Guia do desenvolvedor

Amazon Textract

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

Amazon Textract: Guia do desenvolvedor

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

As marcas comerciais e imagens comerciais da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestigie a Amazon. Todas as outras marcas comerciais que não pertencem à Amazon pertencem a seus respectivos proprietários, que podem ou não ser afiliados, patrocinados pela Amazon ou ter conexão com ela.

Table of Contents

O que é o Amazon Textract? ... 1
Usuários iniciantes do Amazon Textract ... 2
Usando o Amazon Textract com umAWSSDK ... 3
Como funcionam ... 4
Detectar texto ... 5
Analisar documentos ... 6
Analisando faturas e recibos ... 7
Analisar documentos de identidade ... 11
Documentos de entrada ... 12
Objetos de resposta do Amazon Textract ... 14
Objetos de resposta de detecção de texto e análise de ... 14
Objetos de resposta de fatura e recebimento ... 35
Objetos de resposta da documentação ... 38
Localização do item em uma página de documento ... 40
Bounding Box ... 41
Polígono ... 44
Conceitos básicos ... 45
Etapa 1: Configurar uma conta ... 45
Cadastre-se na AWS ... 45
Criar um usuário do IAM ... 46
Próxima etapa ... 47
Etapa 2: Configurar aAWS CLIeAWSSDKs da ... 47
Próxima etapa ... 49
Etapa 3: Conceitos básicos do uso daAWS CLIeAWSAPI SDK ... 49
Formatação dos exemplos de AWS CLI ... 50
Processando documentos com operações síncronas ... 51
Chamando operações síncronas do Amazon Textract ... 52
Solicitação ... 52
Resposta ... 54
Detectar texto do documento ... 125
Análise do texto do documento ... 138
Analisando documentos de fatura e recibo ... 150
Análise de documentos de identificação ... 163
Processando documentos com operações assíncronas ... 169

Chamando operações assíncronas ... 170
Iniciar a detecção de texto ... 171
Obter o status de conclusão de uma solicitação de análise do Amazon Textract ... 173
Obtendo resultados de detecção de texto Amazon Textract ... 174
Configurar operações assíncronas ... 184
Dando acesso ao Amazon Textract ao seu tópico do Amazon SNS ... 185
Detectando ou analisando texto em um documento de várias páginas ... 186
Executando operações assíncronas ... 188
Notificação de resultados do Amazon Textract ... 214
Lidando com chamadas limitadas e conexões descartadas ... 216
Práticas recomendadas para o Amazon Textract ... 222
Forneça um documento de entrada ideal ... 222
Usar escores de confiança ... 223
Considere usar a análise humana ... 223
Tutoriais ... 224
Pré-requisitos ... 224
Extraindo pares de valores-chave de um documento de formulário ... 225
Exportando tabelas para um arquivo CSV ... 228
Como criar umAWS LambdaFunção ... 238
Para chamar a operação DetectDocumentText a partir de uma função do Lambda: ... 238
Exemplos de código adicionais ... 241
Exemplos de código ... 243
Ações ... 243
Analisar um documento ... 244
Detectar texto em um documento ... 247
Obter dados sobre uma tarefa de análise de documentos ... 250
Iniciar a análise assíncrona de um documento ... 252
Inicie a detecção de texto assíncrona ... 255
Exemplos entre serviços ... 257
Criar uma aplicação de exploração do Amazon Textract ... 257
Detectar entidades em texto extraído de uma imagem ... 259
Amazon A2I e Amazon Textract ... 261
Conceitos básicos do Amazon A2I ... 261
Condições de ativação da revisão humana ... 261
Fluxo de trabalho de revisão humana (definição de fluxo) ... 263
Loops humanos ... 264

Comece a usar o Amazon A2I 265
Criar um fluxo de trabalho de revisão humana 266
Analisar o documento 272
Monitorar o loop humano 273
Exibir dados de saída e métricas do trabalhador 275
Segurança 278
Proteção de dados 278
Criptografia no Amazon Textract 279
Privacidade do tráfego entre redes 280
Identity and Access Management 280
Público 281
Autenticação com identidades 282
Gerenciamento do acesso usando políticas 285
Como o Amazon Textract funciona com o IAM 287
Exemplos de políticas baseadas em identidade 292
Solução de problemas 295
Registro em log e monitoramento 298
Monitoramento 298
Métricas do CloudWatch para o Amazon Textract 302
Registro de chamadas de API do Amazon Textract comAWS CloudTrail 304
Informações do Amazon Textract no CloudTrail 305
Noções básicas das entradas dos arquivos de log do Amazon Textract T 306
Validação de conformidade 308
Resiliência 309
Segurança da infraestrutura 310
Análise de configuração e vulnerabilidade 310
VPC endpoints (AWS PrivateLink) 310
Considerações endpoints da VPC do Amazon Textract 311
Criar um VPC endpoint de interface para o Amazon Textract 311
Criar uma política de endpoint da VPC para o Amazon Textract 311
Referência da API 313
Ações 313
AnalyzeDocument 314
AnalyzeExpense 321
AnalyzeID 328
DetectDocumentText 333

GetDocumentAnalysis ... 338
GetDocumentTextDetection ... 345
GetExpenseAnalysis ... 352
StartDocumentAnalysis ... 360
StartDocumentTextDetection ... 367
StartExpenseAnalysis ... 373
Tipos de dados ... 378
AnalyzeIDDetections ... 380
Block ... 382
BoundingBox ... 387
Document ... 389
DocumentLocation ... 391
DocumentMetadata ... 392
ExpenseDetection ... 393
ExpenseDocument ... 395
ExpenseField ... 397
ExpenseType ... 399
Geometry ... 400
HumanLoopActivationOutput ... 401
HumanLoopConfig ... 403
HumanLoopDataAttributes ... 405
IdentityDocument ... 406
IdentityDocumentField ... 407
LineItemFields ... 408
LineItemGroup ... 409
NormalizedValue ... 410
NotificationChannel ... 411
OutputConfig ... 413
Point ... 415
Relationship ... 416
S3Object ... 418
Warning ... 420
Limites ... 421
Amazon Textract ... 421
Histórico do documento ... 423
Glossário da AWS ... 425

..... cdxxvi

O que é o Amazon Textract?

O Amazon Textract facilita a adição de detecção e análise de texto de documentos aos aplicativos. O uso dos clientes do Amazon Textract pode:

- Detecte texto digitado e manuscrito em uma variedade de documentos, incluindo relatórios financeiros, registros médicos e formulários fiscais.
- Extraia texto, formulários e tabelas de documentos com dados estruturados, usando a API Amazon Textract Document Analysis.
- Processe faturas e recebimentos com a API AnalyzeExpense.
- Processar documentos de identificação, como licenças de motorista e passaportes emitidos pelo governo dos EUA, usando a API AnalyzeID.

O Amazon Textract é baseado na mesma tecnologia comprovada e altamente escalável de aprendizado profundo desenvolvida pelos cientistas de visão computadorizada da Amazon para analisar diariamente bilhões de imagens e vídeos. Para usá-lo, não é necessário ter conhecimento de Machine Learning. Amazon Textract inclui APIs simples e fáceis de usar que podem analisar arquivos de imagem e arquivos PDF. O Amazon Textract está sempre aprendendo com novos dados e a Amazon está sempre adicionando novos recursos ao serviço.

Veja a seguir alguns casos de uso comuns para usar o Amazon Textract:

- Criando um índice de pesquisa inteligente— Usando o Amazon Textract, você pode criar bibliotecas de texto detectadas em arquivos de imagem e PDF.
- Usando extração de texto inteligente para processamento de linguagens naturais (NLP)— O Amazon Textract fornece controle sobre como o texto é agrupado como uma entrada para aplicativos de PNL. Ele pode extrair texto como palavras e linhas. Ele também agrupa texto por células de tabela se a análise de tabela de documentos do Amazon Textract estiver ativada.
- Acelerando a captura e a normalização de dados de diferentes fontes— Amazon Textract permite a extração de dados de texto e tabular de uma ampla variedade de documentos, como documentos financeiros, relatórios de pesquisa e anotações médicas. Com as APIs do Amazon Textract Analyze Document, você pode extrair dados não estruturados e estruturados de seus documentos com facilidade e rapidez.
- Automatizando a captura de dados de formulários— Amazon Textract permite que dados estruturados sejam extraídos de formulários. Com as APIs do Amazon Textract Analysis, você

pode criar recursos de extração em fluxos de trabalho de negócios existentes para que os dados do usuário enviados por meio de formulários possam ser extraídos para um formato utilizável.

