSON.stringify(policy)}}))
     console.log(response)
     console.log(sqsQueueUrl, topicArn)
     return [sqsQueueUrl, topicArn]

   } catch (err) {
     console.log("Error", err);

   }
 }

 const deleteTopicAndQueue = async (sqsQueueUrlArg, snsTopicArnArg) => {
   const deleteQueue = await sqsClient.send(new DeleteQueueCommand({QueueUrl:
sqsQueueUrlArg}));
   const deleteTopic = await snsClient.send(new DeleteTopicCommand({TopicArn:
snsTopicArnArg}));
   console.log("Successfully deleted.")
 }

 const displayBlockInfo = async (block) => {
   console.log(`Block ID: ${block.Id}`)
   console.log(`Block Type: ${block.BlockType}`)
   if (String(block).includes(String("EntityTypes"))){
```

```
        console.log(`EntityTypes: ${block.EntityTypes}`)
    }
    if (String(block).includes(String("Text"))){
        console.log(`EntityTypes: ${block.Text}`)
    }
    if (!String(block.BlockType).includes('PAGE')){
        console.log(`Confidence: ${block.Confidence}`)
    }
    console.log(`Page: ${block.Page}`)
    if (String(block.BlockType).includes("CELL")){
        console.log("Cell Information")
        console.log(`Column: ${block.ColumnIndex}`)
        console.log(`Row: ${block.RowIndex}`)
        console.log(`Column Span: ${block.ColumnSpan}`)
        console.log(`Row Span: ${block.RowSpan}`)
        if (String(block).includes("Relationships")){
            console.log(`Relationships: ${block.Relationships}`)
        }
    }

    console.log("Geometry")
    console.log(`Bounding Box: ${JSON.stringify(block.Geometry.BoundingBox)}`)
    console.log(`Polygon: ${JSON.stringify(block.Geometry.Polygon)}`)

    if (String(block.BlockType).includes('SELECTION_ELEMENT')){
      console.log('Selection Element detected:')
      if (String(block.SelectionStatus).includes('SELECTED')){
        console.log('Selected')
      } else {
        console.log('Not Selected')
      }

    }
  }

  const GetResults = async (processType, JobID) => {

    var maxResults = 1000
    var paginationToken = null
    var finished = false

    while (finished == false){
      var response = null
      if (processType == 'ANALYSIS'){
```

```
       if (paginationToken == null){
         response = textractClient.send(new
GetDocumentAnalysisCommand({JobId:JobID, MaxResults:maxResults}))

       }else{
         response = textractClient.send(new
GetDocumentAnalysisCommand({JobId:JobID, MaxResults:maxResults,
NextToken:paginationToken}))
       }
     }

     if(processType == 'DETECTION'){
       if (paginationToken == null){
         response = textractClient.send(new
GetDocumentTextDetectionCommand({JobId:JobID, MaxResults:maxResults}))

       }else{
         response = textractClient.send(new
GetDocumentTextDetectionCommand({JobId:JobID, MaxResults:maxResults,
NextToken:paginationToken}))
       }
     }

     await new Promise(resolve => setTimeout(resolve, 5000));
     console.log("Detected Documented Text")
     console.log(response)
     //console.log(Object.keys(response))
     console.log(typeof(response))
     var blocks = (await response).Blocks
     console.log(blocks)
     console.log(typeof(blocks))
     var docMetadata = (await response).DocumentMetadata
     var blockString = JSON.stringify(blocks)
     var parsed = JSON.parse(JSON.stringify(blocks))
     console.log(Object.keys(blocks))
     console.log(`Pages: ${docMetadata.Pages}`)
     blocks.forEach((block)=> {
       displayBlockInfo(block)
       console.log()
       console.log()
     })

     //console.log(blocks[0].BlockType)
     //console.log(blocks[1].BlockType)
```

```
      if(String(response).includes("NextToken")){
        paginationToken = response.NextToken
      }else{
        finished = true
      }
    }

  }


  // DELETE TOPIC AND QUEUE
  const main = async () => {
    var sqsAndTopic = await createTopicandQueue();
    var process = await processDocumment(processType, bucket, documentName,
 roleArn, sqsAndTopic[0], sqsAndTopic[1])
    var deleteResults = await deleteTopicAndQueue(sqsAndTopic[0],
 sqsAndTopic[1])
  }

main()
```

4. Führen Sie den Code aus. Die Operation kann einige Zeit in Anspruch nehmen. Nach Abschluss des Vorgangs wird eine Liste der Blöcke für erkannten oder analysierten Text angezeigt.

Amazon Textract Ergebnisbenachrichtigung

Amazon Textract veröffentlicht die Ergebnisse einer Amazon Textract Textract-Analyseanfrage, einschließlich des Status der Erledigung, an ein Amazon Simple Notification Service (Amazon SNS) -Thema. Um die Benachrichtigung von einem Amazon-SNS-Thema zu erhalten, verwenden Sie eine Amazon SQS SQS-Warteschlange oder eineAWS LambdaFunktion. Weitere Informationen finden Sie unter Asynchrone Operationen von Amazon Textract aufrufen . Ein Beispiel finden Sie unter Erkennen oder Analysieren von Text in einem mehrseitigen Dokument.

Die Ergebnisse weisen das folgende JSON-Format auf:

```
{
  "JobId": "String",
  "Status": "String",
  "API": "String",
  "JobTag": "String",
```

```
  "Timestamp": Number,
  "DocumentLocation": {
    "S3ObjectName": "String",
    "S3Bucket": "String"
  }
}
```

In dieser Tabelle werden die verschiedenen Parameter innerhalb einer Amazon Textract Textract-Antwort beschrieben.

Parameter	Beschreibung
JobId	Die eindeutige Kennung, die Amazon Textract dem Job zuweist. Sie stimmt mit einer Auftragskennung überein, die von einem`StartBetrieb`, wie[StartDocumentTextDetection](#)aus.
Status	Der Status des Auftrags. Gültige Werte sind Succeed, Failed oder Error.
API	Die Amazon Textract Textract-Operation , die zur Analyse des Eingabedokuments verwendet wird, wie z. B.[StartDocumentTextDetection](#)oder[StartDocumentAnalysis](#)aus.
JobTag	Die vom Benutzer angegebene Kennung für den Auftrag. Sie geben an`JobTag`in einem Anruf bei der`StartBetrieb`, wie[StartDocumentTextDetection](#)aus.
Zeitstempel	Der Unix-Zeitstempel, der angibt, wann der Job abgeschlossen wurde, wird in Millisekunden zurückgegeben.
documentLocation	Details zu dem Dokument, das verarbeitet wurde. Umfasst den Dateinamen und den Amazon S3 S3-Bucket, in dem die Datei gespeichert ist.

Umgang mit gedrosselten Anrufen und gelöschten

Ein Amazon Textract Textract-Vorgang kann fehlschlagen, wenn Sie die maximale Anzahl von Transaktionen pro Sekunde (TPS) überschreiten, wodurch der Service Ihre Anwendung drosselt oder wenn Ihre Verbindung unterbrochen wird. Wenn Sie beispielsweise in kurzer Zeit zu viele Anrufe Amazon Textract Textract-Operationen tätigen, drosselt es Ihre Anrufe und sendet ein`ProvisionedThroughputExceededException`Fehler in der Betriebsantwort. Hinweise zu Amazon Textract TPS-Kontingenten finden Sie unterAmazon Textract Kontingenteaus.

Sie können Drosselung und abgebrochene Verbindungen verwalten, indem Sie den Vorgang automatisch wiederholen. Sie können die Anzahl der Wiederholungen angeben, indem Sie die`Config`-Parameter, wenn Sie den Amazon Textract Textract-Client erstellen. Wir empfehlen eine Wiederholungsanzahl von 5. DieAWSDas SDK versucht eine Operation die angegebene Anzahl von Malen erneut, bevor es fehlschlägt und eine Ausnahme ausgelöst wird. Weitere Informationen finden Sie unter Wiederholversuche bei Fehlern und exponentielles Backoff in AWS.

> **Note**
>
> Automatische Wiederholungen funktionieren sowohl für synchrone als auch für asynchrone Operationen. Stellen Sie vor dem Angeben von automatischen Wiederholungen sicher, dass Sie über die neueste Version des AWS SDKs verfügen. Weitere Informationen finden Sie unter Schritt 2: Einrichten derAWS CLIundAWS-SDKs .

Das folgende Beispiel zeigt, wie Sie Amazon Textract Textract-Vorgänge automatisch erneut versuchen, wenn Sie mehrere Dokumente verarbeiten.

Voraussetzungen

- Wenn Sie dies noch nicht getan haben:
  a. Erstellen oder aktualisieren Sie einen IAM-Benutzer mit`AmazonTextractFullAccess`und`AmazonS3ReadOnlyAccess`-Berechtigungen Weitere Informationen finden Sie unter Schritt 1: Einrichten eines AWS-Kontos und Erstellen eines IAM-Benutzers .
  b. Installieren und konfigurieren Sie die AWS CLI- und AWS-SDKs. Weitere Informationen finden Sie unter Schritt 2: Einrichten derAWS CLIundAWS-SDKs .

So wiederholen Sie Operationen automatisch

1. Laden Sie mehrere Dokumentbilder in Ihren S3-Bucket hoch, um das Synchrone Beispiel auszuführen. Laden Sie ein mehrseitiges Dokument in Ihren S3-Bucket hoch und führen Sie es`StartDocumentTextDetection`darauf, um das Asynchrone Beispiel auszuführen.

   Detaillierte Anweisungen finden Sie unterHochladen von Objekten in Amazon S3imAmazon Simple Storage Service Benutzerhandbuchaus.

2. Die folgenden Beispiele veranschaulichen, wie Sie die`Config`-Parameter, um eine Operation automatisch erneut zu versuchen. Das synchrone Beispiel ruft den`DetectDocumentText`-Operation, während das Asynchrone Beispiel die`GetDocumentTextDetection`verwenden.

   Sync Example

   Verwenden Sie die folgenden Beispiele, um die`DetectDocumentText`-Betrieb auf den Dokumenten in Ihrem Amazon S3 S3-Bucket. In :`main`So ändern Sie den -Wert`bucket`auf Ihrem S3-Bucket hoch (es sollte ein oder mehrere Objekte wie beispielsweise Bäume, Häuser, Boote, etc. enthalten). Ändern Sie den -Wert`documents`auf die Namen der Dokumentbilder, die Sie in Schritt 2 hochgeladen haben.

```
import boto3
from botocore.client import Config
# Documents

def process_multiple_documents(bucket, documents):

    config = Config(retries = dict(max_attempts = 5))

    # Amazon Textract client
    textract = boto3.client('textract', config=config)

    for documentName in documents:

        print("\nProcessing:
 {}\n=========================================".format(documentName))

        # Call Amazon Textract
        response = textract.detect_document_text(
            Document={
                'S3Object': {
                    'Bucket': bucket,
```

```
                    'Name': documentName
                }
            })

        # Print detected text
        for item in response["Blocks"]:
            if item["BlockType"] == "LINE":
                print ('\033[94m' +  item["Text"] + '\033[0m')


def main():
    bucket = ""
    documents = ["document-image-1.png",
    "document-image-2.png", "document-image-3.png",
    "document-image-4.png", "document-image-5.png" ]
    process_multiple_documents(bucket, documents)



if __name__ == "__main__":
    main()
```

Async Example

Verwenden Sie die folgenden Beispiele zum Aufrufen der `GetDocumentTextDetection`-Operation. Es wird davon ausgegangen, dass Sie bereits angerufen habe`StartDocumentTextDetection`auf den Dokumenten in Ihrem Amazon S3 S3-Bucket erhalten und eine`JobId`aus. In :`main`So ändern Sie den -Wert`bucket`auf Ihrem S3-Bucket und dem -Wert`roleArn`zu dem Arn, der Ihrer Textrakt-Rolle zugewiesen wurde. Sie müssen auch den -Wert änder`document`auf den Namen Ihres mehrseitigen Dokuments in Ihrem Amazon S3 S3-Bucket hoch. Ersetzen Sie schließlich den -Wert`region_name`Geben Sie den Namen Ihrer Region an und geben Sie die`GetResults`-Funktion mit dem Namen Ihres`jobId`aus.

```
import boto3
from botocore.client import Config

class DocumentProcessor:
    jobId = ''
    region_name = ''

    roleArn = ''
```

```
    bucket = ''
    document = ''

    sqsQueueUrl = ''
    snsTopicArn = ''
    processType = ''

    def __init__(self, role, bucket, document, region):
        self.roleArn = role
        self.bucket = bucket
        self.document = document
        self.region_name = region
        self.config = Config(retries = dict(max_attempts = 5))

        self.textract = boto3.client('textract', region_name=self.region_name,
 config=self.config)
        self.sqs = boto3.client('sqs')
        self.sns = boto3.client('sns')

# Display information about a block
    def DisplayBlockInfo(self, block):

        print("Block Id: " + block['Id'])
        print("Type: " + block['BlockType'])
        if 'EntityTypes' in block:
            print('EntityTypes: {}'.format(block['EntityTypes']))

        if 'Text' in block:
            print("Text: " + block['Text'])

        if block['BlockType'] != 'PAGE':
            print("Confidence: " + "{:.2f}".format(block['Confidence']) + "%")

        print('Page: {}'.format(block['Page']))

        if block['BlockType'] == 'CELL':
            print('Cell Information')
            print('\tColumn: {} '.format(block['ColumnIndex']))
            print('\tRow: {}'.format(block['RowIndex']))
            print('\tColumn span: {} '.format(block['ColumnSpan']))
            print('\tRow span: {}'.format(block['RowSpan']))

            if 'Relationships' in block:
                print('\tRelationships: {}'.format(block['Relationships']))
```

```
        print('Geometry')
        print('\tBounding Box: {}'.format(block['Geometry']['BoundingBox']))
        print('\tPolygon: {}'.format(block['Geometry']['Polygon']))

        if block['BlockType'] == 'SELECTION_ELEMENT':
            print('    Selection element detected: ', end='')
            if block['SelectionStatus'] == 'SELECTED':
                print('Selected')
            else:
                print('Not selected')

    def GetResults(self, jobId):
        maxResults = 1000
        paginationToken = None
        finished = False

        while finished == False:

            response = None

            if paginationToken == None:
                response =
self.textract.get_document_text_detection(JobId=jobId,

MaxResults=maxResults)
            else:
                response =
self.textract.get_document_text_detection(JobId=jobId,

MaxResults=maxResults,

NextToken=paginationToken)

            blocks = response['Blocks']
            print('Detected Document Text')
            print('Pages: {}'.format(response['DocumentMetadata']['Pages']))

            # Display block information
            for block in blocks:
                self.DisplayBlockInfo(block)
                print()
                print()
```

```
            if 'NextToken' in response:
                paginationToken = response['NextToken']
            else:
                finished = True

def main():
    roleArn = 'role-arn'
    bucket = 'bucket-name'
    document = 'document-name'
    region_name = 'region-name'
    analyzer = DocumentProcessor(roleArn, bucket, document, region_name)
    analyzer.GetResults("job-id")

if __name__ == "__main__":
    main()
```

Bewährte Methoden für Amazon Textract

Amazon Textract verwendet maschinelles Lernen, um Dokumente wie eine Person zu lesen. Es extrahiert Text, Tabellen und Formulare aus Dokumenten. Verwenden Sie die folgenden bewährten Methoden, um optimale Ergebnisse aus Ihren Dokumenten zu erstellen.

Bereitstellen eines optimalen Eingabedokuments

Im Folgenden finden Sie eine Liste mit einigen Möglichkeiten, wie Sie Ihre Eingabedokumente für bessere Ergebnisse optimieren können.

- Stellen Sie sicher, dass Ihr Dokumenttext in einer Sprache vorliegt, die Amazon Textract unterstützt. Derzeit unterstützt Amazon Textract Englisch, Spanisch, Deutsch, Italienisch, Französisch und Portugiesisch.
- Stellen Sie ein Bild in hoher Qualität bereit, idealerweise mindestens 150 DPI.
- Wenn sich Ihr Dokument bereits in einem der Dateiformate befindet, die Amazon Textract unterstützt (PDF, TIFF, JPEG und PNG), konvertieren oder stauen Sie das Dokument nicht herunter, bevor Sie es auf Amazon Textract hochladen.

Um die besten Ergebnisse beim Extrahieren von Text aus Tabellen in Dokumenten zu erzielen, stellen Sie sicher, dass:

- Tabellen in Ihrem Dokument sind visuell von den umgebenden Elementen auf der Seite getrennt. Zum Beispiel wird die Tabelle nicht auf ein Bild oder ein komplexes Muster überlagert.
- Der Text innerhalb der Tabelle ist aufrecht. Zum Beispiel wird der Text nicht relativ zu anderem Text auf der Seite gedreht.

Wenn Sie Text aus Tabellen extrahieren, werden möglicherweise inkonsistente Ergebnisse angezeigt, wenn:

- Zusammengeführte Tabellenzellen, die sich über mehrere Spalten erstrecken.
- Tabellen mit Zellen, Zeilen oder Spalten, die sich von anderen Teilen derselben Tabelle unterscheiden.

Wir empfehlen die Verwendung vonTexterkennungals Behelfslösung.

Verwenden von Zuverlässigkeitswert

Sie sollten die Konfidenzwerte berücksichtigen, die von Amazon Textract Textract-API-Vorgängen zurückgegeben werden, und die Sensibilität ihres Anwendungsfalls. Der Zuverlässigkeitswert ist eine Zahl zwischen 0 und 100, mit der die Wahrscheinlichkeit angegeben wird, dass eine gegebene Vorhersage korrekt ist. Es hilft Ihnen, fundierte Entscheidungen darüber zu treffen, wie Sie die Ergebnisse verwenden.

Erzwingen Sie in Anwendungen, die empfindlich auf Erkennungsfehler reagieren (falsch positiv), einen Schwellenwert für den Mindestkonfidenzwert. Die Anwendung sollte Ergebnisse unterhalb dieses Schwellenwerts verwerfen oder Situationen kennzeichnen, die ein höheres Maß an menschlicher Kontrolle erfordern.

Der optimale Schwellenwert hängt von der Anwendung ab. Für Archivierungszwecke, z. B. das Dokumentieren handschriftlicher Notizen, kann es bis zu 50% betragen. Geschäftsprozesse mit finanziellen Entscheidungen können Schwellenwerte von 90% oder höher erfordern.

Erwägen Sie die Verwendung von

Erwägen Sie auch, menschliche Überprüfung in Ihre Workflows einzubeziehen. Dies ist besonders wichtig für sensible Anwendungen wie Geschäftsprozesse, die finanzielle Entscheidungen beinhalten.

Tutorials

the section called “Block”-Objekte, die von Amazon Textract Textract-Operationen zurückgegeben werden, enthalten die Ergebnisse von Texterkennungs- und Textanalysevorgängen, wiethe section called “AnalyzeDocument”aus. Die folgenden Python-Tutorials zeigen einige der verschiedenen Möglichkeiten, wie Sie Block-Objekte verwenden können. Sie können beispielsweise Tabelleninformationen in eine Komma-getrennte Datei (Comma-getrennte Datei) exportieren.

Die Tutorials verwenden synchrone Amazon Textract Textract-Vorgänge, die alle Ergebnisse zurückgeben. Wenn Sie asynchrone Operationen wiethe section called “StartDocumentAnalysis”müssen Sie den Beispielcode ändern, um mehrere Stapel zurückgegebenen`Block`Objekte. Um das Beispiel für asynchrone Operationen zu verwenden, stellen Sie sicher, dass Sie die Anweisungen unterKonfigurieren von Amazon Textract für asynchrone Vorgängeaus.

Beispiele, die Ihnen andere Möglichkeiten zur Verwendung von Amazon Textract zeigen, finden Sie unterWeitere Codebeispieleaus.

Themen

- Voraussetzungen
- Extrahieren von Schlüssel-Wert-Paaren aus einem Formulardokument
- Exportieren von Tabellen in eine CSV-Datei
- Erstellen einesAWS LambdaFunktion
- Weitere Codebeispiele

Voraussetzungen

Bevor Sie die Beispiele in diesem Abschnitt ausführen, müssen Sie Ihre Umgebung konfigurieren.

So konfigurieren Sie Ihre Umgebung

1. Erstellen oder aktualisieren Sie einen IAM-Benutzer mit`AmazonTextractFullAccess`Berechtigungen Weitere Informationen finden Sie unter Schritt 1: Einrichten eines AWS-Kontos und Erstellen eines IAM-Benutzers .
2. Installieren und konfigurieren Sie die AWS CLI- und AWS-SDKs. Weitere Informationen finden Sie unter Schritt 2: Einrichten derAWS CLIundAWS-SDKs .

Extrahieren von Schlüssel-Wert-Paaren aus einem Formulardokument

Das folgende Python-Beispiel zeigt, wie Schlüssel-Wert-Paare in Formulardokumenten extrahiert werdenthe section called “Block”Objekte, die in einer Map gespeichert sind. Blockobjekte werden von einem Aufruf an zurückgegebenthe section called “AnalyzeDocument”aus. Weitere Informationen finden Sie unter Formulardaten (Schlüssel-Wert-Paare) .

Sie verwenden die folgenden Funktionen:

- `get_kv_map`— RuftAnalyzeDocumentund speichert die KEY- und VALUE-BLOCK-Objekte in einer Map.
- `get_kv_relationshipundfind_value_block`— Konstruiert die Schlüssel-Wert-Beziehungen aus der Map.

So extrahieren Sie Schlüssel-Wert-Paare aus einem Formulardokument

1. Konfigurieren Sie Ihre Umgebung. Weitere Informationen finden Sie unter Voraussetzungen .
2. Speichern Sie den folgenden Beispiel-Code in einer Datei mit dem Namentextract_python_kv_parser.pyaus.

```
import boto3
import sys
import re
import json


def get_kv_map(file_name):

    with open(file_name, 'rb') as file:
        img_test = file.read()
        bytes_test = bytearray(img_test)
        print('Image loaded', file_name)

    # process using image bytes
    client = boto3.client('textract')
    response = client.analyze_document(Document={'Bytes': bytes_test},
 FeatureTypes=['FORMS'])
```

```
    # Get the text blocks
    blocks=response['Blocks']


    # get key and value maps
    key_map = {}
    value_map = {}
    block_map = {}
    for block in blocks:
        block_id = block['Id']
        block_map[block_id] = block
        if block['BlockType'] == "KEY_VALUE_SET":
            if 'KEY' in block['EntityTypes']:
                key_map[block_id] = block
            else:
                value_map[block_id] = block

    return key_map, value_map, block_map


def get_kv_relationship(key_map, value_map, block_map):
    kvs = {}
    for block_id, key_block in key_map.items():
        value_block = find_value_block(key_block, value_map)
        key = get_text(key_block, block_map)
        val = get_text(value_block, block_map)
        kvs[key] = val
    return kvs


def find_value_block(key_block, value_map):
    for relationship in key_block['Relationships']:
        if relationship['Type'] == 'VALUE':
            for value_id in relationship['Ids']:
                value_block = value_map[value_id]
    return value_block


def get_text(result, blocks_map):
    text = ''
    if 'Relationships' in result:
        for relationship in result['Relationships']:
            if relationship['Type'] == 'CHILD':
                for child_id in relationship['Ids']:
```

```
                    word = blocks_map[child_id]
                    if word['BlockType'] == 'WORD':
                        text += word['Text'] + ' '
                    if word['BlockType'] == 'SELECTION_ELEMENT':
                        if word['SelectionStatus'] == 'SELECTED':
                            text += 'X '


    return text


def print_kvs(kvs):
    for key, value in kvs.items():
        print(key, ":", value)


def search_value(kvs, search_key):
    for key, value in kvs.items():
        if re.search(search_key, key, re.IGNORECASE):
            return value

def main(file_name):

    key_map, value_map, block_map = get_kv_map(file_name)

    # Get Key Value relationship
    kvs = get_kv_relationship(key_map, value_map, block_map)
    print("\n\n== FOUND KEY : VALUE pairs ===\n")
    print_kvs(kvs)

    # Start searching a key value
    while input('\n Do you want to search a value for a key? (enter "n" for exit)
 ') != 'n':
        search_key = input('\n Enter a search key:')
        print('The value is:', search_value(kvs, search_key))

if __name__ == "__main__":
    file_name = sys.argv[1]
    main(file_name)
```

3. Geben Sie in der Eingabeaufforderung den folgenden Befehl ein. Ersetzen`file`mit der zu analysierenden Dokumentbilddatei.

```
textract_python_kv_parser.py file
```

4. Wenn Sie dazu aufgefordert werden, geben Sie einen Schlüssel ein, der sich im Eingabedokument befindet. Wenn der Code den Schlüssel erkennt, zeigt er den Wert des Schlüssels an.

Exportieren von Tabellen in eine CSV-Datei

Diese Python-Beispiele zeigen, wie Tabellen aus einem Bild eines Dokuments in eine Komma-getrennte Datei (Comma-getrennte Datei, CSV-Datei) exportiert werden.

Das Beispiel für die synchrone Dokumentenanalyse sammelt Tabelleninformationen aus einem Aufruf anthe section called “AnalyzeDocument”aus. Das Beispiel für die asynchrone Dokumentenanalyse rufthe section called “StartDocumentAnalysis”und ruft dann die Ergebnisse abthe section called “GetDocumentAnalysis”als`Block`Objekte.

Tabelleninformationen werden zurückgegeben alsthe section called “Block”Objekte von einem Anruf anthe section called “AnalyzeDocument”aus. Weitere Informationen finden Sie unter Tabellen . Die`Block`-Objekte werden in einer Kartenstruktur gespeichert, die zum Exportieren der Tabellendaten in eine CSV-Datei verwendet wird.

Synchronous

In diesem Beispiel verwenden Sie die Funktionen:

- `get_table_csv_results`— RuftAnalyzeDocumentund erstellt eine Map von Tabellen, die im Dokument erkannt werden. Erstellt eine CSV-Darstellung aller erkannten Tabellen.
- `generate_table_csv`— Erzeugt die CSV-Datei für eine einzelne Tabelle.
- `get_rows_columns_map`— Ruft die Zeilen und Spalten aus der Map ab.
- `get_text`— Ruft den Text aus einer Zelle ab.

So exportieren Sie Tabellen in eine CSV-Datei

1. Konfigurieren Sie Ihre Umgebung. Weitere Informationen finden Sie unter Voraussetzungen .
2. Speichern Sie den folgenden Beispiel-Code in einer Datei mit dem Namentextract_python_table_parser.pyaus.

```
import webbrowser, os
import json
import boto3
import io
from io import BytesIO
import sys
from pprint import pprint


def get_rows_columns_map(table_result, blocks_map):
    rows = {}
    for relationship in table_result['Relationships']:
        if relationship['Type'] == 'CHILD':
            for child_id in relationship['Ids']:
                cell = blocks_map[child_id]
                if cell['BlockType'] == 'CELL':
                    row_index = cell['RowIndex']
                    col_index = cell['ColumnIndex']
                    if row_index not in rows:
                        # create new row
                        rows[row_index] = {}

                    # get the text value
                    rows[row_index][col_index] = get_text(cell, blocks_map)
    return rows


def get_text(result, blocks_map):
    text = ''
    if 'Relationships' in result:
        for relationship in result['Relationships']:
            if relationship['Type'] == 'CHILD':
                for child_id in relationship['Ids']:
                    word = blocks_map[child_id]
                    if word['BlockType'] == 'WORD':
                        text += word['Text'] + ' '
                    if word['BlockType'] == 'SELECTION_ELEMENT':
                        if word['SelectionStatus'] =='SELECTED':
                            text +=  'X '
    return text


def get_table_csv_results(file_name):
```

```
    with open(file_name, 'rb') as file:
        img_test = file.read()
        bytes_test = bytearray(img_test)
        print('Image loaded', file_name)

    # process using image bytes
    # get the results
    client = boto3.client('textract')

    response = client.analyze_document(Document={'Bytes': bytes_test},
 FeatureTypes=['TABLES'])

    # Get the text blocks
    blocks=response['Blocks']
    pprint(blocks)

    blocks_map = {}
    table_blocks = []
    for block in blocks:
        blocks_map[block['Id']] = block
        if block['BlockType'] == "TABLE":
            table_blocks.append(block)

    if len(table_blocks) <= 0:
        return "<b> NO Table FOUND </b>"

    csv = ''
    for index, table in enumerate(table_blocks):
        csv += generate_table_csv(table, blocks_map, index +1)
        csv += '\n\n'

    return csv

def generate_table_csv(table_result, blocks_map, table_index):
    rows = get_rows_columns_map(table_result, blocks_map)

    table_id = 'Table_' + str(table_index)

    # get cells.
    csv = 'Table: {0}\n\n'.format(table_id)

    for row_index, cols in rows.items():
```

```
            for col_index, text in cols.items():
                csv += '{}'.format(text) + ","
            csv += '\n'

        csv += '\n\n\n'
        return csv

def main(file_name):
    table_csv = get_table_csv_results(file_name)

    output_file = 'output.csv'

    # replace content
    with open(output_file, "wt") as fout:
        fout.write(table_csv)

    # show the results
    print('CSV OUTPUT FILE: ', output_file)


if __name__ == "__main__":
    file_name = sys.argv[1]
    main(file_name)
```

3. Geben Sie in der Eingabeaufforderung den folgenden Befehl ein. Ersetzen`file`mit dem Namen der Dokumentbilddatei, die Sie analysieren möchten.

```
python textract_python_table_parser.py file
```

Wenn Sie das Beispiel ausführen, wird die CSV-Ausgabe in einer Datei mit dem Namen gespeichert`output.csv`aus.

Asynchronous

In diesem Beispiel verwenden Sie zwei verschiedene Skripte. Das erste Skript startet die asynchrone Analyse von Dokumenten mit`StartDocumentAnalysis`und bekommt das`Block`von zurückgegebene Informationen`GetDocumentAnalysis`aus. Das zweite Skript nimmt das zurückgegebene`Block`Informationen für jede Seite, formatiert die Daten als Tabelle und speichert die Tabellen in einer CSV-Datei.

So exportieren Sie Tabellen in eine CSV-Datei

1. Konfigurieren Sie Ihre Umgebung. Weitere Informationen finden Sie unter Voraussetzungen .
2. Stellen Sie sicher, dass Sie die Anweisungen unter sehen befolgt habenKonfigurieren von Amazon Textract für asynchrone Vorgängeaus. Der auf dieser Seite dokumentierte Prozess ermöglicht es Ihnen, Nachrichten über den Abschlussstatus asynchroner Jobs zu senden und zu empfangen.
3. Ersetzen Sie im folgenden Codebeispiel den Wert von`roleArn`wobei der Arn der Rolle zugewiesen wurde, die Sie in Schritt 2 erstellt haben. Ersetzen Sie den Wert von`bucket`mit dem Namen des S3-Buckets, der Ihr Dokument enthält. Ersetzen Sie den Wert von`document`durch den Namen des Dokuments in Ihrem S3-Bucket. Ersetzen Sie den Wert von`region_name`durch den Namen der -Region Ihres Buckets.

   Speichern Sie den folgenden Beispiel-Code in einer Datei mit dem Namenstart_doc_analysis_for_table_extraction.py.aus.

```
import boto3
import time

class DocumentProcessor:

    jobId = ''
    region_name = ''

    roleArn = ''
    bucket = ''
    document = ''

    sqsQueueUrl = ''
    snsTopicArn = ''
    processType = ''

    def __init__(self, role, bucket, document, region):
        self.roleArn = role
        self.bucket = bucket
        self.document = document
        self.region_name = region

        self.textract = boto3.client('textract', region_name=self.region_name)
        self.sqs = boto3.client('sqs')
        self.sns = boto3.client('sns')
```

```
    def ProcessDocument(self):

        jobFound = False

        response =
 self.textract.start_document_analysis(DocumentLocation={'S3Object': {'Bucket':
 self.bucket, 'Name': self.document}},
                FeatureTypes=["TABLES", "FORMS"],
 NotificationChannel={'RoleArn': self.roleArn, 'SNSTopicArn':
 self.snsTopicArn})
        print('Processing type: Analysis')

        print('Start Job Id: ' + response['JobId'])

        print('Done!')

    def CreateTopicandQueue(self):

        millis = str(int(round(time.time() * 1000)))

        # Create SNS topic
        snsTopicName = "AmazonTextractTopic" + millis

        topicResponse = self.sns.create_topic(Name=snsTopicName)
        self.snsTopicArn = topicResponse['TopicArn']

        # create SQS queue
        sqsQueueName = "AmazonTextractQueue" + millis
        self.sqs.create_queue(QueueName=sqsQueueName)
        self.sqsQueueUrl = self.sqs.get_queue_url(QueueName=sqsQueueName)
['QueueUrl']

        attribs = self.sqs.get_queue_attributes(QueueUrl=self.sqsQueueUrl,
                                                AttributeNames=['QueueArn'])
['Attributes']

        sqsQueueArn = attribs['QueueArn']

        # Subscribe SQS queue to SNS topic
        self.sns.subscribe(TopicArn=self.snsTopicArn, Protocol='sqs',
 Endpoint=sqsQueueArn)

        # Authorize SNS to write SQS queue
```

```
        policy = """{{
      "Version":"2012-10-17",
      "Statement":[
        {{
          "Sid":"MyPolicy",
          "Effect":"Allow",
          "Principal" : {{"AWS" : "*"}},
          "Action":"SQS:SendMessage",
          "Resource": "{}",
          "Condition":{{
            "ArnEquals":{{
              "aws:SourceArn": "{}"
            }}
          }}
        }}
      ]
    }}""".format(sqsQueueArn, self.snsTopicArn)

        response = self.sqs.set_queue_attributes(
            QueueUrl=self.sqsQueueUrl,
            Attributes={
                'Policy': policy
            })

def main():
    roleArn = 'role-arn'
    bucket = 'bucket-name'
    document = 'document-name'
    region_name = 'region-name'

    analyzer = DocumentProcessor(roleArn, bucket, document, region_name)
    analyzer.CreateTopicandQueue()
    analyzer.ProcessDocument()

if __name__ == "__main__":
    main()
```

4. Führen Sie den Code aus. Der Code druckt ein JobId. Kopiere dieses JobId nach unten.
5. Warten Sie, bis Ihr Job die Verarbeitung abgeschlossen hat, und kopieren Sie nach Abschluss des Auftrags den folgenden Code in eine Datei namensget_doc_analysis_for_table_extraction.pyaus. Ersetzen Sie den Wert von`jobId`mit der Job-ID, die Sie zuvor kopiert haben. Ersetzen Sie den Wert von`region_name`mit

dem Namen der Region, die mit Ihrer Textrakt-Rolle verknüpft ist. Ersetzen Sie den Wert von`file_name`durch den Namen, den Sie dem Ausgang zuweisen möchten.