Alguns dos benefícios de usar o Amazon Textract incluem:

- Integração da detecção de texto de documentos em seus aplicativos— O Amazon Textract remove a complexidade da criação de recursos de detecção de texto aos aplicativos ao disponibilizar análises eficientes e precisas com uma API simples. Você não precisa de visão computadorizada ou de conhecimento profundo para usar o Amazon Textract para detectar texto de documentos. Com as APIs de texto do Amazon Textract, você pode integrar facilmente a detecção de texto a qualquer aplicativo de dispositivo conectado, da web ou celular.
- Análise de documentos escalável— Amazon Textract permite que você analise e extraia dados rapidamente de milhões de documentos, o que pode acelerar a tomada de decisões.
- Baixo custo— Com o Amazon Textract, você paga apenas pelos documentos analisados. Não há tarifas mínimas nem compromissos antecipados. Comece a usar gratuitamente e economize mais à medida que você expande seus negócios com nosso modelo de definição de preços em níveis do.

Com o processamento síncrono, o Amazon Textract pode analisar documentos de página única para aplicativos em que a latência é crítica. Amazon Textract também fornece operações assíncronas para estender o suporte a documentos de várias páginas.

Usuários iniciantes do Amazon Textract

Caso seja sua primeira vez usando o Amazon Textract, recomendamos que você leia as seções a seguir em ordem:

1. Como funciona o Amazon Textract— Esta seção apresenta os componentes do Amazon Textract e como eles trabalham juntos para uma experiência de ponta a ponta.
2. Conceitos básicos do Amazon Textract— nesta seção, você abre sua conta e testa a API do Amazon Textract.

Usando o Amazon Textract com umAWSSDK

Os kits de desenvolvimento de software (SDKs) da AWS estão disponíveis para muitas linguagens de programação populares. Cada SDK fornece uma API, exemplos de código e documentação que facilitam a criação de aplicativos em seu idioma preferido pelos desenvolvedores.

Documentação do SDK	Exemplos de código
AWS SDK para C++	AWS SDK para C++Exemplos de código do
AWS SDK para Go	AWS SDK para GoExemplos de código do
AWS SDK para Java	AWS SDK para JavaExemplos de código do
AWS SDK para JavaScript	AWS SDK para JavaScriptExemplos de código do
AWS SDK para .NET	AWS SDK para .NETExemplos de código do
AWS SDK para PHP	AWS SDK para PHPExemplos de código do
AWS SDK para Python (Boto3)	AWS SDK para Python (Boto3)Exemplos de código do
AWS SDK para Ruby	AWS SDK para RubyExemplos de código do

> **Exemplo de disponibilidade**
>
> Você não pode encontrar o que precisa? Solicite um código de exemplo no link Forneça comentários na parte inferior desta página.

Como funciona o Amazon Textract

O Amazon Textract permite detectar e analisar texto em documentos de entrada de uma ou várias páginas (consulteDocumentos de entrada).

Amazon Textract fornece operações para as seguintes ações.

- Detectando somente texto. Para obter mais informações, consulteDetectar texto.
- Detectando e analisando relacionamentos entre texto. Para obter mais informações, consulteAnalisar documentos.
- Detectando e analisando texto em faturas e recibos. Para obter mais informações, consulteAnalisando faturas e recibos.
- Detectando e analisando texto em documentos de identidade do governo. Para obter mais informações, consulteAnalisar documentos de identidade.

Amazon Textract fornece operações síncronas para processamento de documentos pequenos, de uma única página e com respostas quase em tempo real. Para obter mais informações, consulte Processando documentos com operações síncronas. O Amazon Textract também fornece operações assíncronas que você pode usar para processar documentos maiores com várias páginas. As respostas assíncronas não são em tempo real. Para obter mais informações, consulte Processando documentos com operações assíncronas.

Quando uma operação Amazon Textract processa um documento, os resultados são retornados em uma matriz dethe section called “Block”objetos ou uma matriz dethe section called “ExpenseDocument”objects. Ambos os objetos contêm informações detectadas sobre itens, incluindo sua localização no documento e sua relação com outros itens no documento. Para obter mais informações, consulte Objetos de resposta do Amazon Textract. Para obter exemplos que mostram como usar`Block`objetos, consulteTutoriais.

Tópicos

- Detectar texto
- Analisar documentos
- Analisando faturas e recibos
- Analisar documentos de identidade
- Documentos de entrada

- Objetos de resposta do Amazon Textract
- Localização do item em uma página de documento

Detectar texto

Amazon Textract fornece operações síncronas e assíncronas que retornam somente o texto detectado em um documento. Para ambos os conjuntos de operações, as seguintes informações são retornadas em váriosthe section called “Block”objects.

- As linhas e palavras do texto detectado
- As relações entre as linhas e as palavras do texto detectado
- A página em que o texto detectado aparece
- A localização das linhas e palavras do texto na página do documento

Para obter mais informações, consulte the section called “Linhas e palavras de texto”.

Para detectar texto de forma síncrona, use oDetectDocumentTextOperação da API e passe um arquivo de documento como entrada. Todo o conjunto de resultados é retornado pela operação. Para obter mais informações e um exemplo, consulte Processando documentos com operações síncronas.

> **Note**
>
> A operação da API do Amazon Rekognition`DetectText`é diferente do`DetectDocumentText`. Você usa`DetectText`para detectar texto em cenas ao vivo, como cartazes ou sinais de trânsito.

Para detectar texto de forma assíncrona, useStartDocumentTextDetectionpara começar a processar um arquivo de documento de entrada. Para obter resultados, ligue paraGetDocumentTextDetection. Os resultados são retornados em uma ou mais respostas de`GetDocumentTextDetection`. Para obter mais informações e um exemplo, consulte Processando documentos com operações assíncronas.

Analisar documentos

Amazon Textract analisa documentos e formulários para relacionamentos entre o texto detectado. As operações de análise Amazon Textract retornam 3 categorias de extração de documentos — texto, formulários e tabelas. A análise de faturas e recibos é tratada por meio de um processo diferente, para obter mais informações consulteAnalisando faturas e recibos.

Extração de texto

O texto bruto extraído de um documento. Para obter mais informações, consulteLinhas e palavras de texto.

Extração de formulário

Os dados do formulário são vinculados a itens de texto extraídos de um documento. Amazon Textract representa dados de formulário como pares de chave-valor. No exemplo a seguir, uma das linhas de texto detectadas pelo Amazon Textract éName (Nome): Jane Doe. Amazon Textract também identifica uma chave (Name (Nome):) e um valor (Jane Doe). Para obter mais informações, consulteDados do formulário (pares de valores-chave).

Name (Nome): Jane Doe

Endereço: 123 Any Street, Anytown, Estados Unidos da América

Data de nascimento: 12-26-1980

Os pares de valores-chave também são usados para representar caixas de seleção ou botões de opção (botões de opção) extraídos dos formulários.

Masculino:☑

Para obter mais informações, consulteElementos de seleção.

Extração de mesa

O Amazon Textract pode extrair tabelas, células de tabela e os itens dentro das células da tabela e pode ser programado para retornar os resultados em um arquivo JSON, .csv ou .txt.

Name (Nome)	Endereço
Ana Carolina	123 Any Town

Para obter mais informações, consulte Tabelas. Elementos de seleção também podem ser extraídos das tabelas. Para obter mais informações, consulteElementos de seleção.

Para itens analisados, Amazon Textract retorna o seguinte em váriosthe section called "Block"objects:

- As linhas e palavras do texto detectado
- O conteúdo dos itens detectados
- A relação entre itens detectados
- A página em que o item foi detectado
- A localização do item na página do documento

Você pode usar operações síncronas ou assíncronas para analisar texto em um documento. Para analisar o texto de forma síncrona, use oAnalyzeDocumente passe um documento como entrada.`AnalyzeDocument`retorna todo o conjunto de resultados. Para obter mais informações, consulte Analisando texto do documento com o Amazon Textract.

Para detectar texto de forma assíncrona, useStartDocumentAnalysispara iniciar o processamento. Para obter resultados, ligue paraGetDocumentAnalysis. Os resultados são retornados em uma ou mais respostas de`GetDocumentAnalysis`. Para obter mais informações e um exemplo, consulte Detectando ou analisando texto em um documento de várias páginas.

Para especificar qual tipo de análise executar, você pode usar o`FeatureTypes`parâmetro de entrada de lista. Adicione TABLES à lista para retornar informações sobre as tabelas detectadas no documento de entrada — por exemplo, células de tabela, texto de célula e elementos de seleção nas células. Adicione FORMS para retornar relacionamentos de palavras, como pares de chave-valor e elementos de seleção. Para executar os dois tipos de análise, adicione TABLES e FORMS ao`FeatureTypes`.

Todas as linhas e palavras detectadas no documento são incluídas na resposta (incluindo texto não relacionado ao valor de`FeatureTypes`).

Analisando faturas e recibos

O Amazon Textract extrai dados relevantes, como informações de contato, itens comprados e nome do fornecedor, de quase qualquer fatura ou recibo sem a necessidade de nenhum modelo ou configuração. As faturas e os recibos costumam usar vários layouts, dificultando e demorado extrair dados manualmente em escala. Amazon Textract usa ML para entender o contexto de faturas e

recibos e extrai automaticamente dados, como data de fatura ou recebimento, número da fatura ou do recibo, preços do item, valor total e condições de pagamento para atender às necessidades da sua empresa.

Amazon Textract também identifica nomes de fornecedores que são críticos para seus fluxos de trabalho, mas podem não ser identificados explicitamente. Por exemplo, o Amazon Textract pode encontrar o nome do fornecedor em um recibo, mesmo que ele seja indicado apenas dentro de um logotipo na parte superior da página sem uma combinação explícita de pares de valores-chave. O Amazon Textract também facilita a consolidação de dados de diversos recibos e faturas que usam palavras diferentes para o mesmo conceito. Por exemplo, o Amazon Textract mapeia relacionamentos entre nomes de campo em documentos diferentes, como número do cliente, número do cliente e ID da conta, gerando taxonomia padrão como`INVOICE_RECEIPT_ID`. Nesse caso, o Amazon Textract representa dados de forma consistente em diferentes tipos de documentos. Campos que não se alinham com a taxonomia padrão são categorizados como`OTHER`.

A seguir há uma lista dos campos padrão que o AnalyzeExpense suporta atualmente:

- Nome do fornecedor:`VENDOR_NAME`
- Total:`TOTAL`
- Endereço do receptor:`RECEIVER_ADDRESS`
- Data de fatura/recebimento:`INVOICE_RECEIPT_DATE`
- ID de fatura/recibo:`INVOICE_RECEIPT_ID`
- Termos de pagamento:`PAYMENT_TERMS`
- Subtotal:`SUBTOTAL`
- Data de vencimento:`DUE_DATE`
- Imposto:`TAX`
- ID do pagador do imposto da fatura (SSN/ITIN ou EIN):`TAX_PAYER_ID`
- Nome do item:`ITEM_NAME`
- Preço do item:`PRICE`
- Quantidade do item:`QUANTITY`

A API AnalyzeExpense retorna os seguintes elementos para uma determinada página de documento:

- O número de recebimentos ou faturas em uma página representada como`ExpenseIndex`
- O nome padronizado para campos individuais representados como`Type`

- O nome real do campo como exibido no documento, representado como`LabelDetection`
- O valor do campo correspondente representado como`ValueDetection`
- O número de páginas dentro do documento enviado representado como`Pages`
- O número da página em que o campo, o valor ou os itens de linha foram detectados, representado como`PageNumber`
- A geometria, que inclui a caixa delimitadora e a localização das coordenadas dos itens de campo, valor ou linha individuais na página, representados como`Geometry`
- A pontuação de confiança associada a cada pedaço de dados detectado no documento, representado como`Confidence`
- A linha inteira de itens de linha individuais comprados, representada como`EXPENSE_ROW`

Veja a seguir uma parte da saída da API para um recebimento processado pelo AnalyzeExpense que mostra o Total: US$55,64 no campo documento extraído como padrão`TOTAL`, texto real no documento como “Total”, Pontuação de Confiança de “97,1”, Número da Página “1”, O valor total como “$55,64" e a caixa delimitadora e as coordenadas do polígono:

```
{
    "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
```

```
}
```

Você pode usar operações síncronas para analisar uma fatura ou um recibo. Para analisar esses documentos, você usa a operação AnalyzeExpense e passa um recibo ou fatura para ela.`AnalyzeExpense`retorna todo o conjunto de resultados. Para obter mais informações, consulte Analisando faturas e recibos com o Amazon Textract.

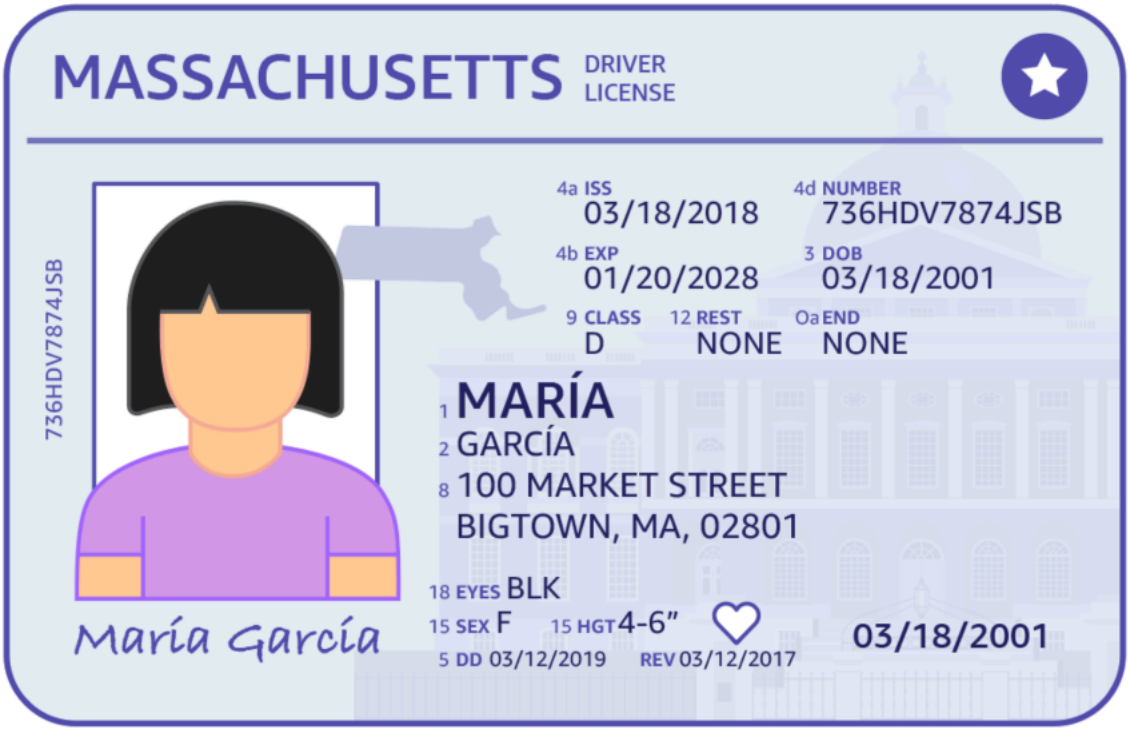
Para analisar faturas e recebimentos de forma assíncrona, useStartExpenseAnalysispara começar a processar um arquivo de documento de entrada. Para obter resultados, ligue paraGetExpenseAnalysis. Os resultados de uma determinada chamada paraStartExpenseAnalysissão retornados por`GetExpenseAnalysis`. Para obter mais informações e um exemplo, consulte Processando documentos com operações assíncronas.