```
import boto3
from pprint import pprint

jobId = 'job-id'
region_name = 'region-name'
file_name = "output-file-name.csv"

textract = boto3.client('textract', region_name=region_name)

# Display information about a block
def DisplayBlockInfo(block):
    print("Block Id: " + block['Id'])
    print("Type: " + block['BlockType'])
    if 'EntityTypes' in block:
        print('EntityTypes: {}'.format(block['EntityTypes']))

    if 'Text' in block:
        print("Text: " + block['Text'])

    if block['BlockType'] != 'PAGE':
        print("Confidence: " + "{:.2f}".format(block['Confidence']) + "%")

def GetResults(jobId, file_name):
    maxResults = 1000
    paginationToken = None
    finished = False

    while finished == False:

        response = None

        if paginationToken == None:
            response = textract.get_document_analysis(JobId=jobId,
 MaxResults=maxResults)
        else:
            response = textract.get_document_analysis(JobId=jobId,
 MaxResults=maxResults,

 NextToken=paginationToken)
```

```
        blocks = response['Blocks']
        table_csv = get_table_csv_results(blocks)
        output_file = file_name
        # replace content
        with open(output_file, "at") as fout:
            fout.write(table_csv)
        # show the results
        print('Detected Document Text')
        print('Pages: {}'.format(response['DocumentMetadata']['Pages']))
        print('OUTPUT TO CSV FILE: ', output_file)

        # Display block information
        for block in blocks:
            DisplayBlockInfo(block)
            print()
            print()

        if 'NextToken' in response:
            paginationToken = response['NextToken']
        else:
            finished = True


def get_rows_columns_map(table_result, blocks_map):
    rows = {}
    for relationship in table_result['Relationships']:
        if relationship['Type'] == 'CHILD':
            for child_id in relationship['Ids']:
                try:
                    cell = blocks_map[child_id]
                    if cell['BlockType'] == 'CELL':
                        row_index = cell['RowIndex']
                        col_index = cell['ColumnIndex']
                        if row_index not in rows:
                            # create new row
                            rows[row_index] = {}

                        # get the text value
                        rows[row_index][col_index] = get_text(cell, blocks_map)
                except KeyError:
                    print("Error extracting Table data - {}:".format(KeyError))
                    pass
    return rows
```

```
def get_text(result, blocks_map):
    text = ''
    if 'Relationships' in result:
        for relationship in result['Relationships']:
            if relationship['Type'] == 'CHILD':
                for child_id in relationship['Ids']:
                    try:
                        word = blocks_map[child_id]
                        if word['BlockType'] == 'WORD':
                            text += word['Text'] + ' '
                        if word['BlockType'] == 'SELECTION_ELEMENT':
                            if word['SelectionStatus'] == 'SELECTED':
                                text += 'X '
                    except KeyError:
                        print("Error extracting Table data -
 {}:".format(KeyError))

    return text


def get_table_csv_results(blocks):

    pprint(blocks)

    blocks_map = {}
    table_blocks = []
    for block in blocks:
        blocks_map[block['Id']] = block
        if block['BlockType'] == "TABLE":
            table_blocks.append(block)

    if len(table_blocks) <= 0:
        return "<b> NO Table FOUND </b>"

    csv = ''
    for index, table in enumerate(table_blocks):
        csv += generate_table_csv(table, blocks_map, index + 1)
        csv += '\n\n'

    return csv


def generate_table_csv(table_result, blocks_map, table_index):
```

```
    rows = get_rows_columns_map(table_result, blocks_map)

    table_id = 'Table_' + str(table_index)

    # get cells.
    csv = 'Table: {0}\n\n'.format(table_id)

    for row_index, cols in rows.items():

        for col_index, text in cols.items():
            csv += '{}'.format(text) + ","
        csv += '\n'

    csv += '\n\n\n'
    return csv

response_blocks = GetResults(jobId, file_name)
```

6. Führen Sie den Code aus.

   Nachdem Sie Ihre Ergebnisse erhalten haben, löschen Sie unbedingt die zugehörigen SNS- und SQS-Ressourcen, sonst können Sie Gebühren für diese anfallen.

Erstellen einesAWS LambdaFunktion

Sie können Amazon Textract Textract-API-Vorgänge von einemAWS LambdaFunktion. Die folgenden Anweisungen zeigen, wie Sie eine Lambda-Funktion in Python erstellen, die aufrufthe section called “DetectDocumentText”aus. Es gibt eine Liste von zurückthe section called “Block”Objekte. Um dieses Beispiel auszuführen, benötigen Sie einen Amazon S3 S3-Bucket, der ein Dokument im PNG- oder JPEG-Format enthält. Verwenden Sie die Konsole, um die Funktion zu erstellen.

Ein Beispiel, das Lambda-Funktionen verwendet, um Dokumente in großem Maßstab zu verarbeiten, finden Sie unterDokumentenverarbeitung im großen Maßstab mit Amazon Textractaus.

So rufen Sie den DetectDocumentText-Vorgang von einer Lambda-Funktion aus auf:

Schritt 1: Erstellen eines Lambda-Bereitstellungspakets

1. Öffnen Sie ein Befehlsfenster.

2. Geben Sie die folgenden Befehle ein, um ein Bereitstellungspaket mit der neuesten Version derAWS-SDK.

```
pip install boto3 --target python/.
zip boto3-layer.zip -r python/
```

Schritt 2: Erstellen einer Lambda-Funktion

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die AWS Lambda-Konsole an https://console.aws.amazon.com/lambda.
2. Wählen Sie Create function (Funktion erstellen).
3. Geben Sie Folgendes an.
   - Wählen Sie Author from scratch aus.
   - Geben Sie als Function name (Funktionsname) einen Namen ein.
   - FürLaufzeit, wählenPython 3.7oderPython 3.6aus.
   - FürWählen oder erstellen Sie eine Ausführungsrolle, wählenErstellen einer neuen Rolle mit den grundlegenden Lambda-Berechtigungenaus.
4. Klicken Sie aufCreate -Funktionum die Lambda-Funktion zu erstellen.
5. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
6. Wählen Sie im NavigationsbereichRollenaus.
7. Wählen Sie aus der Ressourcenliste die IAM-Rolle aus, die Lambda für Sie erstellt hat. Der Rollenname beginnt mit dem Namen Ihrer Lambda-Funktion.
8. Wählen Sie das SymbolBerechtigungenund wählen Sie dannRichtlinien anfügenaus.
9. Wählen Sie die AmazonTextractFullAccess- und Amazons3ReadOnlyAccess-Richtlinien aus.
10. SelectRichtlinie anfügenaus.

Weitere Informationen finden Sie unterErstellen einer Lambda-Funktion mit der Konsole

Schritt 3: Erstellen und fügen Sie eine -Ebene hinzu

1. Öffnen Sie die AWS Lambda-Konsole unter https://console.aws.amazon.com/lambda/.
2. Wählen Sie im Navigationsbereich Layers aus.

3. Wählen Sie Create Layer (Ebene erstellen) aus.
4. FürNameGeben Sie einen Namen ein.
5. Geben Sie im Feld Description (Beschreibung) eine Beschreibung ein.
6. FürCodeeingabetyp, wählenHochladen der ZIP-Dateiund wählen Sie ausHochladenaus.
7. Wählen Sie im Dialogfeld die ZIP-Datei (boto3-layer.zip) aus, die Zip, in der Sie erstellt habenSchritt 1: Erstellen eines Lambda-Bereitstellungspaketsaus.
8. FürKompatible Laufzeiten, wählen Sie die Version der Laufzeit aus, in der Sie ausgewählt habenSchritt 2: Erstellen einer Lambda-Funktionaus.
9. Klicken Sie aufGeben Sie einen Namen für den Benutzer ein und klicken Sie dann aufum den Layer zu erstellen.
10. Wählen Sie das Menüsymbol des Navigationsbereichs.
11. Wählen Sie im Navigationsbereich Functions aus.
12. Wählen Sie in der Ressourcenliste die Funktion aus, die Sie in erstellt habenSchritt 2: Erstellen einer Lambda-Funktionaus.
13. Klicken Sie aufKonfigurationund imDesigner-Bereich wählenEbenen(unter Ihrem Lambda-Funktionsnamen).
14. In derEbenen-Bereich wählenHinzufügen einer Ebeneaus.
15. Klicken Sie aufWählen Sie aus der Liste der Laufzeit-kompatiblen Layeraus.
16. In :Kompatible LayerWählen Sie denNameundVersionder -Ebene, die Sie in Schritt 3 erstellt haben.
17. Wählen Sie Add (Hinzufügen) aus.

Schritt 4: Fügen Sie der Funktion Python-Code hinzu

1. In :DesignerWählen Sie Ihre Funktion aus.
2. Fügen Sie im Funktionscode-Editor der Datei Folgendes hinzulambda_function.pyaus. Ändern Sie die Werte von`bucket`und`document`zu deinem Bucket und Dokument.

```
import json
import boto3

def lambda_handler(event, context):
```

```
    bucket="bucket"
    document="document"
    client = boto3.client('textract')



    #process using S3 object
    response = client.detect_document_text(
        Document={'S3Object': {'Bucket': bucket, 'Name': document}})

    #Get the text blocks
    blocks=response['Blocks']

    return {
        'statusCode': 200,
        'body': json.dumps(blocks)
    }
```

3. Klicken Sie aufSaveum Ihre Lambda-Funktion zu speichern.

Schritt 5: Testen Sie Ihr Lambda

1. SelectTestaus.
2. Geben Sie einen Wert für einEvent name (Ereignisname)aus.
3. Wählen Sie Create (Erstellen) aus.
4. Die Ausgabe, eine Liste vonthe section called “Block”-Objekte werden im Bereich Ausführungsergebnisse angezeigt.

Wenn das SymbolAWS Lambdagibt einen Timeout-Fehler zurück, ein Amazon Textract Textract-API-Betriebsaufruf könnte die Ursache sein. Weitere Informationen über die Verlängerung des Timout-Zeitraums für eineAWS Lambda-Funktion, sieheAWS Lambda-Funktionskonfigurationaus.

Weitere Informationen zum Aufrufen einer Lambda-Funktion aus Ihrem Code finden Sie unterAufrufenAWS LambdaFunktionenaus.

Weitere Codebeispiele

Die folgenden Tabelle enthält Links zu weiteren Codebeispielen für Amazon Textract.

Beispiel	Beschreibung
Beispiele für Amazon Textract Codebeispiele	Zeigen Sie verschiedene Möglichkeiten auf, wie Sie Amazon Textract verwenden können.
Dokumentenverarbeitung im großen Maßstab mit Amazon Textract	Zeigt eine serverlose Referenzarchitektur an, die Dokumente in großem Maßstab verarbeitet.
Amazon Textract Parser	Zeigt, wie das analysiert wirdthe section called “Block”Objekte, die von Amazon Textract Textract-Operationen zurückgegeben werden.
Beispiele für Amazon Textract Textract-Dokumentationscode	In diesem Handbuch verwendete Codebeisp iele.
Textraktor	Zeigt, wie die Amazon Textract Textract-Ausgabe in mehrere Formate konvertiert wird.
Generieren Sie durchsuchbare PDF-Dokum ente mit Amazon Textract	Zeigt, wie ein durchsuchbares PDF-Dokum ent aus verschiedenen Arten von Eingabedo kumenten wie Bildern im JPG/PNG-Format und gescannten PDF-Dokumenten erstellt wird.

Code-Beispiele für Amazon Textract

Die folgenden Code-Beispiele zeigen, wie man Amazon Textract mit einemAWSSoftware-Entwicklungskit (SDK).

Die Beispiele lassen sich in die folgenden Kategorien einteilen:

Aktionen

Code-Auszüge, die Ihnen zeigen, wie Sie einzelne Servicefunktionen aufrufen.

Serviceübergreifende Beispiele

Beispielanwendungen, die über mehrere hinweg arbeitenAWS-Services.

Eine vollständige Liste vonAWSSDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von Amazon Textract mit einemAWSSDKaus. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Codebeispiele

- Aktionen für Amazon Textract
  - Analysieren Sie ein Dokument mit Amazon Textract und einemAWSSDK
  - Erkennen Sie Text in einem Dokument mit Amazon Textract und einemAWSSDK
  - Abrufen von Daten über einen Amazon Textract Textract-Dokumentenanalyseauftrag mit einemAWSSDK
  - Starten Sie die asynchrone Analyse eines Dokuments mit Amazon Textract und einemAWSSDK
  - Starten Sie die asynchrone Texterkennung mit Amazon Textract und einemAWSSDK
- Serviceübergreifende Beispiele für Amazon Textract
  - Erstellen Sie eine Amazon-Textract-Explorer-Anwendung
  - Erkennen Sie Entitäten in Text, der aus einem Bild extrahiert wurdeAWSSDK

Aktionen für Amazon Textract

In den folgenden Codebeispielen wird demonstriert, wie Sie einzelne Amazon Textract Aktionen ausführen könnenAWS-SDKs. Diese Auszüge rufen die Amazon Textract Textract-API auf und sollen

nicht isoliert ausgeführt werden. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes im Kontext finden.

Die folgenden Beispiele enthalten nur die am häufigsten verwendeten Aktionen. Eine vollständige Liste finden Sie unterAmazon Textract API-Referenzaus.

Beispiele

- Analysieren Sie ein Dokument mit Amazon Textract und einemAWSSDK
- Erkennen Sie Text in einem Dokument mit Amazon Textract und einemAWSSDK
- Abrufen von Daten über einen Amazon Textract Textract-Dokumentenanalyseauftrag mit einemAWSSDK
- Starten Sie die asynchrone Analyse eines Dokuments mit Amazon Textract und einemAWSSDK
- Starten Sie die asynchrone Texterkennung mit Amazon Textract und einemAWSSDK

Analysieren Sie ein Dokument mit Amazon Textract und einemAWSSDK

In den folgenden Codebeispielen wird demonstriert, wie Sie ein Dokument mit Amazon Textract analysieren.

Java

SDK for Java 2.x

```
   public static void analyzeDoc(TextractClient textractClient, String
sourceDoc) {

       try {
           InputStream sourceStream = new FileInputStream(new File(sourceDoc));
           SdkBytes sourceBytes = SdkBytes.fromInputStream(sourceStream);

           // Get the input Document object as bytes
           Document myDoc = Document.builder()
                   .bytes(sourceBytes)
                   .build();

           List<FeatureType> featureTypes = new ArrayList<FeatureType>();
           featureTypes.add(FeatureType.FORMS);
           featureTypes.add(FeatureType.TABLES);
```

```
            AnalyzeDocumentRequest analyzeDocumentRequest =
AnalyzeDocumentRequest.builder()
                    .featureTypes(featureTypes)
                    .document(myDoc)
                    .build();

            AnalyzeDocumentResponse analyzeDocument =
textractClient.analyzeDocument(analyzeDocumentRequest);
            List<Block> docInfo = analyzeDocument.blocks();
            Iterator<Block> blockIterator = docInfo.iterator();

            while(blockIterator.hasNext()) {
                Block block = blockIterator.next();
                System.out.println("The block type is "
+block.blockType().toString());
            }

        } catch (TextractException | FileNotFoundException e) {

            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

- Weitere Anleitungen und mehr Code finden Sie im GitHub.
- API-Details dazu finden Sie unter.AnalyzeDocumentinAWS SDK for Java 2.x-API-Referenzaus.

Python

SDK for Python (Boto3)

```
class TextractWrapper:
    """Encapsulates Textract functions."""
    def __init__(self, textract_client, s3_resource, sqs_resource):
        """
        :param textract_client: A Boto3 Textract client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        :param sqs_resource: A Boto3 Amazon SQS resource.
        """
```

```
        self.textract_client = textract_client
        self.s3_resource = s3_resource
        self.sqs_resource = sqs_resource

    def analyze_file(
            self, feature_types, *, document_file_name=None,
document_bytes=None):
        """
        Detects text and additional elements, such as forms or tables, in a local
image
        file or from in-memory byte data.
        The image must be in PNG or JPG format.

        :param feature_types: The types of additional document features to
detect.
        :param document_file_name: The name of a document image file.
        :param document_bytes: In-memory byte data of a document image.
        :return: The response from Amazon Textract, including a list of blocks
                 that describe elements detected in the image.
        """
        if document_file_name is not None:
            with open(document_file_name, 'rb') as document_file:
                document_bytes = document_file.read()
        try:
            response = self.textract_client.analyze_document(
                Document={'Bytes': document_bytes}, FeatureTypes=feature_types)
            logger.info(
                "Detected %s blocks.", len(response['Blocks']))
        except ClientError:
            logger.exception("Couldn't detect text.")
            raise
        else:
            return response
```

- Finden Sie Anweisungen und mehr Code im GitHub.
- API-Details dazu finden Sie unter.AnalyzeDocumentinAWSAPI-Referenz für SDK for Python (Boto3)aus.

Eine vollständige Liste vonAWSSDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von Amazon Textract mit einemAWSSDKaus. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erkennen Sie Text in einem Dokument mit Amazon Textract und einemAWSSDK

In den folgenden Codebeispielen wird demonstriert, wie man Text in einem Dokument mithilfe von Amazon Textract erkennt.

Java

SDK für Java 2.x

Erkennt Text aus einem Eingabedokument.

```
   public static void detectDocText(TextractClient textractClient,String
sourceDoc) {

       try {

           InputStream sourceStream = new FileInputStream(new File(sourceDoc));
           SdkBytes sourceBytes = SdkBytes.fromInputStream(sourceStream);

           // Get the input Document object as bytes
           Document myDoc = Document.builder()
                   .bytes(sourceBytes)
                   .build();

           DetectDocumentTextRequest detectDocumentTextRequest =
DetectDocumentTextRequest.builder()
                   .document(myDoc)
                   .build();

           // Invoke the Detect operation
           DetectDocumentTextResponse textResponse =
textractClient.detectDocumentText(detectDocumentTextRequest);

           List<Block> docInfo = textResponse.blocks();

           Iterator<Block> blockIterator = docInfo.iterator();

           while(blockIterator.hasNext()) {
               Block block = blockIterator.next();
               System.out.println("The block type is "
+block.blockType().toString());
           }
```

```
            DocumentMetadata documentMetadata = textResponse.documentMetadata();
            System.out.println("The number of pages in the document is "
+documentMetadata.pages());

        } catch (TextractException | FileNotFoundException e) {

            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

Erkennen von Text aus einem Dokument in einem Amazon S3 S3-Bucket.

```
    public static void detectDocTextS3 (TextractClient textractClient, String
bucketName, String docName) {

        try {
            S3Object s3Object = S3Object.builder()
                    .bucket(bucketName)
                    .name(docName)
                    .build();

            // Create a Document object and reference the s3Object instance
            Document myDoc = Document.builder()
                    .s3Object(s3Object)
                    .build();

            // Create a DetectDocumentTextRequest object
            DetectDocumentTextRequest detectDocumentTextRequest =
DetectDocumentTextRequest.builder()
                    .document(myDoc)
                    .build();

            // Invoke the detectDocumentText method
            DetectDocumentTextResponse textResponse =
textractClient.detectDocumentText(detectDocumentTextRequest);

            List<Block> docInfo = textResponse.blocks();

            Iterator<Block> blockIterator = docInfo.iterator();
```

```
            while(blockIterator.hasNext()) {
                Block block = blockIterator.next();
                System.out.println("The block type is "
 +block.blockType().toString());
            }

            DocumentMetadata documentMetadata = textResponse.documentMetadata();
            System.out.println("The number of pages in the document is "
 +documentMetadata.pages());

        } catch (TextractException e) {

            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

- Weitere Anleitungen und mehr Code finden Sie im GitHub.
- API-Details dazu finden Sie unter.DetectDocumentTextinAWS SDK for Java 2.x-API-Referenzaus.

Python

SDK for Python (Boto3)

```
class TextractWrapper:
    """Encapsulates Textract functions."""
    def __init__(self, textract_client, s3_resource, sqs_resource):
        """
        :param textract_client: A Boto3 Textract client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        :param sqs_resource: A Boto3 Amazon SQS resource.
        """
        self.textract_client = textract_client
        self.s3_resource = s3_resource
        self.sqs_resource = sqs_resource

    def detect_file_text(self, *, document_file_name=None, document_bytes=None):
        """
        Detects text elements in a local image file or from in-memory byte data.
```

```
        The image must be in PNG or JPG format.

        :param document_file_name: The name of a document image file.
        :param document_bytes: In-memory byte data of a document image.
        :return: The response from Amazon Textract, including a list of blocks
                 that describe elements detected in the image.
        """
        if document_file_name is not None:
            with open(document_file_name, 'rb') as document_file:
                document_bytes = document_file.read()
        try:
            response = self.textract_client.detect_document_text(
                Document={'Bytes': document_bytes})
            logger.info(
                "Detected %s blocks.", len(response['Blocks']))
        except ClientError:
            logger.exception("Couldn't detect text.")
            raise
        else:
            return response
```

- Finden Sie Anweisungen und mehr Code im GitHub.
- API-Details dazu finden Sie unter.DetectDocumentTextinAWSAPI-Referenz für SDK for Python (Boto3)aus.

Eine vollständige Liste vonAWSSDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von Amazon Textract mit einemAWSSDKaus. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Abrufen von Daten über einen Amazon Textract Textract-Dokumentenanalyseauftrag mit einemAWSSDK

Im folgenden Codebeispiel wird veranschaulicht, wie Sie Daten zu einem Amazon Textract Dokumentanalyse-Auftrag abrufen.

Python

SDK for Python (Boto3)

```
class TextractWrapper:
    """Encapsulates Textract functions."""
    def __init__(self, textract_client, s3_resource, sqs_resource):
        """
        :param textract_client: A Boto3 Textract client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        :param sqs_resource: A Boto3 Amazon SQS resource.
        """
        self.textract_client = textract_client
        self.s3_resource = s3_resource
        self.sqs_resource = sqs_resource

    def get_analysis_job(self, job_id):
        """
        Gets data for a previously started detection job that includes additional
        elements.

        :param job_id: The ID of the job to retrieve.
        :return: The job data, including a list of blocks that describe elements
                 detected in the image.
        """
        try:
            response = self.textract_client.get_document_analysis(
                JobId=job_id)
            job_status = response['JobStatus']
            logger.info("Job %s status is %s.", job_id, job_status)
        except ClientError:
            logger.exception("Couldn't get data for job %s.", job_id)
            raise
        else:
            return response
```

- Finden Sie Anweisungen und mehr Code im GitHub.
- API-Details dazu finden Sie unter.GetDocumentAnalysisinAWSAPI-Referenz für SDK for Python (Boto3)aus.

Eine vollständige Liste vonAWSSDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von Amazon Textract mit einemAWSSDKaus. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Starten Sie die asynchrone Analyse eines Dokuments mit Amazon Textract und einemAWSSDK

In den folgenden Codebeispielen wird demonstriert, wie Sie eine asynchrone Analyse eines Dokuments mit Amazon Textract starten.

Java

SDK for Java 2.x

```
   public static String startDocAnalysisS3 (TextractClient textractClient,
String bucketName, String docName) {

       try {

           List<FeatureType> myList = new ArrayList<FeatureType>();
           myList.add(FeatureType.TABLES);
           myList.add(FeatureType.FORMS);

           S3Object s3Object = S3Object.builder()
                   .bucket(bucketName)
                   .name(docName)
                   .build();

           DocumentLocation location = DocumentLocation.builder()
                   .s3Object(s3Object)
                   .build();

           StartDocumentAnalysisRequest documentAnalysisRequest =
StartDocumentAnalysisRequest.builder()
                   .documentLocation(location)
                   .featureTypes(myList)
                   .build();

           StartDocumentAnalysisResponse response =
textractClient.startDocumentAnalysis(documentAnalysisRequest);

           // Get the job ID
           String jobId = response.jobId();
           return jobId;

       } catch (TextractException e) {
```

```
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "" ;
    }

    private static String getJobResults(TextractClient textractClient, String
jobId) {

        boolean finished = false;
        int index = 0 ;
        String status = "" ;

       try {
        while (!finished) {
            GetDocumentAnalysisRequest analysisRequest =
GetDocumentAnalysisRequest.builder()
                    .jobId(jobId)
                    .maxResults(1000)
                    .build();

            GetDocumentAnalysisResponse response =
textractClient.getDocumentAnalysis(analysisRequest);
            status = response.jobStatus().toString();

            if (status.compareTo("SUCCEEDED") == 0)
                finished = true;
            else {
                System.out.println(index + " status is: " + status);
                Thread.sleep(1000);
            }
            index++ ;
        }
        return status;

       } catch( InterruptedException e) {
           System.out.println(e.getMessage());
           System.exit(1);
       }
       return "";
    }
```

- Weitere Anleitungen und mehr Code finden Sie im GitHub.
- API-Details dazu finden Sie unter.StartDocumentAnalysisinAWS SDK for Java 2.x-API-Referenzaus.

Python

SDK for Python (Boto3)

Starten Sie einen asynchronen Job, um ein Dokument zu analysieren.

```
class TextractWrapper:
    """Encapsulates Textract functions."""
    def __init__(self, textract_client, s3_resource, sqs_resource):
        """
        :param textract_client: A Boto3 Textract client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        :param sqs_resource: A Boto3 Amazon SQS resource.
        """
        self.textract_client = textract_client
        self.s3_resource = s3_resource
        self.sqs_resource = sqs_resource

    def start_analysis_job(
            self, bucket_name, document_file_name, feature_types, sns_topic_arn,
            sns_role_arn):
        """
        Starts an asynchronous job to detect text and additional elements, such
 as
        forms or tables, in an image stored in an Amazon S3 bucket. Textract
 publishes
        a notification to the specified Amazon SNS topic when the job completes.
        The image must be in PNG, JPG, or PDF format.

        :param bucket_name: The name of the Amazon S3 bucket that contains the
 image.
        :param document_file_name: The name of the document image stored in
 Amazon S3.
        :param feature_types: The types of additional document features to
 detect.
        :param sns_topic_arn: The Amazon Resource Name (ARN) of an Amazon SNS
 topic
                              where job completion notification is published.
```

```
        :param sns_role_arn: The ARN of an AWS Identity and Access Management
(IAM)
                             role that can be assumed by Textract and grants
permission
                             to publish to the Amazon SNS topic.
        :return: The ID of the job.
        """
        try:
            response = self.textract_client.start_document_analysis(
                DocumentLocation={
                    'S3Object': {'Bucket': bucket_name, 'Name':
document_file_name}},
                NotificationChannel={
                    'SNSTopicArn': sns_topic_arn, 'RoleArn': sns_role_arn},
                FeatureTypes=feature_types)
            job_id = response['JobId']
            logger.info(
                "Started text analysis job %s on %s.", job_id,
document_file_name)
        except ClientError:
            logger.exception("Couldn't analyze text in %s.", document_file_name)
            raise
        else:
            return job_id
```

- Weitere Anleitungen und mehr Code finden Sie im GitHub.
- API-Details dazu finden Sie unter.StartDocumentAnalysisinAWSAPI-Referenz für SDK for Python (Boto3)aus.

Eine vollständige Liste vonAWSSDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von Amazon Textract mit einemAWSSDKaus. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Starten Sie die asynchrone Texterkennung mit Amazon Textract und einemAWSSDK

Im folgenden Codebeispiel wird veranschaulicht, wie Sie die asynchrone Texterkennung in einem Dokument mit Amazon Textract starten.

Python

SDK for Python (Boto3)

Starten Sie einen asynchronen Auftrag, um Text in einem Dokument zu erkennen.

```
class TextractWrapper:
    """Encapsulates Textract functions."""
    def __init__(self, textract_client, s3_resource, sqs_resource):
        """
        :param textract_client: A Boto3 Textract client.
        :param s3_resource: A Boto3 Amazon S3 resource.
        :param sqs_resource: A Boto3 Amazon SQS resource.
        """
        self.textract_client = textract_client
        self.s3_resource = s3_resource
        self.sqs_resource = sqs_resource

    def start_detection_job(
            self, bucket_name, document_file_name, sns_topic_arn, sns_role_arn):
        """
        Starts an asynchronous job to detect text elements in an image stored in
 an
        Amazon S3 bucket. Textract publishes a notification to the specified
 Amazon SNS
        topic when the job completes.
        The image must be in PNG, JPG, or PDF format.

        :param bucket_name: The name of the Amazon S3 bucket that contains the
 image.
        :param document_file_name: The name of the document image stored in
 Amazon S3.
        :param sns_topic_arn: The Amazon Resource Name (ARN) of an Amazon SNS
 topic
                              where the job completion notification is published.
        :param sns_role_arn: The ARN of an AWS Identity and Access Management
 (IAM)
                             role that can be assumed by Textract and grants
 permission
                             to publish to the Amazon SNS topic.
        :return: The ID of the job.
        """
        try:
            response = self.textract_client.start_document_text_detection(
```

```
                DocumentLocation={
                    'S3Object': {'Bucket': bucket_name, 'Name':
document_file_name}},
                NotificationChannel={
                    'SNSTopicArn': sns_topic_arn, 'RoleArn': sns_role_arn})
            job_id = response['JobId']
            logger.info(
                "Started text detection job %s on %s.", job_id,
document_file_name)
        except ClientError:
            logger.exception("Couldn't detect text in %s.", document_file_name)
            raise
        else:
            return job_id
```

- Weitere Anleitungen und mehr Code finden Sie im GitHub.
- API-Details dazu finden Sie unter.StartDocumentTextDetectioninAWSAPI-Referenz für SDK for Python (Boto3)aus.

Eine vollständige Liste vonAWSSDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von Amazon Textract mit einemAWSSDKaus. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Serviceübergreifende Beispiele für Amazon Textract

Die folgenden Beispielanwendungen verwendenAWSSDKs zur Kombination von Amazon Textract mit anderenAWS-Services. Jedes Beispiel enthält einen Link zu GitHub, in dem Sie Anweisungen zum Einrichten und Ausführen der Anwendung finden.

Beispiele

- Erstellen Sie eine Amazon-Textract-Explorer-Anwendung
- Erkennen Sie Entitäten in Text, der aus einem Bild extrahiert wurdeAWSSDK

Erstellen Sie eine Amazon-Textract-Explorer-Anwendung

Die folgenden Code-Beispiele zeigen, wie man die Amazon-Textract-Ausgabe in einer interaktiven Anwendung untersuchen kann.

JavaScript

SDK for JavaScript V3

Veranschaulicht, wie Sie das AWS SDK für JavaScript verwenden, um eine React-Anwendung zu erstellen, die Amazon Textract verwendet, um Daten aus einem Dokument-Image zu extrahieren und auf einer interaktiven Webseite anzuzeigen. Dieses Beispiel wird in einem Webbrowser ausgeführt und erfordert eine authentifizierte Amazon-Cognito-Identität für Anmeldeinformationen. Es verwendet Amazon Simple Storage Service (Amazon S3) zur Speicherung und fragt für Benachrichtigungen eine Amazon Simple Queue Service (Amazon SQS)-Warteschlange ab, die ein Amazon Simple Notification Service (Amazon SNS)-Thema abonniert hat.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel unter GitHub.

In diesem Beispiel verwendete Dienste

- Amazon Cognito Identity
- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Python

SDK for Python (Boto3)

Zeigt, wie man AWS SDK für Python (Boto3) mit Amazon Textract verwendet, um Text-, Formular- und Tabellenelemente in einem Dokument-Image zu erkennen. Das Eingabe-Image und die Amazon-Textract-Ausgabe werden in einer Tkinter-Anwendung angezeigt, mit der Sie die erkannten Elemente untersuchen können.

- Senden Sie ein Dokument-Image an Amazon Textract und untersuchen Sie die Ausgabe erkannter Elemente.
- Senden Sie Images direkt an Amazon Textract oder über einen Amazon Simple Storage Service (Amazon S3)-Bucket.
- Verwenden Sie asynchrone APIs, um einen Auftrag zu starten, der eine Benachrichtigung an ein Amazon Simple Notification Service (Amazon SNS)-Thema veröffentlicht.

- Stellen Sie eine Amazon Simple Queue Service (Amazon SQS)-Warteschlange ab, um eine Meldung zum Abschluss des Auftrags zu erhalten.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel unter GitHub.

In diesem Beispiel verwendete Dienste

- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Eine vollständige Liste vonAWSSDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von Amazon Textract mit einemAWSSDKaus. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erkennen Sie Entitäten in Text, der aus einem Bild extrahiert wurdeAWSSDK

Das folgende Codebeispiel zeigt, wie Amazon Comprehend verwendet wird, um Entitäten in Text zu erkennen, der von Amazon Textract aus einem Bild extrahiert wurde, das in Amazon S3 gespeichert ist.

Python

SDK for Python (Boto3)

Veranschaulicht, wie man das benutztAWS SDK für Python (Boto3)in einem Jupyter-Notizbuch, um Entitäten in Text zu erkennen, der aus einem Bild extrahiert wird. In diesem Beispiel extrahiert Amazon Textract Text aus einem Bild, das in Amazon Simple Storage Service (Amazon S3) und Amazon Comprehend gespeichert ist, um Entitäten im extrahierten Text zu erkennen.

Dieses Beispiel ist ein Jupyter-Notebook und muss in einer Umgebung ausgeführt werden, die Notebooks hosten kann. Anweisungen zum Ausführen des Beispiels mit Amazon SageMaker finden Sie in den Anweisungen untertextractandComprehendNotebook.iPynbaus.

Vollständiger Quellcode und Anweisungen zum Einrichten und Ausführen finden Sie im vollständigen Beispiel unter GitHub.

In diesem Beispiel verwendete Dienste

- Amazon Comprehend
- Amazon S3
- Amazon Textract

Eine vollständige Liste vonAWSSDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden von Amazon Textract mit einemAWSSDKaus. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwenden von Amazon Augmented AI, um eine menschliche Bewertung zu Amazon Textract Output hinzuzufügen

Amazon Augmented AI (Amazon A2I) ist ein Machine Learning (ML) -Service, der es einfach macht, Workflows für die Prüfung von ML-Analysen durch Menschen zu erstellen.

Amazon Textract ist in Amazon A2I integriert. Sie können damit Dokumentanalyseergebnisse mit einem niedrigen Konfidenzwert an menschliche Gutachter weiterleiten.

Sie können Amazon Textract verwenden`AnalyzeDocument`API zum Extrahieren von Daten aus Formularen und der Amazon A2I-Konsole. Sie können die Bedingungen angeben, unter denen Amazon A2I Prognosen an Prüfer weiterleitet. Sie legen Bedingungen basierend auf der Konfidenzschwelle wichtiger Formularschlüssel fest. Zum Beispiel können Sie ein Dokument an einen menschlichen Prüfer senden, wenn der SchlüsselNameoder sein zugehöriger WertJane Doewurde mit geringem Vertrauen festgestellt.

Themen

- Kernkonzepte von Amazon A2I
- Verwenden von Amazon A2I

Kernkonzepte von Amazon A2I

Lesen Sie die folgenden Begriffe, um sich mit den Kernkonzepten von Amazon A2I vertraut zu machen.

Aktivierungsbedingungen der Prüfung durch Menschen

Sie können Amazon A2I verwendenAktivierungsbedingungenum anzugeben, wann ein Dokument zur Überprüfung an Menschen gesendet wird und welchen Formularinhalt, den die Arbeitnehmer überprüfen sollen.

Sie können beispielsweise eine Aktivierungsbedingung festlegen, damit Amazon Textract Formulare an Amazon A2I weiterleitet, wenn ein wichtiger Schlüssel mit geringem Vertrauen erkannt wird, wie z.Telefonnummeraus. In diesem Beispiel werden menschliche Gutachter gebeten, dieTelefonnummerFeld und Wert, der von Amazon Textract erkannt wurde.

Mithilfe der folgenden Aktivierungsbedingungen können Sie angeben, wann Formulare zur Überprüfung an Menschen gesendet werden:

- Eine Prüfung durch Menschen für bestimmte Formularschlüssel basierend auf dem Konfidenzwert des Formulars auslösen. Menschliche Prüfer werden aufgefordert, diese Formularschlüssel und dazugehörige Werte zu überprüfen.
- Eine Prüfung durch Menschen auslösen, wenn bestimmte Formularschlüssel fehlen. Menschliche Gutachter werden gebeten, diese Formularschlüssel und die zugehörigen Werte zu identifizieren.
- Eine Prüfung durch Menschen für alle von Amazon Textract identifizierte Formularschlüssel mit Konfidenzwerten in einem bestimmten Bereich auslösen.
- Senden Sie eine Stichprobe von Formularen nach dem Zufallsprinzip an Personen für die Prüfung durch Menschen. Menschliche Prüfer werden aufgefordert, alle von Amazon Textract erkannten Formularschlüssel und -Werte zu überprüfen.

Wenn Ihre Aktivierungsbedingung von den Schlüsselkonfidenzwerten des Formulars abhängt, können Sie zwei Arten von Schwellenwerten für das Konfidenzkenntnissen des Formulars verwenden, um

- Identifizierungs-Vertrauen— Der Konfidenzwert für Schlüssel-Wert-Paare, die in einem Formular erkannt wurden.
- Qualifikation Vertrauen— Der Konfidenzwert für Text, der in einem Schlüssel-Wert-Paar in einem Formular enthalten ist.

Wenn Sie einen Konfidenzschwellenwert angeben, leitet Amazon A2I nur die Vorhersagen, die innerhalb des Schwellenwerts liegen, an menschliche Gutachter weiter. Sie können diese Schwellenwerte jederzeit anpassen, um das richtige Gleichgewicht zwischen Genauigkeit und Wirtschaftlichkeit zu erreichen. Dies kann Ihnen helfen, Audits zu implementieren, um die Genauigkeit der Vorhersagen regelmäßig zu überwachen.

> **Note**
>
> Sie können die Bedingungen, unter denen Dokumente zur Überprüfung an Menschen gesendet werden, weiter anpassen, indem Sie den benutzerdefinierten Aufgabentyp Amazon A2I verwenden. Mit diesem Aufgabentyp geben Sie direkt in Ihrer Bewerbung Bedingungen für eine menschliche Überprüfung an. Weitere Informationen finden Sie unterVerwenden

von Amazon Augmented AI mit benutzerdefinierten Aufgabentypenim Amazon SageMaker Developer Guide.

Arbeitsablauf für menschliche Überprüfung (Flow-Definition)

Du benutzt einArbeitsablauf für menschliche Überprüfung, auch alsFlow-Definition, um Ressourcen anzugeben, die zum Erstellen Ihres Personalprüfungsworkflows verwendet werden, und um Ihre Aktivierungsbedingungen festzulegen.