Analisar documentos de identidade

Amazon Textract pode extrair informações relevantes de passaportes, carteiras de motorista e outras documentações de identidade emitidas pelo governo dos EUA usando a API AnalyzeID. Com o Analyze ID, as empresas podem extrair informações de IDs com rapidez e precisão, como carteiras de motorista dos EUA, IDs estaduais e passaportes com modelo ou formato diferente. A API AnalyzeID retorna duas categorias de tipos de dados:

- Pares de valores-chave disponíveis em ID, como Data de nascimento, Data de emissão, número de ID, Classe e Restrições.
- Campos implícitos no documento que podem não ter chaves explícitas associadas a eles, como Nome, Endereço e Emitido por.

Os nomes das chaves são padronizados dentro da resposta. Por exemplo, se sua carteira de motorista diz LIC# (número da licença) e passaporte diz Passaporte Não, a resposta do Analyze ID retornará a chave padronizada como “ID do documento” junto com a chave bruta (por exemplo, LIC#). Essa padronização permite que os clientes combinem facilmente informações em muitas IDs que usam termos diferentes para o mesmo conceito.



Analisar ID retorna informações nas estruturas chamadas`IdentityDocumentFields`. Estes são`JSO`Nestruturas contendo duas informações: o Tipo normalizado e o Valor associado ao Tipo. Ambos também têm uma pontuação de confiança. Para obter mais informações, consulte Objetos de resposta da documentação.

Você pode usar operações síncronas para analisar uma carteira de motorista ou passaporte. Para analisar esses documentos, você usa a operação AnalyzeID e passa um documento de identidade para ela.`AnalyzeID`retorna todo o conjunto de resultados. Para obter mais informações, consulte Analisando a documentação de identidade com o Amazon Textract.

> **Note**
>
> Alguns documentos de identidade, como carteiras de motorista, têm dois lados. Você pode passar as imagens de frente e verso das carteiras de driver como imagens separadas dentro da mesma solicitação da API Analyze ID.

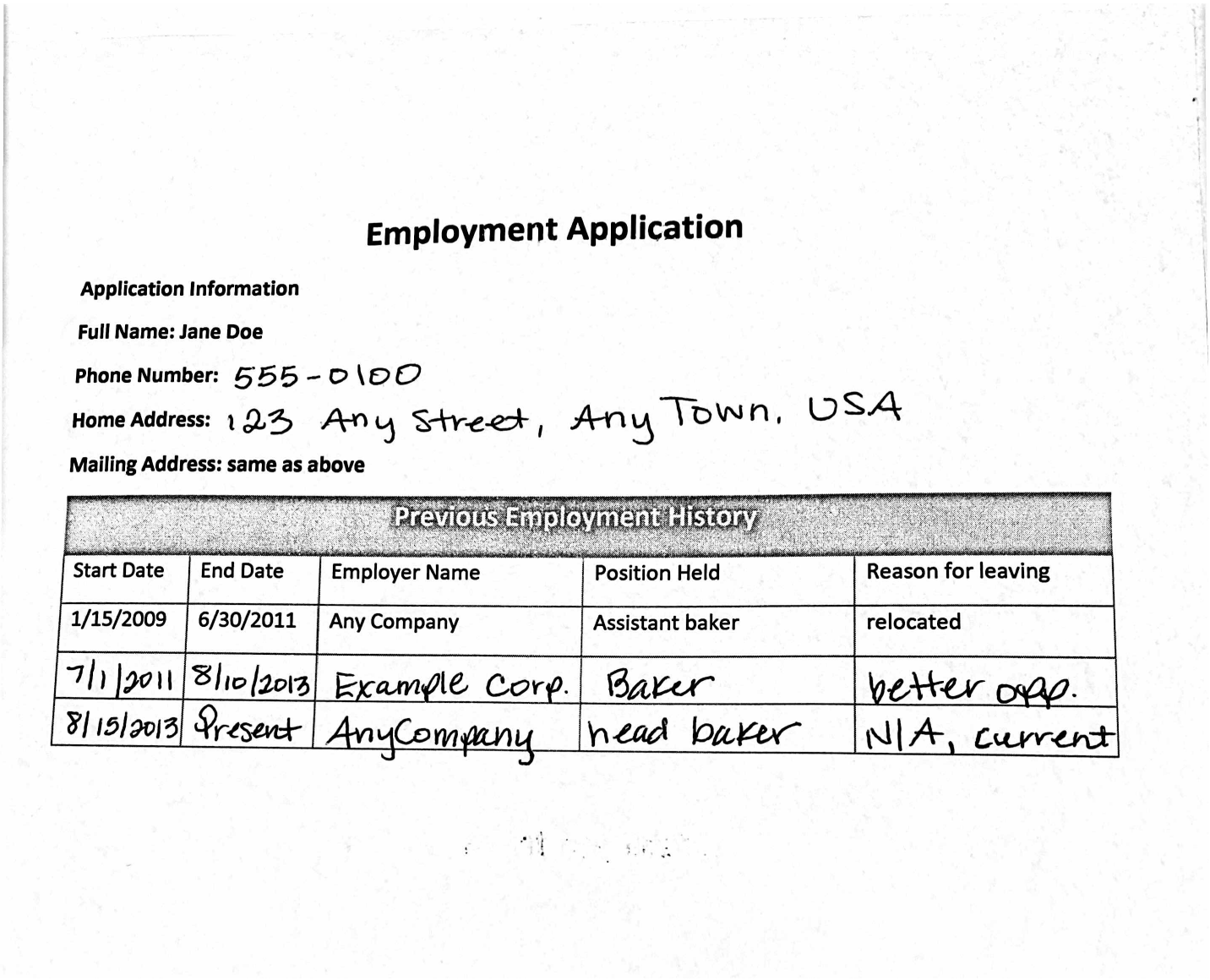
Documentos de entrada

Uma entrada adequada para uma operação Amazon Textract é um documento de uma ou várias páginas. Alguns exemplos são um documento legal, um formulário, um ID ou uma carta. Um

formulário é um documento com perguntas ou solicitações para que um usuário forneça respostas. Alguns exemplos são um formulário de registro de paciente, um formulário fiscal ou um formulário de reivindicação de seguro.

Um documento pode estar no formato JPEG, PNG, PDF ou TIFF. Com arquivos em formato PDF e TIFF, você pode processar documentos de várias páginas. Para obter informações sobre como Amazon Textract representa documentos como`Block`objetos, consulte[Objetos de resposta de detecção de texto e análise de](#).

A seguir há um exemplo de documento de entrada aceitável.



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

Para obter informações sobre os limites de documentos, consulte[Limites rígidos no Amazon Textract](#).

Para operações síncronas do Amazon Textract, você pode usar documentos de entrada armazenados em um bucket do Amazon S3 ou passar bytes de imagem codificados em base64. Para obter mais informações, consulte Chamando operações síncronas do Amazon Textract. Para operações assíncronas, você precisa fornecer documentos de entrada em um bucket do Amazon S3. Para obter mais informações, consulte Chamando operações assíncronas do Amazon Textract.

Objetos de resposta do Amazon Textract

As operações do Amazon Textract retornam diferentes tipos de objetos, dependendo da execução das operações. Para detectar texto e analisar um documento genérico, a operação retorna um objeto Block. Para analisar uma fatura ou recebimento, a operação retorna um objeto ExpenseDocuments. Para analisar a documentação de identidade, a operação retorna um objeto IdentityDocumentFields. Para obter mais informações sobre esses objetos de resposta, consulte as seguintes seções:

Tópicos

- Objetos de resposta de detecção de texto e análise de
- Objetos de resposta de fatura e recebimento
- Objetos de resposta da documentação

Objetos de resposta de detecção de texto e análise de

Quando o Amazon Textract processa um documento, ele cria uma lista deBlockobjetos para o texto detectado ou analisado. Cada bloco contém informações sobre um item detectado, onde ele está localizado e a confiança que o Amazon Textract tem na precisão do processamento.

Um documento é composto pelos seguintes tipos de`Block`objects.

- Páginas
- Linhas e palavras de texto
- Dados do formulário (pares de valores-chave)
- Tabelas e células
- Elementos de seleção

O conteúdo de um bloco depende da operação que você chama. Se você chamar uma das operações de detecção de texto, as páginas, as linhas e as palavras do texto detectado serão

retornadas. Para obter mais informações, consulte Detectar texto. Se você chamar uma das operações de análise de documentos, informações sobre páginas detectadas, pares de valores-chave, tabelas, elementos de seleção e texto serão retornadas. Para obter mais informações, consulte Analisar documentos.

AlgunsBlockOs campos de objeto são comuns a ambos os tipos de processamento. Por exemplo, cada bloco tem um identificador exclusivo.

Para obter exemplos que mostram como usarBlockobjetos, consulteTutoriais.

Layout do documento

Amazon Textract retorna uma representação de um documento como uma lista de diferentes tipos deBlockobjetos que estão vinculados em um relacionamento pai-filho ou em um par de valores-chave. Metadados que fornecem o número de páginas em um documento também são retornados. A seguir está a JSON para um típicoBlockobjeto do tipoPAGE.

```
{
    "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
    }
}
```

Um documento é feito de um ou mais`PAGE`Blocos. Cada página contém uma lista de blocos filhos para os itens primários detectados na página, como linhas de texto e tabelas. Para obter mais informações, consulte Páginas.

Você pode determinar o tipo de um`Block`objeto inspecionando o`BlockType`campo.

UMA`Block`Object contém uma lista de relacionados`Block`Objetos no`Relationships`campo, que é uma matriz deRelationshipobjects. UMA`Relationships`array é do tipo CHILD ou do tipo VALUE. Uma matriz do tipo CHILD é usada para listar os itens que são filhos do bloco atual. Por exemplo, se o bloco atual for do tipo LINE,`Relationships`contém uma lista de IDs para os blocos WORD que compõem a linha de texto. Uma matriz do tipo VALUE é usada para conter pares chave-valor. Você pode determinar o tipo de relacionamento inspecionando o`Type`campo do`Relationship`objeto.

Blocos filhos não têm informações sobre seus objetos Bloco pai.

Para exemplos que mostram`Block`Informações, consulteProcessando documentos com operações síncronas.

Confiança

As operações do Amazon Textract retornam a confiança percentual que o Amazon Textract tem sobre a precisão do item detectado. Para obter a confiança, use o`Confidence`campo do`Block`objeto. Um valor mais alto indica uma confiança mais alta. Dependendo do cenário, detecções com baixa confiança podem precisar de confirmação visual por um humano.

Geometria

As operações Amazon Textract, com exceção da análise de identidade, retornam informações de localização sobre a localização dos itens detectados em uma página de documento. Para obter a localização, use o`Geometry`campo do`Block`objeto. Para obter mais informações, consulteLocalização do item em uma página de documento

Páginas

Um documento consiste em uma ou mais páginas. UMAthe section called “Block”objeto do tipo`PAGE`existe para cada página do documento. UMA`PAGE`objeto de bloco contém uma lista das IDs filho para as linhas de texto, pares de valores-chave e tabelas detectadas na página do documento.

O JSON para um`PAGE`O bloco será semelhante à seguinte.

```
{

    "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
},
```

Se você estiver usando operações assíncronas com um documento de várias páginas em formato PDF, você pode determinar a página em que um bloco está localizado inspecionando o`Page`campo do`Block`objeto. Uma imagem digitalizada (uma imagem em formato JPEG, PNG, PDF ou TIFF) é considerada um documento de página única, mesmo se houver mais de uma página de documento na imagem. Operações assíncronas sempre retornam um`Page`valor de 1 para imagens digitalizadas.

O número total de páginas é retornado no`Pages`campo de`DocumentMetadata`.`DocumentMetadata`é retornado com cada lista de`Block`objetos retornados por uma operação do Amazon Textract.

Linhas e palavras de texto

O texto detectado retornado pelas operações Amazon Textract é retornado em uma lista dethe section called “Block”objects. Esses objetos representam linhas de texto ou palavras textuais detectadas em uma página de documento. O texto a seguir mostra duas linhas de texto que são feitas de várias palavras.

Isso é texto.

Em duas linhas separadas.

O texto detectado é retornado na`Text`campo de um`Block`objeto. O`BlockType`campo determina se o texto é uma linha de texto (LINE) ou uma palavra (WORD). UMAPALAVRAUm ou mais caracteres latinos ISO básicos não separados por espaços. UMALINHAé uma série de palavras delimitadas por tabulação e contíguas.

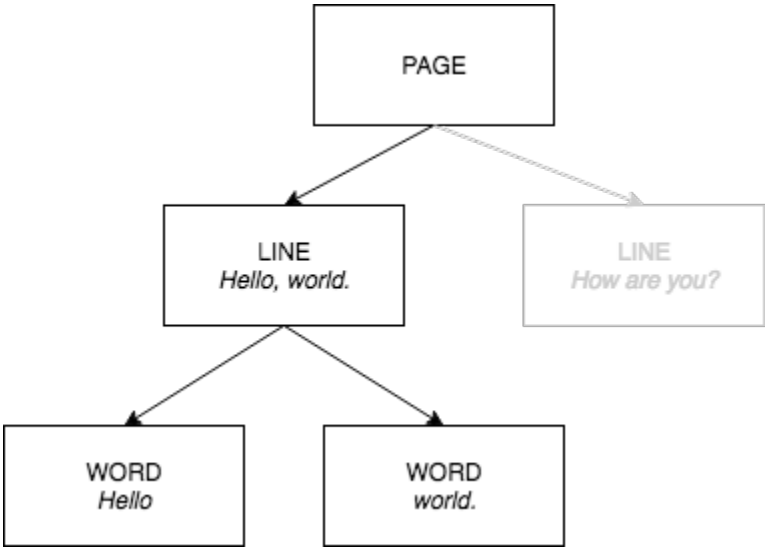
Além disso, o Amazon Textract determinará se um pedaço de texto foi escrito à mão ou impresso usando o`TextTypes`campo. Eles retornam como MANUSCRITA E IMPRESSA, respectivamente.

O outro`Block`as propriedades são comuns a todos os tipos de blocos, como informações de ID, confiança e geometria. Para obter mais informações, consulte the section called “Objetos de resposta de detecção de texto e análise de”.

Para detectar apenas linhas e palavras, você pode usarDetectDocumentTextouStartDocumentTextDetection. Para obter mais informações, consulte Detectar texto. Para obter o texto detectado (linhas e palavras) e informações sobre como ele se relaciona com outras partes do documento, como tabelas, você pode usarAnalyzeDocumentouStartDocumentAnalysis. Para obter mais informações, consulte Analisar documentos.

`PAGE`,`LINE`, e`WORD`blocos estão relacionados entre si em um relacionamento pai-filho. UMA`PAGE`block é o pai de todos`LINE`blocos de objetos em uma página de documento. Como um LINE pode ter uma ou mais palavras, a`Relationships`array para um bloco LINE armazena as IDs para blocos WORD filhos que compõem a linha de texto.

O diagrama a seguir mostra como a linhaOlá, mundo.no textoOlá, mundo. Como você está?é representado por`Block`objects.



A seguir está a saída JSON de`DetectDocumentText`quando a fraseOlá, mundo. Como você está?é detectado. O primeiro exemplo é o JSON para a página do documento. Observe como as IDs CHILD permitem que você navegue pelo documento.

```
{
    "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"
},
```

Veja a seguir o JSON para os blocos LINE que compõem a linha “Hello, World”:

```
{
    "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"
},
```

Veja a seguir o JSON para o bloco WORD para a palavraOlá,:

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

O JSON final é o bloco WORD para a palavramundo.:

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

Dados do formulário (pares de chave-valor)

Amazon Textract pode extrair dados de formulário de documentos como pares de chave-valor. Por exemplo, no texto a seguir, o Amazon Textract pode identificar uma chave (Name (Nome):) e um valor (Ana Carolina).