Die von Ihnen angegebenen Ressourcen sind:

- Eine IAM-Rolle mit der Berechtigung zum Aufrufen von Amazon A2I-API-Operationen
- Ein Amazon S3 S3-Bucket, in dem Sie die Ausgabe des Human Review speichern möchten
- Ihr MenschArbeitsteam-Arbeit
- EINVorlage für Auftragnehmeraufgabendas enthält Anweisungen und Beispiele, um Mitarbeitern beim Abschließen der Bewertungsaufgabe zu helfen

Sie verwenden auch den Arbeitsablauf für menschliche Überprüfung, um Aktivierungsbedingungen festzulegen. Weitere Informationen finden Sie unter Aktivierungsbedingungen der Prüfung durch Menschen .

Sie können einen einzelnen Workflow für die Prüfung durch Menschen verwenden, um mehrereMenschliche Schleifenaus.

Sie können einen Workflow für die Prüfung durch Menschen in der SageMaker-Konsole oder mit der SageMaker-API erstellen. Weitere Informationen finden Sie unter Erstellen eines Workflows für die Prüfung durch Menschen.

Vorlage für Worker-Aufgaben

Du benutzt einVorlage für Auftragnehmeraufgabenum eine Worker-Benutzeroberfläche zu erstellen, die für Ihre menschlichen Überprüfungsaufgaben verwendet wird.

Die Worker-Benutzeroberfläche zeigt Ihre Dokumente und Arbeitsanweisungen an. Es bietet auch Tools, die Mitarbeiter verwenden, um Ihre Aufgaben abzuschließen.

Wenn Sie die SageMaker-Konsole verwenden, um die Worker-Aufgabenvorlage zu konfigurieren, wenn Sie einen Workflow für die Prüfung durch Menschen erstellen. Weitere Informationen finden Sie unter Erstellen eines Workflows für die Prüfung durch Menschen.

Arbeitsteam-Arbeit

EINArbeitsteam-Arbeitist eine Gruppe menschlicher Arbeiter, an die Sie Ihre menschlichen Überprüfungsaufgaben senden.

Wenn Sie einen Workflow für die Prüfung durch Menschen erstellen, geben Sie ein einzelnes Arbeitsteam an.

Mit Amazon A2I können Sie einen Pool von Prüfern innerhalb Ihrer eigenen Organisation verwenden. Sie können auch auf die Belegschaft zugreifen, die aus über 500.000 unabhängigen Auftragnehmern besteht, die bereits Aufgaben des maschinellen Lernens über Amazon Mechanical Turk ausführen. Eine weitere Möglichkeit besteht darin, von AWS vorab geprüfte Belegschaftsanbieter für Qualität und Einhaltung von Sicherheitsverfahren einzusetzen.

Amazon A2I bietet Rezensenten auch ein Webinterface, das aus allen Anweisungen und Tools besteht, die sie zum Abschließen ihrer Überprüfungsaufgaben benötigen.

Für jede Art von Belegschaft (Privat, Vendor und Mechanical Turk) können Sie mehrere Arbeitsteams erstellen. Sie können jedes Arbeitsteam in mehreren Arbeitsabläufen für menschliche Überprüfungen verwenden. Weitere Informationen zum Erstellen von Arbeitskräften und Arbeitsteams finden Sie unterErstellen und Verwalten von Arbeitskräftenim Amazon SageMaker Developer Guide.

> **Important**
>
> Klicken aufhierum die Compliance-Programme zu sehen, die derzeit Amazon Augmented AI abdecken. Wenn Sie Amazon Augmented AI in Verbindung mit anderen AWS-Services (wie Amazon Rekognition und Amazon Textract) verwenden, beachten Sie bitte, dass Amazon Augmented AI möglicherweise nicht für dieselben Compliance-Programmen wie diese anderen Services verfügbar ist. Sie sind dafür verantwortlich, wie Sie Amazon Augmented AI nutzen, einschließlich des Verstehens, wie der Service Kundendaten verarbeitet oder speichert, und für alle Auswirkungen auf die Compliance Ihrer Datenumgebung. Sie sollten Ihre Workload-Vorgaben und -Ziele mit Ihrem AWS-Konto-Team besprechen, damit Sie prüfen können, ob der Service für Ihren vorgeschlagenen Anwendungsfall und die dazugehörige Architektur geeignet ist.
> Derzeit ist Amazon Augmented AI PCI-konform mit Ausnahme von öffentlichen und Anbieterbelegschaftsfällen.

Informationen zur Amazon Augmented AI HIPAA-Compliance erhalten Sie, wenn Sie aufhieraus.

Menschliche Schleifen

Sie verwenden eine menschliche Schleife, um eine Aufgabe mit Prüfung durch Menschen (Human Loop)

Sie weisen einem Mitarbeiter in Ihrem menschlichen Arbeiterteam eine menschliche Überprüfungsaufgabe zu. Die Arbeitskraft wird aufgefordert, Schlüssel-Wert-Paare zu überprüfen, die von Amazon Textract in Ihrem Eingabedokument erkannt wurden, das Sie unter Ihren Aktivierungsbedingungen angegeben haben.

Angenommen, ein Bild fällt zwischen fünfzig und sechzig Prozent darauf, dass es einen Apfel enthält. Dies liegt innerhalb Ihrer Vertrauensschwelle für menschliche Überprüfung und wird an einen Arbeitnehmer gesendet. Der Arbeiter überprüft das Dokument auf einen Apfel und markiert es so, dass er einen Apfel enthält oder nicht enthält. Dann sendet Amazon A2I das Dokument zurück in den Workflow.

Wenn Sie Amazon Textract anrufen`AnalyzeDocument`und geben Sie einen menschlichen Review-Workflow (Flow-Definition) und einen menschlichen Schleifennamen an, eine menschliche Überprüfungsaufgabe wird erstellt, wenn die in Ihrem Arbeitsablauf für menschliche Überprüfung angegebenen Aktivierungsbedingungen erfüllt sind. Diese Aufgaben werden mit den Ressourcen erstellt, die Sie in Ihrem Arbeitsablauf für menschliche Überprüfung angeben.

Verwenden von Amazon A2I

Die folgenden Schritte helfen Ihnen, Amazon A2I in eine einseitige Dokumentenanalyseaufgabe von Amazon Textract zu integrieren. Dazu führen Sie die folgenden Schritte aus:

1. Erstellen Sie mit der Amazon A2I-Konsole (empfohlen für neue Benutzer) oder die Amazon A2I-API einen Arbeitsablauf für menschliche Überprüfung.
2. Um ein Formular zu analysieren und bei Bedarf eine menschliche Überprüfung einzubeziehen, verwenden Sie die`AnalyzeDocument`-Betrieb und geben Sie den Amazon-Ressourcennamen (ARN) des Workflows für die Prüfung durch Menschen an. Die Antwort sagt Ihnen, ob menschliche Überprüfung erforderlich ist.

3. Überwachen Sie Ihre menschliche Schleife mit der Amazon A2I Konsole und der API.
4. Überprüfen Sie die Ergebnisse der menschlichen Überprüfung in einem Amazon S3 S3-Bucket, in den die Ergebnisse gesendet werden.

So richten Sie eine SageMaker-Notebook-Instanz ein und verwenden Sie ein Beispiel-Notebook unterEnd-to-End-Demo mit Amazon Textract und Augmented AIimAmazon SageMaker Entwicklerhandbuch

> **Note**
>
> In diesem Abschnitt wird erläutert, wie Sie einen Arbeitsablauf für die Überprüfung von Menschen für den Aufgabentyp Amazon A2I, Amazon Textract, erstellen. Um die Integration von Amazon A2I und Amazon Textract weiter anzupassen, können Sie den benutzerdefinierten Aufgabentyp Amazon A2I verwenden. Mit dieser Option stellen Sie eine benutzerdefinierte Worker-Aufgabenvorlage zur Verfügung und geben die Bedingungen an, unter denen ein Dokument direkt in Ihrer Anwendung zur menschlichen Überprüfung gesendet wird. Weitere Informationen finden Sie unterVerwenden von Amazon Augmented AI mit benutzerdefinierten AufgabentypenimAmazon SageMaker Entwicklerhandbuchaus.

Themen

- Erstellen eines Workflows für die Prüfung durch Menschen
- Analysieren des Dokuments
- Überwachen von Human L
- Ausgabedaten und Worker-Metriken anzeigen

Erstellen eines Workflows für die Prüfung durch Menschen

Sie können mit der Amazon A2I-Konsole (empfohlen für neue Benutzer) oder der Amazon A2I einen Arbeitsablauf für menschliche Überprüfung erstellen`CreateFlowDefinition`verwenden.

Themen

- Erstellen eines Workflows für die Prüfung durch Menschen (Human Review) (Konsole)
- Erstellen eines Workflows für die Prüfung durch Menschen (Human Review) (API)

Erstellen eines Workflows für die Prüfung durch Menschen (Human Review) (Konsole)

Sie können dieses Beispiel entweder mit Ihrem eigenen Dokument in Amazon S3 ausfüllen oder herunterladendieses Beispieldokumentund legen Sie es in Ihren Amazon S3 S3-Bucket.

Stellen Sie sicher, dass sich Ihr S3-Bucket im selben befindetAWSRegion, in der Sie Amazon Textract verwenden. Weitere Informationen zur Erstellung eines Buckets finden Sie unterBucket erstellenimAmazon Simple Storage Service Console — Benutzerhandbuchaus.

> **Note**
>
> Die Amazon A2I-Konsole ist in die SageMaker-Konsole eingebettet. Um die Konsole verwenden zu können, benötigen Sie Berechtigungen für den Zugriff auf die SageMaker-Konsole und zum Erstellen eines Arbeitsteams. Um loszulegen, können Sie dieAmazonSageMakerFullAccessEine von IAM verwaltete Richtlinie enthält, um die meisten Aktionen in SageMaker durchzuführen. Weitere Informationen finden Sie unterIdentity and Access Management for Amazon SageMakerimAmazon SageMaker Entwicklerhandbuchaus.

Themen

- Schritt 1: Erstellen eines Arbeitsteams (-Konsole)
- Schritt 2: Erstellen eines Workflows für die Prüfung durch Menschen (Human Review) (Konsole)

Schritt 1: Erstellen eines Arbeitsteams (-Konsole)

Erstellen Sie zunächst ein Arbeitsteam in der Amazon A2I-Konsole und fügen Sie sich als Mitarbeiter hinzu, damit Sie eine Vorschau der menschlichen Überprüfungsaufgabe im Arbeiterportal anzeigen können. Mitglieder des Arbeitsteams können sich verschiedene Aufgaben und Dokumente ansehen, die ihnen zugewiesen sind.

So erstellen Sie private Arbeitskräfte mit Worker-E-Mails (Console)

1. Öffnen Sie die SageMaker-Konsole unterhttps://console.aws.amazon.com/sagemaker/aus.
2. Klicken Sie im Navigationsbereich unterGround Truth, wählenKennzeichnung von Belegschaftenaus.
3. Wählen Sie Private (Privat) und anschließend Create private team (Privatteam erstellen) aus.
4. Wählen Sie Invite new workers by email (Neue Auftragnehmer per E-Mail einladen).

5. Geben Sie in diesem Beispiel Ihre E-Mail-Adresse und die E-Mail-Adresse aller anderen ein, die Sie in der Lage sein möchten, eine Vorschau des Arbeiterportals anzuzeigen. Sie können eine Liste von bis zu 50 E-Mail-Adressen, getrennt durch Kommas, in dieE-Mail-Adressenaus.
6. Geben Sie einen Organisationsnamen und eine Kontakt-E-Mail ein.
7. Wählen Sie Create private team (Privatteam erstellen).

Wenn du dich einem privaten Arbeitsteam hinzufügst, erhältst du eine E-Mail vonno-reply@verificationemail.commit Anmeldeinformationen. Verwenden Sie den Link in dieser E-Mail, um Ihr Passwort zurückzusetzen und sich beim Arbeiterportal anzumelden. Hier werden Ihre menschlichen Überprüfungsaufgaben angezeigt, nachdem Sie angerufen habenAnalyzeDocumentaus.

Schritt 2: Erstellen eines Workflows für die Prüfung durch Menschen (Human Review) (Konsole)

In diesem Schritt erstellen Sie einen Amazon Textract Human Review Workflow.

So erstellen Sie einen Workflow (Human Review) (Konsole)

1. Öffnen Sie die Amazon A2I Konsole unterhttps://console.aws.amazon.com/a2iZugriff auf denArbeitsabläufe für menschliche Überprüfungangezeigten.
2. Klicken Sie aufWorkflow für menschliche Überprüfung erstellenaus.
3. FürNameGeben Sie einen Workflow-Namen ein.
4. FürS3 bucketWählen Sie den Bucket aus, in dem Amazon A2I die Ergebnisse Ihrer Aufgaben für die Prüfung durch Menschen speichern soll. Wenn Sie keinen Bucket auswählen, ändern Sie diesen, um den Namen des Buckets einzugeben.
5. UNDERIAM-Rollewählen vonCreate a new role (Neue Rolle erstellen)aus. Es erscheint ein Fenster mit dem TitelErstellen einer IAM-Rolleaus. In diesem Fenster geben Sie die Amazon S3 S3-Buckets an, auf die diese Rolle Zugriff haben soll. Wenn du nicht wählstJeder S3-Bucket, geben Sie den Ausgabe-Bucket an, den Sie in Schritt 4 angegeben haben, und den Bucket, der Ihr Eingabedokument enthält.
6. FürAufgabentyp, wählenTextract - Extraktion des Schlüssel-Wert-Paares.
7. In :Amazon Textract Textract-Formularextraktion - Bedingungen für den Aufruf von menschlicher Überprüfung, geben Sie Aktivierungsbedingungen an. Wir empfehlen Ihnen, einen hohen Schwellenwert für den Konfidenzwert für mindestens einen Schlüssel in Ihrem Dokument festzulegen, um eine menschliche Überprüfung auszulösen, damit Sie eine Arbeitsaufgabe im Arbeiterportal in der Vorschau anzeigen können.

Wenn Sie das in dieser exemplarische Vorgehensweise bereitgestellte Beispieldokument verwendet haben, geben Sie die Aktivierungsbedingungen wie folgt an:

a. Klicken Sie aufEine Prüfung durch Menschen für bestimmte Formularschlüssel basierend auf dem Konfidenzwert des Formulars oder wenn bestimmte Formularschlüssel fehlen.
b. FürTastenname, geben Sie ein**`Mail Address`**aus.
c. Legen Sie den Wert fürIdentifizierungs-VertrauenSchwellenwert zwischen0und99aus.
d. Legen Sie den Wert fürQualifikation VertrauenSchwellenwert zwischen0und99aus.
e. Klicken Sie aufEine Prüfung durch Menschen für alle von Amazon Textract identifizierte Formularschlüssel mit Konfidenzwerten in einem bestimmten Bereich auslösen.
f. Für**`identification confidence`**Wählen Sie eine beliebige Zahl zwischen 0 und 90 aus.
g. Für**`qualification confidence`**Wählen Sie eine beliebige Zahl zwischen 0 und 90 aus.

Dies löst eine menschliche Überprüfung aus, wenn Amazon Textract einen Konfidenzwert zurückgibt, der weniger als 99 beträgtPostanschriftund seinen Wert oder wenn es für jedes im Dokument erkannte Schlüssel-Wert-Paar einen Konfidenzwert von weniger als 90 zurückgibt.

8. UNDERErstellung von Arbeitsaufgabenvorlagenwählen vonAus einer Standardvorlage erstellenaus.
9. FürName der VorlageGeben Sie einen aussagekräftigen Namen ein..
10. FürTask description, fügen Sie etwas Ähnliches wie folgt hinzu:

    `Read the instructions and review the document.`
11. FürArbeitnehmerInnenwählenPrivataus.
12. Wählen Sie aus dem Menü das private Team aus, das Sie erstellt haben.
13. Wählen Sie Erstellen.

Nachdem Ihr Arbeitsablauf für menschliche Überprüfung erstellt wurde, wird er in der Tabelle auf derArbeitsabläufe für menschliche Überprüfungangezeigten. Wenn derStatusistAktiv, kopieren und speichern Sie den Workflow-ARN.

Erstellen eines Workflows für die Prüfung durch Menschen (Human Review) (API)

Sie können einen Workflow für die Prüfung durch Menschen (Human Review) oderFlow-DefinitionVerwenden des Amazon A2I,`CreateFlowDefinition`verwenden.

In diesem Beispiel können Sie entweder Ihr eigenes Dokument in Amazon S3 verwenden oder herunterladendieses Beispieldokumentund lagere es in deinem S3-Bucket.

Stellen Sie sicher, dass sich Ihr Amazon S3 S3-Bucket im selben befindetAWSRegion, die Sie zum Anrufen verwenden möchten`AnalyzeDocument`aus. Um einen Bucket zu erstellen, befolgen Sie die Anweisungen unter Erstellen eines Buckets im Amazon Simple Storage Service Console Benutzerhandbuch.

Voraussetzungen

Um die Amazon A2I-API zum Erstellen eines Workflows für menschliche Überprüfung zu verwenden, müssen Sie die folgenden Voraussetzungen erfüllen:

- Konfigurieren Sie eine IAM-Rolle mit der Berechtigung, sowohl Amazon A2I- als auch Amazon Textract Textract-API-Vorgänge aufzurufen. Zu Beginn können Sie die AWS-Richtlinien AmazonAugmentedAIFullAccess und AmazonTextractFullAccess an eine IAM-Rolle anhängen. Notieren Sie die IAM-Rolle Amazon Resources Name (ARN), da Sie sie später benötigen.

  Weitere detailliertere Berechtigungen bei der Verwendung von Amazon Textract finden Sie unterBeispiele für Amazon Textract identitätsbasierte -Richtlinienaus. Weitere Informationen zur Amazon A2I finden Sie unterBerechtigungen und Sicherheit in Amazon Augmented AIimAmazon SageMaker Entwicklerhandbuchaus.
- Erstellen Sie ein privates Arbeitsteam und notieren Sie den ARN des Arbeitsteams. Wenn Sie ein neuer Benutzer von Amazon A2I sind, folgen Sie den Anweisungen unterSchritt 1: Erstellen eines Arbeitsteams (-Konsole)aus.
- Erstellen Sie eine Vorlage für Auftragnehmeraufgaben. Folgen Sie den Anweisungen unterErstellen einer Vorlage für Auftragnehmeraufgabenum eine Vorlage mit der Amazon A2I-Konsole zu erstellen. Wenn Sie die Vorlage erstellen, wählen SieTextrakt-Form-ExtraktionzumVorlagentypaus. Ersetzen Sie in der Vorlage`s3_arn`mit dem Amazon S3 S3-ARN Ihres Dokuments. Fügen Sie zusätzliche Arbeiteranweisungen hinzu`<full-instructions header="Instructions"></full-instructions>`aus.

  Wenn Sie eine Vorschau der Vorlage anzeigen möchten, stellen Sie sicher, dass Ihre IAM-Rolle über die in beschriebenen Berechtigungen verfügtAktivieren der Vorschau von Vorlagen für Auftragnehmeraufgabenaus.

  Nachdem Sie Ihre Vorlage erstellt haben, notieren Sie den ARN der Worker-Aufgabenvorlage.

Sie verwenden die in erstellten RessourcenVoraussetzungenKonfigurieren des`CreateFlowDefinition`request. In dieser Anforderung geben Sie auch Aktivierungsbedingungen im JSON-Format an. Weitere Informationen zum Konfigurieren der Aktivierungsbedingungen finden Sie unterVerwenden Sie Human Loop Aktivierungsbedingungen JSON-Schema mit Amazon Textractaus.

Erstellen eines Workflows für die Prüfung durch Menschen (AWS SDK for Python (Boto3))

Um dieses Beispiel zu verwenden, ersetzen Sie die*rot*Text mit Ihren Spezifikationen und Ressourcen.

Kodieren Sie zuerst Ihre Aktivierungsbedingungen mit dem folgenden Code in ein JSON-Objekt. Dies löst eine menschliche Überprüfung aus, wenn Amazon Textract einen Konfidenzwert zurückgibt, der weniger als 99 beträgtPostanschriftund seinen Wert oder wenn es für jedes im Dokument erkannte Schlüssel-Wert-Paar einen Konfidenzwert von weniger als 90 zurückgibt. Wenn Sie das in diesem Beispiel bereitgestellte Beispieldokument verwenden, erstellen diese Aktivierungsbedingungen eine menschliche Überprüfungsaufgabe.

```
import json

humanLoopActivationConditions = json.dumps("{
                "Conditions": [
                    {
                        "ConditionType": "ImportantFormKeyConfidenceCheck",
                        "ConditionParameters": {
                            "ImportantFormKey": "Mail Address",
                            "KeyValueBlockConfidenceLessThan": 99,
                            "WordBlockConfidenceLessThan": 99
                        }
                    },
                    {
                        "ConditionType": "ImportantFormKeyConfidenceCheck",
                        "ConditionParameters": {
                            "ImportantFormKey": "*",
                            "KeyValueBlockConfidenceLessThan": 90,
                            "WordBlockConfidenceLessThan": 90
                        }
                    }
                ]
            }"
)
```

Verwenden vonhumanLoopActivationConditionsum dascreate_flow_definitionrequest. Im folgenden Beispiel wird das SDK for Python (Boto3) zum Aufrufencreate_flow_definitionIn us-west-2 AWS-Region. Es spezifiziert die Verwendung eines privaten Arbeitsteams.

```
response = client.create_flow_definition(
    FlowDefinitionName='string',
    HumanLoopRequestSource={
         'AwsManagedHumanLoopRequestSource': "AWS/Textract/AnalyzeDocument/Forms/V1"
    },
    HumanLoopActivationConfig={
        'HumanLoopActivationConditionsConfig': {
            'HumanLoopActivationConditions': humanLoopActivationConditions
        }
    },
    HumanLoopConfig={
        'WorkteamArn': "arn:aws:sagemaker:us-west-2:111122223333:workteam/private-
crowd/work-team-name",
        'HumanTaskUiArn': "arn:aws:sagemaker:us-west-2:111122223333:human-task-
ui/worker-task-template-name",
        'TaskTitle': "Add a task title",
        'TaskDescription': "Describe your task",
        'TaskCount': 1,
        'TaskAvailabilityLifetimeInSeconds': 3600,
        'TaskTimeLimitInSeconds': 86400,
        'TaskKeywords': ["Document Review", "Content Review"]
        }
    },
    OutputConfig={
        'S3OutputPath': "s3://DOC-EXAMPLE-BUCKET/prefix/",
    },
    RoleArn="arn:aws:iam::111122223333:role/role-name"
)
```

Analysieren des Dokuments

Um Amazon A2I in einen Amazon Textract Textract-Dokumentenanalyse-Workflow zu integrieren, konfigurieren SieHumanLoopConfigimAnalyzeDocumentverwenden.

In :HumanLoopConfigangeben, geben Sie Ihren Workflow für die Prüfung durch Menschen (Flow-Definition) -ARN inFlowDefinitionArnund geben Sie Ihrer menschlichen Schleife einen Namen inHumanLoopNameaus.

Analyze the Document (AWS SDK for Python (Boto3))

Im folgenden Beispiel wird das SDK for Python (Boto3) zum Aufrufen`analyze_document`in us-west-2. Ersetzen Sie das*rot, kursiv*Text mit Ihren Ressourcen. Weitere Informationen finden Sie unteranalyze_documentimAWSAPI-Referenz für SDK for Python (Boto)aus.

```
client.analyze_document(Document={'S3Object': {"Bucket": "DOC-EXAMPLE-BUCKET",
 "Name": "document-name.png"}},
        HumanLoopConfig={"FlowDefinitionArn":"arn:aws:sagemaker:us-
west-2:111122223333:flow-definition/flow-definition-name",
                         "HumanLoopName":"human-loop-name",
                         "DataAttributes":{"ContentClassifiers":
["FreeOfPersonallyIdentifiableInformation"|"FreeOfAdultContent",]}},
        FeatureTypes=["FORMS"])
```

Analyze the Document (AWS CLI)

Im folgenden Beispiel wird verwendetAWSCLI zum Anrufen`analyze_document`aus. Diese Beispiele sind kompatibel mitAWSCLI Version 2. Die erste ist die Kurzschriftensyntax, die zweite in der JSON-Syntax. Weitere Informationen finden Sie unterAnalyse-DokumentimAWS CLIBefehlsreferenzaus.

```
aws textract analyze-document \
     --document '{"S3Object":{"Bucket":"bucket_name","Name":"file_name"}}' \
     --human-loop-config
 HumanLoopName="test",FlowDefinitionArn="arn:aws:sagemaker:eu-west-1:xyz:flow-
definition/
hl_name",DataAttributes='{ContentClassifiers=["FreeOfPersonallyIdentifiableInformation","Fre
     --feature-types '["FORMS"]'
```

```
aws textract analyze-document \
     --document '{"S3Object":{"Bucket":"bucket_name","Name":"file_name"}}' \
     --human-loop-config \
          '{"HumanLoopName":"test","FlowDefinitionArn":"arn:aws:sagemaker:eu-
west-1:xyz:flow-definition/hl_name","DataAttributes": {"ContentClassifiers":
["FreeOfPersonallyIdentifiableInformation","FreeOfAdultContent"]}}' \
     --feature-types '["FORMS"]'
```

> **Note**
>
> Vermeiden Sie Leerzeichen in Ihrem —human-loop-config -Parameter, da dies zu Verarbeitungsproblemen für Ihren Code führen kann.

Die Antwort auf diese Anfrage enthältHumanLoopActivationOutput, was angibt, ob eine menschliche Schleife erstellt wurde und ob ja, warum. Wenn eine menschliche Schleife erstellt wurde, enthält dieses Objekt auch`HumanLoopArn`aus.

Weitere Informationen zu und Beispielen unter Verwendung der`AnalyzeDocument`Betrieb, sieheAnalysieren von Dokumenttext mit Amazon Textractaus.

Überwachen von Human L

Mit der Amazon A2I-Konsole und API können Sie Details zu Ihrer menschlichen Schleife anzeigen und im Falle eines Fehlers eine aktive menschliche Schleife stoppen.

Anzeigen von Human Loop

Sie können Ihren Human Loop-Status in der Amazon A2I-Konsole anzeigen und indem Sie dieAmazon A2I Runtime-APIaus.

So finden Sie Details zu Ihrem Human Loop (Konsole)

1. Öffnen Sie die Amazon A2I Konsole unterhttps://console.aws.amazon.com/a2iZugriff auf denArbeitsabläufe für menschliche Überprüfungangezeigten.
2. Wählen Sie den Workflow für die Prüfung durch Menschen, den Sie auch zur Konfiguration verwendet haben`HumanLoopConfig`in`AnalyzeDocument`aus.
3. In derMenschliche Schleifenwählen, wählen Sie die Schleife für die Prüfung durch Menschen aus, deren Details Sie anzeigen möchten.

So finden Sie Details zu Ihrer Human Loop (API):

Verwenden des Amazon A2I`DescribeHumanLoop`verwenden. Geben Sie den Namen der menschlichen Schleife an, den Sie benutzt haben`AnalyzeDocument`aus.

Aufrufe im FollowingSDK for Python (Boto3)`describe_human_loop`aus.

```
response = client.describe_human_loop(HumanLoopName="human-loop-name")
```

Stoppen einer Human L

Nachdem eine Schleife für Menschen gestartet wurde, können Sie sie mit der Amazon A2I-Konsole und -API beenden.

So stoppen Sie Ihre menschliche Schleife (Konsole)

1. Öffnen Sie die Amazon A2I Konsole unterhttps://console.aws.amazon.com/a2iZugriff auf denArbeitsabläufe für menschliche Überprüfungangezeigten.
2. Wählen Sie den Workflow für die Prüfung durch Menschen, den Sie verwendet haben, um`HumanLoopConfig`im`AnalyzeDocument`verwenden.
3. In derMenschliche SchleifenWählen Sie die menschliche Schleife aus, die Sie anhalten möchten.
4. Wählen Sie Stop (Anhalten) aus.

So stoppen Sie Ihre menschliche Schleife (API)

Verwenden des Amazon A2I`StopHumanLoop`verwenden. Geben Sie den Namen der Human Loop an, die Sie aufgerufen haben`AnalyzeDocument`aus.

Aufrufen des folgenden SDK for Python (Boto3) -Beispielaufrufe`stop_human_loop`aus.

```
response = client.stop_human_loop(HumanLoopName="human-loop-name")
```

Ausgabedaten und Worker-Metriken anzeigen

Wenn eine menschliche Überprüfungsaufgabe von einem Mitarbeiter abgeschlossen wird, speichert Amazon A2I Ihre Ausgabedaten im Amazon S3 S3-Bucket, den Sie in Ihrem Arbeitsablauf für menschliche Überprüfung angegeben haben.

Wenn Sie eine private Belegschaft verwenden, enthalten Ihre Ausgabedaten Worker-Metadaten, mit denen Sie einzelne Mitarbeiteraktivitäten verfolgen können.

Finden von Ausgabespeicherdaten in Amazon S3

Amazon A2I verwendet Ihren Namen des Arbeitsablaufs für die menschliche Überprüfung als Präfix für den Namen der Datei, die die Ausgabedaten von menschlichen Schleifen speichert, die mit diesem Human Review Workflow erstellt wurden.

Der Pfad zu einer Human-Loop-Ausgabe verwendet das folgende Muster, in dem*YYYY*/*MM*/*DD*/*hh*/*mm*/*ss*repräsentiert das Erstellungsdatum der menschlichen Schleife mit Jahr (YYYY), Monat (MM), und Tag (DD) und die Erstellungszeit mit Stunde (hh), Minute (mm) und zweitens (ss) enthalten.

```
s3://output-bucket-specified-in-flow-definition/flow-definition-
name/YYYY/MM/DD/hh/mm/ss/human-loop-name/output.json
```

Verwenden Sie die Amazon A2I-Konsole, um die Ausgabe für eine menschliche Schleife anzuzeigen.

So sehen Sie Human Loop-Ausgabe

1. Öffnen Sie die Amazon A2I Konsole unterhttps://console.aws.amazon.com/a2iZugriff auf denArbeitsabläufe für menschliche Überprüfungangezeigten.
2. Wählen Sie den Workflow für die Prüfung durch Menschen, den Sie zum Konfigurieren verwendenHumanLoopConfiginAnalyzeDocumentaus.
3. In derMenschliche Schleifenwählen Sie die menschliche Schleife aus, deren Ausgabe Sie überprüfen möchten.
4. UNDEROutput-StandortWählen Sie die Verknüpfung zu den Ausgabedaten aus.

Verfolgen der Aktivität privater Mitarbeiter

Wenn Sie eine private Belegschaft für menschliche Überprüfungsaufgaben verwenden, enthalten die Ausgabedaten folgende Informationen über den Mitarbeiter, der die Überprüfung abgeschlossen hat:

- Das Tool `workerId`.
- In `workerMetadata`:
  - `identityProviderType`— Der Service, der zur Verwaltung der privaten Arbeitskräfte verwendet wird.
  - `issuer`— Der Amazon Cognito Cognito-Benutzerpool oder OIDC Identity Provider (IdP) - Aussteller, der mit dem Arbeitsteam verknüpft ist, das dieser Aufgabe für die Prüfung durch Menschen zugeordnet ist.
  - `sub`— Eine eindeutige Kennung, die sich auf den Worker bezieht. Wenn Sie eine Belegschaft mit Amazon Cognito erstellt haben, können Sie mithilfe dieser ID mithilfe von Amazon Cognito Details zu diesem Worker (z. B. Namen oder Benutzernamen) abrufen. Weitere Informationen

erhalten Sie unterVerwalten von und Suchen nach BenutzerkontenimAmazon Cognito-Entwicklerhandbuchaus.

Im Folgenden finden Sie ein Beispiel für die Ausgabe, die Sie sehen könnten, wenn Sie Amazon Cognito zum Erstellen einer privaten Belegschaft verwendet haben.

```
"workerId": "a12b3cdefg4h5i67",
            "workerMetadata": {
                "identityData": {
                    "identityProviderType": "Cognito",
                    "issuer": "https://cognito-idp.aws-region.amazonaws.com/aws-
region_123456789",
                    "sub": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee"
```

Im Folgenden finden Sie ein Beispiel für die Ausgabe, die Sie sehen könnten, wenn Sie Ihren eigenen OIDC-IdP verwendet haben, um eine private Belegschaft zu erstellen:

```
"workerId": "a12b3cdefg4h5i67",
            "workerMetadata": {
                "identityData": {
                    "identityProviderType": "Oidc",
                    "issuer": "https://example-oidc-ipd.com/adfs",
                    "sub": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee"
```

Weitere Informationen zum Einsatz privater Arbeitskräfte finden Sie unterVerwenden von privaten ArbeitskräftenimAmazon SageMaker Entwicklerhandbuchaus.

Sicherheit in Amazon Textract

Cloud-Sicherheit hat bei AWS höchste Priorität. Als AWS-Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die eingerichtet wurde, um die Anforderungen der anspruchsvollsten Organisationen in puncto Sicherheit zu erfüllen.

In den folgenden Themen wird beschrieben, wie Ihre Amazon Textract Textract-Ressourcen geschützt werden.

Themen

- Datenschutz in Amazon Textract
- Identity and Access Management für Amazon Textract
- Protokollieren und überwachen
- Protokollieren von Amazon-Textract-API-Aufrufen mitAWS CloudTrail
- Compliance-Validierung für Amazon Textract
- Ausfallsicherheit in Amazon Textract
- Infrastruktursicherheit in Amazon Textract
- Konfigurations- und Schwachstellenanalyse in Amazon Textract
- Amazon Textract und Schnittstellen-VPC-Endpunkte (AWS PrivateLink)

Datenschutz in Amazon Textract

Amazon Textract entspricht demAWS Modell der geteilten VerantwortungDies umfasst Vorschriften und Richtlinien für den Datenschutz.AWSist zuständig für den Schutz der globalen Infrastruktur, die alleAWS-Services.AWSbehält die Kontrolle über die Daten, die in dieser Infrastruktur gehostet werden, einschließlich der Steuerelemente zur Sicherheitskonfiguration für den Umgang mit Kundeninhalten und persönlichen Daten.AWSKunden undAPNPartner, die entweder als Datenverantwortliche oder als Datenverarbeiter agieren, sind für alle personenbezogenen Daten verantwortlich, die sie in derAWSCloud.

Wir empfehlen aus Gründen des Datenschutzes, dass Sie AWS-Anmeldeinformationen schützen und die Benutzerkonten jeweils mit AWS Identity and Access Management (IAM) einrichten. Auf diese Weise erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem sollten Sie die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Factor Authentication (MFA).
- Verwenden Sie SSL/TLS für die Kommunikation mit AWS-Ressourcen.
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit ein AWS CloudTrail.
- Verwenden Sie AWS-Verschlüsselungslösungen zusammen mit allen Standardsicherheitskontrollen in AWS-Services.
- Verwenden Sie erweiterte verwaltete Sicherheitsservices wie Amazon Macie, die dabei helfen, in Amazon S3 gespeicherte persönliche Daten zu erkennen und zu sichern.

Wir empfehlen dringend, in Freitextfeldern wie z. B. im Feld Name keine sensiblen, identifizierenden Informationen wie Kontonummern von Kunden einzugeben. Dies gilt auch, wenn Sie mit Amazon Textract oder anderen arbeitenAWSDienste, die die Konsole verwenden, API,AWS CLI, oderAWSSDKs. Alle Daten, die Sie in Amazon Textract oder andere Services eingeben, können in Diagnoseprotokolle aufgenommen werden. Wenn Sie eine URL für einen externen Server bereitstellen, schließen Sie keine Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL ein.

Weitere Informationen zum Datenschutz enthält der Blog-Beitrag AWS Shared Responsibility Model and GDPR im AWS-Sicherheitsblog.

Verschlüsselung in Amazon Textract

Datenverschlüsselung bezieht sich auf den Schutz von Daten während der Übertragung und im Ruhezustand. Sie können Ihre Daten schützen, indem Sie Amazon S3-Managed Keys oderAWS KMS keyin Ruhe, neben der Standard-Transportschichtsicherheit während des Transports.