Name (Nome): Ana Carolina

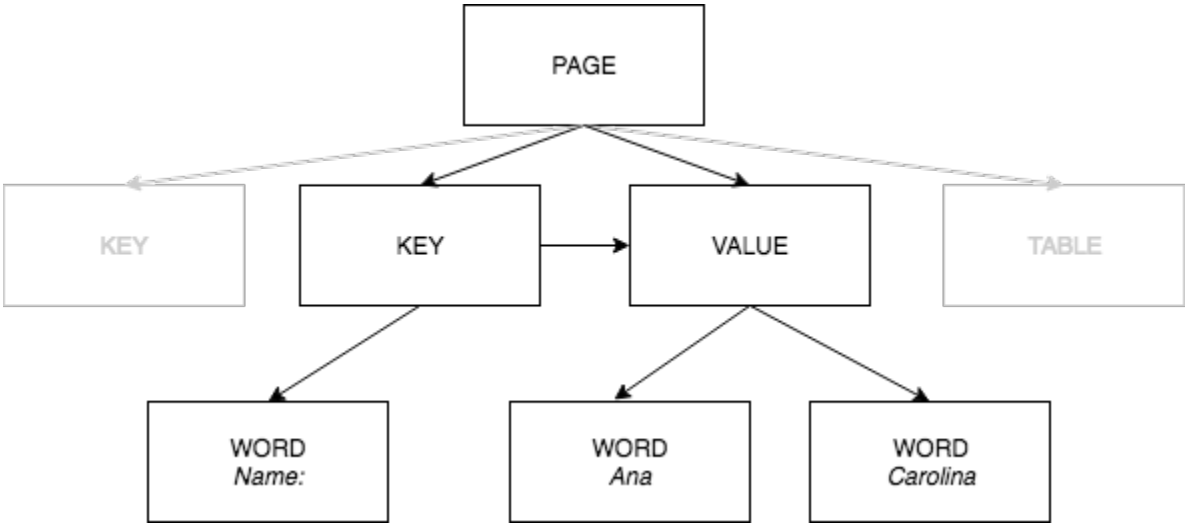
Os pares de chave-valor são retornados comoBlockobjetos nas respostas deAnalyzeDocumenteGetDocumentAnalysis. Você pode usar o`FeatureTypes`parâmetro de entrada para recuperar informações sobre pares de valores-chave, tabelas ou ambos. Apenas para pares de chave-valor, use o valor`FORMS`. Para ver um exemplo, consulte Extraindo pares de valores-chave de um documento de formulário. Para obter informações gerais sobre como um documento é representado por`Block`objetos, consulteObjetos de resposta de detecção de texto e análise de.

Objetos de bloco com o tipo KEY_VALUE_SET são os contêineres para objetos KEY ou VALUE Block que armazenam informações sobre itens de texto vinculados detectados em um documento. Você pode usar o`EntityType`atributo para determinar se um bloco é KEY ou VALUE.

- UMACHAVEO objeto contém informações sobre a chave do texto vinculado. Por exemplo,Name (Nome):. Um bloco KEY tem duas listas de relacionamento. Um relacionamento do tipo VALUE é uma lista que contém o ID do bloco VALUE associado à chave. Um relacionamento do tipo CHILD é uma lista de IDs para os blocos WORD que compõem o texto da chave.
- UMAVALORO objeto contém informações sobre o texto associado a uma chave. No exemplo anterior,Ana Carolinaé o valor da chaveName (Nome):. Um bloco VALUE tem um relacionamento com uma lista de blocos CHILD que identificam blocos WORD. Cada bloco WORD contém uma das palavras que compõem o texto do valor. UMA`VALUE`Objetos também podem conter informações sobre elementos selecionados. Para obter mais informações, consulte Elementos de seleção.

Cada instância de um KEY_VALUE_SET`Block`O objeto é um filho do PAGE`Block`objeto que corresponde à página atual.

O diagrama a seguir mostra como o par de chave-valorName (Nome): Ana Carolinaé representado por`Block`objects.



Os exemplos a seguir mostram como o par de chave-valorName (Nome): Ana Carolinaé representado por JSON.

O bloco PAGE tem blocos CHILD do tipo`KEY_VALUE_SET`para cada bloco KEY e VALUE detectado no documento.

```
{
    "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
},
```

O JSON a seguir mostra que o bloco KEY (52be1777-53f7-42f6-a7cf-6d09bdc15a30) tem uma relação com o bloco VALUE (7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c). Ele também tem um bloco CHILD para o bloco WORD (c734fca6-c4c4-415c-b6c1-30f7510b72ee) que contém o texto da chave (Name (Nome):).

```
{
    "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
},
```

O JSON a seguir mostra que o bloco VALUE 7ca7caa6-00ef-4cda-b1aa-5571dfed1a7c tem uma lista CHILD de IDs para os blocos WORD que compõem o texto do valor (AnaeCarolina).

```
{
    "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
}
```

O JSON a seguir mostra o`Block`objetos para as palavrasName (Nome):,Ana, eCarolina.

```
{
    "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"
},
```

Tabelas

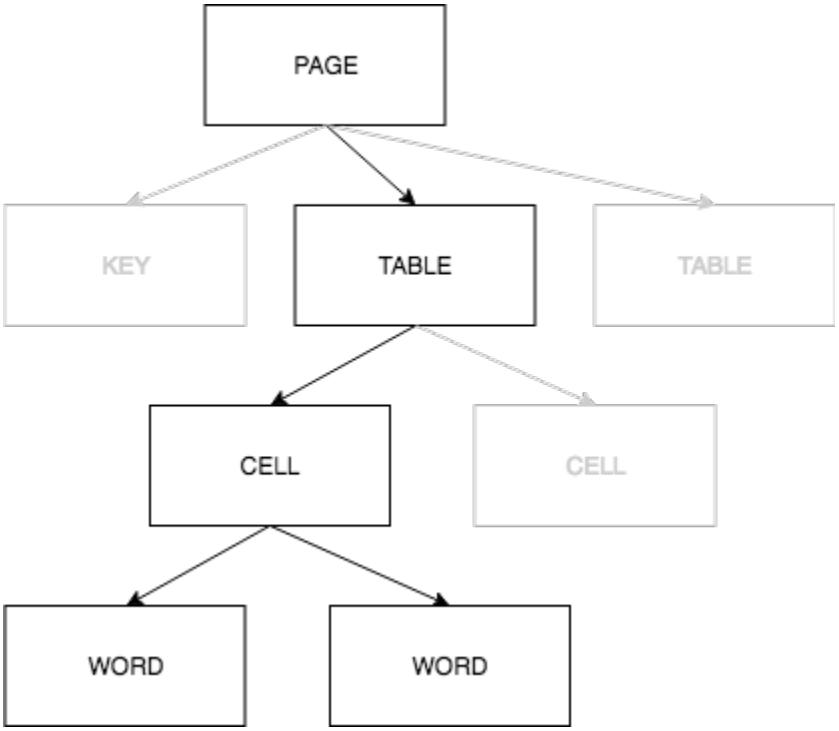
Amazon Textract pode extrair tabelas e as células em uma tabela. Por exemplo, quando a tabela a seguir é detectada em um formulário, o Amazon Textract detecta uma tabela com quatro células.

Name (Nome)	Endereço
Ana Carolina	123 Any Town

As tabelas detectadas são retornadas comoBlockobjetos nas respostas deAnalyzeDocumenteGetDocumentAnalysis. Você pode usar o`FeatureTypes`parâmetro de

entrada para recuperar informações sobre pares de valores-chave, tabelas ou ambos. Somente para tabelas, use o valor`TABLES`. Para ver um exemplo, consulte Exportando tabelas para um arquivo CSV. Para obter informações gerais sobre como um documento é representado por`Block`objetos, consulteObjetos de resposta de detecção de texto e análise de.

O diagrama a seguir mostra como uma única célula em uma tabela é representada por`Block`objects.



Uma célula contém`WORD`blocos para palavras detectadas e`SELECTION_ELEMENT`blocos para elementos de seleção, como caixas de seleção.

O seguinte é JSON parcial para a tabela anterior, que tem quatro células.

O objeto PAGE Block tem uma lista de IDs de bloco FILHO para o bloco TABLE e cada LINHA de texto detectada.

```
{
    "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"
},
```

O bloco TABLE inclui uma lista de IDs filhos para as células dentro da tabela. Um bloco TABLE também inclui informações de geometria para a localização da tabela no documento. O JSON a seguir mostra que a tabela tem quatro células, que estão listadas na`Ids`matriz.

```
{
    "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"
},
```

O tipo de bloco para as células da tabela é CELL. O`Block`objeto para cada célula inclui informações sobre a localização da célula em comparação com outras células na tabela. Ele também inclui informações de geometria para a localização da célula no documento. No exemplo anterior,`505e9581-0d1c-42fb-a214-6ff736822e8c`é o ID filho da célula que contém a palavraName (Nome). O exemplo a seguir são as informações para a célula.

```
{
    "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"
},
```

Cada célula tem uma localização em uma tabela, com a primeira célula sendo 1,1. No exemplo anterior, a célula com o valorName (Nome)está na linha 1, coluna 1. A célula com o valor123 Any Townestá na linha 2, coluna 2. Um objeto de bloco de células contém essas informações na`RowIndex`e`ColumnIndex`campos. A lista filho contém os IDs dos objetos do bloco WORD que contêm o texto que está dentro da célula. As palavras na lista estão na ordem em que são detectadas, do canto superior esquerdo da célula para o canto inferior direito da célula. No exemplo anterior, a célula tem um ID filho com o valor e9108c8e-0167-4482-989e-8b6cd3c3653e. A seguinte saída é para o Bloco WORD com o valor ID de e9108c8e-0167-4482-989e-8b6cd3c3653e:

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

Elementos de seleção

Amazon Textract pode detectar elementos de seleção, como botões de opção (botões de opção) e caixas de seleção em uma página de documento. Elementos de seleção podem ser detectados emDados do formulárioe emtabelas. Por exemplo, quando a tabela a seguir é detectada em um formulário, o Amazon Textract detecta as caixas de seleção nas células da tabela.

	Concordo	Neutral	Discordo

Bom serviço	☑	☐	☐
Fácil de usar	☐	☑	☐
Preço justo	☑	☐	☐

Os elementos de seleção detectados são devolvidos comoBlockobjetos nas respostas deAnalyzeDocumenteGetDocumentAnalysis.

> **Note**
>
> Você pode usar o`FeatureTypes`parâmetro de entrada para recuperar informações sobre pares de valores-chave, tabelas ou ambos. Por exemplo, se você filtrar em tabelas, a resposta incluirá os elementos de seleção detectados nas tabelas. Elementos de seleção detectados em pares de valores-chave não estão incluídos na resposta.

As informações sobre um elemento de seleção estão contidas em um`Block`objeto do tipo`SELECTION_ELEMENT`. Para determinar o status de um elemento selecionável, use a`SelectionStatus`campo do`SELECTION_ELEMENT`bloco. O status pode serSELECIONADOouNOT_SELECTED. Por exemplo, o valor de`SelectionStatus`para a imagem anterior éSELECIONADO.

UMA`SELECTION_ELEMENT` `Block`objeto está associado a um par de valores-chave ou a uma célula de tabela. UMA`SELECTION_ELEMENT` `Block`objeto contém informações da caixa delimitadora para um elemento de seleção na`Geometry`campo. UMA`SELECTION_ELEMENT` `Block`objeto não é filho de um`PAGE` `Block`objeto.

Dados do formulário (pares de chave-valor)

Um par de valores-chave é usado para representar um elemento de seleção detectado em um formulário. O`KEY`bloco contém o texto para o elemento de seleção. O`VALUE`bloco contém o bloco SELECTION_ELEMENT. O diagrama a seguir mostra como os elementos de seleção são representados porthe section called “Block”objects.

Para obter mais informações sobre pares de chave-valor, consulteDados do formulário (pares de chave-valor).

O trecho JSON a seguir mostra a chave para um par de valores-chave que contém um elemento de seleção (male ☑). O ID filho (Id bd14cfd5-9005-498b-a7f3-45ceb171f0ff) é o ID do bloco WORD que contém o texto para o elemento de seleção (macho). O valor ID (Id 24aaac7f-fcce-49c7-a4f0-3688b05586d4) é o ID do`VALUE`bloco que contém o`SELECTION_ELEMENT`objeto de bloco.

```
{
    "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"
}
```

O trecho JSON a seguir é o bloco WORD para a palavraMasculino. O bloco WORD também tem um bloco LINE pai.

```
{
    "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"
},
```

O bloco VALUE tem um filho (Id f2f5e8cd-e73a-4e99-a095-053acd3b6bfb) que é o bloco SELECTION_ELEMENT.

```
{
    "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"
},
}
```

O JSON a seguir é o bloco SELECTION_ELEMENT. O valor de`SelectionStatus`Indica que a caixa de seleção está marcada.

```
{
    "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"
```

```
}
```

Células do

Amazon Textract pode detectar elementos de seleção dentro de uma célula de tabela. Por exemplo, as células na tabela a seguir têm caixas de seleção.

	Concordo	Neutral	Discordo
Bom serviço	☑	☐	☐
Fácil de usar	☐	☑	☐
Preço justo	☑	☐	☐

UMA`CELL`bloco pode conter filho`SELECTION_ELEMENT`objetos para elementos de seleção, bem como filho`WORD`blocos para texto detectado.

Para obter mais informações sobre tabelas, consulteTabelas.

O TABLE`Block`objeto para a tabela anterior é semelhante a este.

```
{
    "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"
}
```

O CELLBLOCKobjeto (Id c63ad40d-5a14-4646-a8df-2d4304213dbc) para a célula que contém a caixa de seleçãoBom serviçoParece o seguinte. Inclui uma criançaBlock(Id = 26d122fd-c5f4-4b53-92c4-0ae92730ee1e) que é oSELECTION_ELEMENT BlockObjeto da caixa de seleção.

```
{
    "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"
}
```

O SELECTION_ELEMENTBlockobjeto para a caixa de seleção é o seguinte. O valor deSelectionStatusIndica que a caixa de seleção está marcada.

```
{
    "Geometry": {.......},
    "BlockType": "SELECTION_ELEMENT",
    "SelectionStatus": "SELECTED",
```

```
    "Confidence": 88.79517364501953,
    "Id": "26d122fd-c5f4-4b53-92c4-0ae92730ee1e"
}
```

Objetos de resposta de fatura e recebimento

Quando você envia uma fatura ou um recibo para a API AnalyzeExpense, ela retorna uma série de objetos ExpenseDocuments. Cada documento de despesa é ainda separado em`LineItemGroupseSummaryFields`. A maioria das faturas e recibos contém informações como o nome do fornecedor, o número do recebimento, a data do recebimento ou o valor total. AnalyzeExpense retorna essas informações em`SummaryFields`. Recibos e faturas também contêm detalhes sobre os itens comprados. A API AnalyzeExpense retorna essas informações em`LineItemGroups`. O`ExpenseIndex`campo identifica exclusivamente a despesa e associa o apropriado`SummaryFieldseLineItemGroups`detectado nessa despesa.

O nível mais granular de dados na resposta AnalyzeExpense consiste em`Type`,`ValueDetection`, e`LabelDetection`(Optional). As entidades individuais são:

- Type: Refere-se a que tipo de informação é detectada em um nível alto.
- Detecção de etiquetas: Refere-se ao rótulo de um valor associado dentro do texto do documento.`LabelDetection`é opcional e retornado somente se o rótulo for escrito.
- ValueDetecção: Refere-se ao valor do rótulo ou tipo retornado.