Verschlüsselung im Ruhezustand

Die primäre Methode zur Verschlüsselung von Daten in Amazon Textract ist die serverseitige Verschlüsselung. Von Amazon S3-Buckets übergebene Eingabedokumente werden von Amazon S3 verschlüsselt und beim Zugriff entschlüsselt. Wenn Sie Ihre Anforderung authentifizieren und Zugriffsberechtigungen besitzen, gibt es in Bezug auf die Art und Weise, wie Sie auf verschlüsselte oder nicht verschlüsselte Objekte zugreifen, keinen Unterschied. Wenn Sie beispielsweise Ihre Objekte unter Verwendung einer vorsignierten URL teilen, verhält sich die URL für verschlüsselte und unverschlüsselte Objekte gleich. Wenn Sie außerdem Objekte in Ihrem -Bucket auflisten, müssen Sie`List`Die API gibt eine Liste aller Objekte zurück, unabhängig davon, ob sie verschlüsselt sind.

Amazon Textract verwendet zwei gegenseitig ausschließende Methoden der serverseitigen Verschlüsselung

Serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)

Wenn Sie eine serverseitige Verschlüsselung mit Amazon S3 verwalteten Schlüsseln (SSE-S3) verwenden, wird jedes Objekt wird mit einem eindeutigen Schlüssel verschlüsselt. Als zusätzlicher Schutz wird der eigentliche Schlüssel mit einem Master-Schlüssel verschlüsselt, der regelmäßig rotiert wird. Die serverseitige Amazon S3-Verschlüsselung verwendet eine der stärksten Blockverschlüsselungen, die verfügbar sind, 256-bit Advanced Encryption Standard (AES-256), um Ihre Daten zu verschlüsseln. Weitere Informationen finden Sie unter Protecting Data Using Server-Side Encryption with Amazon S3-Managed Encryption Keys (SSE-S3).

Serverseitige Verschlüsselung mit KMS-Schlüsseln, die im AWS Key Management Service (SSE-KMS) gespeichert sind

Serverseitige Verschlüsselung mit KMS-Schlüsseln, die im AWS Key Management Service (SSE-KMS) gespeichert sind, ist mit SSE-S3 vergleichbar, jedoch mit einigen zusätzlichen Vorteilen und Kosten für die Nutzung dieses Services. Es gibt separate Berechtigungen für die Verwendung eines KMS-Schlüssels, das einen zusätzlichen Schutz vor unbefugtem Zugriff auf Ihre Objekte in Amazon S3 bietet. SSE-KMS bietet Ihnen außerdem ein Prüfprotokoll, das anzeigt, wann und von wem Ihr KMS-Schlüssel verwendet wurde. Darüber hinaus können Sie KMS-Schlüssel erstellen und verwalten oder verwendenVon AWS verwaltete SchlüsselDiese sind einzigartig für Sie, Ihren Service und Ihre Region. Weitere Informationen finden Sie unterSchützen von Daten mithilfe serverseitiger VerschlüsselungMit KMS-Schlüsseln, die im AWS Key Management Service (SSE-KMS) gespeichert sind.

Verschlüsselung während der Übertragung

Amazon Textract verwendet Transport Layer Security (TLS), um Daten zu verschlüsseln, die zwischen dem Service und dem Agenten gesendet wurden. Darüber hinaus verwendet Amazon Textract VPC-Endpunkte, um Daten zwischen den verschiedenen Microservices zu senden, die verwendet werden, wenn Amazon Textract ein Dokument verarbeitet.

Richtlinie für den Datenverkehr zwischen Netzwerken

Amazon Textract kommuniziert ausschließlich über HTTPS-Endpunkte, die in allen von Amazon Textract unterstützten Regionen unterstützt werden

Identity and Access Management für Amazon Textract

AWS Identity and Access Management (IAM) ist ein AWS-Service, mit dem ein Administrator den Zugriff auf AWS-Ressourcen sicher steuern kann. IAM-Administratoren kontrollieren, wer sein kannauthentifiziert(angemeldet) undautorisiert(haben Berechtigungen) um Amazon Textract Textract-Ressourcen zu verwenden. IAM ist ein AWS-Service, den Sie ohne zusätzliche Kosten verwenden können.

Themen

- Zielgruppe
- Authentifizierung mit Identitäten
- Verwalten des Zugriffs mit Richtlinien
- Funktionsweise von Amazon Textract mit IAM
- Beispiele für Amazon Textract identitätsbasierte -Richtlinien
- Problembehandlung bei Amazon Textract Identity and Access Access Access

Zielgruppe

Verwendung vonAWS Identity and Access Management(IAM) unterscheidet sich je nach Ihrer Arbeit in Amazon Textract.

Service-Nutzer— Wenn Sie den Amazon Textract Textract--Service verwenden, stellt Ihnen Ihr Administrator die erforderlichen Anmeldeinformationen und Berechtigungen bereit. Möglicherweise benötigen Sie für Ihre Arbeit weitere Amazon Textract Textract-Funktionen, desto mehr Berechtigungen benötigen Sie möglicherweise. Wenn Sie die Funktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anzufordern müssen. Informationen dazu, dass Sie keinen Zugriff auf eine Funktion in Amazon Textract haben, informieren Sie sich inProblembehandlung bei Amazon Textract Identity and Access Access Accessaus.

Service-Administrator— Wenn Sie in Ihrem Unternehmen die Verantwortung für Amazon Textract-Ressourcen haben, haben Sie wahrscheinlich vollständigen Zugriff auf Amazon Textract. Ihre Aufgabe besteht darin, die Amazon Textract Textract-Funktionen und -Ressourcen festzulegen, auf die Mitarbeiter zugreifen sollen. Sie müssen dann Anträge an Ihren IAM-Administrator stellen, um die Berechtigungen Ihrer Dienstnutzer zu ändern. Lesen Sie die Informationen auf dieser Seite, um

die Grundkonzepte von IAM nachzuvollziehen. Weitere Informationen dazu, wie Ihr Unternehmen IAM mit Amazon Textract verwenden kann, finden Sie unterFunktionsweise von Amazon Textract mit IAMaus.

IAM-Administrator— Wenn Sie als IAM-Administrator fungieren, sollten Sie Einzelheiten dazu kennen, wie Sie Richtlinien zur Verwaltung des Zugriffs auf Amazon Textract verfassen können. Beispiele für identitätsbasierte Amazon Textract Textract-Richtlinien, die Sie in IAM verwenden können, finden Sie unterBeispiele für Amazon Textract identitätsbasierte -Richtlinienaus.

Authentifizierung mit Identitäten

Die Authentifizierung ist die Art und Weise, wie Sie sich mit Ihren Anmeldeinformationen bei AWS anmelden. Weitere Informationen zum Anmelden mit der AWS-Managementkonsole finden Sie unter Anmelden bei der AWS-Managementkonsole als IAM-Benutzer oder Stammbenutzer im IAM-Benutzerhandbuch.

Die Authentifizierung (Anmeldung bei AWS) muss als AWS-Konto-Stammbenutzer, als IAM-Benutzer oder durch Übernahme einer IAM-Rolle erfolgen. Sie können auch die Single-Sign-on-Authentifizierung Ihres Unternehmens verwenden oder sich sogar über Google oder Facebook anmelden. In diesen Fällen hat Ihr Administrator vorher einen Identitätsverbund unter Verwendung von IAM-Rollen eingerichtet. Wenn Sie mit Anmeldeinformationen eines anderen Unternehmens auf AWS zugreifen, nehmen Sie indirekt eine Rolle an.

Um sich direkt bei der AWS-Managementkonsole anzumelden, verwenden Sie Ihr Passwort mit der E-Mail-Adresse Ihres Stammbenutzers oder den Namen Ihres IAM-Benutzers. Sie können auf AWS programmgesteuert oder mit Ihren Stamm- oder IAM-Benutzerzugriffsschlüsseln zugreifen. AWS stellt SDK- und Befehlszeilen-Tools bereit, mit denen Ihre Anforderung anhand Ihrer Anmeldeinformationen kryptografisch signiert wird. Wenn Sie keine AWS-Tools verwenden, müssen Sie die Anforderung selbst signieren. Hierzu verwenden Sie Signature Version 4, ein Protokoll für die Authentifizierung eingehender API-Anforderungen. Weitere Informationen zur Authentifizierung von Anforderungen finden Sie unter Signature Version 4-Signaturvorgang in der Allgemeinen AWS-Referenz.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise auch zusätzliche Sicherheitsinformationen angeben. AWS empfiehlt beispielsweise die Verwendung von Multi-Factor Authentication (MFA), um die Sicherheit Ihres Kontos zu verbessern. Weitere Informationen finden Sie unter Verwenden der Multi-Faktor-Authentifizierung (MFA) in AWS im IAM-Benutzerhandbuch.

AWS-Konto-Stammbenutzer

Wenn Sie ein AWS-Konto neu erstellen, enthält es zunächst nur eine einzelne Anmeldeidentität, die über Vollzugriff auf sämtliche AWS-Services und -Ressourcen im Konto verfügt. Diese Identität wird als AWS-Konto Stammbenutzer bezeichnet. Um auf den Stammbenutzer zuzugreifen, müssen Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, die zur Erstellung des Kontos verwendet wurden. Wir raten ausdrücklich davon ab, den Stammbenutzer für Alltagsaufgaben zu verwenden, auch nicht für administrative Aufgaben. Bleiben Sie stattdessen bei dem bewährten -Verfahren, den Stammbenutzer nur zu verwenden, um Ihren ersten IAM-Benutzer zu erstellen. Anschließend legen Sie die Anmeldedaten für den Stammbenutzer an einem sicheren Ort ab und verwenden sie nur, um einige Konto- und Service-Verwaltungsaufgaben durchzuführen.

IAM-Benutzer und -Gruppen

Ein IAM-Benutzer ist eine Entität in Ihrem AWS-Konto mit bestimmten Berechtigungen für eine einzelne Person oder eine einzelne Anwendung. Ein IAM-Benutzer kann langfristige Anmeldeinformationen wie Benutzername und Passwort oder einen Satz von Zugriffsschlüsseln haben. Informationen zum Generieren von Zugriffsschlüsseln finden Sie unter Verwalten von Zugriffsschlüsseln für IAM-Benutzer im IAM-Benutzerhandbuch. Achten Sie beim Generieren von Zugriffsschlüsseln für einen IAM-Benutzer darauf, dass Sie das Schlüsselpaar anzeigen und sicher speichern. Sie können den geheimen Zugriffsschlüssel später nicht wiederherstellen. Stattdessen müssen Sie ein neues Zugriffsschlüsselpaar generieren.

Eine IAM-Gruppe ist eine Identität, die eine Sammlung von IAM-Benutzern angibt. Sie können sich nicht als Gruppe anmelden. Mithilfe von Gruppen können Sie Berechtigungen für mehrere Benutzer gleichzeitig angeben. Gruppen vereinfachen die Verwaltung von Berechtigungen, wenn es zahlreiche Benutzer gibt. Sie könnten beispielsweise einer Gruppe mit dem Namen IAMAdmins Berechtigungen zum Verwalten von IAM-Ressourcen erteilen.

Benutzer unterscheiden sich von Rollen. Ein Benutzer ist einer einzigen Person oder Anwendung eindeutig zugeordnet. Eine Rolle kann von allen Personen angenommen werden, die sie benötigen. Benutzer besitzen dauerhafte Anmeldeinformationen. Rollen stellen temporäre Anmeldeinformationen bereit. Weitere Informationen finden Sie unter Erstellen eines IAM-Benutzers (anstatt einer Rolle) im IAM-Benutzerhandbuch.

IAM-Rollen

Eine IAM-Rolle ist eine Entität in Ihrem AWS-Konto mit spezifischen Berechtigungen. Sie ähnelt einem IAM-Benutzer, ist aber nicht mit einer bestimmten Person verknüpft. Sie können eine IAM-

Rolle vorübergehend in der AWS-Managementkonsole annehmen, indem Sie Rollen wechseln. Sie können eine Rolle annehmen, indem Sie eine AWS CLI oder AWS-API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter Verwenden von IAM-Rollen im IAM-Benutzerhandbuch.

IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

- Temporäre IAM-Benutzerberechtigungen – Ein IAM-Benutzer kann eine IAM-Rolle annehmen, um vorübergehend verschiedene Berechtigungen für eine bestimmte Aufgabe zu übernehmen.
- Verbundbenutzerzugriff – Anstatt einen IAM-Benutzer zu erstellen, können Sie bereits vorhandene Identitäten in Directory Service, im Benutzerverzeichnis Ihres Unternehmens oder von einem Web-Identitätsanbieter verwenden. Diese werden als verbundene Benutzer bezeichnet. AWS weist einem verbundenen Benutzer eine Rolle zu, wenn der Zugriff über einen Identitätsanbieter angefordert wird. Weitere Informationen zu verbundenen Benutzern finden Sie unter Verbundene Benutzer und Rollen im IAM-Benutzerhandbuch.
- Kontenübergreifender Zugriff – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen sind die primäre Möglichkeit zum Gewähren von kontoübergreifendem Zugriff. In einigen AWS-Services können Sie jedoch eine Richtlinie direkt an eine Ressource anfügen (anstatt eine Rolle als Proxy zu verwenden). Informationen zu den Unterschieden zwischen Rollen und ressourcenbasierten Richtlinien für den kontenübergreifenden Zugriff finden Sie unter So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien im IAM-Benutzerhandbuch.
- Serviceübergreifender Zugriff – Einige AWS-Services verwenden Funktionen in anderen AWS-Services. Wenn Sie beispielsweise einen Aufruf in einem Service tätigen, ist es üblich, dass dieser Service Anwendungen in Amazon EC2 ausführt oder Objekte in Amazon S3 speichert. Ein Service kann dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servicerolle oder mit einer serviceverknüpften Rolle tun.
  - Prinzipalberechtigungen – Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle zum Ausführen von Aktionen in AWS verwenden, gelten Sie als Prinzipal. Richtlinien erteilen einem Prinzipal Berechtigungen. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service auslösen. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Informationen dazu, ob eine Aktion zusätzliche abhängige Aktionen in einer -Richtlinie erfordert, finden Sie unterAktionen, Ressourcen und Bedingungsschlüssel für Amazon TextractimService Authorization-Referenzaus.

  - Servicerolle – Eine Servicerolle ist eine IAM-Rolle, die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle in IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.
  - Serviceverknüpfte Rolle – Eine serviceverknüpfte Rolle ist ein Typ von Servicerolle, die mit einem AWS-Service verknüpft ist. Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Serviceverknüpfte Rollen werden in Ihrem IAM-Konto angezeigt und gehören zum Service. Ein IAM-Administrator kann die Berechtigungen für serviceverknüpfte Rollen anzeigen, aber nicht bearbeiten.
- Anwendungen in Amazon EC2 – Sie können eine IAM-Rolle nutzen, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2-Instance ausgeführt werden und AWS CLI- oder AWS-API-Anforderungen durchführen. Das ist eher zu empfehlen, als Zugriffsschlüssel innerhalb der EC2-Instance zu speichern. Erstellen Sie ein Instance-Profil, das an die Instance angefügt ist, um eine AWS-Rolle einer EC2-Instance zuzuweisen und die Rolle für sämtliche Anwendungen der Instance bereitzustellen. Ein Instance-Profil enthält die Rolle und ermöglicht, dass Programme, die in der EC2-Instance ausgeführt werden, temporäre Anmeldeinformationen erhalten. Weitere Informationen finden Sie unter Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon EC2-Instances ausgeführt werden im IAM-Benutzerhandbuch.

Informationen dazu, wann Sie IAM-Rollen oder IAM-Benutzer verwenden sollten, finden Sie unter Erstellen einer IAM-Rolle (anstatt eines Benutzers) im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Für die Zugriffssteuerung in AWS erstellen Sie Richtlinien und fügen sie den IAM-Identitäten oder AWS-Ressourcen an. Eine Richtlinie ist ein Objekt in AWS, das, einer Identität oder Ressource zugeordnet, deren Berechtigungen definiert. Sie können sich als Root-Benutzer oder IAM-Benutzer anmelden oder eine IAM-Rolle übernehmen. Wenn Sie dann eine Anforderung durchführen, wertet AWS die zugehörigen identitätsbasierten oder ressourcenbasierten Richtlinien aus. Berechtigungen in den Richtlinien bestimmen, ob die Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden in AWS als JSON-Dokumente gespeichert. Weitere Informationen zu Struktur und Inhalten von JSON-Richtliniendokumenten finden Sie unter Übersicht über JSON-Richtlinien im IAM-Benutzerhandbuch.

Administratoren können mithilfe von AWS-JSON-Richtlinien festlegen, wer zum Zugriff auf was berechtigt ist. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Eine IAM-Entität (Benutzer oder Rolle) besitzt zunächst keine Berechtigungen. Anders ausgedrückt, können Benutzer standardmäßig keine Aktionen ausführen und nicht einmal ihr Passwort ändern. Um einem Benutzer die Berechtigung für eine Aktion zu erteilen, muss ein Administrator einem Benutzer eine Berechtigungsrichtlinie zuweisen. Alternativ kann der Administrator den Benutzer zu einer Gruppe hinzufügen, die über die gewünschten Berechtigungen verfügt. Wenn ein Administrator einer Gruppe Berechtigungen erteilt, erhalten alle Benutzer in dieser Gruppe diese Berechtigungen.

IAM-Richtlinien definieren Berechtigungen für eine Aktion unabhängig von der Methode, die Sie zur Ausführung der Aktion verwenden. Angenommen, es gibt eine Richtlinie, die Berechtigungen für die `iam:GetRole`-Aktion erteilt. Ein Benutzer mit dieser Richtlinie kann Benutzerinformationen über die AWS-Managementkonsole, die AWS CLI oder die AWS-API abrufen.

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien können weiter als Inline-Richtlinien oder verwaltete Richtlinien kategorisiert werden. Inline-Richtlinien sind direkt in einen einzelnen Benutzer, eine einzelne Gruppe oder eine einzelne Rolle eingebettet. Verwaltete Richtlinien sind eigenständige Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem AWS-Konto anfügen können. Verwaltete Richtlinien umfassen von AWS verwaltete und von Kunden verwaltete Richtlinien. Informationen dazu, wie Sie zwischen einer verwalteten Richtlinie und einer eingebundenen Richtlinie wählen, finden Sie unter Auswahl zwischen verwalteten und eingebundenen Richtlinien im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern.

Für die Ressource, an die die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Prinzipale können Konten, Benutzer, Rollen, Verbundbenutzer oder AWS-Services umfassen.

Ressourcenbasierte Richtlinien sind eingebundene Richtlinien, die sich in diesem Service befinden. Sie können verwaltete AWS-Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Zugriffskontrolllisten (ACLs)

Zugriffskontrolllisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Amazon S3, AWS WAF und Amazon VPC sind Beispiele für Services, die ACLs unterstützen. Weitere Informationen“ zu ACLs finden Sie unter Zugriffskontrollliste (ACL) – Übersicht (Access Control List) im Amazon-Simple-Storage-Service-Entwicklerhandbuch.

Weitere Richtlinientypen

AWS unterstützt zusätzliche, weniger häufig verwendete Richtlinientypen. Diese Richtlinientypen können die maximalen Berechtigungen festlegen, die Ihnen von den häufiger verwendeten Richtlinientypen erteilt werden können.

- Berechtigungsgrenzen – Eine Berechtigungsgrenze ist eine erweiterte Funktion, mit der Sie die maximalen Berechtigungen festlegen können, die eine identitätsbasierte Richtlinie einer IAM-Entität (IAM-Benutzer oder -Rolle) erteilen kann. Sie können eine Berechtigungsgrenze für eine Entität festlegen. Die resultierenden Berechtigungen sind eine Schnittmenge der identitätsbasierten Richtlinien der Entität und ihrer Berechtigungsgrenzen. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle im Feld `Principal` angeben, werden nicht durch Berechtigungsgrenzen eingeschränkt. Ein ausdrückliches Ablehnen in einer dieser Richtlinien setzt das Zulassen außer Kraft. Weitere Informationen über Berechtigungsgrenzen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im IAM-Benutzerhandbuch.
- Service-Kontrollrichtlinien (SCPs) – SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OE) in AWS Organizations angeben. AWS Organizations ist ein Service für die Gruppierung und zentrale Verwaltung der AWS-Konten Ihres Unternehmens. Wenn Sie innerhalb einer Organisation alle Funktionen

aktivieren, können Sie Service-Kontrollrichtlinien (SCPs) auf alle oder einzelne Ihrer Konten anwenden. Eine SCP beschränkt die Berechtigungen für Entitäten in Mitgliedskonten, einschließlich aller AWS-Konto-Stammbenutzer. Weitere Informationen über Organizations und SCPs finden Sie unter Funktionsweise von SCPs im AWS Organizations-Benutzerhandbuch.

- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter Sitzungsrichtlinien im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn mehrere auf eine Anforderung mehrere Richtlinientypen angewendet werden können, sind die entsprechenden Berechtigungen komplizierter. Informationen dazu, wie AWS die Zulässigkeit einer Anforderung ermittelt, wenn mehrere Richtlinientypen beteiligt sind, finden Sie unter Logik für die Richtlinienauswertung im IAM-Benutzerhandbuch.

Funktionsweise von Amazon Textract mit IAM

Bevor Sie mit IAM den Zugriff auf Amazon Textract verwalten können, sollten Sie sich darüber informieren, welche IAM-Funktionen Sie mit Amazon Textract verwenden können. So erhalten Sie eine allgemeine Übersicht über Amazon Textract und andereAWS-Services funktionieren mit IAM, sieheAWSServices, die mit IAM funktionierenimIAM User Guideaus.

Themen

- Amazon Textract Identitätsbasierte Richtlinien
- Amazon Textract Ressourcen-basierte Richtlinien
- Autorisierung auf der Basis von Amazon Textract Tags
- Amazon Textract IAM-Rollen

Amazon Textract Identitätsbasierte Richtlinien

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen erteilt oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen

Aktionen zugelassen oder abgelehnt werden. Amazon Textract unterstützt bestimmte Aktionen, Ressourcen und Bedingungsschlüssel. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der IAM-Referenz für JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

Aktionen

Administratoren können mithilfe von AWS-JSON-Richtlinien festlegen, wer zum Zugriff auf was berechtigt ist. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Richtlinienaktionen haben normalerweise denselben Namen wie die zugehörige AWS-API-Operation. Es gibt einige Ausnahmen, z. B. Aktionen, die nur mit Genehmigung durchgeführt werden können und für die es keinen passenden API-Vorgang gibt. Es gibt auch einige Operationen, die mehrere Aktionen in einer Richtlinie erfordern. Diese zusätzlichen Aktionen werden als abhängige Aktionen bezeichnet.

Schließen Sie Aktionen in eine Richtlinie ein, um Berechtigungen zur Durchführung der zugeordneten Operation zu erteilen.

Für asynchrone Aktionen in Amazon Textract müssen zwei Aktionsberechtigungen erteilt werden, eine für Startaktionen und eine für Aktionen abrufen. Wenn Sie einen Amazon S3 S3-Bucket zum Übergeben von Dokumenten verwenden, müssen Sie Ihrem Konto außerdem Lesezugriff gewähren.

In Amazon Textract beginnen alle Richtlinienaktionen mit: `textract:`aus. Um einem Benutzer beispielsweise die Berechtigung zum Ausführen einer Amazon Textract-Operation mit Amazon Textract zu erteilen`AnalyzeDocument`-Operation schließen Sie die`textract:AnalyzeDocument`Maßnahmen in ihrer Politik. Richtlinienanweisungen müssen entweder ein `Action`- oder ein `NotAction`-Element enthalten. Amazon Textract definiert eine eigene Gruppe von Aktionen, die Aufgaben beschreiben, die Sie mit diesem Service durchführen können.

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie folgendermaßen durch Kommas.

```
"Action": [
      "textract:action1",
      "textract:action2"
```

Sie können auch Platzhalter (*) verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen festlegen, die mit dem Wort `Describe` beginnen, einschließlich der folgenden Aktion:

```
"Action": "textract:Describe*"
```

Eine Liste der Amazon Textract -Aktionen finden Sie unterVon Amazon Textract definierte AktionenimIAM User Guideaus.

Ressourcen

Administratoren können mithilfe von AWS-JSON-Richtlinien festlegen, wer zum Zugriff auf was berechtigt ist. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf die die Aktion angewendet wird. Anweisungen müssen entweder ein `Resource`- oder ein `NotResource`-Element enthalten. Als bewährte Methode geben Sie eine Ressource mit dem zugehörigen Amazon-Ressourcennamen (ARN) an. Sie können dies für Aktionen tun, die einen bestimmten Ressourcentyp unterstützen, der als Berechtigungen auf Ressourcenebene bezeichnet wird.

Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, z. B. Auflistungsoperationen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Amazon Textract unterstützt die Angabe von Ressourcen-ARNs in einer Richtlinie nicht.

Bedingungsschlüssel

Administratoren können mithilfe von AWS-JSON-Richtlinien festlegen, wer zum Zugriff auf was berechtigt ist. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Mithilfe des Elements `Condition`(oder des Blocks `Condition`) können Sie die Bedingungen angeben, unter denen eine Anweisung wirksam ist. Das Element `Condition` ist optional. Sie können bedingte Ausdrücke erstellen, die Bedingungsoperatoren verwenden, z. B. ist gleich oder kleiner als, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt.

Wenn Sie mehrere `Condition`-Elemente in einer Anweisung oder mehrere Schlüssel in einem einzelnen `Condition`-Element angeben, wertet AWS diese mittels einer logischen AND-Operation aus. Wenn Sie mehrere Werte für einen einzelnen Bedingungsschlüssel angeben, wertet AWS die Bedingung mittels einer logischen OR-Operation aus. Alle Bedingungen müssen erfüllt werden, bevor die Berechtigungen der Anweisung gewährt werden.

Sie können auch Platzhaltervariablen verwenden, wenn Sie Bedingungen angeben. Beispielsweise können Sie einem IAM-Benutzer die Berechtigung für den Zugriff auf eine Ressource nur dann gewähren, wenn sie mit dessen IAM-Benutzernamen gekennzeichnet ist. Weitere Informationen finden Sie unter IAM-Richtlinienelemente: Variablen und Markierungen im IAM-Benutzerhandbuch.

AWS unterstützt globale Bedingungsschlüssel und servicespezifische Bedingungsschlüssel. Eine Liste aller globalen AWS-Bedingungsschlüssel finden Sie unter Globale AWS-Bedingungskontextschlüssel im IAM-Benutzerhandbuch.

Amazon Textract stellt keine servicespezifischen Bedingungsschlüssel bereit, unterstützt jedoch die Verwendung einiger globaler Bedingungsschlüssel. Für eine Liste allerAWSGlobale Bedingungsschlüssel finden Sie unterAWSGlobale -BedingungskontextschlüsselimIAM User Guideaus.

Beispiele

Beispiele für identitätsbasierte Amazon Textract Textract-Richtlinien finden Sie unterBeispiele für Amazon Textract identitätsbasierte -Richtlinienaus.

Amazon Textract Ressourcen-basierte Richtlinien

Amazon Textract unterstützt keine ressourcenbasierten Richtlinien.

Autorisierung auf der Basis von Amazon Textract Tags

Amazon Textract unterstützt das Tagging von Ressourcen und die Steuerung des Zugriffs anhand von Tags nicht.

Amazon Textract IAM-Rollen

Eine IAM-Rolle ist eine Entität in Ihrem AWS-Konto mit spezifischen Berechtigungen.

Verwenden temporärer Anmeldeinformationen mit Amazon Textract

Sie können temporäre Anmeldeinformationen verwenden, um sich über einen Verbund anzumelden, eine IAM-Rolle anzunehmen oder eine kontenübergreifende Rolle anzunehmen. Temporäre Sicherheitsanmeldeinformationen erhalten Sie durch Aufrufen von AWS STS-API-Vorgängen wie AssumeRole oder GetFederationToken.

Amazon Textract unterstützt die Verwendung temporärer Anmeldeinformationen.

Serviceverknüpfte Rollen

Serviceverknüpfte Rollen erlauben AWS-Services den Zugriff auf Ressourcen in anderen Services, um eine Aktion in Ihrem Auftrag auszuführen. Serviceverknüpfte Rollen werden in Ihrem IAM-Konto angezeigt und gehören zum Service. Ein IAM-Administrator kann die Berechtigungen für serviceverknüpfte Rollen anzeigen, aber nicht bearbeiten.

Amazon Textract unterstützt serviceverknüpfte Rollen nicht.

> **Note**
>
> Da Amazon Textract keine serviceverknüpften Rollen unterstützt, unterstützt es keine AWS-Service-Prinzipale. Weitere Informationen zu Service-Prinzipals finden Sie unterAWS-Service-PrinzipaleimIAM User Guide

Servicerollen

Diese Funktion ermöglicht einem Service das Annehmen einer Servicerolle in Ihrem Namen. Diese Rolle gewährt dem Service Zugriff auf Ressourcen in anderen Services, um eine Aktion in Ihrem Namen auszuführen. Servicerollen werden in Ihrem IAM-Konto angezeigt und gehören zum Konto. Dies bedeutet, dass ein IAM-Administrator die Berechtigungen für diese Rolle ändern kann. Dies kann jedoch die Funktionalität des Services beeinträchtigen.

Amazon Textract unterstützt Servicerollen.

Beispiele für Amazon Textract identitätsbasierte -Richtlinien

IAM-Benutzer besitzen standardmäßig keine Berechtigungen zum Erstellen oder Ändern von Amazon Textract Textract-Ressourcen. Sie können auch keine Aufgaben ausführen, die die AWS-Managementkonsole-, AWS CLI- oder AWS-API benutzen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern und Rollen die Berechtigung zum Ausführen bestimmter API-

Operationen für die angegebenen Ressourcen gewähren, die diese benötigen. Der Administrator muss diese Richtlinien anschließend den IAM-Benutzern oder -Gruppen anfügen, die diese Berechtigungen benötigen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter Erstellen von Richtlinien auf der JSON-Registerkarte im IAM-Benutzerhandbuch.

Themen

- Bewährte Methoden für Richtlinien
- Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer
- Zugriff auf synchronen Betrieb in Amazon Textract gewähren
- Zugriff auf asynchrone Vorgänge in Amazon Textract gewähren

Bewährte Methoden für Richtlinien

Identitätsbasierte Richtlinien sind sehr leistungsfähig. Sie bestimmen, ob jemand Amazon Textract Textract-Ressourcen in Ihrem Konto erstellen oder löschen oder auf sie zugreifen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Erste Schritte mitAWSVerwaltete Richtlinien— Um Amazon Textract schnell zu verwenden, verwenden SieAWSverwaltete Richtlinien, um Ihren Mitarbeitern die von ihnen benötigten Berechtigungen zu gewähren. Diese Richtlinien sind bereits in Ihrem Konto verfügbar und werden von AWS. Weitere Informationen finden Sie unter Erste Schritte zur Verwendung von Berechtigungen mit AWS von verwalteten Richtlinien im IAM-Benutzerhandbuch.
- Gewähren von geringsten Rechten – Gewähren Sie beim Erstellen benutzerdefinierter Richtlinien nur die Berechtigungen, die zum Ausführen einer Aufgabe erforderlich sind. Beginnen Sie mit einem Mindestsatz von Berechtigungen und gewähren Sie zusätzliche Berechtigungen wie erforderlich. Dies ist sicherer, als mit Berechtigungen zu beginnen, die zu weit gefasst sind, und dann später zu versuchen, sie zu begrenzen. Weitere Informationen finden Sie unter Gewähren von geringsten Rechten im IAM-Benutzerhandbuch.
- Aktivieren von MFA für sensible Vorgänge – Fordern Sie von IAM-Benutzern die Verwendung von Multi-Factor Authentication (MFA), um zusätzliche Sicherheit beim Zugriff auf sensible Ressourcen oder API-Operationen zu bieten. Weitere Informationen finden Sie unter Verwenden der Multi-Faktor-Authentifizierung (MFA) in AWS im IAM-Benutzerhandbuch.

- Verwenden von Richtlinienbedingungen für zusätzliche Sicherheit – Definieren Sie die Bedingungen, unter denen Ihre identitätsbasierten Richtlinien den Zugriff auf eine Ressource zulassen, soweit praktikabel. Beispielsweise können Sie Bedingungen schreiben, die eine Reihe von zulässigen IP-Adressen festlegen, von denen eine Anforderung stammen muss. Sie können auch Bedingungen schreiben, die Anforderungen nur innerhalb eines bestimmten Datums- oder Zeitbereichs zulassen oder die Verwendung von SSL oder MFA fordern. Weitere Informationen finden Sie unterIAM-JSON-Richtlinienelemente: BedingungimIAM User Guideaus.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie enthält Berechtigungen für die Ausführung dieser Aktion auf der Konsole oder für die programmgesteuerte Ausführung über die AWS CLI oder die AWS-API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
```

```
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Zugriff auf synchronen Betrieb in Amazon Textract gewähren

Diese Beispielrichtlinie gewährt einem IAM-Benutzer auf Ihrem IAM-Benutzer Zugriff auf die synchronen Aktionen in Amazon TextractAWSKonto.

```
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "textract:DetectDocumentText",
                "textract:AnalyzeDocument"
            ],
            "Resource": "*"
        }
    ]
```

Zugriff auf asynchrone Vorgänge in Amazon Textract gewähren

Die folgende Beispielrichtlinie gibt einen IAM-Benutzer auf IhremAWSKontozugriff auf alle asynchronen Vorgänge, die in Amazon Textract verwendet werden.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "textract:StartDocumentTextDetection",
                "textract:StartDocumentAnalysis",
                "textract:GetDocumentTextDetection",
                "textract:GetDocumentAnalysis"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

Problembehandlung bei Amazon Textract Identity and Access Access Access

Diagnostizieren und beheben Sie mithilfe der folgenden Informationen gängige Probleme, die bei der Verwendung von Amazon Textract und IAM auftreten können.

Themen

- Ich bin nicht befugt, eine Aktion in Amazon Textract auszuführen
- Ich bin nicht zur Ausführung von iam:PassRole autorisiert
- Ich möchte meine Zugriffsschlüssel anzeigen
- Ich bin Administrator und möchte anderen den Zugriff auf Amazon Textract ermöglichen
- Ich möchte Menschen außerhalb meinesAWSKonto für den Zugriff auf meine Amazon Textract Textract-Ressourcen

Ich bin nicht befugt, eine Aktion in Amazon Textract auszuführen

Wenn die AWS-Managementkonsole Ihnen mitteilt, dass Sie nicht zur Ausführung einer Aktion autorisiert sind, müssen Sie sich an Ihren Administrator wenden, um Unterstützung zu erhalten. Ihr Administrator ist die Person, die Ihnen Ihren Benutzernamen und Ihr Passwort bereitgestellt hat.

Der folgende Beispielfehler tritt auf, wenn der`mateojackson`IAM-Benutzer versucht zu laufen`DetectDocumentText`auf einem Testbild hat aber nicht`textract:DetectDocumentText`Berechtigungen

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 textract:DetectDocumentText on resource: textimage.png
```

In diesem Fall bittet Mateo seinen Administrator um die Aktualisierung seiner Richtlinien, um unter Verwendung der Aktion `textimage.png` auf die Ressource `textract:DetectDocumentText` zugreifen zu können.

Ich bin nicht zur Ausführung von iam:PassRole autorisiert

Wenn Sie die Fehlermeldung erhalten, dass Sie nicht zur Ausführung der `iam:PassRole` Aktion autorisiert sind, müssen Sie sich an Ihren Administrator wenden, um Unterstützung zu erhalten. Ihr Administrator ist die Person, die Ihnen Ihren Benutzernamen und Ihr Passwort bereitgestellt hat. Bitten Sie diese Person, Ihre Richtlinien zu aktualisieren, damit Sie eine Rolle an Amazon Textract übergeben können.

Einige AWS-Services gewähren Ihnen die Berechtigung zur Übergabe einer vorhandenen Rolle an diesen Service, statt eine neue Service-Rolle oder serviceverknüpfte Rolle erstellen zu müssen. Hierfür benötigen Sie Berechtigungen zur Übergabe der Rolle an den Service.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen`marymajor`versucht, die Konsole zu verwenden, um eine Aktion in Amazon Textract auszuführen. Um die Aktion ausführen zu können, müssen dem Service jedoch von einer Service-Rolle Berechtigungen gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In diesem Fall bittet Mary ihren Administrator um die Aktualisierung ihrer Richtlinien, um die Aktion `iam:PassRole` ausführen zu können.

Ich möchte meine Zugriffsschlüssel anzeigen

Nachdem Sie Ihre IAM-Benutzerzugriffsschlüssel erstellt haben, können Sie Ihre Zugriffsschlüssel-ID jederzeit anzeigen. Sie können Ihren geheimen Zugriffsschlüssel jedoch nicht erneut anzeigen. Wenn Sie den geheimen Zugriffsschlüssel verlieren, müssen Sie ein neues Zugriffsschlüsselpaar erstellen.

Zugriffsschlüssel bestehen aus zwei Teilen: einer Zugriffsschlüssel-ID (z. B. `AKIAIOSFODNN7EXAMPLE`) und einem geheimen Zugriffsschlüssel (z. B. `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Ähnlich wie bei Benutzernamen und Passwörtern müssen Sie die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel zusammen verwenden, um Ihre Anfragen zu authentifizieren. Verwalten Sie Ihre Zugriffsschlüssel so sicher wie Ihren Benutzernamen und Ihr Passwort.

> **Important**
>
> Geben Sie Ihre Zugriffsschlüssel nicht an Dritte weiter, auch nicht für die Suche nach Ihrer kanonischen Benutzer-ID. Wenn Sie dies tun, gewähren Sie anderen Personen möglicherweise den permanenten Zugriff auf Ihr Konto.

Während der Erstellung eines Zugriffsschlüsselpaars werden Sie aufgefordert, die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel an einem sicheren Speicherort zu speichern. Der geheime Zugriffsschlüssel ist nur zu dem Zeitpunkt verfügbar, an dem Sie ihn erstellen. Wenn Sie Ihren geheimen Zugriffsschlüssel verlieren, müssen Sie Ihrem IAM-Benutzer neue Zugriffsschlüssel hinzufügen. Sie können maximal zwei Zugriffsschlüssel besitzen. Wenn Sie bereits zwei Zugriffschlüssel besitzen, müssen Sie ein Schlüsselpaar löschen, bevor Sie ein neues erstellen. Anweisungen hierfür finden Sie unter Verwalten von Zugriffsschlüsseln im IAM-Benutzerhandbuch.

Ich bin Administrator und möchte anderen den Zugriff auf Amazon Textract ermöglichen

Um anderen Personen oder einer Anwendung Zugriff auf Amazon Textract zu gewähren, müssen Sie eine juristische Stelle von IAM (Benutzer oder Rolle) für die Person oder Anwendung erstellen, die Zugriff benötigt. Sie werden die Anmeldeinformationen für diese Einrichtung verwenden, um auf zuzugreifen AWS. Anschließend müssen Sie der Entität eine Richtlinie anfügen, die ihnen die richtigen Berechtigungen in Amazon Textract gewährt.

Informationen zum Einstieg finden Sie unter Erstellen Ihrer ersten delegierten IAM-Benutzer und -Gruppen im IAM-Benutzerhandbuch.

Ich möchte Menschen außerhalb meinesAWSKonto für den Zugriff auf meine Amazon Textract Textract-Ressourcen

Sie können eine Rolle erstellen, die Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation für den Zugriff auf Ihre Ressourcen verwenden können. Sie können angeben, welchen Personen vertraut werden darf, damit diese die Rolle übernehmen können. Im Fall von Services, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (Access Control Lists, ACLs) verwenden, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen finden Sie hier:

- Informationen dazu, ob Amazon Textract diese Funktionen unterstützt, finden Sie unterFunktionsweise von Amazon Textract mit IAMaus.
- Informationen zum Gewähren des Zugriffs auf Ihre Ressourcen für alle Ihre AWS-Konten finden Sie unter Gewähren des Zugriffs für einen IAM-Benutzer in einem anderen Ihrer AWS-Konto im IAM-Benutzerhandbuch.
- Informationen dazu, wie Sie AWS-Konten-Drittanbieter Zugriff auf Ihre Ressourcen bereitstellen, finden Sie unter Gewähren des Zugriffs auf AWS-Konten von externen Benutzern im IAM-Benutzerhandbuch.
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontenübergreifenden Zugriff finden Sie unter So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien im IAM-Benutzerhandbuch.

Protokollieren und überwachen

Verwenden Sie Amazon CloudWatch, um Amazon Textract zu überwachen. In diesem Abschnitt erfahren Sie, wie Sie die Überwachung für Amazon Textract einrichten. Es bietet auch Referenzinhalte für Amazon Textract Textract-Metriken.

Themen

- Überwachen von Amazon Textract
- CloudWatch-Metriken für Amazon Textract

Überwachen von Amazon Textract

Mit CloudWatch können Sie Metriken für individuelle Amazon Textract Textract-Operationen oder globale Amazon Textract Textract-Metriken für das Konto abrufen. Sie können Metriken verwenden, um den Zustand der Amazon Texttract-basierten Lösung zu überwachen und Alarme einzurichten, damit Sie benachrichtigt werden, falls einzelne oder mehrere Metriken einen definierten Grenzwert überschreiten. Sie können beispielsweise Metriken für die Anzahl aufgetretener Serverfehler anzeigen. Außerdem können Sie Metriken für die Anzahl von erfolgreichen spezifischen Amazon Textract Textract-Operationen anzeigen. Verwenden Sie, um Metriken anzuzeigenAmazon CloudWatch, derAWS CLIoder dasCloudWatch APIaus.

Verwenden von CloudWatch-Metriken für Amazon Textract

Um Metriken zu verwenden, müssen Sie die folgenden Informationen angeben:

- Die Metrikdimension oder keine Dimension. Eine Dimension ist ein Name-Wert-Paar, mit dem Sie eine Metrik eindeutig identifizieren. Amazon Textract verfügt über eine Dimension namensOperationaus. Es stellt Metriken für eine bestimmte Operation zur Verfügung. Wenn Sie keine Dimension angeben, gilt die Metrik für alle Amazon Textract Textract-Operationen innerhalb Ihres Kontos.
- Der Metrikname, beispielsweise `UserErrorCount`.

Sie können Überwachungsdaten für Amazon Textract mithilfe derAWS-Managementkonsole, derAWS CLIoder die CloudWatch API. Sie können die CloudWatch-API auch über eines der Amazon AWS Software Development Kits (SDKs) oder der CloudWatch-API-Tools verwenden. Die Konsole zeigt eine Reihe von Graphen an, die auf den unformatierten Daten aus der CloudWatch-API basieren. Je nach Anforderungen können Sie entweder die in der Konsole angezeigten oder die mit der API aufgerufenen Graphen verwenden.

In der folgenden Liste finden Sie einige häufige Verwendungszwecke für die Metriken. Es handelt sich dabei um Vorschläge für den Einstieg und nicht um eine umfassende Liste.

Wie gehe ich vor?	Relevante Metriken
Wie kann ich erkennen, ob meine Anwendung die maximale Anzahl an Anfragen pro Sekunde erreicht hat?	Überwachen Sie die `Sum`-Statistik der `ThrottledCount` -Metrik.
Wie überwache ich die Anforderungsfehler?	Verwenden Sie die `Sum`-Statistik der `UserErrorCount` -Metrik.
Wie finde ich die Gesamtanzahl der Anforderungen?	Verwenden Sie die `SampleCount` -Statistik der `ResponseTime` -Metrik. Dies umfasst jegliche Anfrage, die zu einem Fehler geführt hat. Um nur erfolgreiche Operationsaufrufe anzuzeigen, verwenden Sie die`SuccessfulRequestCount` -Metrik.

Wie gehe ich vor?	Relevante Metriken
Wie überwache ich die Latenz der Amazon Textract Textract-Operationsaufrufe?	Verwenden Sie die `ResponseTime` -Metrik.

Sie müssen zur Überwachung von Amazon Textract mit CloudWatch über die entsprechenden CloudWatch-Berechtigungen verfügen. Weitere Informationen finden Sie unter Authentifizierung und Zugriffssteuerung für Amazon CloudWatch.

Zugriff auf Amazon Textract -Metriken

Die folgenden Beispiele zeigen, wie Sie mithilfe der CloudWatch-Konsole, auf Amazon Textract Textract-Metriken zugreifen.AWS CLIund die CloudWatch API.

So zeigen Sie Metriken an (Konsole)

1. Öffnen Sie die CloudWatch-Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Klicken Sie aufMetriken, wähle dasAlle -Metrikenund wählen Sie dannAmazon Textractaus.
3. Klicken Sie aufNach dem Betriebund wählen Sie dann eine Metrik.

   Wählen Sie z. B.StartDocumentAnalysisMetrik, um zu messen, wie oft die asynchrone Dokumentenanalyse gestartet wurde.
4. Wählen Sie einen Wert für den Datumsbereich aus. Die Metrikanzahl, die im Graph angezeigt wird.

So zeigen Sie Metriken an, um erfolgreich**`StartDocumentAnalysis`**Operationsaufrufe, die über einen bestimmten Zeitraum getätigt wurden (CLI)

- Öffnen Sie die AWS CLI und geben Sie den folgenden Befehl ein:

```
aws cloudwatch get-metric-statistics \
    --metric-name SuccessfulRequestCount \
    --start-time 2019-02-01T00:00:00Z \
    --period 3600 \
    --end-time 2019-03-01T00:00:00Z \
    --namespace AWS/Textract \
    --dimensions Name=Operation,Value=StartDocumentAnalysis \
    --statistics Sum
```

Dieses Beispiel zeigt die erfolgreichen Aufrufe der `StartDocumentAnalysis`-Operation für einen bestimmten Zeitraum an. Weitere Informationen finden Sie unter get-metric-statistics.

So greifen Sie auf Metriken zu (CloudWatch-API)

- Rufen Sie die folgende Seite auf `GetMetricStatistics`. Weitere Informationen finden Sie im .Amazon CloudWatch CloudWatch-API-Referenzaus.

Einrichten eines Alarms

Sie können einen CloudWatch-Alarm erstellen, der eine Amazon Simple Notification Service (Amazon SNS) sendet, sobald sich der Status des Alarms ändert. Ein Alarm überwacht eine Metrik über einen bestimmten, von Ihnen definierten Zeitraum. Der Alarm führt eine oder mehrere Aktionen durch, basierend auf dem Wert der Metrik im Vergleich zu einem bestimmten Schwellenwert in einer Reihe von Zeiträumen. Die Aktion ist eine Benachrichtigung, die an ein Amazon-SNS-Thema oder eine Auto-Scaling-Richtlinie gesendet wird.

Alarme rufen nur Aktionen für nachhaltige Statusänderungen auf. CloudWatch-Alarme rufen keine Aktionen auf, nur weil sie sich in einem bestimmten Status befinden. Der Status muss sich geändert haben und für eine festgelegte Anzahl an Zeiträumen aufrechterhalten worden sein.

So richten Sie einen Alarm ein (Konsole)

1. Melden Sie sich bei AWS-Managementkonsole an und öffnen Sie die CloudWatch-Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im NavigationsbereichAlarme, und wählen SieAlarm erstellenaus. Dies öffnetCreate Alarm Wizardaus.
3. Wählen Sie Select metric (Metrik auswählen) aus.
4. In derAlle Metriken-Tab wählenTextractaus.
5. Klicken Sie aufNach Operationund wählen Sie dann eine Metrik.

   Wählen Sie z. B.StartDocumentAnalysisum einen Alarm für eine maximale Anzahl an asynchronen Dokumentanalysevorgängen einzurichten.
6. Wählen Sie die Registerkarte Graphed metrics aus.
7. Wählen Sie für Statistic (Statistik) Sum (Summe) aus.

8. Wählen Sie Select metric (Metrik auswählen) aus.
9. Geben Sie Name und Description an. Für Whenever wählen Sie >= aus und geben einen maximalen Wert Ihrer Wahl an.
10. Wenn CloudWatch Ihnen eine E-Mail senden soll, sobald der Alarmstatus erreicht ist, fürWann immer dieser Alarm:, wählenStaat ist ALARMaus. So senden Sie Alarme für ein bestehendes Amazon SNS SNS-ThemaBenachrichtigung senden an:Wählen Sie ein vorhandenes SNS-Thema. Um den Namen und die E-Mail-Adressen für eine neue E-Mail-Abonnementliste einzurichten, wählen Sie ausNeue Listeaus. CloudWatch speichert die Liste und zeigt sie im Feld an, damit Sie sie für die Einrichtung zukünftiger Alarme nutzen können.

   > **Note**
   >
   > Bei VerwendungNeue ListeUm ein neues Amazon SNS-Thema einzurichten, müssen die E-Mail-Adressen überprüft werden, bevor die gewünschten Empfänger Benachrichtigungen erhalten. Amazon SNS sendet nur dann eine E-Mail, wenn der Alarm einen Alarmzustand auslöst. Wenn es zu dieser Änderung des Alarmzustands kommt, bevor die E-Mail-Adressen überprüft wurden, erhalten die vorgesehenen Empfänger keine Benachrichtigung.

11. Wählen Sie Create Alarm (Alarm erstellen) aus.

So richten Sie einen Alarm ein (AWS CLI)

- Öffnen Sie die AWS CLI und geben Sie den folgenden Befehl ein. Ändern Sie den -Wert`alarm-actions`-Parameter, um auf ein zuvor erstelltes Amazon SNS SNS-Thema zu verweisen.

```
aws cloudwatch put-metric-alarm \
    --alarm-name StartDocumentAnalysisUserErrors \
    --alarm-description "Alarm when more than 10 StartDocumentAnalysys user errors
 occur within 5 minutes" \
    --metric-name UserErrorCount \
    --namespace AWS/Textract \
    --statistic Sum \
    --period 300 \
    --threshold 10 \
    --comparison-operator GreaterThanThreshold \
    --evaluation-periods 1 \
    --unit Count \
    --dimensions Name=Operation,Value=StartDocumentAnalysis \
```

```
    --alarm-actions arn:aws:sns:us-east-1:111111111111:alarmtopic
```

Dieses Beispiel zeigt, wie Sie einen Alarm erstellen, wenn mehr als 10 Benutzerfehler innerhalb von 5 Minuten auftreten, damit Aufrufe von`StartDocumentAnalysis`aus. Weitere Informationen finden Sie unter put-metric-alarm.

So richten Sie einen Alarm ein (CloudWatch-API)

- Rufen Sie die folgende Seite auf `PutMetricAlarm`. Weitere Informationen finden Sie unterAmazon CloudWatch CloudWatch-API-Referenzaus.

CloudWatch-Metriken für Amazon Textract

Dieser Abschnitt enthält Informationen über die Amazon CloudWatch CloudWatch-Metriken und dieOperationdimension, die für Amazon Textract verfügbar sind.

Sie können auch eine Gesamtansicht der Amazon Textract Textract-Metriken aus der Amazon Textract Textract-Konsole sehen.

CloudWatch-Metriken für Amazon Textract

In der folgenden Tabelle finden Sie eine Übersicht über die Amazon Textract Textract-Metriken.

Metrik	Beschreibung
SuccessfulRequestCount	die Anzahl erfolgreicher Anforderungen. Der Antwortcode-Bereich für eine erfolgreiche Anfrage ist 200 bis 299. Einheit: Anzahl Gültige Statistiken: `Sum,Average`
ThrottledCount	Die Anzahl der gedrosselten Anforderungen. Amazon Textract drosselt eine Anforderung, wenn mehr Anforderungen eingehen, als das Transaktionslimit pro Sekunde für das Konto erlaubt. Wenn der Grenzwert für Ihr Konto häufig überschritten wird, können Sie eine Erweiterung des Limits beantragen. Informationen zum Anfordern einer Erweiterung finden Sie unter AWS Service Limits.

Metrik	Beschreibung
	Einheit: Anzahl Gültige Statistiken: `Sum,Average`
ResponseTime	Die Zeit in Millisekunden, in der Amazon Textract die Antwort berechnet. Einheiten: 1. Anzahl für `Data Samples`-Statistiken 2. Millisekunden für die `Average` Statistik Gültige Statistiken: `Data Samples,Average` **Note** Die`ResponseTime` Die Metrik ist im Metrikbereich von Amazon Textract nicht enthalten.
ServerErrorCount	Die Anzahl von Server-Fehlern. Der Antwortcode-Bereich für eine erfolgreiche Anfrage ist 500 bis 599. Einheit: Anzahl Gültige Statistiken: `Sum,Average`
UserErrorCount	Die Anzahl der Benutzerfehler (ungültige Parameter, ungültiges Bild, keine Berechtigung und so weiter). Der Antwortcode-Bereich für einen Benutzerfehler ist 400 bis 499. Einheit: Anzahl Gültige Statistiken: `Sum,Average`

CloudWatch Dimension for Amazon Textract

Um operationsspezifische Metriken aufzurufen, verwenden Sie den `AWS/Textract`-Namespace und geben Sie eine Operationsdimension an. Weitere Informationen zu Dimensionen finden Sie unterDimensionenimAmazon CloudWatch-Benutzerhandbuchaus.

Protokollieren von Amazon-Textract-API-Aufrufen mitAWS CloudTrail

Amazon Textract ist integriert mitAWS CloudTrail, ein Service, der eine Aufzeichnung der von einem Benutzer, einer Rolle oder einer durchgeführten Aktionen bietet.AWSService in Amazon Textract. CloudTrail erfasst alle API-Aufrufe für Amazon Textract als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der Amazon Textract Textract-Konsole und Code-Aufrufe der Amazon Textract Textract-API-Operationen.

Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail-Ereignissen an einen Amazon S3 Bucket, einschließlich Ereignissen für Amazon Textract, aktivieren. Wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse in der CloudTrail-Konsole trotzdem in Ereignisverlauf anzeigen. Mit den von CloudTrail erfassten Informationen können Sie die an Amazon Textract gestellte Anfrage, die IP-Adresse, von der die Anfrage gestellt wurde, den Zeitpunkt der Anfrage und zusätzliche Details bestimmen.

Weitere Informationen zu CloudTrail finden Sie im AWS CloudTrail-Benutzerhandbuch.

Amazon Textract Informationen in CloudTrail

CloudTrail wird beim Erstellen Ihres AWS-Kontos für Sie aktiviert. Wenn eine Aktivität in Amazon Textract auftritt, wird diese Aktivität zusammen mit anderen in einem CloudTrail-Ereignis aufgezeichnetAWS-Service-Ereignisse inEreignisverlaufaus. Sie können die neusten Ereignisse in Ihr AWS-Konto herunterladen und dort suchen und anzeigen. Weitere Informationen finden Sie unter Anzeigen von Ereignissen mit dem CloudTrail-Ereignisverlauf.

Für eine fortlaufende Aufzeichnung von Ereignissen in IhremAWS-Konto, einschließlich Ereignissen für Amazon Textract, erstellen Sie einen Trail. Ein Trail ermöglicht es CloudTrail, Protokolldateien in einem Amazon-S3-Bucket bereitzustellen. Wenn Sie einen Pfad in der Konsole anlegen, gilt dieser für alle AWS-Regionen. Der Trail protokolliert Ereignisse aus allen Regionen in der AWS-Partition und stellt die Protokolldateien in dem von Ihnen angegebenen Amazon S3 Bucket bereit. Darüber

hinaus können Sie andere konfigurierenAWS-Services zur weiteren Analyse und Bekämpfung der in den CloudTrail-Protokollen erfassten Ereignisdaten. Weitere Informationen finden Sie unter:

- Übersicht zum Erstellen eines Trails
- Von CloudTrail unterstützte Dienste und Integrationen
- Konfigurieren von Amazon-SNS-Benachrichtigungen für CloudTrail
- Empfangen von CloudTrail-Protokolldateien aus mehreren Regionen und Empfangen von CloudTrail-Protokolldateien aus mehreren Konten

Alle Operationen von Amazon Textract werden von CloudTrail protokolliert und in dokumentier-API-Referenzaus. Zum Beispiel generieren Aufrufe der Aktionen `DetectDocumentText`, `AnalyzeDocument` und `GetDocumentText` Einträge in den CloudTrail-Protokolldateien.

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Anhand der Identitätsinformationen zur Benutzeridentität können Sie Folgendes bestimmen:

- Ob die Anfrage mit Stammbenutzer- oder AWS Identity and Access Management (IAM)-Anmeldeinformationen ausgeführt wurde.
- Ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen Verbundbenutzer ausgeführt wurde.
- Ob die Anforderung aus einem anderen AWS-Service gesendet wurde

Weitere Informationen finden Sie unter dem CloudTrail userIdentity-Element.

Anforderungsparameter und Antwortfelder, die nicht protokolliert werden

Aus Datenschutzgründen werden bestimmte Anforderungsparameter und Antwortfelder nicht protokolliert, z. B. Amazon S3 S3-Bucket-Namen und Dateinamen, die in Anforderungsparametern angegeben sind, werden in CloudTrail-Protokolleinträgen In einem CloudTrail-Protokoll finden Sie keine Informationen über Bildbytes, die in einer Anforderung übergeben werden. Die folgende Tabelle zeigt die Eingabeparameter und Antwortparameter, die nicht für jeden Amazon Textract Textract-Vorgang protokolliert werden.

Operation	Anfrageparameter	Antwortfelder
AnalyzeDocument	`Bytes`	Alle
DetectDocumentText	`Bytes`	Alle
StartDocumentAnalysis	Keine	Keine
GetDocumentAnalysis	Keine	Alle
StartDocumentTextDetection	Keine	Keine
GetDocumentTextDetection	Keine	Alle

Grundlagen zu Amazon Textract Textract-Protokolldateieinträgen

Ein Trail ist eine Konfiguration, durch die Ereignisse als Protokolldateien an den von Ihnen angegebenen Amazon-S3-Bucket übermittelt werden. CloudTrail-Protokolldateien können einen oder mehrere Einträge enthalten. Ein Ereignis stellt eine einzelne Anforderung aus einer beliebigen Quelle dar und enthält Informationen über die angeforderte Aktion, Datum und Uhrzeit der Operation, Anforderungsparameter usw. CloudTrail-Protokolleinträge sind kein geordnetes Stack-Trace der öffentlichen API-Aufrufe und erscheinen daher in keiner bestimmten Reihenfolge.

Das folgende Beispiel zeigt einen CloudTrail-Protokolleintrag, der die `AnalyzeDocument`-Operation demonstriert: Die Bildbytes für die Eingabe`document`und die Analyseergebnisse (`responseElements`) werden nicht protokolliert.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111111111111:user/janedoe",
        "accountId": "111111111111",
        "accessKeyId": "AIDACKCEVSQ6C2EXAMPLE",
        "userName": "janedoe"
    },
    "eventTime": "2019-04-03T23:56:31Z",
    "eventSource": "textract.amazonaws.com",
    "eventName": "AnalyzeDocument",
```

```
    "awsRegion": "us-east-1",
    "sourceIPAddress": "198.51.100.0",
    "userAgent": "",
    "requestParameters": {
        "document": {},
        "featureTypes": [
            "TABLES"
        ]
    },
    "responseElements": null,
    "requestID": "e387676b-d1f0-4ea7-85d6-f5a344052dce",
    "eventID": "c5db79ce-e4ea-4401-8517-784481d559f7",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111111111111"
}
```

Das folgende Beispiel zeigt einen CloudTrail-Protokolleintrag für`StartDocumentAnalysis`verwenden. Der Protokolleintrag enthält den Amazon S3 S3-Bucket-Namen und den Image-Dateinamen in`documentLocation`aus. Das Protokoll enthält auch die Operationsantwort.

```
{
    "Records": [
        {
            "eventVersion": "1.05",
            "userIdentity": {
                "type": "IAMUser",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111111111111:user/janedoe",
                "accountId": "11111111111",
                "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
                "userName": "janedoe"
            },
            "eventTime": "2019-04-04T01:42:24Z",
            "eventSource": "textract.amazonaws.com",
            "eventName": "StartDocumentAnalysis",
            "awsRegion": "us-east-1",
            "sourceIPAddress": "198.51.100.0",
            "userAgent": "",
            "requestParameters": {
                "documentLocation": {
                    "s3Object": {
                        "bucket": "bucket",
```

```
                        "name": "document.png"
                    }
                },
                "featureTypes": [
                    "TABLES"
                ]
            },
            "responseElements": {
                "jobId":
 "f3c718b444fa603d5d625ab967008f4b620d4650c9db8ca1cae01ef7efe51373"
            },
            "requestID": "9ae352e8-9de1-41ad-b77b-85aa348c2e82",
            "eventID": "f741bca0-c3cb-4805-82ea-baf76439deef",
            "eventType": "AwsApiCall",
            "recipientAccountId": "111111111111"
        }

    ]
}
```


Compliance-Validierung für Amazon Textract

Externe Auditoren bewerten im Rahmen von mehreren die Sicherheit und Compliance von Amazon TextractAWSCompliance-Programme Dazu gehören HIPAA, SOC, ISO und PCI.

> **Note**
>
> Wenn Sie Daten über den Textract-Dienst verarbeiten, der der PCI-DSS-Konformität unterliegen, müssen Sie Ihr Konto abmelden, indem Sie sich an den AWS Support wenden und den Ihnen zur Verfügung gestellten Prozess folgen.

Eine Liste der AWS-Services im Bereich bestimmter Compliance-Programme finden Sie unter AWS-Services im Bereich nach Compliance-Programm. Allgemeine Informationen finden Sie unter AWS-Compliance-Programme.

Sie können Auditberichte von Drittanbietern unter AWS Artifact herunterladen. Weitere Informationen finden Sie unter Herunterladen von Berichten in AWS Artifact.

Ihre Compliance-Verantwortung bei Verwendung von Amazon Textract hängt von der Vertraulichkeit der Daten, den Compliance-Zielen des Unternehmens und den geltenden Gesetzen und Vorschriften ab.AWSstellt folgende Ressourcen bereit, um Sie bei der Compliance zu unterstützen:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.
- Whitepaper zur Erstellung einer Architektur mit HIPAA-konformer Sicherheit und Compliance – In diesem Whitepaper wird beschrieben, wie Unternehmen mithilfe von AWS HIPAA-konforme Anwendungen erstellen können.
- AWS-Compliance-Ressourcen – Diese Arbeitsbücher und Leitfäden könnten für Ihre Branche und Ihren Standort relevant sein.
- Auswertung von Ressourcen mit Regeln im AWS ConfigDeveloper Guide – Der AWS Config-Service bewertet, wie gut Ihre Ressourcenkonfigurationen mit internen Praktiken, Branchenrichtlinien und Vorschriften übereinstimmen.
- AWS Security Hub CSPM— DasAWSDer -Service bietet einen umfassenden Überblick über Ihren Sicherheitsstatus innerhalbAWSaus. Die Security Hub hilft Ihnen, Ihre Einhaltung branchenweit geltender Sicherheitsstandards und bewährter Methoden zu überprüfen.

Ausfallsicherheit in Amazon Textract

Im Zentrum der globalen AWS Infrastruktur stehen die AWS-Regionen und Availability Zones (Verfügbarkeitszonen, AZs). AWS -Regionen stellen mehrere physisch getrennte und isolierte Availability Zones bereit, die über hoch redundante Netzwerke mit niedriger Latenz und hohen Durchsätzen verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Zonen ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Weitere Informationen über AWS Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

> **Note**
>
> Die regionsübergreifende Datenübermittlung ist aufgrund der Datenschutz-Grundverordnung (DSGVO) nicht gestattet.

Infrastruktursicherheit in Amazon Textract

Als verwalteter Service ist Amazon Textract durchAWSglobale Verfahren zur Gewährleistung der Netzwerksicherheit, die imAmazon Web Services: Übersicht über SicherheitsprozesseWhitepaper.

Du benutztAWSveröffentlichte API-Aufrufe, um über das Netzwerk auf Amazon Textract zuzugreifen. Kunden müssen Transport Layer Security (TLS) 1.0 oder neuer unterstützen. Wir empfehlen TLS 1.2 oder höher. Clients müssen außerdem Cipher Suites mit PFS (Perfect Forward Secrecy) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman) unterstützen. Die meisten modernen Systemen wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit AWS -Security-Token-Service (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Konfigurations- und Schwachstellenanalyse in Amazon Textract

Konfiguration und IT-Steuerung unterliegen der übergreifenden Verantwortlichkeit von AWS und Ihnen, unserem Kunden. Weitere Informationen finden Sie unter AWSModell der übergreifenden Verantwortlichkeit.

Amazon Textract und Schnittstellen-VPC-Endpunkte (AWS PrivateLink)

Sie können eine private Verbindung zwischen Ihrer VPC und Amazon Textract herstellen, indem Sie eine erstellenSchnittstellen-VPC-Endpunktaus. Schnittstellenendpunkte werden unterstützt vonAWS PrivateLink, einer Technologie, die es Ihnen ermöglicht, ohne Internet-Gateway, NAT-Gerät, VPN-Verbindung oder AWS-Direct-Connect-Verbindung privat auf Amazon Textract Textract-APIs zuzugreifen. Die Instances in Ihrer VPC benötigen für die Kommunikation mit Amazon Textract Textract-APIs keine öffentlichen IP-Adressen. Datenverkehr zwischen Ihrer VPC und Amazon Textract verlässt das AWS-Netzwerk nicht.

Jeder Schnittstellenendpunkt wird durch eine oder mehrere Elastic Network-Schnittstellen in Ihren Subnetzen dargestellt.

Weitere Informationen finden Sie unter Schnittstellen-VPC-Endpunkte (AWS-PrivateLink) im Amazon-VPC-Benutzerhandbuch.

Überlegungen zu Amazon Textract VPC-Endpunkten

Bevor Sie einen Schnittstellen-VPC-Endpunkt für Amazon Textract einrichten, stellen Sie sicher, dass Sie die Überprüfung überprüfen.Eigenschaften und Beschränkungen von Schnittstellenendpunktenim Amazon VPC User Guideaus.

Amazon Textract unterstützt Aufrufe all seiner API-Aktionen aus der VPC.

Erstellen eines Schnittstellen-VPC-Endpunkts für Amazon Textract

Sie können einen VPC-Endpunkt für den Amazon Textract Textract-Dienst mithilfe der Amazon VPC-Konsole oder derAWS Command Line Interface(AWS CLI) enthalten. Weitere Informationen finden Sie unter Erstellung eines Schnittstellenendpunkts im Amazon VPC Benutzerhandbuch.

Erstellen Sie einen VPC-Endpunkt für Amazon Textract mit dem folgenden Servicenamen:

- com.amazonaws.*Region*.textract - Zum Erstellen eines Endpunkts für die meisten Amazon Textract Textract-Vorgänge.
- com.amazonaws.*Region*.textract-fips - Zum Erstellen eines Endpunkts für Amazon Textract, der dem US-Regierungsstandard Federal Information Processing Standard (FIPS) Publication 140-2 entspricht.

Wenn Sie einen privaten DNS für den Endpunkt aktivieren, können Sie mit seinem standardmäßigen DNS-Namen für die Region API-Anforderungen an Amazon Textract senden, beispielsweise`textract.us-east-1.amazonaws.com`aus.

Weitere Informationen finden Sie unter Zugriff auf einen Service über einen Schnittstellenendpunkt im Amazon VPC Benutzerhandbuch.

Erstellen einer VPC-Endpunktrichtlinie für Amazon Textract

Sie können eine Endpunktrichtlinie an Ihren VPC-Endpunkt anhängen, der den Zugriff auf Amazon Textract steuert. Die Richtlinie gibt die folgenden Informationen an:

- Prinzipal, der die Aktionen ausführen kann
- Aktionen, die ausgeführt werden können
- Die Ressourcen, für die Aktionen ausgeführt werden können.

Weitere Informationen finden Sie unter Steuerung des Zugriffs auf Services mit VPC-Endpunkten im Amazon-VPC-Benutzerhandbuch.

Beispiel: VPC-Endpunktrichtlinie für Amazon Textract Textract-Aktionen

Im Folgenden finden Sie ein Beispiel für eine Endpunktrichtlinie für Amazon Textract. Wenn diese Richtlinie an einen Endpunkt angefügt wird, gewährt sie Zugriff auf die aufgelisteten Amazon Textract Textract-Aktionen für alle Prinzipale auf allen Ressourcen.

Diese Beispielrichtlinie ermöglicht den Zugriff auf nur die - Vorgänge`DetectDocumentText`und`AnalyzeDocument`aus. Benutzer können Amazon Textract Textract-Operationen weiterhin von außerhalb des VPC-Endpunkts aufrufen.

```
   "Statement":[
      {
         "Principal":"*",
         "Effect":"Allow",
         "Action":[
            "textract:DetectDocumentText",
            "textract:AnalyzeDocument",
         ],
         "Resource":"*"
      }
   ]
}
```

API-Referenz

In diesem Abschnitt finden Sie Informationen zu den -API-Vorgängen von Amazon Textract.

Themen

- Aktionen
- Datentypen

Aktionen

Folgende Aktionen werden unterstützt:

- AnalyzeDocument
- AnalyzeExpense
- AnalyzeID
- DetectDocumentText
- GetDocumentAnalysis
- GetDocumentTextDetection
- GetExpenseAnalysis
- StartDocumentAnalysis
- StartDocumentTextDetection
- StartExpenseAnalysis

AnalyzeDocument

Analysiert ein Eingabedokument auf Beziehungen zwischen erkannten Elementen.

Die Arten der zurückgegebenen Informationen lauten wie folgt:

- Formulardaten (Schlüssel-Wert-Paare). Die zugehörigen Informationen werden in zweiBlockObjekte, jeweils vom TypKEY_VALUE_SET: Ein SCHLÜSSELBlockObjekt und ein VALUEBlock-Objekt. Beispiel,Name: Ana Silva Carolinaenthält einen Schlüssel und einen Wert. Name:ist der Schlüssel. Ana Silva Carolinaist der Wert.
- Tabellen- und Tabellenzellendaten. EIN TISCHBlock-Objekt enthält Informationen über eine erkannte Tabelle. EINE ZELLEBlock-Objekt wird für jede Zelle in einer Tabelle zurückgegeben.
- Textzeilen und Wörter. EINE ZEILEBlockObjekt enthält ein oder mehrereBlockObjekte. Alle Zeilen und Wörter, die im Dokument erkannt werden, werden zurückgegeben (einschließlich Text, der keine Beziehung zum Wert vonFeatureTypes) enthalten.

Auswahlelemente wie Kontrollkästchen und Optionsfelder (Optionsfelder) können in Formulardaten und in Tabellen erkannt werden. EIN SELECTION_ELEMENTBlock-Objekt enthält Informationen über ein Selektionselement, einschließlich des Auswahlstatus.

Sie können auswählen, welche Art von Analyse durchgeführt werden soll, indem Sie dieFeatureTypes-Liste.

Die Ausgabe wird in einer Liste vonBlockObjekte.

AnalyzeDocumentist ein synchroner Vorgang. Um Dokumente asynchron zu analysieren, verwenden SieStartDocumentAnalysisaus.

Weitere Informationen finden Sie unterTextanalyse von Dokumentenaus.

Anforderungssyntax

```
{
   "Document": {
      "Bytes": blob,
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
```

```
    },
    "FeatureTypes": [ "string" ],
    "HumanLoopConfig": {
       "DataAttributes": {
          "ContentClassifiers": [ "string" ]
       },
       "FlowDefinitionArn": "string",
       "HumanLoopName": "string"
    }
}
```

Anfrageparameter

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

Document

Das Eingabedokument als Base64-codierte Bytes oder ein Amazon S3 S3-Objekt. Wenn Sie die AWS CLI verwenden, um Amazon Textract Textract-Vorgänge aufzurufen, können Sie keine Bildbytes übergeben. Das Dokument muss ein Bild im JPEG-, PNG-, PDF- oder TIFF-Format sein.

Wenn Sie ein AWS SDK zum Aufrufen von Amazon Textract verwenden, müssen Sie möglicherweise keine Base64-Codierungsbytes codieren, die mit der`Bytes`field.

Typ: Document Objekt

Erforderlich Ja

FeatureTypes

Eine Liste der durchzuführenden Analysetypen. Fügen Sie TABLES zur Liste hinzu, um Informationen zu den Tabellen zurückzugeben, die im Eingabedokument erkannt werden. Fügen Sie FORMS hinzu, um erkannte Formulardaten zurückzugeben Um beide Analysetypen durchzuführen, fügen Sie TABLES und FORMS hinzu`FeatureTypes`aus. Alle im Dokument erkannten Zeilen und Wörter sind in der Antwort enthalten (einschließlich Text, der nicht mit dem Wert von`FeatureTypes`) enthalten.

Type: Zeichenfolgen-Array

Zulässige Werte: `TABLES | FORMS`

Erforderlich Ja

HumanLoopConfig

Legt die Konfiguration für den Menschen im Loop-Workflow zur Analyse von Dokumenten fest.

Typ: HumanLoopConfig Objekt

Erforderlich Nein

Antwortsyntax

```
{
   "AnalyzeDocumentModelVersion": "string",
   "Blocks": [
      {
         "BlockType": "string",
         "ColumnIndex": number,
         "ColumnSpan": number,
         "Confidence": number,
         "EntityTypes": [ "string" ],
         "Geometry": {
            "BoundingBox": {
               "Height": number,
               "Left": number,
               "Top": number,
               "Width": number
            },
            "Polygon": [
               {
                  "X": number,
                  "Y": number
               }
            ]
         },
         "Id": "string",
         "Page": number,
         "Relationships": [
            {
               "Ids": [ "string" ],
               "Type": "string"
            }
         ],
         "RowIndex": number,
         "RowSpan": number,
```

```
            "SelectionStatus": "string",
            "Text": "string",
            "TextType": "string"
         }
      ],
      "DocumentMetadata": {
         "Pages": number
      },
      "HumanLoopActivationOutput": {
         "HumanLoopActivationConditionsEvaluationResults": "string",
         "HumanLoopActivationReasons": [ "string" ],
         "HumanLoopArn": "string"
      }
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

AnalyzeDocumentModelVersion

Die Version des für die Analyse des Dokuments verwendeten Modells.

Type: String (Zeichenfolge)

Blocks

Die Elemente, die erkannt und analysiert werden von`AnalyzeDocument`aus.

Type: Array vonBlockObjekte

DocumentMetadata

Metadaten über das analysierte Dokument. Ein Beispiel ist die Anzahl der Seiten.

Typ: DocumentMetadata Objekt

HumanLoopActivationOutput

Zeigt die Ergebnisse des Menschen in der Schleifenauswertung an.

Typ: HumanLoopActivationOutput Objekt

Fehler

AccessDeniedException

Sie sind nicht berechtigt, die Aktion auszuführen. Verwenden Sie den Amazon-Ressourcennamen (ARN) der IAM-Rolle oder eines autorisierten Benutzers, um den Vorgang auszuführen.

HTTP-Statuscode: 400

BadDocumentException

Amazon Textract kann das Dokument nicht lesen. Weitere Informationen zu den Beleglimits in Amazon Textract finden Sie unterHard Limits in Amazon Textractaus.

HTTP-Statuscode: 400

DocumentTooLargeException

Das Dokument kann nicht verarbeitet werden, da es zu groß ist. Die maximale Dokumentgröße für synchrone Operationen 10 MB. Die maximale Dokumentgröße für asynchrone Vorgänge beträgt 500 MB für PDF-Dateien.

HTTP-Statuscode: 400

HumanLoopQuotaExceededException

Zeigt an, dass Sie die maximal zulässige Anzahl an aktiven Menschen in den verfügbaren Loop-Workflows überschritten haben

HTTP-Statuscode: 400

InternalServerError

Amazon Textract hat ein Service-Problem festgestellt. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

InvalidParameterException

Ein Eingabeparameter verletzt eine Beschränkung. Zum Beispiel wird bei synchronen Operationen ein`InvalidParameterException`Ausnahme tritt auf, wenn keiner der`S3Object`oder`Bytes`Werte werden im`Document`Anforderungsparameter. Validieren Sie den Parameter, bevor Sie die API-Operation erneut aufrufen.

HTTP-Statuscode: 400

InvalidS3ObjectException

Amazon Textract kann auf das in der Anforderung angegebene S3-Objekt nicht zugreifen.Konfigurieren des Zugriffs auf Amazon S3Informationen zur Problembehebung finden Sie unterFehlerbehebung für Amazon S3

HTTP-Statuscode: 400

ProvisionedThroughputExceededException

Die Anzahl der Anforderungen hat das Durchsatzlimit überschritten. Wenn Sie dieses Limit erhöhen müssen, wenden Sie sich an Amazon Textract.

HTTP-Statuscode: 400

ThrottlingException

Amazon Textract kann die Anforderung vorübergehend nicht verarbeiten. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

UnsupportedDocumentException

Das Format des Eingabedokuments wird nicht unterstützt. Dokumente für Operationen können im PNG-, JPEG-, PDF- oder TIFF-Format vorliegen.

HTTP-Statuscode: 400

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-Befehlszeilenschnittstelle
- AWS-SDK für .NET
- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS-SDK für JavaScript
- AWS SDK für PHP V3

- AWS SDK für Python
- AWS SDK für Ruby V3

AnalyzeExpense

`AnalyzeExpense`analysiert synchron ein Eingabedokument auf finanziell verwandte Beziehungen zwischen Text.

Informationen werden zurückgegeben als`ExpenseDocuments`und zerstreut sich wie folgt.

- `LineItemGroups`- Ein Datensatz mit`LineItems`welche Informationen über die Textzeilen speichern, z. B. einen gekauften Artikel und seinen Preis auf einer Quittung.
- `SummaryFields`- Enthält alle anderen Informationen einer Quittung, wie Kopfdaten oder den Namen des Lieferanten.

Anforderungssyntax

```
{
   "Document": {
      "Bytes": blob,
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   }
}
```

Anfrageparameter

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

Document

Das Eingabedokument, entweder als Byte oder als S3-Objekt.

Sie übergeben Bild-Bytes an eine Amazon Textract Textract-API-Operation, indem Sie die`Bytes`eigentum. Verwenden Sie zum Beispiel die`Bytes`-Eigenschaft, um ein aus einem lokalen Dateisystem geladenes Dokument zu übergeben. Bildbytes, die mit dem`Bytes`Die Eigenschaft muss base64-codiert werden. Ihr Code muss möglicherweise keine Dokumentdateibytes codieren, wenn Sie ein AWS SDK zum Aufrufen von Amazon Textract Textract-API-Operationen verwenden.

Sie übergeben Bilder, die in einem S3-Bucket Amazon Textract, indem Sie die`S3Object`eigentum. Dokumente, die in einem S3-Bucket gespeichert sind, müssen nicht base64-codiert werden.

Die AWS-Region für den S3-Bucket, der das S3-Objekt enthält, muss mit der von Ihnen für Amazon Textract Textract-Operationen verwendeten AWS-Region übereinstimmen.

Wenn Sie Amazon Textract Textract-Operationen mithilfe der AWS CLI aufrufen, wird die Weitergabe von Bild-Bytes mit der Bytes -Eigenschaft nicht unterstützt. Sie müssen zuerst das Dokument auf einen Amazon-S3-Bucket hochladen und anschließend die Operation mithilfe der S3Object-Eigenschaft aufrufen.

Damit Amazon Textract ein S3-Objekt verarbeiten kann, muss der Benutzer über die Berechtigung für den Zugriff auf das S3-Objekt verfügen.

Typ: Document Objekt

Erforderlich: Ja

Antwortsyntax

```
{
   "DocumentMetadata": {
      "Pages": number
   },
   "ExpenseDocuments": [
      {
         "ExpenseIndex": number,
         "LineItemGroups": [
            {
               "LineItemGroupIndex": number,
               "LineItems": [
                  {
                     "LineItemExpenseFields": [
                        {
                           "LabelDetection": {
                              "Confidence": number,
                              "Geometry": {
                                 "BoundingBox": {
                                    "Height": number,
                                    "Left": number,
                                    "Top": number,
```

```
                                    "Width": number
                                },
                                "Polygon": [
                                    {
                                        "X": number,
                                        "Y": number
                                    }
                                ]
                            },
                            "Text": "string"
                        },
                        "PageNumber": number,
                        "Type": {
                            "Confidence": number,
                            "Text": "string"
                        },
                        "ValueDetection": {
                            "Confidence": number,
                            "Geometry": {
                                "BoundingBox": {
                                    "Height": number,
                                    "Left": number,
                                    "Top": number,
                                    "Width": number
                                },
                                "Polygon": [
                                    {
                                        "X": number,
                                        "Y": number
                                    }
                                ]
                            },
                            "Text": "string"
                        }
                    }
                ]
            }
        ]
    }
],
"SummaryFields": [
    {
        "LabelDetection": {
            "Confidence": number,
```

```
                  "Geometry": {
                     "BoundingBox": {
                        "Height": number,
                        "Left": number,
                        "Top": number,
                        "Width": number
                     },
                     "Polygon": [
                        {
                           "X": number,
                           "Y": number
                        }
                     ]
                  },
                  "Text": "string"
               },
               "PageNumber": number,
               "Type": {
                  "Confidence": number,
                  "Text": "string"
               },
               "ValueDetection": {
                  "Confidence": number,
                  "Geometry": {
                     "BoundingBox": {
                        "Height": number,
                        "Left": number,
                        "Top": number,
                        "Width": number
                     },
                     "Polygon": [
                        {
                           "X": number,
                           "Y": number
                        }
                     ]
                  },
                  "Text": "string"
               }
            }
         ]
      }
   ]
```

```
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

DocumentMetadata

Informationen über das Eingabedokument.

Typ: DocumentMetadata Objekt

ExpenseDocuments

Die von Amazon Textract festgestellten Kosten.

Type: Array vonExpenseDocumentObjekte

Fehler

AccessDeniedException

Sie sind nicht berechtigt, die Aktion auszuführen. Verwenden Sie den Amazon-Ressourcennamen (ARN) der IAM-Rolle oder eines autorisierten Benutzers, um den Vorgang auszuführen.

HTTP-Statuscode: 400

BadDocumentException

Amazon Textract kann das Dokument nicht lesen. Weitere Informationen zu den Beleglimits in Amazon Textract finden Sie unterHard Limits in Amazon Textractaus.

HTTP-Statuscode: 400

DocumentTooLargeException

Das Dokument kann nicht verarbeitet werden, da es zu groß ist. Die maximale Dokumentgröße für synchrone Operationen 10 MB. Die maximale Dokumentgröße für asynchrone Vorgänge beträgt 500 MB für PDF-Dateien.

HTTP-Statuscode: 400

InternalServerError

Amazon Textract hat ein Service-Problem festgestellt. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

InvalidParameterException

Ein Eingabeparameter verletzt eine Beschränkung. Zum Beispiel wird bei synchronen Operationen ein`InvalidParameterException`Ausnahme tritt auf, wenn keiner der`S3Object`oder`Bytes`Werte werden im`Document`Anforderungsparameter. Validieren Sie den Parameter, bevor Sie die API-Operation erneut aufrufen.

HTTP-Statuscode: 400

InvalidS3ObjectException

Amazon Textract kann auf das in der Anforderung angegebene S3-Objekt nicht zugreifen.Konfigurieren des Zugriffs auf Amazon S3Informationen zur Problembehebung finden Sie unterFehlerbehebung für Amazon S3

HTTP-Statuscode: 400

ProvisionedThroughputExceededException

Die Anzahl der Anforderungen hat das Durchsatzlimit überschritten. Wenn Sie dieses Limit erhöhen müssen, wenden Sie sich an Amazon Textract.

HTTP-Statuscode: 400

ThrottlingException

Amazon Textract kann die Anforderung vorübergehend nicht verarbeiten. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

UnsupportedDocumentException

Das Format des Eingabedokuments wird nicht unterstützt. Dokumente für Operationen können im PNG-, JPEG-, PDF- oder TIFF-Format vorliegen.

HTTP-Statuscode: 400

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-Befehlszeilenschnittstelle
- AWS-SDK für .NET
- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS-SDK für JavaScript
- AWS SDK für PHP V3
- AWS SDK für Python
- AWS SDK für Ruby V3

AnalyzeID

Analysiert Ausweisdokumente auf relevante Informationen. Diese Informationen werden extrahiert und als`IdentityDocumentFields`, das sowohl das normalisierte Feld als auch den Wert des extrahierten Textes aufzeichnet. Im Gegensatz zu anderen Amazon Textract Textract-Vorgängen`AnalyzeID`gibt keine Geometrie-Daten zurück.

Anforderungssyntax

```
{
   "DocumentPages": [
      {
         "Bytes": blob,
         "S3Object": {
            "Bucket": "string",
            "Name": "string",
            "Version": "string"
         }
      }
   ]
}
```

Anfrageparameter

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

DocumentPages

Das Dokument wird an AnalyeID übergeben.

Type: Array-Reihe[Document](#)Objekte

Array-Mitglieder: Die Mindestanzahl beträgt 1 Element. Maximale Anzahl von 2 Elementen.

Erforderlich: Ja

Antwortsyntax

```
{
   "AnalyzeIDModelVersion": "string",
   "DocumentMetadata": {
```

```
      "Pages": number
   },
   "IdentityDocuments": [
      {
         "DocumentIndex": number,
         "IdentityDocumentFields": [
            {
               "Type": {
                  "Confidence": number,
                  "NormalizedValue": {
                     "Value": "string",
                     "ValueType": "string"
                  },
                  "Text": "string"
               },
               "ValueDetection": {
                  "Confidence": number,
                  "NormalizedValue": {
                     "Value": "string",
                     "ValueType": "string"
                  },
                  "Text": "string"
               }
            }
         ]
      }
   ]
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

AnalyzeIDModelVersion

Die Version der AnalyeIdentity API, die zur Verarbeitung von Dokumenten verwendet wird.

Type: String (Zeichenfolge)

DocumentMetadata

Informationen über das Eingabedokument.

Typ: DocumentMetadata Objekt

IdentityDocuments

Die Liste der von AnalyeID verarbeiteten Dokumente. Schließt eine Zahl ein, die ihren Platz in der Liste und die Antwortstruktur für das Dokument angibt.

Type: Array-ReiheIdentityDocumentObjekte

Fehler

AccessDeniedException

Sie sind nicht berechtigt, die Aktion auszuführen. Verwenden Sie den Amazon-Ressourcennamen (ARN) der IAM-Rolle oder eines autorisierten Benutzers, um den Vorgang auszuführen.

HTTP-Statuscode: 400

BadDocumentException

Amazon Textract kann das Dokument nicht lesen. Weitere Informationen zu den Beleglimits in Amazon Textract finden Sie unterHard Limits in Amazon Textractaus.

HTTP-Statuscode: 400

DocumentTooLargeException

Das Dokument kann nicht verarbeitet werden, da es zu groß ist. Die maximale Dokumentgröße für synchrone Operationen 10 MB. Die maximale Dokumentgröße für asynchrone Vorgänge beträgt 500 MB für PDF-Dateien.

HTTP-Statuscode: 400

InternalServerError

Amazon Textract hat ein Service-Problem festgestellt. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

InvalidParameterException

Ein Eingabeparameter verletzt eine Beschränkung. Zum Beispiel wird bei synchronen Operationen ein`InvalidParameterException`Ausnahme tritt auf, wenn keiner der`S3Object`oder`Bytes`Werte werden im`Document`Anforderungsparameter. Validieren Sie den Parameter, bevor Sie die API-Operation erneut aufrufen.

HTTP-Statuscode: 400

InvalidS3ObjectException

Amazon Textract kann auf das in der Anforderung angegebene S3-Objekt nicht zugreifen.Konfigurieren des Zugriffs auf Amazon S3Informationen zur Problembehebung finden Sie unterFehlerbehebung für Amazon S3

HTTP-Statuscode: 400

ProvisionedThroughputExceededException

Die Anzahl der Anforderungen hat das Durchsatzlimit überschritten. Wenn Sie dieses Limit erhöhen müssen, wenden Sie sich an Amazon Textract.

HTTP-Statuscode: 400

ThrottlingException

Amazon Textract kann die Anforderung vorübergehend nicht verarbeiten. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

UnsupportedDocumentException

Das Format des Eingabedokuments wird nicht unterstützt. Dokumente für Operationen können im PNG-, JPEG-, PDF- oder TIFF-Format vorliegen.

HTTP-Statuscode: 400

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-Befehlszeilenschnittstelle
- AWS-SDK für .NET
- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS-SDK für JavaScript

- AWS SDK für PHP V3
- AWS SDK für Python
- AWS SDK für Ruby V3

DetectDocumentText

Erkennt Text im Eingabedokument. Amazon Textract kann Textzeilen und Wörter erkennen, aus denen eine Textzeile besteht. Das Eingabedokument muss ein Bild im JPEG-, PNG-, PDF- oder TIFF-Format sein.`DetectDocumentText`gibt den erkannten Text in einem Array vonBlock-Objekte.

Jede Dokumentseite ist als verknüpft`Block`vom Typ PAGE. Jede SEITE`Block`object ist das übergeordnete Element von LINE`Block`-Objekte, die die Zeilen des erkannten Textes auf einer Seite darstellen. EINE ZEILE`Block`object ist ein übergeordnetes Element für jedes Wort, aus dem die Zeile besteht. Wörter werden dargestellt durch`Block`Objekte des Typs WORD.

`DetectDocumentText`ist ein synchroner Vorgang. Um Dokumente asynchron zu analysieren, verwenden SieStartDocumentTextDetectionaus.

Weitere Informationen finden Sie unterTexterkennung von Dokumentenaus.

Anforderungssyntax

```
{
   "Document": {
      "Bytes": blob,
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   }
}
```

Anfrageparameter

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

Document

Das Eingabedokument als Base64-codierte Bytes oder ein Amazon S3 S3-Objekt. Wenn Sie die AWS CLI verwenden, um Amazon Textract Textract-Vorgänge aufzurufen, können Sie keine Bildbytes übergeben. Das Dokument muss ein Bild im JPEG- oder PNG-Format sein.

Wenn Sie ein AWS SDK zum Aufrufen von Amazon Textract verwenden, müssen Sie möglicherweise keine Base64-Codierung von Image-Bytes, die mit dem`Bytes`field.

Typ: Document Objekt

Erforderlich: Ja

Antwortsyntax

```
{
   "Blocks": [
      {
         "BlockType": "string",
         "ColumnIndex": number,
         "ColumnSpan": number,
         "Confidence": number,
         "EntityTypes": [ "string" ],
         "Geometry": {
            "BoundingBox": {
               "Height": number,
               "Left": number,
               "Top": number,
               "Width": number
            },
            "Polygon": [
               {
                  "X": number,
                  "Y": number
               }
            ]
         },
         "Id": "string",
         "Page": number,
         "Relationships": [
            {
               "Ids": [ "string" ],
               "Type": "string"
            }
         ],
         "RowIndex": number,
         "RowSpan": number,
         "SelectionStatus": "string",
         "Text": "string",
         "TextType": "string"
      }
   ],
```

```
   "DetectDocumentTextModelVersion": "string",
   "DocumentMetadata": {
      "Pages": number
   }
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

Blocks

Ein Array von`Block`-Objekte, die den Text enthalten, der im Dokument erkannt wurde.

Type: Array-ReiheBlockObjekte

DetectDocumentTextModelVersion

Type: String (Zeichenfolge)

DocumentMetadata

Metadaten über das Dokument. Es enthält die Anzahl der Seiten, die im Dokument erkannt werden.

Typ: DocumentMetadata Objekt

Fehler

AccessDeniedException

Sie sind nicht berechtigt, die Aktion auszuführen. Verwenden Sie den Amazon-Ressourcennamen (ARN) der IAM-Rolle oder eines autorisierten Benutzers, um den Vorgang auszuführen.

HTTP-Statuscode: 400

BadDocumentException

Amazon Textract kann das Dokument nicht lesen. Weitere Informationen zu den Beleglimits in Amazon Textract finden Sie unterHard Limits in Amazon Textractaus.

HTTP-Statuscode: 400

DocumentTooLargeException

Das Dokument kann nicht verarbeitet werden, da es zu groß ist. Die maximale Dokumentgröße für synchrone Operationen 10 MB. Die maximale Dokumentgröße für asynchrone Vorgänge beträgt 500 MB für PDF-Dateien.

HTTP-Statuscode: 400

InternalServerError

Amazon Textract hat ein Service-Problem festgestellt. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

InvalidParameterException

Ein Eingabeparameter verletzt eine Beschränkung. Zum Beispiel wird bei synchronen Operationen ein`InvalidParameterException`Ausnahme tritt auf, wenn keiner der`S3Object`oder`Bytes`Werte werden im`Document`Anforderungsparameter. Validieren Sie den Parameter, bevor Sie die API-Operation erneut aufrufen.

HTTP-Statuscode: 400

InvalidS3ObjectException

Amazon Textract kann auf das in der Anforderung angegebene S3-Objekt nicht zugreifen. für weitere Informationen,Konfigurieren des Zugriffs auf Amazon S3Informationen zur Problembehebung finden Sie unterFehlerbehebung für Amazon S3

HTTP-Statuscode: 400

ProvisionedThroughputExceededException

Die Anzahl der Anforderungen hat das Durchsatzlimit überschritten. Wenn Sie dieses Limit erhöhen müssen, wenden Sie sich an Amazon Textract.

HTTP-Statuscode: 400

ThrottlingException

Amazon Textract kann die Anforderung vorübergehend nicht verarbeiten. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

UnsupportedDocumentException

Das Format des Eingabedokuments wird nicht unterstützt. Dokumente für Operationen können im PNG-, JPEG-, PDF- oder TIFF-Format vorliegen.

HTTP-Statuscode: 400

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-Befehlszeilenschnittstelle
- AWS-SDK für .NET
- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS-SDK für JavaScript
- AWS SDK für PHP V3
- AWS SDK für Python
- AWS SDK für Ruby V3

GetDocumentAnalysis

Ruft die Ergebnisse für einen asynchronen Amazon Textract Textract-Vorgang ab, der Text in einem Dokument analysiert.

Sie starten die asynchrone Textanalyse durch AufrufenStartDocumentAnalysis, die eine Job-ID zurückgibt (`JobId`) enthalten. Wenn der Textanalysevorgang abgeschlossen ist, veröffentlicht Amazon Textract einen Abschlussstatus für das Thema Amazon Simple Notification Service (Amazon SNS), das im ersten Aufruf von registriert ist`StartDocumentAnalysis`aus. Um die Ergebnisse des Texterkennungsvorgangs zu erhalten, überprüfen Sie zunächst, ob der im Amazon SNS SNS-Thema veröffentlichte Statuswert lautet`SUCCEEDED`aus. Wenn ja, ruf an`GetDocumentAnalysis`und übergeben Sie die Job-ID (`JobId`) vom ersten Anruf an`StartDocumentAnalysis`aus.

`GetDocumentAnalysis`gibt ein Array vonBlock-Objekte. Die folgenden Arten von Informationen werden zurückgegeben:

- Formulardaten (Schlüssel-Wert-Paare). Die zugehörigen Informationen werden in zweiBlockObjekte, jeweils vom Typ`KEY_VALUE_SET`: Ein SCHLÜSSEL`Block`Objekt und ein VALUE`Block`-Objekt. Beispiel,Name: Ana Silva Carolinaenthält einen Schlüssel und einen Wert. Name:ist der Schlüssel. Ana Silva Carolinaist der Wert.
- Tabellen- und Tabellenzellendaten. EIN TISCH`Block`-Objekt enthält Informationen über eine erkannte Tabelle. EINE ZELLE`Block`-Objekt wird für jede Zelle in einer Tabelle zurückgegeben.
- Textzeilen und Wörter. EINE ZEILE`Block`-Objekt enthält ein oder mehrere WORD`Block`-Objekte. Alle Zeilen und Wörter, die im Dokument erkannt werden, werden zurückgegeben (einschließlich Text, der keine Beziehung zum Wert des`StartDocumentAnalysis` `FeatureTypes`Eingabeparameter).

Auswahlelemente wie Kontrollkästchen und Optionsfelder (Optionsfelder) können in Formulardaten und in Tabellen erkannt werden. EIN SELECTION_ELEMENT`Block`-Objekt enthält Informationen über ein Selektionselement, einschließlich des Auswahlstatus.

Verwenden der`MaxResults`Parameter, um die Anzahl der zurückgegebenen Blöcke einzuschränken. Wenn es mehr Ergebnisse gibt als angegeben in`MaxResults`, der Wert von`NextToken`In der Operationsantwort enthält ein Paginierungs-Token für den Abruf des nächsten Ergebnissatzes. Um die nächste Ergebnisseite abzurufen, rufen Sie`GetDocumentAnalysis`und bevölkern`NextToken`Anforderungsparameter mit dem Token-Wert, der vom vorherigen Aufruf an`GetDocumentAnalysis`aus.

Weitere Informationen finden Sie unterTextanalyse für Dokumenteaus.

Anforderungssyntax

```
{
   "JobId": "string",
   "MaxResults": number,
   "NextToken": "string"
}
```

Anfrageparameter

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

JobId

Ein eindeutiger Bezeichner für den Text-Erkennungs-Job. Die`JobId`wird von zurückgegeben`StartDocumentAnalysis`aus. EIN`JobId`Der Wert ist nur 7 Tage lang gültig.

Type: String (Zeichenfolge)

Längenbeschränkungen: Mindestlänge 1. Höchstlänge = 64 Zeichen.

Pattern: `^[a-zA-Z0-9-_]+$`

Erforderlich Ja

MaxResults

Die maximale Anzahl der Ergebnisse, die pro paginierten Aufruf zurückgegeben werden sollen. Der größte Wert, den Sie angeben können, ist 1.000. Wenn Sie einen Wert größer als 1.000 angeben, werden maximal 1.000 Ergebnisse zurückgegeben. Der Standardwert lautet 1.000.

Type: Ganzzahl

Gültiger Bereich: Der Mindestwert ist 1.

Erforderlich Nein

NextToken

Wenn die vorherige Antwort unvollständig war (da mehr Blöcke abgerufen werden müssen), gibt Amazon Textract ein Paginierungstoken in der Antwort zurück. Sie können dieses Paginierungs-Token verwenden, um den nächsten Satz von Blöcken abzurufen.

Type: String (Zeichenfolge)

Längenbeschränkungen: Mindestlänge 1. Höchstlänge = 255 Zeichen.

Pattern: `.*\S.*`

Erforderlich Nein

Antwortsyntax

```
{
   "AnalyzeDocumentModelVersion": "string",
   "Blocks": [
      {
         "BlockType": "string",
         "ColumnIndex": number,
         "ColumnSpan": number,
         "Confidence": number,
         "EntityTypes": [ "string" ],
         "Geometry": {
            "BoundingBox": {
               "Height": number,
               "Left": number,
               "Top": number,
               "Width": number
            },
            "Polygon": [
               {
                  "X": number,
                  "Y": number
               }
            ]
         },
         "Id": "string",
         "Page": number,
         "Relationships": [
            {
               "Ids": [ "string" ],
               "Type": "string"
            }
         ],
         "RowIndex": number,
         "RowSpan": number,
```

```
            "SelectionStatus": "string",
            "Text": "string",
            "TextType": "string"
         }
      ],
      "DocumentMetadata": {
         "Pages": number
      },
      "JobStatus": "string",
      "NextToken": "string",
      "StatusMessage": "string",
      "Warnings": [
         {
            "ErrorCode": "string",
            "Pages": [ number ]
         }
      ]
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

AnalyzeDocumentModelVersion

Type: String (Zeichenfolge)

Blocks

Die Ergebnisse der Textanalyse-Operation.

Type: Array vonBlockObjekte

DocumentMetadata

Informationen über ein Dokument, das Amazon Textract verarbeitet hat.`DocumentMetadata`wird auf jeder Seite mit paginierten Antworten eines Amazon Textract Textract-Videovorgangs zurückgegeben.

Typ: DocumentMetadata Objekt

JobStatus

Der aktuelle Status des Texterkennungsauftrags.

Type: String (Zeichenfolge)

Zulässige Werte: `IN_PROGRESS | SUCCEEDED | FAILED | PARTIAL_SUCCESS`

NextToken

Wenn die Antwort abgeschnitten wird, gibt Amazon Textract dieses Token zurück. Sie können dieses Token in der nachfolgenden Anforderung verwenden, um den nächsten Satz von Texterkennungsergebnissen abzurufen.

Type: String (Zeichenfolge)

Längenbeschränkungen: Mindestlänge 1. Höchstlänge = 255 Zeichen.

Pattern: `.*\S.*`

StatusMessage

Gibt zurück, wenn der Erkennungsauftrag nicht abgeschlossen werden konnte. Enthält eine Erklärung dafür, welcher Fehler aufgetreten ist.

Type: String (Zeichenfolge)

Warnings

Eine Liste der Warnungen, die während des Dokumentenanalysevorgangs aufgetreten sind.

Type: Array vonWarningObjekte

Fehler

AccessDeniedException

Sie sind nicht berechtigt, die Aktion auszuführen. Verwenden Sie den Amazon-Ressourcennamen (ARN) der IAM-Rolle oder eines autorisierten Benutzers, um den Vorgang auszuführen.

HTTP-Statuscode: 400

InternalServerError

Amazon Textract hat ein Service-Problem festgestellt. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

InvalidJobIdException

Eine ungültige Job-ID wurde übergeben anGetDocumentAnalysisoder zuGetDocumentAnalysisaus.

HTTP-Statuscode: 400

InvalidKMSKeyException

Zeigt an, dass Sie keine Entschlüsselungsberechtigungen mit dem eingegebenen KMS-Schlüssel haben oder der KMS-Schlüssel falsch eingegeben wurde.

HTTP-Statuscode: 400

InvalidParameterException

Ein Eingabeparameter verletzt eine Beschränkung. Zum Beispiel wird bei synchronen Operationen ein`InvalidParameterException`Ausnahme tritt auf, wenn keiner der`S3Object`oder`Bytes`Werte werden im`Document`Anforderungsparameter. Validieren Sie den Parameter, bevor Sie den API-Vorgang erneut aufrufen.

HTTP-Statuscode: 400

InvalidS3ObjectException

Amazon Textract kann auf das in der Anforderung angegebene S3-Objekt nicht zugreifen. für weitere Informationen,Konfigurieren des Zugriffs auf Amazon S3Informationen zur Problembehebung finden Sie unterFehlerbehebung für Amazon S3

HTTP-Statuscode: 400

ProvisionedThroughputExceededException

Die Anzahl der Anforderungen hat das Durchsatzlimit überschritten. Wenn Sie dieses Limit erhöhen müssen, wenden Sie sich an Amazon Textract.

HTTP-Statuscode: 400

ThrottlingException

Amazon Textract kann die Anforderung vorübergehend nicht verarbeiten. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-Befehlszeilenschnittstelle
- AWS-SDK für .NET
- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS-SDK für JavaScript
- AWS SDK für PHP V3
- AWS SDK für Python
- AWS SDK für Ruby V3

GetDocumentTextDetection

Ruft die Ergebnisse für einen asynchronen Amazon Textract Textract-Vorgang ab, der Text in einem Dokument erkennt. Amazon Textract kann Textzeilen und Wörter erkennen, aus denen eine Textzeile besteht.

Sie starten die asynchrone Texterkennung durch AufrufenStartDocumentTextDetection, die eine Job-ID zurückgibt (`JobId`) enthalten. Wenn der Vorgang zur Texterkennung abgeschlossen ist, veröffentlicht Amazon Textract einen Abschlussstatus für das Thema Amazon Simple Notification Service (Amazon SNS), das im ersten Aufruf von registriert ist`StartDocumentTextDetection`aus. Um die Ergebnisse des Texterkennungsvorgangs zu erhalten, überprüfen Sie zunächst, ob der im Amazon SNS SNS-Thema veröffentlichte Statuswert lautet`SUCCEEDED`aus. Wenn ja, ruf an`GetDocumentTextDetection`und übergeben Sie die Job-ID (`JobId`) vom ersten Anruf an`StartDocumentTextDetection`aus.

`GetDocumentTextDetection`gibt ein Array vonBlock-Objekte.

Jede Dokumentseite ist als verknüpft`Block`vom Typ PAGE. Jede SEITE`Block`Objekt ist das übergeordnete Element von LINE`Block`-Objekte, die die Zeilen des erkannten Textes auf einer Seite darstellen. EINE ZEILE`Block`object ist ein übergeordnetes Element für jedes Wort, aus dem die Zeile besteht. Wörter werden dargestellt durch`Block`Objekte vom Typ WORD.

Verwenden Sie den MaxResults-Parameter, um die Anzahl der zurückgegebenen Blöcke einzuschränken. Wenn es mehr Ergebnisse gibt als in angegeben`MaxResults`, der Wert von`NextToken`In der Betriebsantwort enthält ein Paginierungs-Token für den Abruf des nächsten Ergebnissatzes. Um die nächste Ergebnisseite abzurufen, rufen Sie`GetDocumentTextDetection`und bevölkern`NextToken`Anforderungsparameter mit dem Token-Wert, der vom vorherigen Aufruf an zurückgegeben wurde`GetDocumentTextDetection`aus.

Weitere Informationen finden Sie unterErkennung von Text-Dokumentenaus.

Anforderungssyntax

```
{
   "JobId": "string",
   "MaxResults": number,
   "NextToken": "string"
}
```

Anfrageparameter

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

JobId

Ein eindeutiger Bezeichner für den Texterkennungsauftrag. Die`JobId`wird zurückgegeben von`StartDocumentTextDetection`aus. EIN`JobId`Der Wert ist nur 7 Tage lang gültig.

Type: String (Zeichenfolge)

Einschränkungen für die Länge: Mindestlänge 1. Höchstlänge = 64 Zeichen.

Pattern: `^[a-zA-Z0-9-_]+$`

Erforderlich: Ja

MaxResults

Die maximale Anzahl der Ergebnisse, die pro paginierten Aufruf zurückgegeben werden sollen. Der größte Wert, den Sie angeben können, ist 1.000. Wenn Sie einen größeren Wert als 1.000 angeben, wird die maximale Anzahl von 1.000 Ergebnissen zurückgegeben. Der Standardwert lautet 1.000.

Type: Ganzzahl

Gültiger Bereich: Der Mindestwert ist 1.

Erforderlich: Nein

NextToken

Wenn die vorherige Antwort unvollständig war (da mehr Blöcke abgerufen werden müssen), gibt Amazon Textract ein Paginierungstoken in der Antwort zurück. Sie können dieses Paginierungs-Token verwenden, um den nächsten Satz von Blöcken abzurufen.

Type: String (Zeichenfolge)

Einschränkungen für die Länge: Mindestlänge 1. Höchstlänge = 255 Zeichen.

Pattern: `.*\S.*`

Erforderlich: Nein

Antwortsyntax

```
{
   "Blocks": [
      {
         "BlockType": "string",
         "ColumnIndex": number,
         "ColumnSpan": number,
         "Confidence": number,
         "EntityTypes": [ "string" ],
         "Geometry": {
            "BoundingBox": {
               "Height": number,
               "Left": number,
               "Top": number,
               "Width": number
            },
            "Polygon": [
               {
                  "X": number,
                  "Y": number
               }
            ]
         },
         "Id": "string",
         "Page": number,
         "Relationships": [
            {
               "Ids": [ "string" ],
               "Type": "string"
            }
         ],
         "RowIndex": number,
         "RowSpan": number,
         "SelectionStatus": "string",
         "Text": "string",
         "TextType": "string"
      }
   ],
   "DetectDocumentTextModelVersion": "string",
   "DocumentMetadata": {
      "Pages": number
   },
```

```
   "JobStatus": "string",
   "NextToken": "string",
   "StatusMessage": "string",
   "Warnings": [
      {
         "ErrorCode": "string",
         "Pages": [ number ]
      }
   ]
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

Blocks

Die Ergebnisse des Texterkennungsvorgangs.

Type: Array vonBlockObjekte

DetectDocumentTextModelVersion

Type: String (Zeichenfolge)

DocumentMetadata

Informationen über ein Dokument, das Amazon Textract verarbeitet hat.`DocumentMetadata`wird auf jeder Seite mit paginierten Antworten eines Amazon Textract Textract-Videovorgangs zurückgegeben.

Typ: DocumentMetadata Objekt

JobStatus

Der aktuelle Status des Texterkennungsauftrags.

Type: String (Zeichenfolge)

Zulässige Werte: `IN_PROGRESS | SUCCEEDED | FAILED | PARTIAL_SUCCESS`

NextToken

Wenn die Antwort abgeschnitten wird, gibt Amazon Textract dieses Token zurück. Sie können dieses Token in der nachfolgenden Anforderung verwenden, um den nächsten Satz von Texterkennungs-Ergebnissen abzurufen.

Type: String (Zeichenfolge)

Einschränkungen für die Länge: Mindestlänge 1. Höchstlänge = 255 Zeichen.

Pattern: `.*\S.*`

StatusMessage

Gibt zurück, wenn der Erkennungsauftrag nicht abgeschlossen werden konnte. Enthält eine Erklärung dafür, welcher Fehler aufgetreten ist.

Type: String (Zeichenfolge)

Warnings

Eine Liste der Warnungen, die während des Texterkennungsvorgangs für das Dokument aufgetreten sind.

Type: Array vonWarningObjekte

Fehler

AccessDeniedException

Sie sind nicht berechtigt, die Aktion auszuführen. Verwenden Sie den Amazon-Ressourcennamen (ARN) der IAM-Rolle oder eines autorisierten Benutzers, um den Vorgang auszuführen.

HTTP-Statuscode: 400

InternalServerError

Amazon Textract hat ein Service-Problem festgestellt. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

InvalidJobIdException

Eine ungültige Job-ID wurde übergeben anGetDocumentAnalysisoder zuGetDocumentAnalysisaus.

HTTP-Statuscode: 400

InvalidKMSKeyException

Zeigt an, dass Sie keine Entschlüsselungsberechtigungen mit dem eingegebenen KMS-Schlüssel haben oder der KMS-Schlüssel falsch eingegeben wurde.

HTTP-Statuscode: 400

InvalidParameterException

Ein Eingabeparameter verletzt eine Beschränkung. Zum Beispiel wird bei synchronen Operationen ein`InvalidParameterException`Ausnahme tritt auf, wenn keiner der`S3Object`oder`Bytes`Werte werden im`Document`Anforderungsparameter. Validieren Sie den Parameter, bevor Sie den API-Vorgang erneut aufrufen.

HTTP-Statuscode: 400

InvalidS3ObjectException

Amazon Textract kann auf das in der Anforderung angegebene S3-Objekt nicht zugreifen.Konfigurieren des Zugriffs auf Amazon S3Informationen zur Problembehebung finden Sie unterFehlerbehebung für Amazon S3

HTTP-Statuscode: 400

ProvisionedThroughputExceededException

Die Anzahl der Anforderungen hat das Durchsatzlimit überschritten. Wenn Sie dieses Limit erhöhen müssen, wenden Sie sich an Amazon Textract.

HTTP-Statuscode: 400

ThrottlingException

Amazon Textract kann die Anforderung vorübergehend nicht verarbeiten. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-Befehlszeilenschnittstelle
- AWS-SDK für .NET
- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS-SDK für JavaScript
- AWS SDK für PHP V3
- AWS SDK für Python
- AWS SDK für Ruby V3

GetExpenseAnalysis

Ruft die Ergebnisse für einen asynchronen Amazon Textract Textract-Vorgang ab, der Rechnungen und Belege analysiert. Amazon Textract findet Kontaktinformationen, gekaufte Artikel und den Namen des Lieferanten aus Eingaberechnungen und Quittungen.

Sie starten die asynchrone Rechnungs-/Beleganalyse, indem Sie anrufenStartExpenseAnalysis, die eine Job-ID zurückgibt (`JobId`) enthalten. Nach Abschluss der Rechnungs-/Belegungsanalyse veröffentlicht Amazon Textract den Erledigungsstatus im Amazon Simple Notification Service (Amazon SNS) -Thema. Dieses Thema muss im ersten Aufruf von`StartExpenseAnalysis`aus. Um die Ergebnisse des Rechnungs-/Beleganalysevorgangs zu erhalten, stellen Sie zunächst sicher, dass der für das Amazon SNS-Thema veröffentlichte Status Wert lautet`SUCCEEDED`aus. Wenn ja, ruf an`GetExpenseAnalysis`und übergeben Sie die Job-ID (`JobId`) vom ersten Anruf an`StartExpenseAnalysis`aus.

Verwenden Sie den MaxResults-Parameter, um die Anzahl der zurückgegebenen Blöcke einzuschränken. Wenn es mehr Ergebnisse gibt als angegeben in`MaxResults`, der Wert von`NextToken`In der Operationsantwort enthält ein Paginierungs-Token für den Abruf des nächsten Ergebnissatzes. Um die nächste Ergebnisseite abzurufen, rufen Sie`GetExpenseAnalysis`und bevölkern Sie das`NextToken`Anforderungsparameter mit dem Token-Wert, der vom vorherigen Aufruf an`GetExpenseAnalysis`aus.

Weitere Informationen finden Sie unterRechnungen und Belege analysierenaus.

Anforderungssyntax

```
{
   "JobId": "string",
   "MaxResults": number,
   "NextToken": "string"
}
```

Anfrageparameter

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

JobId

Ein eindeutiger Bezeichner für den Texterkennungsauftrag. Die`JobId`wird zurückgegeben`StartExpenseAnalysis`aus. EIN`JobId`Wert ist nur 7 Tage lang gültig.

Type: String (Zeichenfolge)

Längenbeschränkungen: Mindestlänge 1. Höchstlänge = 64 Zeichen.

Pattern: `^[a-zA-Z0-9-_]+$`

Erforderlich: Ja

MaxResults

Die maximale Anzahl der Ergebnisse, die pro paginierten Aufruf zurückgegeben werden sollen. Der größte Wert, den Sie angeben können, ist 20. Wenn Sie einen Wert größer als 20 angeben, werden maximal 20 Ergebnisse zurückgegeben. Der Standardwert ist 20.

Type: Ganzzahl

Gültiger Bereich: Der Mindestwert ist 1.

Erforderlich: Nein

NextToken

Wenn die vorherige Antwort unvollständig war (da mehr Blöcke abgerufen werden müssen), gibt Amazon Textract ein Paginierungstoken in der Antwort zurück. Sie können dieses Paginierungs-Token verwenden, um den nächsten Satz von Blöcken abzurufen.

Type: String (Zeichenfolge)

Längenbeschränkungen: Mindestlänge 1. Höchstlänge = 255 Zeichen.

Pattern: `.*\S.*`

Erforderlich: Nein

Antwortsyntax

```
{
   "AnalyzeExpenseModelVersion": "string",
   "DocumentMetadata": {
      "Pages": number
   },
   "ExpenseDocuments": [
      {
         "ExpenseIndex": number,
         "LineItemGroups": [
```

```
            {
               "LineItemGroupIndex": number,
               "LineItems": [
                  {
                     "LineItemExpenseFields": [
                        {
                           "LabelDetection": {
                              "Confidence": number,
                              "Geometry": {
                                 "BoundingBox": {
                                    "Height": number,
                                    "Left": number,
                                    "Top": number,
                                    "Width": number
                                 },
                                 "Polygon": [
                                    {
                                       "X": number,
                                       "Y": number
                                    }
                                 ]
                              },
                              "Text": "string"
                           },
                           "PageNumber": number,
                           "Type": {
                              "Confidence": number,
                              "Text": "string"
                           },
                           "ValueDetection": {
                              "Confidence": number,
                              "Geometry": {
                                 "BoundingBox": {
                                    "Height": number,
                                    "Left": number,
                                    "Top": number,
                                    "Width": number
                                 },
                                 "Polygon": [
                                    {
                                       "X": number,
                                       "Y": number
                                    }
                                 ]
```

```
                        },
                        "Text": "string"
                     }
                  }
               ]
            }
         ]
      }
   ],
   "SummaryFields": [
      {
         "LabelDetection": {
            "Confidence": number,
            "Geometry": {
               "BoundingBox": {
                  "Height": number,
                  "Left": number,
                  "Top": number,
                  "Width": number
               },
               "Polygon": [
                  {
                     "X": number,
                     "Y": number
                  }
               ]
            },
            "Text": "string"
         },
         "PageNumber": number,
         "Type": {
            "Confidence": number,
            "Text": "string"
         },
         "ValueDetection": {
            "Confidence": number,
            "Geometry": {
               "BoundingBox": {
                  "Height": number,
                  "Left": number,
                  "Top": number,
                  "Width": number
               },
               "Polygon": [
```

```
                        {
                           "X": number,
                           "Y": number
                        }
                     ]
                  },
                  "Text": "string"
               }
            }
         ]
      }
   ],
   "JobStatus": "string",
   "NextToken": "string",
   "StatusMessage": "string",
   "Warnings": [
      {
         "ErrorCode": "string",
         "Pages": [ number ]
      }
   ]
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

AnalyzeExpenseModelVersion

Die aktuelle Modellversion von AnalyzeExpense.

Type: String (Zeichenfolge)

DocumentMetadata

Informationen über ein Dokument, das Amazon Textract verarbeitet hat.`DocumentMetadata`wird auf jeder Seite mit paginierten Antworten eines Amazon Textract Textract-Vorgangs zurückgegeben.

Typ: DocumentMetadata Objekt

ExpenseDocuments

Die von Amazon Textract festgestellten Kosten.

Type: Array-AnordnungExpenseDocumentObjekte

JobStatus

Der aktuelle Status des Texterkennungsauftrags.

Type: String (Zeichenfolge)

Zulässige Werte: `IN_PROGRESS | SUCCEEDED | FAILED | PARTIAL_SUCCESS`

NextToken

Wenn die Antwort abgeschnitten wird, gibt Amazon Textract dieses Token zurück. Sie können dieses Token in der nachfolgenden Anforderung verwenden, um den nächsten Satz von Ergebnissen zur Texterkennung abzurufen.

Type: String (Zeichenfolge)

Längenbeschränkungen: Mindestlänge 1. Höchstlänge = 255 Zeichen.

Pattern: `.*\S.*`

StatusMessage

Gibt zurück, wenn der Erkennungsauftrag nicht abgeschlossen werden konnte. Enthält eine Erklärung dafür, welcher Fehler aufgetreten ist.

Type: String (Zeichenfolge)

Warnings

Eine Liste der Warnungen, die während des Texterkennungsvorgangs für das Dokument aufgetreten sind.

Type: Array-AnordnungWarningObjekte

Fehler

AccessDeniedException

Sie sind nicht berechtigt, die Aktion auszuführen. Verwenden Sie den Amazon-Ressourcennamen (ARN) der IAM-Rolle oder eines autorisierten Benutzers, um den Vorgang auszuführen.

HTTP-Statuscode: 400

InternalServerError

Amazon Textract hat ein Service-Problem festgestellt. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

InvalidJobIdException

Eine ungültige Job-ID wurde übergeben anGetDocumentAnalysisoder zuGetDocumentAnalysisaus.

HTTP-Statuscode: 400

InvalidKMSKeyException

Zeigt an, dass Sie keine Entschlüsselungsberechtigungen mit dem eingegebenen KMS-Schlüssel haben oder der KMS-Schlüssel falsch eingegeben wurde.

HTTP-Statuscode: 400

InvalidParameterException

Ein Eingabeparameter verletzt eine Beschränkung. Zum Beispiel wird bei synchronen Operationen ein`InvalidParameterException`Ausnahme tritt auf, wenn keiner der`S3Object`oder`Bytes`Werte werden im`Document`Anforderungsparameter. Validieren Sie den Parameter, bevor Sie den API-Vorgang erneut aufrufen.

HTTP-Statuscode: 400

InvalidS3ObjectException

Amazon Textract kann auf das in der Anforderung angegebene S3-Objekt nicht zugreifen. für weitere Informationen,Konfigurieren des Zugriffs auf Amazon S3Informationen zur Problembehebung finden Sie unterFehlerbehebung für Amazon S3

HTTP-Statuscode: 400

ProvisionedThroughputExceededException

Die Anzahl der Anforderungen hat das Durchsatzlimit überschritten. Wenn Sie dieses Limit erhöhen müssen, wenden Sie sich an Amazon Textract.

HTTP-Statuscode: 400

ThrottlingException

Amazon Textract kann die Anforderung vorübergehend nicht verarbeiten. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-Befehlszeilenschnittstelle
- AWS-SDK für .NET
- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS-SDK für JavaScript
- AWS SDK für PHP V3
- AWS SDK für Python
- AWS SDK für Ruby V3

StartDocumentAnalysis

Startet die asynchrone Analyse eines Eingabedokuments für Beziehungen zwischen erkannten Elementen wie Schlüssel-Wert-Paare, Tabellen und Selektionselemente.

`StartDocumentAnalysis`kann Text in Dokumenten analysieren, die im JPEG-, PNG-, TIFF- und PDF-Format vorliegen. Die Dokumente werden in einem Amazon S3 S3-Bucket gespeichert. Verwenden vonDocumentLocationum den -Bucket-Namen und den Dateinamen des Dokuments anzugeben.

`StartDocumentAnalysis`gibt eine Job-ID zurück (`JobId`) die Sie verwenden, um die Ergebnisse der Operation zu erhalten. Wenn die Textanalyse abgeschlossen ist, veröffentlicht Amazon Textract einen Abschlussstatus im Amazon Simple Notification Service (Amazon SNS) -Thema, das Sie in angeben`NotificationChannel`aus. Um die Ergebnisse des Textanalysevorgangs zu erhalten, überprüfen Sie zunächst, ob der im Amazon SNS SNS-Thema veröffentlichte Statuswert lautet`SUCCEEDED`aus. Wenn ja, ruf anGetDocumentAnalysisund übergeben Sie die Job-ID (`JobId`) vom ersten Anruf an`StartDocumentAnalysis`aus.

Weitere Informationen finden Sie unterTextanalyse von Dokumentenaus.

Anforderungssyntax

```
{
   "ClientRequestToken": "string",
   "DocumentLocation": {
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   },
   "FeatureTypes": [ "string" ],
   "JobTag": "string",
   "KMSKeyId": "string",
   "NotificationChannel": {
      "RoleArn": "string",
      "SNSTopicArn": "string"
   },
   "OutputConfig": {
      "S3Bucket": "string",
      "S3Prefix": "string"
   }
```

```
}
```

Anfrageparameter

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

ClientRequestToken

Das idempotent-Token, mit dem Sie die Startanfrage identifizieren. Wenn Sie dasselbe Token mit mehreren verwenden`StartDocumentAnalysis`Anfragen, das gleiche`JobId`wird zurückgegeben. Verwenden von`ClientRequestToken`um zu verhindern, dass derselbe Job versehentlich mehr als einmal gestartet wird. Weitere Informationen finden Sie unterAsynchrone Operationen von Amazon Textract aufrufenaus.

Type: String (Zeichenfolge)

Einschränkungen der Länge: Mindestlänge 1. Höchstlänge = 64 Zeichen.

Pattern: `^[a-zA-Z0-9-_]+$`

Erforderlich: Nein

DocumentLocation

Der Speicherort des zu verarbeitenden Dokuments.

Typ: DocumentLocation Objekt

Erforderlich: Ja

FeatureTypes

Eine Liste der durchzuführenden Analysetypen. Fügen Sie TABLES zur Liste hinzu, um Informationen zu den Tabellen zurückzugeben, die im Eingabedokument erkannt werden. Fügen Sie FORMS hinzu, um erkannte Formulardaten zurückzugeben Um beide Analysetypen durchzuführen, fügen Sie TABLES und FORMS hinzu`FeatureTypes`aus. Alle im Dokument erkannten Zeilen und Wörter sind in der Antwort enthalten (einschließlich Text, der nicht mit dem Wert von`FeatureTypes`) enthalten.

Type: Zeichenfolgen-Array

Zulässige Werte: `TABLES | FORMS`

Erforderlich: Ja

JobTag

Eine Kennung, die Sie angeben, die in der zum Amazon SNS SNS-Thema veröffentlichten Abschlussbenachrichtigung enthalten ist. Sie können beispielsweise die Datei verwendenJobTagum die Art des Dokuments anzugeben, dem die Abschlussmitteilung entspricht (z. B. ein Steuerformular oder eine Quittung).

Type: String (Zeichenfolge)

Einschränkungen der Länge: Mindestlänge 1. Höchstlänge = 64 Zeichen.

Pattern: `[a-zA-Z0-9_.\-:]+`

Erforderlich: Nein

KMSKeyId

Der KMS-Schlüssel, der zum Verschlüsseln der Inferenzergebnisse verwendet wird. Dies kann entweder im Key ID- oder Key Alias Format vorliegen. Wenn ein KMS-Schlüssel bereitgestellt wird, wird der KMS-Schlüssel zur serverseitigen Verschlüsselung der Objekte im Kunden-Bucket verwendet. Wenn dieser Parameter nicht aktiviert ist, wird das Ergebnis serverseitig mit SSE-S3 verschlüsselt.

Type: String (Zeichenfolge)

Einschränkungen der Länge: Mindestlänge 1. Maximale Länge beträgt 2048 Zeichen.

Pattern: `^[A-Za-z0-9][A-Za-z0-9:_/+=,@.-]{0,2048}$`

Erforderlich: Nein

NotificationChannel

Das Amazon SNS SNS-Thema ARN, in dem Amazon Textract den Abschlussstatus des Vorgangs veröffentlichen soll.

Typ: NotificationChannel Objekt

Erforderlich: Nein

OutputConfig

Legt fest, ob die Ausgabe an einen vom Kunden definierten Bucket gesendet wird. Standardmäßig speichert Amazon Textract die Ergebnisse intern, auf die der GetDocumentAnalysis Vorgang zugegriffen werden soll.

Typ: OutputConfig Objekt

Erforderlich: Nein

Antwortsyntax

```
{
   "JobId": "string"
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

JobId

Der Bezeichner für den Dokumenttexterkennungs-Auftrag. Verwenden vonJobIdum den Job in einem nachfolgenden Aufruf zu identifizierenGetDocumentAnalysisaus. EINJobIdDer Wert ist nur 7 Tage lang gültig.

Type: String (Zeichenfolge)

Einschränkungen der Länge: Mindestlänge 1. Höchstlänge = 64 Zeichen.

Pattern: `^[a-zA-Z0-9-_]+$`

Fehler

AccessDeniedException

Sie sind nicht berechtigt, die Aktion auszuführen. Verwenden Sie den Amazon-Ressourcennamen (ARN) der IAM-Rolle oder eines autorisierten Benutzers, um den Vorgang auszuführen.

HTTP-Statuscode: 400

BadDocumentException

Amazon Textract kann das Dokument nicht lesen. Weitere Informationen zu den Beleglimits in Amazon Textract finden Sie unterHard Limits in Amazon Textractaus.

HTTP-Statuscode: 400

DocumentTooLargeException

Das Dokument kann nicht verarbeitet werden, da es zu groß ist. Die maximale Dokumentgröße für synchrone Operationen 10 MB. Die maximale Dokumentgröße für asynchrone Vorgänge beträgt 500 MB für PDF-Dateien.

HTTP-Statuscode: 400

IdempotentParameterMismatchException

EIN`ClientRequestToken`Der Eingabeparameter wurde mit einer Operation wiederverwendet, aber mindestens eine der anderen Eingabeparameter ist anders als im vorherigen Aufruf der Operation.

HTTP-Statuscode: 400

InternalServerError

Amazon Textract hat ein Service-Problem festgestellt. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

InvalidKMSKeyException

Zeigt an, dass Sie keine Entschlüsselungsberechtigungen mit dem eingegebenen KMS-Schlüssel haben oder der KMS-Schlüssel falsch eingegeben wurde.

HTTP-Statuscode: 400

InvalidParameterException

Ein Eingabeparameter verletzt eine Beschränkung. Zum Beispiel wird bei synchronen Operationen ein`InvalidParameterException`Ausnahme tritt auf, wenn keiner der`S3Object`oder`Bytes`Werte werden im`Document`Parameter anfordern. Validieren Sie den Parameter, bevor Sie die API-Operation erneut aufrufen.

HTTP-Statuscode: 400

InvalidS3ObjectException

Amazon Textract kann auf das in der Anforderung angegebene S3-Objekt nicht zugreifen.Konfigurieren des Zugriffs auf Amazon S3Informationen zur Problembehebung finden Sie unterFehlerbehebung für Amazon S3

HTTP-Statuscode: 400

LimitExceededException

Ein Amazon Textract Textract-Service-Limit wurde überschritten. Wenn Sie beispielsweise zu viele asynchrone Jobs gleichzeitig starten, rufen Sie den Betrieb auf (`StartDocumentTextDetection`Wenn Sie beispielsweise eine LimitExceededException - Ausnahme (HTTP-Statuscode: 400) lösen, bis die Anzahl der gleichzeitig ausgeführten Aufträge unter dem Amazon Textract Textract-Service-Limit liegt.

HTTP-Statuscode: 400

ProvisionedThroughputExceededException

Die Anzahl der Anforderungen hat das Durchsatzlimit überschritten. Wenn Sie dieses Limit erhöhen müssen, wenden Sie sich an Amazon Textract.

HTTP-Statuscode: 400

ThrottlingException

Amazon Textract kann die Anforderung vorübergehend nicht verarbeiten. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

UnsupportedDocumentException

Das Format des Eingabedokuments wird nicht unterstützt. Dokumente für Operationen können im PNG-, JPEG-, PDF- oder TIFF-Format vorliegen.

HTTP-Statuscode: 400

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-Befehlszeilenschnittstelle
- AWS-SDK für .NET
- AWS-SDK für C++
- AWS-SDK for Go

- AWSSDK for Java V2
- AWS-SDK für JavaScript
- AWS SDK für PHP V3
- AWS SDK für Python
- AWS SDK für Ruby V3

StartDocumentTextDetection

Startet die asynchrone Erkennung von Text in einem Dokument. Amazon Textract kann Textzeilen und Wörter erkennen, aus denen eine Textzeile besteht.

`StartDocumentTextDetection`kann Text in Dokumenten analysieren, die im JPEG-, PNG-, TIFF- und PDF-Format vorliegen. Die Dokumente werden in einem Amazon S3 S3-Bucket gespeichert. Verwenden vonDocumentLocationum den -Bucket-Namen und den Dateinamen des Dokuments anzugeben.

`StartTextDetection`gibt eine Job-ID zurück (`JobId`), die Sie verwenden, um die Ergebnisse der Operation zu erhalten. Wenn die Texterkennung abgeschlossen ist, veröffentlicht Amazon Textract einen Abschlussstatus im Amazon Simple Notification Service (Amazon SNS) -Thema, das Sie in`NotificationChannel`aus. Um die Ergebnisse des Texterkennungsvorgangs zu erhalten, überprüfen Sie zunächst, ob der im Amazon SNS SNS-Thema veröffentlichte Statuswert lautet`SUCCEEDED`aus. Wenn ja, ruf anGetDocumentTextDetectionund übergeben Sie die Job-ID (`JobId`) vom ersten Anruf an`StartDocumentTextDetection`aus.

Weitere Informationen finden Sie unterTexterkennung von Dokumentenaus.

Anforderungssyntax

```
{
   "ClientRequestToken": "string",
   "DocumentLocation": {
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   },
   "JobTag": "string",
   "KMSKeyId": "string",
   "NotificationChannel": {
      "RoleArn": "string",
      "SNSTopicArn": "string"
   },
   "OutputConfig": {
      "S3Bucket": "string",
      "S3Prefix": "string"
   }
```

```
}
```

Anfrageparameter

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

ClientRequestToken

Das idempotente Token, das verwendet wird, um die Startanfrage zu identifizieren. Wenn Sie dasselbe Token mit mehreren verwenden`StartDocumentTextDetection`Anfragen, das gleiche`JobId`wird zurückgegeben. Verwenden von`ClientRequestToken`um zu verhindern, dass derselbe Job versehentlich mehr als einmal gestartet wird. Weitere Informationen finden Sie unterAsynchrone Operationen von Amazon Textract aufrufenaus.

Type: String (Zeichenfolge)

Einschränkungen der Länge: Mindestlänge 1. Höchstlänge = 64 Zeichen.

Pattern: `^[a-zA-Z0-9-_]+$`

: Erforderlich Nein

DocumentLocation

Der Speicherort des zu verarbeitenden Dokuments.

Typ: DocumentLocation Objekt

: Erforderlich Ja

JobTag

Eine Kennung, die Sie angeben, die in der zum Amazon SNS SNS-Thema veröffentlichten Abschlussbenachrichtigung enthalten ist. Sie können beispielsweise die Datei verwenden`JobTag`um die Art des Dokuments anzugeben, dem die Abschlussmitteilung entspricht (z. B. ein Steuerformular oder eine Quittung).

Type: String (Zeichenfolge)

Einschränkungen der Länge: Mindestlänge 1. Höchstlänge = 64 Zeichen.

Pattern: `[a-zA-Z0-9_.\-:]+`

: Erforderlich Nein

KMSKeyId

Der KMS-Schlüssel, der zum Verschlüsseln der Inferenzergebnisse verwendet wird. Dies kann entweder im Key ID- oder Key Alias Format vorliegen. Wenn ein KMS-Schlüssel bereitgestellt wird, wird der KMS-Schlüssel zur serverseitigen Verschlüsselung der Objekte im Kunden-Bucket verwendet. Wenn dieser Parameter nicht aktiviert ist, wird das Ergebnis serverseitig mit SSE-S3 verschlüsselt.

Type: String (Zeichenfolge)

Einschränkungen der Länge: Mindestlänge 1. Maximale Länge beträgt 2048 Zeichen.

Pattern: `^[A-Za-z0-9][A-Za-z0-9:_/+=,@.-]{0,2048}$`

: Erforderlich Nein

NotificationChannel

Das Amazon SNS SNS-Thema ARN, in dem Amazon Textract den Abschlussstatus des Vorgangs veröffentlichen soll.

Typ: NotificationChannel Objekt

: Erforderlich Nein

OutputConfig

Legt fest, ob die Ausgabe an einen vom Kunden definierten Bucket gesendet wird. Standardmäßig speichert Amazon Textract die Ergebnisse intern, auf die mit dem GetDocumentTextDetection - Vorgang zugegriffen werden soll.

Typ: OutputConfig Objekt

: Erforderlich Nein

Antwortsyntax

```
{
   "JobId": "string"
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

JobId

Die Kennung des Texterkennungsauftrags für das Dokument. Verwenden von`JobId`um den Job in einem nachfolgenden Aufruf zu identifizieren`GetDocumentTextDetection`aus. EIN`JobId`Wert ist nur 7 Tage lang gültig.

Type: String (Zeichenfolge)

Einschränkungen der Länge: Mindestlänge 1. Höchstlänge = 64 Zeichen.

Pattern: `^[a-zA-Z0-9-_]+$`

Fehler

AccessDeniedException

Sie sind nicht berechtigt, die Aktion auszuführen. Verwenden Sie den Amazon-Ressourcennamen (ARN) der IAM-Rolle oder eines autorisierten Benutzers, um den Vorgang auszuführen.

HTTP-Statuscode: 400

BadDocumentException

Amazon Textract kann das Dokument nicht lesen. Weitere Informationen zu den Beleglimits in Amazon Textract finden Sie unterHard Limits in Amazon Textractaus.

HTTP-Statuscode: 400

DocumentTooLargeException

Das Dokument kann nicht verarbeitet werden, da es zu groß ist. Die maximale Dokumentgröße für synchrone Operationen 10 MB. Die maximale Dokumentgröße für asynchrone Vorgänge beträgt 500 MB für PDF-Dateien.

HTTP-Statuscode: 400

IdempotentParameterMismatchException

EINClientRequestTokenDer Eingabeparameter wurde mit einer Operation wiederverwendet, aber mindestens eine der anderen Eingabeparameter ist anders als im vorherigen Aufruf der Operation.

HTTP-Statuscode: 400

InternalServerError

Amazon Textract hat ein Service-Problem festgestellt. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

InvalidKMSKeyException

Zeigt an, dass Sie keine Entschlüsselungsberechtigungen mit dem eingegebenen KMS-Schlüssel haben oder der KMS-Schlüssel falsch eingegeben wurde.

HTTP-Statuscode: 400

InvalidParameterException

Ein Eingabeparameter verletzt eine Beschränkung. Zum Beispiel wird bei synchronen Operationen ein`InvalidParameterException`Ausnahme tritt auf, wenn keiner der`S3Object`oder`Bytes`Werte werden im`Document`Anforderungsparameter. Validieren Sie den Parameter, bevor Sie die API-Operation erneut aufrufen.

HTTP-Statuscode: 400

InvalidS3ObjectException

Amazon Textract kann auf das in der Anforderung angegebene S3-Objekt nicht zugreifen.Konfigurieren des Zugriffs auf Amazon S3Informationen zur Problembehebung finden Sie unterFehlerbehebung für Amazon S3

HTTP-Statuscode: 400

LimitExceededException

Ein Amazon Textract Textract-Service-Limit wurde überschritten. Wenn Sie beispielsweise zu viele asynchrone Jobs gleichzeitig starten, rufen Sie den Betrieb auf (`StartDocumentTextDetection`lösen beispielsweise eine LimitExcedeException-Ausnahme (HTTP-Statuscode: 400) aus, bis die Anzahl der gleichzeitig ausgeführten Aufträge unter dem Amazon Textract Textract-Service-Limit liegt.

HTTP-Statuscode: 400

ProvisionedThroughputExceededException

Die Anzahl der Anforderungen hat das Durchsatzlimit überschritten. Wenn Sie dieses Limit erhöhen müssen, wenden Sie sich an Amazon Textract.

HTTP-Statuscode: 400

ThrottlingException

Amazon Textract kann die Anforderung vorübergehend nicht verarbeiten. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

UnsupportedDocumentException

Das Format des Eingabedokuments wird nicht unterstützt. Dokumente für Operationen können im PNG-, JPEG-, PDF- oder TIFF-Format vorliegen.

HTTP-Statuscode: 400

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-Befehlszeilenschnittstelle
- AWS-SDK für .NET
- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS-SDK für JavaScript
- AWS SDK für PHP V3
- AWS SDK für Python
- AWS SDK für Ruby V3

StartExpenseAnalysis

Startet die asynchrone Analyse von Rechnungen oder Belegen für Daten wie Kontaktinformationen, gekaufte Artikel und Kreditorennamen.

`StartExpenseAnalysis`kann Text in Dokumenten analysieren, die im JPEG-, PNG- und PDF-Format vorliegen. Die Dokumente müssen in einem Amazon S3 S3-Bucket gespeichert werden. Verwenden derDocumentLocationParameter, um den Namen Ihres S3-Buckets und den Namen des Dokuments in diesem Bucket anzugeben.

`StartExpenseAnalysis`gibt eine Job-ID zurück (`JobId`) die Sie zur Verfügung stellen`GetExpenseAnalysis`um die Ergebnisse der Operation abzurufen. Wenn die Analyse der Eingaberechnungen/Quittungen abgeschlossen ist, veröffentlicht Amazon Textract einen Abschlussstatus für das Thema Amazon Simple Notification Service (Amazon SNS), das Sie dem`NotificationChannel`aus. Um die Ergebnisse des Rechnungs- und Empfangsanalysevorgangs zu erhalten, stellen Sie sicher, dass der im Amazon SNS SNS-Thema veröffentlichte Statuswert lautet`SUCCEEDED`aus. Wenn ja, ruf anGetExpenseAnalysisund übergeben Sie die Job-ID (`JobId`) das wurde von Ihrem Anruf an zurückgegeben`StartExpenseAnalysis`aus.

Weitere Informationen finden Sie unterRechnungen und Belege analysierenaus.

Anforderungssyntax

```
{
   "ClientRequestToken": "string",
   "DocumentLocation": {
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   },
   "JobTag": "string",
   "KMSKeyId": "string",
   "NotificationChannel": {
      "RoleArn": "string",
      "SNSTopicArn": "string"
   },
   "OutputConfig": {
      "S3Bucket": "string",
      "S3Prefix": "string"
```

```
    }
}
```

Anfrageparameter

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

ClientRequestToken

Das idempotente Token, das verwendet wird, um die Startanfrage zu identifizieren. Wenn Sie dasselbe Token mit mehreren verwenden`StartDocumentTextDetection`Anfragen, das gleiche`JobId`wird zurückgegeben. Verwenden von`ClientRequestToken`um zu verhindern, dass derselbe Job versehentlich mehr als einmal gestartet wird. Weitere Informationen finden Sie unterAsynchrone Operationen von Amazon Textract aufrufen

Type: String (Zeichenfolge)

Einschränkungen der Länge: Mindestlänge 1. Höchstlänge = 64 Zeichen.

Pattern: `^[a-zA-Z0-9-_]+$`

Erforderlich. Nein

DocumentLocation

Der Speicherort des zu verarbeitenden Dokuments.

Typ: DocumentLocation Objekt

Erforderlich. Ja

JobTag

Eine von Ihnen angegebene Kennung, die in der zum Amazon SNS SNS-Thema veröffentlichten Abschlussbenachrichtigung enthalten ist. Sie können beispielsweise die Datei verwenden`JobTag`um die Art des Dokuments anzugeben, dem die Abschlussmitteilung entspricht (z. B. ein Steuerformular oder eine Quittung).

Type: String (Zeichenfolge)

Einschränkungen der Länge: Mindestlänge 1. Höchstlänge = 64 Zeichen.

Pattern: `[a-zA-Z0-9_.\-:]+`

Erforderlich. Nein

KMSKeyId

Der KMS-Schlüssel, der zum Verschlüsseln der Inferenzergebnisse verwendet wird. Dies kann entweder im Key ID- oder Key Alias Format vorliegen. Wenn ein KMS-Schlüssel bereitgestellt wird, wird der KMS-Schlüssel zur serverseitigen Verschlüsselung der Objekte im Kunden-Bucket verwendet. Wenn dieser Parameter nicht aktiviert ist, wird das Ergebnis serverseitig mit SSE-S3 verschlüsselt.

Type: String (Zeichenfolge)

Einschränkungen der Länge: Mindestlänge 1. Maximale Länge beträgt 2048 Zeichen.

Pattern: `^[A-Za-z0-9][A-Za-z0-9:_/+=,@.-]{0,2048}$`

Erforderlich. Nein

NotificationChannel

Das Amazon SNS SNS-Thema ARN, in dem Amazon Textract den Abschlussstatus des Vorgangs veröffentlichen soll.

Typ: NotificationChannel Objekt

Erforderlich. Nein

OutputConfig

Legt fest, ob die Ausgabe an einen vom Kunden definierten Bucket gesendet wird. Standardmäßig speichert Amazon Textract die Ergebnisse intern, auf die von der`GetExpenseAnalysis`verwenden.

Typ: OutputConfig Objekt

Erforderlich. Nein

Antwortsyntax

```
{
   "JobId": "string"
```

```
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

JobId

Eine eindeutige Kennung für den Texterkennungsauftrag. Die`JobId`wird zurückgegeben von`StartExpenseAnalysis`aus. EIN`JobId`Wert ist nur 7 Tage lang gültig.

Type: String (Zeichenfolge)

Einschränkungen der Länge: Mindestlänge 1. Höchstlänge = 64 Zeichen.

Pattern: `^[a-zA-Z0-9-_]+$`

Fehler

AccessDeniedException

Sie sind nicht berechtigt, die Aktion auszuführen. Verwenden Sie den Amazon-Ressourcennamen (ARN) der IAM-Rolle oder eines autorisierten Benutzers, um den Vorgang auszuführen.

HTTP-Statuscode: 400

BadDocumentException

Amazon Textract kann das Dokument nicht lesen. Weitere Informationen zu den Beleglimits in Amazon Textract finden Sie unterHard Limits in Amazon Textractaus.

HTTP-Statuscode: 400

DocumentTooLargeException

Das Dokument kann nicht verarbeitet werden, da es zu groß ist. Die maximale Dokumentgröße für synchrone Operationen 10 MB. Die maximale Dokumentgröße für asynchrone Vorgänge beträgt 500 MB für PDF-Dateien.

HTTP-Statuscode: 400

IdempotentParameterMismatchException

EINClientRequestTokenDer Eingabeparameter wurde mit einer Operation wiederverwendet, aber mindestens eine der anderen Eingabeparameter ist anders als im vorherigen Aufruf der Operation.

HTTP-Statuscode: 400

InternalServerError

Amazon Textract hat ein Service-Problem festgestellt. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

InvalidKMSKeyException

Zeigt an, dass Sie keine Entschlüsselungsberechtigungen mit dem eingegebenen KMS-Schlüssel haben oder der KMS-Schlüssel falsch eingegeben wurde.

HTTP-Statuscode: 400

InvalidParameterException

Ein Eingabeparameter verletzt eine Beschränkung. Zum Beispiel wird bei synchronen Operationen einInvalidParameterExceptionAusnahme tritt auf, wenn keiner derS3ObjectoderBytesWerte werden imDocumentAnforderungsparameter. Validieren Sie den Parameter, bevor Sie die API-Operation erneut aufrufen.

HTTP-Statuscode: 400

InvalidS3ObjectException

Amazon Textract kann auf das in der Anforderung angegebene S3-Objekt nicht zugreifen.Konfigurieren des Zugriffs auf Amazon S3Informationen zur Problembehebung finden Sie unterFehlerbehebung für Amazon S3

HTTP-Statuscode: 400

LimitExceededException

Ein Amazon Textract Textract-Service-Limit wurde überschritten. Wenn Sie beispielsweise zu viele asynchrone Jobs gleichzeitig starten, rufen Sie den Betrieb auf (StartDocumentTextDetectionWenn Sie beispielsweise eine LimitExcedeDException-

Ausnahme (HTTP-Statuscode: 400) aus, bis die Anzahl der gleichzeitig ausgeführten Aufträge unter dem Amazon-Textract-Service-Limit liegt.

HTTP-Statuscode: 400

ProvisionedThroughputExceededException

Die Anzahl der Anforderungen hat das Durchsatzlimit überschritten. Wenn Sie dieses Limit erhöhen möchten, wenden Sie sich an Amazon Textract.

HTTP-Statuscode: 400

ThrottlingException

Amazon Textract kann die Anforderung vorübergehend nicht verarbeiten. Wiederholen Sie den Aufruf.

HTTP-Statuscode: 500

UnsupportedDocumentException

Das Format des Eingabedokuments wird nicht unterstützt. Dokumente für Operationen können im PNG-, JPEG-, PDF- oder TIFF-Format vorliegen.

HTTP-Statuscode: 400

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-Befehlszeilenschnittstelle
- AWS-SDK für .NET
- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS-SDK für JavaScript
- AWS SDK für PHP V3
- AWS SDK für Python
- AWS SDK für Ruby V3

Datentypen

Die folgenden Datentypen werden unterstützt:

- AnalyzeIDDetections
- Block
- BoundingBox
- Document
- DocumentLocation
- DocumentMetadata
- ExpenseDetection
- ExpenseDocument
- ExpenseField
- ExpenseType
- Geometry
- HumanLoopActivationOutput
- HumanLoopConfig
- HumanLoopDataAttributes
- IdentityDocument
- IdentityDocumentField
- LineItemFields
- LineItemGroup
- NormalizedValue
- NotificationChannel
- OutputConfig
- Point
- Relationship
- S3Object
- Warning

AnalyzeIDDetections

Wird verwendet, um die von einer AnalyeID-Operation erkannten Informationen zu enthalten.

Inhalt

Confidence

Der Konfidenzwert ist.

Type: Gleitkommazahl

Gültiger Bereich: Der Mindestwert 0 ist. Maximalwert 100.

: Erforderlich Nein

NormalizedValue

Wird nur für Datumsangaben zurückgegeben, gibt den Typ des erkannten Wertes und das Datum zurück, das maschinenlesbar geschrieben wird.

Typ: NormalizedValue Objekt

: Erforderlich Nein

Text

Text des normalisierten Feldes oder des damit verbundenen Wertes.

Type: String (Zeichenfolge)

: Erforderlich Ja

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

Block

EIN`Block`repräsentiert Elemente, die in einem Dokument innerhalb einer Gruppe von Pixeln nahe beieinander erkannt werden. Die Informationen, die in einem`Block`-Objekt hängt von der Art der Operation ab. Bei der Texterkennung für Dokumente (z.DetectDocumentText) erhalten Sie Informationen über die erkannten Wörter und Textzeilen. In der Textanalyse (zum BeispielAnalyzeDocument) können Sie auch Informationen über die Felder, Tabellen und Auswahlelemente abrufen, die im Dokument erkannt werden.

Ein Array von`Block`-Objekte werden sowohl durch synchrone als auch durch asynchrone Operationen zurückgegeben. Bei synchronen Operationen wieDetectDocumentText, das Array von`Block`object ist der gesamte Ergebnissatz. Bei asynchronen Operationen wieGetDocumentAnalysiswird das Array über eine oder mehrere Antworten zurückgegeben.

Weitere Informationen finden Sie unterFunktionsweise von Amazon Textractaus.

Inhalt

BlockType

Der Typ des erkannten Textelements. Bei Operationen zur Texterkennung werden die folgenden Typen zurückgegeben:

- SEITE- Enthält eine Liste der LINE`Block`Objekte, die auf einer Dokumentseite erkannt werden.
- WORT- Ein Wort wurde auf einer Dokumentseite erkannt. Ein Wort besteht aus einem oder mehreren lateinischen ISO-Basiszeichen, die nicht durch Leerzeichen getrennt sind.
- LINIE- Eine Reihe von tabulatorgetrennten, zusammenhängenden Wörtern, die auf einer Dokumentseite erkannt werden.

In Textanalyseoperationen werden die folgenden Typen zurückgegeben:

- SEITE- Enthält eine Liste von untergeordnetenBlockObjekte, die auf einer Dokumentseite erkannt werden.
- KEY_VALUE_SET- Speichert den KEY und VALUE`Block`-Objekte für verknüpften Text, der auf einer Dokumentseite erkannt wird. Verwenden der`EntityType`-Feld, um festzustellen, ob ein KEY_VALUE_SET-Objekt ein KEY ist`Block`Objekt oder ein VALUE`Block`-Objekt.
- WORT- Ein Wort, das auf einer Dokumentseite erkannt wird. Ein Wort besteht aus einem oder mehreren lateinischen ISO-Basiszeichen, die nicht durch Leerzeichen getrennt sind.
- LINIE- Eine Reihe von tabulatorgetrennten, zusammenhängenden Wörtern, die auf einer Dokumentseite erkannt werden.

- TABELLE- Eine Tabelle, die auf einer Dokumentseite erkannt wird. Eine Tabelle ist rasterbasierte Informationen mit zwei oder mehr Zeilen oder Spalten mit einer Zellenspanne von jeweils einer Zeile und einer Spalte.
- ZELLE- Eine Zelle innerhalb einer erkannten Tabelle. Die Zelle ist das übergeordnete Element des Blocks, der den Text in der Zelle enthält.
- SELECTION_ELEMENT- Ein Auswahlelement wie ein Optionsfeld (Optionsfeld) oder ein Kontrollkästchen, das auf einer Dokumentseite erkannt wird. Verwenden Sie den Wert von`SelectionStatus`um den Status des Selektionselements zu bestimmen.

Type: String (Zeichenfolge)

Zulässige Werte: `KEY_VALUE_SET | PAGE | LINE | WORD | TABLE | CELL | SELECTION_ELEMENT`

: Erforderlich Nein

ColumnIndex

Die Spalte, in der eine Tabellenzelle angezeigt wird. Die erste Spaltenposition ist 1.`ColumnIndex`wird nicht zurückgegeben von`DetectDocumentText`und`GetDocumentTextDetection`aus.

Type: Ganzzahl

Gültiger Bereich: Der Mindestwert ist.

: Erforderlich Nein

ColumnSpan

Die Anzahl der Spalten, die eine Tabellenzelle umfasst. Derzeit ist dieser Wert immer 1, auch wenn die Anzahl der gespeicherten Spalten größer als 1 ist.`ColumnSpan`wird nicht zurückgegeben von`DetectDocumentText`und`GetDocumentTextDetection`aus.

Type: Ganzzahl

Gültiger Bereich: Der Mindestwert ist.

: Erforderlich Nein

Confidence

Der Konfidenzwert, den Amazon Textract in der Genauigkeit des erkannten Textes und die Genauigkeit der Geometrie aufweist, zeigt um den erkannten Text.

Type: Gleitkommazahl

Gültiger Bereich: Der Mindestwert ist. Maximalwert 100.

: Erforderlich Nein

EntityTypes

Der Typ von Entität. Folgendes kann zurückgegeben werden:

- SCHLÜSSEL- Eine Kennung für ein Feld im Dokument.
- WERT- Der Feldtext.

`EntityTypes`wird nicht zurückgegeben von`DetectDocumentText`und`GetDocumentTextDetection`aus.

Type: Zeichenfolgen-Array

Zulässige Werte: `KEY | VALUE`

: Erforderlich Nein

Geometry

Die Position des erkannten Textes auf dem Bild. Es enthält einen achsorientierten, groben Begrenzungsrahmen, der den Text umgibt, und ein feinkörniges Polygon für genauere räumliche Informationen.

Typ: Geometry Objekt

: Erforderlich Nein

Id

Die Kennung für den erkannten Text. Die Kennung ist nur für eine einzelne Operation eindeutig.

Type: String (Zeichenfolge)

Pattern: `.*\S.*`

: Erforderlich Nein

Page

Die Seite, auf der ein Block erkannt wurde.`Page`wird durch asynchrone Operationen zurückgegeben. Seitenwerte größer als 1 werden nur für mehrseitige Dokumente zurückgegeben,

die im PDF- oder TIFF-Format vorliegen. Ein gescanntes Bild (JPEG/PNG), selbst wenn es mehrere Dokumentseiten enthält, gilt als einseitiges Dokument. Der Wert von`Page`ist immer 1. Synchrone Vorgänge kehren nicht zurück`Page`weil jedes Eingabedokument als einseitiges Dokument angesehen wird.

Type: Ganzzahl

Gültiger Bereich: Der Mindestwert ist.

: Erforderlich Nein

Relationships

Eine Liste der untergeordneten Blöcke des aktuellen Blocks. Ein LINE-Objekt verfügt beispielsweise über untergeordnete Blöcke für jeden WORD-Block, der Teil der Textzeile ist. Es gibt keine Relationship-Objekte in der Liste für Beziehungen, die nicht existieren, z. B. wenn der aktuelle Block keine untergeordneten Blöcke enthält. Die Listengröße kann wie folgt sein:

- 0 - Der Block hat keine untergeordneten Blöcke.
- 1 - Der Block hat untergeordnete Blöcke.

Type: Array von[Relationship](#)Objekte

: Erforderlich Nein

RowIndex

Die Zeile, in der sich eine Tabellenzelle befindet. Die erste Zeilenposition ist 1.`RowIndex`wird nicht zurückgegeben von`DetectDocumentText`und`GetDocumentTextDetection`aus.

Type: Ganzzahl

Gültiger Bereich: Der Mindestwert ist.

: Erforderlich Nein

RowSpan

Die Anzahl der Zeilen, die eine Tabellenzelle umfasst. Derzeit ist dieser Wert immer 1, auch wenn die Anzahl der überspannten Zeilen größer als 1 ist.`RowSpan`wird nicht zurückgegeben von`DetectDocumentText`und`GetDocumentTextDetection`aus.

Type: Ganzzahl

Gültiger Bereich: Der Mindestwert ist.

: Erforderlich Nein

SelectionStatus

Der Auswahlstatus eines Auswahlelements, z. B. ein Optionsfeld oder ein Kontrollkästchen.

Type: String (Zeichenfolge)

Zulässige Werte: `SELECTED | NOT_SELECTED`

: Erforderlich Nein

Text

Das Wort oder die Textzeile, die von Amazon Textract erkannt wird.

Type: String (Zeichenfolge)

: Erforderlich Nein

TextType

Die Art von Text, den Amazon Textract erkannt hat. Kann nach handgeschriebenem Text und gedrucktem Text suchen.

Type: String (Zeichenfolge)

Zulässige Werte: `HANDWRITING | PRINTED`

: Erforderlich Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

BoundingBox

Der Begrenzungsrahmen um die erkannte Seite, den Text, das Schlüssel-Wert-Paar, die Tabelle, die Tabellenzelle oder das Auswahlelement auf einer Dokumentseite. Die`left`(x-Koordinate) und`top`(y-Koordinate) sind Koordinaten, die obere und linke Seite des Begrenzungsrahmens darstellen. Beachten Sie, dass die obere linke Ecke des Bildes der Ursprung (0,0) ist.

Die`top`und`left`Die zurückgegebenen Werte sind Verhältnisse der Gesamtseitengröße des Dokuments. Beispiel: Bei einem Eingangsbild mit 700 x 200 Pixeln und einer linken oberen Koordinate des Begrenzungsrahmens von 350 x 50 Pixeln gibt die API einen `left`-Wert von 0,5 (350/700) und einen `top`-Wert von 0,25 (50/200) zurück.

Die`width`und`height`Die Werte stellen die Abmessungen des Begrenzungsrahmens als Verhältnis der gesamten Dokumentseitendimension dar. Wenn die Seitengröße des Dokuments beispielsweise 700 x 200 Pixel beträgt und die Breite des Begrenzungsrahmens 70 Pixel beträgt, beträgt die zurückgegebene Breite 0,1.

Inhalt

Height

Die Höhe des Begrenzungsrahmens als Verhältnis der Gesamtseitenhöhe des Dokuments.

Type: Gleitkommazahl

: Erforderlich Nein

Left

Die linke Koordinate des Begrenzungsrahmens als Verhältnis der Gesamtseitenbreite des Dokuments.

Type: Gleitkommazahl

: Erforderlich Nein

Top

Die obere Koordinate des Begrenzungsrahmens als Verhältnis der Gesamtseitenhöhe des Dokuments.

Type: Gleitkommazahl

: Erforderlich Nein

Width

Die Breite des Begrenzungsrahmens als Verhältnis der Gesamtseitenbreite des Dokuments.

Type: Gleitkommazahl

: Erforderlich Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

Document

Das Eingabedokument, entweder als Byte oder als S3-Objekt.

Sie übergeben Bild-Bytes an eine Amazon Textract Textract-API-Operation, indem Sie die`Bytes`eigentum. Verwenden Sie zum Beispiel die`Bytes`-Eigenschaft, um ein aus einem lokalen Dateisystem geladenes Dokument zu übergeben. Bildbytes, die mit dem`Bytes`-Eigenschaft muss base64-codiert werden. Ihr Code muss möglicherweise keine Dokumentdateibytes codieren, wenn Sie ein AWS SDK verwenden, um Amazon Textract Textract-API-Operationen aufzurufen.

Sie übergeben Bilder, die in einem S3-Bucket gespeichert sind, an eine Amazon Textract Textract-API-Operation, indem Sie die`S3Object`eigentum. Dokumente, die in einem S3-Bucket gespeichert sind, müssen nicht base64-codiert werden.

Die AWS-Region für den S3-Bucket, der das S3-Objekt enthält, muss mit der von Ihnen für Amazon Textract Textract-Operationen verwendeten AWS-Region übereinstimmen.

Wenn Sie Amazon Textract Textract-Operationen mithilfe der AWS CLI aufrufen, wird die Weitergabe von Bild-Bytes mithilfe der Bytes -Eigenschaft nicht unterstützt. Sie müssen zuerst das Dokument auf einen Amazon S3 S3-Bucket hochladen und anschließend die Operation mithilfe der S3Object-Eigenschaft aufrufen.

Damit Amazon Textract ein S3-Objekt verarbeiten kann, muss der Benutzer über die Berechtigung für den Zugriff auf das S3-Objekt verfügen.

Inhalt

Bytes

Ein Blob von base64-codierten Dokumentbytes. Die maximale Größe eines Dokuments, das in einem Byte-Blob bereitgestellt wird, beträgt 5 MB. Die Dokumentbytes müssen im PNG- oder JPEG-Format vorliegen.

Wenn Sie ein AWS SDK zum Aufrufen von Amazon Textract verwenden, müssen Sie möglicherweise keine Base64-Codierungsbytes codieren, die mit der`Bytes`field.

Type: Base64-kodiertes Binärdatenobjekt

Längenbeschränkungen: Mindestlänge 1. Maximale Länge beträgt 10485760 Zeichen.

Erforderlich: Nein

S3Object

Identifiziert ein S3-Objekt als Dokumentquelle. Die maximale Größe eines Dokuments, das in einem S3-Bucket gespeichert ist, beträgt 5 MB.

Typ: S3Object Objekt

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

DocumentLocation

Der Amazon S3 S3-Bucket, der das zu verarbeitende Dokument enthält. Es wird von asynchronen Operationen wieStartDocumentTextDetectionaus.

Das Eingabedokument kann eine Bilddatei im JPEG- oder PNG-Format sein. Es kann sich auch um eine Datei im PDF-Format handeln.

Inhalt

S3Object

Der Amazon S3 S3-Bucket, der das Eingabedokument enthält.

Typ: S3Object Objekt

Erforderlich Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

DocumentMetadata

Informationen über das Eingabedokument.

Inhalt

Pages

Die Anzahl der Seiten, die im Dokument erkannt werden.

Type: Ganzzahl

Gültiger Bereich: Der Mindestwert ist.

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

ExpenseDetection

Ein Objekt, das zum Speichern von Informationen über den von Amazon Textract erkannten Wert oder Label verwendet wird.

Inhalt

Confidence

Das Vertrauen in die Erkennung als Prozentsatz

Type: Gleitkommazahl

Gültiger Bereich: Der Mindestwert 0 ist. Maximalwert 100.

: Erforderlich Nein

Geometry

Informationen darüber, wo sich die folgenden Elemente auf einer Dokumentseite befinden: Erkannte Seite, Text, Schlüssel-Wert-Paare, Tabellen, Tabellenzellen und Selektionselemente.

Typ: Geometry Objekt

: Erforderlich Nein

Text

Das von Amazon Textract erkannte Wort oder die Textzeile

Type: String (Zeichenfolge)

: Erforderlich Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2

- AWS SDK für Ruby V3

ExpenseDocument

Die Struktur, die alle von AnalyzeExpense zurückgegebenen Informationen enthält

Inhalt

ExpenseIndex

Gibt an, aus welcher Rechnung oder Wareneingang im Dokument die Informationen stammen. Das erste Dokument ist 1, das zweite 2 usw.

Type: Ganzzahl

Gültiger Bereich: Der Mindestwert ist.

Erforderlich Nein

LineItemGroups

Informationen, die in jeder Tabelle eines Dokuments erkannt wurden, unterteilt in`LineItems`aus.

Type: ArrayLineItemGroupObjekte

Erforderlich Nein

SummaryFields

Alle Informationen, die Amazon Textract außerhalb einer Tabelle gefunden hat.

Type: ArrayExpenseFieldObjekte

Erforderlich Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

ExpenseField

Aufschlüsselung erkannter Informationen, unterteilt in die Katagorien Typ, LabelDetection und ValueDetection

Inhalt

LabelDetection

Die explizit angegebene Bezeichnung eines erkannten Elements.

Typ: ExpenseDetection Objekt

Erforderlich Nein

PageNumber

Die Seitenzahl, auf der der Wert erkannt wurde.

Type: Ganzzahl

Gültiger Bereich: Der Mindestwert 0 ist.

Erforderlich Nein

Type

Die implizierte Bezeichnung eines erkannten Elements. Präsentieren Sie neben LabelDetection für explizite Elemente.

Typ: ExpenseType Objekt

Erforderlich Nein

ValueDetection

Der Wert eines erkannten Elements. Präsentiert in expliziten und impliziten Elementen.

Typ: ExpenseDetection Objekt

Erforderlich Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

ExpenseType

Ein Objekt, das zum Speichern von Informationen über den von Amazon Textract erkannten Typ verwendet wird.

Inhalt

Confidence

Das Vertrauen der Genauigkeit in Prozent.

Type: Gleitkommazahl

Gültiger Bereich: Der Mindestwert ist. Maximalwert 100.

Erforderlich: Nein

Text

Das Wort oder die Textzeile, die von Amazon Textract erkannt wurde.

Type: String (Zeichenfolge)

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

Geometry

Informationen darüber, wo sich die folgenden Elemente auf einer Dokumentseite befinden: Erkannte Seite, Text, Schlüssel-Wert-Paare, Tabellen, Tabellenzellen und Selektionselemente.

Inhalt

BoundingBox

Eine achsausgerichtete grobe Darstellung der Position des erkannten Elements auf der Dokumentseite.

Typ: BoundingBox Objekt

Erforderlich Nein

Polygon

Innerhalb des Begrenzungsrahmens ein feinkörniges Polygon um das erkannte Element herum.

Type: Array-ReihePointObjekte

Erforderlich Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

HumanLoopActivationOutput

Zeigt die Ergebnisse des Menschen in der Schleifenauswertung an. Wenn es keinen HumanLooparn gibt, hat der Input keine menschliche Überprüfung ausgelöst.

Inhalt

HumanLoopActivationConditionsEvaluationResults

Zeigt das Ergebnis von Zustandsbewertungen, einschließlich derjenigen Bedingungen, die eine menschliche Überprüfung aktiviert haben.

Type: String (Zeichenfolge)

Einschränkungen: Maximale Länge beträgt 10240 Zeichen.

: Erforderlich Nein

HumanLoopActivationReasons

Zeigt an, ob und warum eine menschliche Überprüfung erforderlich war.

Type: Zeichenfolgen-Array

Array-Mitglieder: Die Mindestanzahl beträgt 1 Element.

: Erforderlich Nein

HumanLoopArn

Der Amazon-Ressourcenname (ARN) vom erstellten HumanLoop.

Type: String (Zeichenfolge)

Einschränkungen: Maximale Länge beträgt 256 Zeichen.

: Erforderlich Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++

- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

HumanLoopConfig

Richtet den Arbeitsablauf für menschliche Überprüfung ein, an den das Dokument gesendet wird, wenn eine der Bedingungen erfüllt ist. Sie können vor der Überprüfung auch bestimmte Attribute des Bildes festlegen.

Inhalt

DataAttributes

Legt die Attribute der Eingabedaten fest.

Typ: HumanLoopDataAttributes Objekt

: Erforderlich Nein

FlowDefinitionArn

Der Amazon-Ressourcenname (ARN) der Flowdefinition.

Type: String (Zeichenfolge)

Einschränkungen für die Länge: Maximale Länge beträgt 256 Zeichen.

: Erforderlich Ja

HumanLoopName

Der Name des menschliche Workflows, der für dieses Bild verwendet wird. Dies sollte innerhalb einer Region eindeutig bleiben.

Type: String (Zeichenfolge)

Einschränkungen für die Länge: Mindestlänge 1. Maximale Länge beträgt 63 Zeichen.

Pattern: `^[a-z0-9](-*[a-z0-9])*`

: Erforderlich Ja

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

HumanLoopDataAttributes

Ermöglicht das Festlegen von Attributen des Bildes. Derzeit können Sie ein Bild als frei von persönlich identifizierbaren Informationen und Inhalten für Erwachsene erklären.

Inhalt

ContentClassifiers

Legt fest, ob das Eingabebild frei von personenbezogenen Daten oder Inhalten für Erwachsene ist.

Type: Zeichenfolgen-Array

Array-Mitglieder: Die maximale Anzahl beträgt 256 Elemente.

Zulässige Werte: `FreeOfPersonallyIdentifiableInformation | FreeOfAdultContent`

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

IdentityDocument

Die Struktur, die jedes in einer AnalyseID-Operation verarbeitete Dokument auflistet.

Inhalt

DocumentIndex

Gibt die Platzierung eines Dokuments in der IdentityDocument-Liste an. Das erste Dokument ist mit 1, das zweite 2 usw. gekennzeichnet.

Type: Ganzzahl

Gültiger Bereich: Mindestwert 0.

Erforderlich: Nein

IdentityDocumentFields

Die Struktur zur Aufzeichnung von Informationen, die aus Ausweisdokumenten extrahiert wurden. Enthält sowohl ein normalisiertes Feld als auch den Wert des extrahierten Textes.

Type: ArrayIdentityDocumentFieldObjekte

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

IdentityDocumentField

Struktur, die sowohl den normalisierten Typ der extrahierten Informationen als auch den damit verbundenen Text enthält. Diese werden als Typ bzw. Value extrahiert.

Inhalt

Type

Wird verwendet, um die von einer AnalyeID-Operation erkannten Informationen zu enthalten.

Typ: AnalyzeIDDetections Objekt

: Erforderlich Nein

ValueDetection

Wird verwendet, um die von einer AnalyeID-Operation erkannten Informationen zu enthalten.

Typ: AnalyzeIDDetections Objekt

: Erforderlich Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

LineItemFields

Eine Struktur, die Informationen über die verschiedenen Zeilen in den Tabellen eines Dokuments enthält.

Inhalt

LineItemExpenseFields

ExpenseFields werden verwendet, um Informationen aus erkannten Zeilen in einer Tabelle anzuzeigen.

Type: Array vonExpenseFieldObjekte

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

LineItemGroup

Eine Gruppierung von Tabellen, die LineIteMS enthalten, wobei jede Tabelle durch die`LineItemGroupIndex`aus.

Inhalt

LineItemGroupIndex

Die Nummer, mit der eine bestimmte Tabelle in einem Dokument identifiziert wird. Die erste Tabelle wird einen LineItemGroupIndex von 1, die zweite 2 usw. haben.

Type: Ganzzahl

Gültiger Bereich: Mindestwert 0.

Erforderlich: Nein

LineItems

Die Aufschlüsselung der Informationen in einer bestimmten Zeile einer Tabelle.

Type: Array-BereichLineItemFieldsObjekte

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

NormalizedValue

Enthält Informationen zu Datumsangaben in einem Dokument, einschließlich der Art des Wertes und des Wertes.

Inhalt

Value

Der Wert des Datums, geschrieben als Jahr-Monat-Tagthour:Minute:Sekunde.

Type: String (Zeichenfolge)

Erforderlich: Nein

ValueType

Der normalisierte Typ des erkannten Wertes. In diesem Fall DATE.

Type: String (Zeichenfolge)

Zulässige Werte: `DATE`

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

NotificationChannel

Das Amazon Simple Notification Service (Amazon SNS) -Thema, für das Amazon Textract den Abschlussstatus eines asynchronen Dokumentvorgangs veröffentlicht, wie z.StartDocumentTextDetectionaus.

Inhalt

RoleArn

Der Amazon Resource Name (ARN) einer IAM-Rolle, die Amazon Textract Textract-Veröffentlichungsberechtigungen für das Amazon SNS SNS-Thema erteilt.

Type: String (Zeichenfolge)

Einschränkungen: Mindestlänge 20. Maximale Länge beträgt 2048 Zeichen.

Pattern: `arn:([a-z\d-]+):iam::\d{12}:role/?[a-zA-Z_0-9+=,.@\-_/]+`

Erforderlich: Ja

SNSTopicArn

Das Amazon SNS SNS-Thema, bei dem Amazon Textract den Abschlussstatus veröffentlicht.

Type: String (Zeichenfolge)

Einschränkungen: Mindestlänge 20. Maximale Länge beträgt 1024 Zeichen.

Pattern: `(^arn:([a-z\d-]+):sns:[a-zA-Z\d-]{1,20}:\w{12}:.+$)`

Erforderlich: Ja

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2

- AWS SDK für Ruby V3

OutputConfig

Legt fest, ob Ihre Ausgabe an einen vom Benutzer erstellten Bucket gesendet wird oder nicht. Wird verwendet, um den Namen des Buckets und das Präfix für die Ausgabedatei festzulegen.

`OutputConfig`ist ein optionaler Parameter, mit dem Sie einstellen können, wo Ihre Ausgabe platziert wird. Standardmäßig speichert Amazon Textract die Ergebnisse intern und kann nur von den API-Vorgängen abrufen aufgerufen werden. Wenn OutputConfig aktiviert ist, können Sie den Namen des Buckets festlegen, an den die Ausgabe gesendet wird, und das Dateipräfix der Ergebnisse, in denen Sie Ihre Ergebnisse herunterladen können. Darüber hinaus können Sie die`KMSKeyID`Parameter zu einem Kundenmasterschlüssel (Customer Master Key, CMK), um Ihre Ausgabe zu verschlüsselt. Ohne diesen Parametersatz verschlüsselt Amazon Textract serverseitig mit dem von AWS verwalteten CMK für Amazon S3.

Die Entschlüsselung von Kundeninhalten ist für die Verarbeitung der Dokumente durch Amazon Textract erforderlich. Wenn Ihr Konto im Rahmen einer Opt-Out-Richtlinie für KI-Dienste abgesagt wurde, werden alle unverschlüsselten Kundeninhalte sofort und dauerhaft gelöscht, nachdem der Kundeninhalt vom Service verarbeitet wurde. Keine Kopie der Ausgabe wird von Amazon Textract aufbewahrt. Weitere Informationen zum Abmelden finden Sie unterRichtlinien zur Abmelden von KI-Services verwalten.

Weitere Informationen zum Datenschutz finden Sie unterHäufig gestellte Fragen zum Datenschutzaus.

Inhalt

S3Bucket

Der Name des Buckets, an den Ihre Ausgabe weitergegeben werden soll.

Type: String (Zeichenfolge)

Einschränkungen für die Länge: Mindestlänge 3. Höchstlänge = 255 Zeichen.

Pattern: `[0-9A-Za-z\.\-_]*`

Erforderlich: Ja

S3Prefix

Das Präfix des Objektschlüssels, in dem die Ausgabe gespeichert wird. Wenn dies nicht aktiviert ist, lautet das Präfix „textract_output“.

Type: String (Zeichenfolge)

Einschränkungen für die Länge: Mindestlänge 1. Maximale Länge beträgt 1024 Zeichen.

Pattern: `.*\S.*`

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

Point

Die X- und Y-Koordinaten eines Punktes auf einer Dokumentseite. Die zurückgegebenen X- und Y-Werte sind Verhältnisse der Gesamtseitengröße des Dokuments. Wenn das Eingabedokument beispielsweise 700 x 200 ist und der Vorgang X = 0,5 und Y = 0,25 zurückgibt, liegt der Punkt auf der (350,50) Pixelkoordinate auf der Dokumentseite.

Ein Array von`Point`-Objekte,`Polygon`, wird als Teil desGeometryObjekt, das in einem zurückgegeben wirdBlock-Objekt. EIN`Polygon`-Objekt stellt ein feinkörniges Polygon um erkannten Text, ein Schlüssel-Wert-Paar, eine Tabelle, eine Tabellenzelle oder ein Auswahlelement dar.

Inhalt

X

Der Wert der X-Koordinate für einen Punkt auf einem`Polygon`aus.

Type: Gleitkommazahl

Erforderlich: Nein

Y

Der Wert der Y-Koordinate für einen Punkt auf einem`Polygon`aus.

Type: Gleitkommazahl

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

Relationship

Informationen darüber, wie Blöcke miteinander verwandt sind. EINBlockobject enthält 0 oder höherRelationObjekte in einer Liste,Relationshipsaus. Weitere Informationen finden Sie unter Block .

DieType-Element stellt den Typ der Beziehung für alle Blöcke imIDsArray.

Inhalt

Ids

Ein Array von IDs für verwandte Blöcke. Sie können den Typ der Beziehung mit demTypeelement.

Type: Zeichenfolgen-Array

Pattern: `.*\S.*`

Erforderlich Nein

Type

Die Art der Beziehung, die die Blöcke im IDS-Array mit dem aktuellen Block haben. Die Beziehung kannVALUEoderCHILDaus. Eine Beziehung vom Typ VALUE ist eine Liste, die die ID des VALUE-Blocks enthält, der mit dem KEY eines Schlüssel-Wert-Paares verknüpft ist. Eine Beziehung vom Typ CHILD ist eine Liste von IDs, die WORD-Blöcke im Fall von Linien Zellblöcke bei Tabellen und WORD-Blöcken im Fall von Auswahlelementen identifizieren.

Type: String (Zeichenfolge)

Zulässige Werte: `VALUE | CHILD | COMPLEX_FEATURES`

Erforderlich Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++

- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

S3Object

Der S3-Bucket-Name und der Dateiname, der das Dokument identifiziert.

Die AWS-Region für den S3-Bucket, der das Dokument enthält, muss mit der Region übereinstimmen, die Sie für Amazon Textract Textract-Vorgänge verwenden.

Damit Amazon Textract eine Datei in einem S3-Bucket verarbeiten kann, muss der Benutzer über die Berechtigung verfügen, auf den S3-Bucket und die Datei zuzugreifen.

Inhalt

Bucket

Der Name des S3-Buckets. Beachten Sie, dass das Zeichen # im Dateinamen nicht gültig ist.

Type: String (Zeichenfolge)

Einschränkungen für die Länge: Mindestlänge 3. Höchstlänge = 255 Zeichen.

Pattern: `[0-9A-Za-z\.\-_]*`

Erforderlich Nein

Name

Der Dateiname des Eingabedokuments. Synchrone Vorgänge können Bilddateien im JPEG- oder PNG-Format verwenden. Asynchrone Vorgänge unterstützen auch Dateien im PDF- und TIFF-Format.

Type: String (Zeichenfolge)

Einschränkungen für die Länge: Mindestlänge 1. Maximale Länge beträgt 1024 Zeichen.

Pattern: `.*\S.*`

Erforderlich Nein

Version

Wenn für den Bucket das Versioning aktiviert ist, können Sie die Objektversion angeben.

Type: String (Zeichenfolge)

Einschränkungen für die Länge: Mindestlänge 1. Maximale Länge beträgt 1024 Zeichen.

Pattern: `.*\S.*`

Erforderlich Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

Warning

Eine Warnung vor einem Problem, das während der asynchronen Textanalyse aufgetreten ist (StartDocumentAnalysis) oder asynchrone Dokumenttexterkennung (StartDocumentTextDetection) enthalten.

Inhalt

ErrorCode

Der Fehlercode für die Warnung.

Type: String (Zeichenfolge)

Erforderlich: Nein

Pages

Eine Liste der Seiten, für die die Warnung gilt.

Type: Array von Ganzzahlen

Gültiger Bereich: Der Mindestwert 0 ist.

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen AWS-SDKs finden Sie unter:

- AWS-SDK für C++
- AWS-SDK for Go
- AWSSDK for Java V2
- AWS SDK für Ruby V3

Hard Limits in Amazon Textract

Die folgende Liste enthält Hard Limits in Amazon Textract, die nicht geändert werden können. Informationen dazu, welche Limits geändert werden können, finden Sie unterAmazon Textract Sie Endpunkte und Kontingenteaus. Informationen dazu, welche Limits geändert werden können, finden Sie unter AWS Service Limits. Wenn Sie ein Limit ändern möchten, lesen Sie unter CreateCase weiter.

Amazon Textract

Limit	Beschreibung
Akzeptierte Dateiformate	Operationen unterstützen JPEG-, PNG-, PDF- und TIFF-Date ien. (JPEG-2000-codierte Bilder in PDFs werden unterstützt)..
Grenzwerte für Dateigröße und Seitenzahl	Für synchrone Operationen haben JPEG-, PNG-, PDF- und TIFF-Dateien eine Größenbeschränkung von 10 MB. PDF- und TIFF-Dateien haben ebenfalls ein Limit von 1 Seite. Für asynchrone Operationen haben JPEG- und PNG-Dateien eine Größenbeschränkung von 10 MB. PDF- und TIFF-Dateien haben ein Limit von 500 MB. PDF- und TIFF-Dateien haben ein Limit von 3.000 Seiten.
Spezifische PDF-Grenzwerte	Die maximale Höhe und Breite beträgt 40 Zoll und 2880 Punkte. PDFs können nicht passwortgeschützt werden. PDFs können formatierte JPEG 2000-Bilder enthalten.
Dokumentdrehung und Bildgröße	Amazon Textract unterstützt alle Dokumentdrehungen in der Ebene, z. B. eine 45-Grad-Drehung in der Ebene. Amazon Textract unterstützt Bilder mit einer Auflösung von weniger als oder gleich 10000 Pixel auf allen Seiten.
Textausrichtung	Text kann innerhalb des Dokuments horizontal ausgerichtet sein. Amazon Textract unterstützt keine vertikale Textausri chtung innerhalb des Dokuments.

Limit	Beschreibung
Sprachen	Amazon Textract unterstützt Texterkennung in Englisch, Französisch, Deutsch, Italienisch, Portugiesisch und Spanisch. Amazon Textract gibt die in der Ausgabe erkannte Sprache nicht zurück.
Charakter-Größe	Die Mindesthöhe für den zu erkennenden Text beträgt 15 Pixel. Bei 150 DPI wäre dies das gleiche wie eine 8-Punkt-Schriftart.
Zeichentyp	Amazon Textract unterstützt sowohl die handschriftliche als auch die gedruckte Zeichenerkennung.
CHARACTERS	Amazon Textract erkennt die folgenden Zeichen: • a-z • A-Z • 0-9 • Ã ö ö ü ü ü ü ü ç é é é é Ã Ã è ù Ù Ë Ï Ã Ã Ã

Dokumentverlauf für Amazon Textract

In der folgenden Tabelle sind wichtige Änderungen in jeder Version desEntwicklerhandbuch für Amazon Textractaus. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

- Letzte Aktualisierung der Dokumentation: 29. Mai 2019

Aktualisierung des Änderungsverlaufs	Aktualisierung der Verlaufsbeschreibung	Aktualisierung des Verlaufsdatums
Integrieren von Codebeispielen ausAWSDocs SDK Codebeispiele GitHub Repo	Der Amazon Textract Textract-Handbuch enthält jetzt zusätzliche Codebeispiele. Vorheriger Beispielabschnitt wurde in Tutorials umbenannt.	30. Januar 2022
AnalyzeExpense wurde hinzugefügt	Amazon Textract unterstützt jetzt die Analyse von Rechnungs- und Belegdokumenten mithilfe der AnalyzeExpense-API. Diese Funktion ist nur in unseren Regionen Asien-Pazifik (Mumbai), Asien-Pazifik (Singapur), Asien-Pazifik (Sydney), Kanada (Central), Europa (Irland), Europa (London), US-Osten (N.Virginia), US-Osten (Ohio) USA (N.Kalifornien) und US-West (Oregon) verfügbar.	26. Juli 2021
Augmented AI KI-Unterstützung	Amazon Textract unterstützt jetzt Amazon Augmented AI zur Implementierung von Human Review.	3. Dezember 2019

Neuer Dienst mit dazugehörigem Handbuch	Amazon Textract ist jetzt zur allgemeinen Verwendung verfügbar.	29. Mai 2019
Support für Selektionselemente	Amazon Textract kann jetzt Auswahlelemente (Optionsfelder und Kontrollkästchen) erkennen.	24. April 2019
Veröffentlichung von Amazon Textract	Dies ist die erste Version der Dokumentation für Amazon Textract.	28. November 2018

AWS-Glossar

Die neueste AWS-Terminologie finden Sie im AWS-Glossar in der allgemeinen AWS-Referenz.

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.