A API AnalyzeExpense também detecta`ITEM`,`QUANTITY`, e`PRICE`dentro de itens de linha como campos normalizados. Se houver outro texto em um item de linha na imagem do recibo, como SKU ou descrição detalhada, ele será incluído no JSON como`EXPENSE_ROW`como mostrado no exemplo abaixo:

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

```
                                        "Confidence": 98.11214447021484
                                    }
```

O exemplo acima mostra como a API AnalyzeExpense retorna a linha inteira em um recibo que contém informações de item de linha sobre 5 bananas vendidas por US$2,5.

Type

A seguir há um exemplo do tipo padrão ou normalizado do par de chave-valor:

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

O recibo não tinha “Nome do fornecedor” listado explicitamente. No entanto, a API Analyze Despense reconheceu o documento como um recibo e categorizou o valor “AMAZON” como Tipo`VENDOR_NAME`.

Detecção de etiquetas

Veja a seguir um exemplo de texto, conforme mostrado em uma página de documento do cliente:

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

```
                "Text": "CASHIER",
                "Confidence": 88.19171142578125
            },
            "ValueDetection": {
                "Geometry": { ... },
                "Text": "Mina",
                "Confidence": 87.89806365966797
            }
        }
```

O documento de exemplo continha “CAIXA Mina”. A API Analyze Expense extraiu o valor como está e o retorna em`LabelDetection`. Para valores implícitos, como “Nome do fornecedor”, onde a “chave” não é mostrada explicitamente no recibo,`LabelDetection`não será incluído no elemento AnalyzeExpense. Nesses casos, a API AnalyzeExpense não retorna`LabelDetection`.

ValueDetecção

A seguir há um exemplo que mostra o “valor” do par chave-valor.

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

No exemplo, o documento continha “CAIXA Mina”. A API AnalyzeExpense detectou o valor do Caixa como Mina e o retornou em`ValueDetection`.

Objetos de resposta da documentação

Quando você envia um documento de identidade para a API AnalyzeID, ele retorna uma série de`IdentityDocumentField`objects. Cada um desses objetos contém`Type`, e`Value`.`Type`registra o campo normalizado que o Amazon Textract detecta e`Value`registra o texto associado ao campo normalizado.

Abaixo está um exemplo de um`IdentityDocumentField`, encurtado por brevidade.

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

Esses são dois exemplos de IdentityDocumentFields cortados de uma resposta mais longa.
Há uma separação entre o tipo detectado e o valor desse tipo. Aqui, é o nome e sobrenome, respectivamente. Essa estrutura se repete com todas as informações contidas. Se um tipo não for reconhecido como um campo normalizado, ele será listado como “outro”.

Veja a seguir uma lista de campos normalizados para Carteiras de Motorista:

- Nome
- sobrenome
- Nome do meio
- sufixo
- city in address
- CEP no endereço
- estado no endereço
- Condado
- número do documento
- Data de validade
- Data de nascimento
- Nome do estado
- Data de emissão
- classe
- restrições
- endossos
- ID do tipo
- veterano
- address

Veja a seguir uma lista de campos normalizados para passaportes dos EUA:

- Nome
- sobrenome
- Nome do meio
- número do documento
- Data de validade
- Data de nascimento
- local de nascimento
- Data de emissão
- ID do tipo

Localização do item em uma página de documento

As operações do Amazon Textract retornam a localização e a geometria dos itens encontrados em uma página de documento.DetectDocumentTexteGetDocumentTextDetectionretorna a localização e a geometria para linhas e palavras, enquantoAnalyzeDocumenteGetDocumentAnalysisRetorna a localização e a geometria de pares de chave-valor, tabelas, células e elementos de seleção.

Para determinar onde um item está em uma página de documento, use a caixa delimitadora (Geometry) informações retornadas pela operação do Amazon Textract em umBlockobjeto. O`Geometry`O objeto contém dois tipos de localização e informações geométricas para os itens detectados:

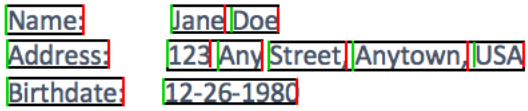
- Um eixo alinhadoBoundingBoxO objeto que contém a coordenada superior esquerda e a largura e a altura do item.
- Um objeto polígono que descreve o contorno do item, especificado como uma matriz dePointobjetos que contêm`X`(eixo horizontal) e`Y`(eixo vertical) coordenadas da página do documento de cada ponto.

O JSON para um`Block`O objeto será semelhante à seguinte. Observe o`BoundingBox`e`Polygon`campos.

```
{
    "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"
},
```

Você pode usar informações de geometria para desenhar caixas delimitadoras em torno de itens detectados. Para um exemplo que usa`BoundingBox`e`Polygon`informações para desenhar caixas em torno de linhas e linhas verticais no início e no final de cada palavra, consulte[Detectando texto do documento com o Amazon Textract](). A saída de exemplo é semelhante à seguinte.



Bounding Box

Uma caixa delimitadora (`BoundingBox`) tem as seguintes propriedades:

- Altura — Altura a altura da caixa delimitadora como uma proporção da altura total da página do documento.
- Esquerda — A coordenada X do ponto superior esquerdo da caixa delimitadora como uma proporção da largura total da página do documento.
- Superior — A coordenada Y do ponto superior esquerdo da caixa delimitadora como uma proporção da altura total da página do documento.
- Largura — a largura da caixa delimitadora como uma proporção da largura total da página do documento.

Cada propriedade BoundingBox tem um valor entre 0 e 1. O valor é uma proporção da largura total da imagem (aplica-se a`Left`e`Width`) ou altura (aplica-se a`Height`e`Top`). Por exemplo, se a

imagem de entrada tiver 700 x 200 pixels e a coordenada superior esquerda da caixa delimitadora tiver (350,50) pixels, a API retornará um`Left`valor de 0,5 (350/700) e um`Top`valor de 0,25 (50/200).

O diagrama a seguir mostra o intervalo de uma página de documento que cada propriedade BoundingBox abrange.

Para exibir a caixa delimitadora com o local e o tamanho corretos, você precisa multiplicar os valores da BoundingBox pela largura ou altura da página do documento (dependendo do valor que deseja) para obter os valores de pixels. Você pode usar os valores de pixel para exibir a caixa delimitadora. Um exemplo é usar uma página de documento de 608 pixels de largura x 588 pixels de altura e os seguintes valores de caixa delimitadora para texto analisado:

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

A localização da caixa delimitadora de texto em pixels é calculada da seguinte forma:

```
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
```

Você pode usar esses valores para exibir uma caixa delimitadora em torno do texto analisado. Os exemplos de Java e Python a seguir demonstram como exibir uma caixa delimitadora.

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

Este exemplo em Python leva na`response`retornado pelo[DetectDocumentText](#)Operação da API.

```
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)
```

Polígono

O polígono retornado por`AnalyzeDocumente`é uma matriz de`Point`objects. EACH`Point`tem uma coordenada X e Y para um local específico na página do documento. Como as coordenadas BoundingBox, as coordenadas do polígono são normalizadas para a largura e a altura do documento, e estão entre 0 e 1.

Você pode usar pontos na matriz de polígonos para exibir uma caixa delimitadora de grãos mais finos em torno de um`Block`objeto. Você calcula a posição de cada ponto de polígono na página do documento usando a mesma técnica usada para`BoundingBoxes`. Multiplique a coordenada X pela largura da página do documento e multiplique a coordenada Y pela altura da página do documento.

O exemplo a seguir mostra como exibir as linhas verticais de um polígono.

```
   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));

   }
```

Conceitos básicos do Amazon Textract

Esta seção apresenta tópicos sobre conceitos básicos do uso do Amazon Textract. Caso você seja novo no Amazon Textract, recomendamos analisar primeiramente a terminologia e os conceitos emComo funciona o Amazon Textract.

Você pode tentar a API usando a demonstração no console do Amazon Textract. Para obter mais informações, consultehttps://console.aws.amazon.com/textract/.

Tópicos

- Etapa 1: Configurar uma conta da AWS e criar um usuário do IAM
- Etapa 2: Configurar aAWS CLIeAWSSDKs da
- Etapa 3: Conceitos básicos do uso daAWS CLIeAWSAPI SDK

Etapa 1: Configurar uma conta da AWS e criar um usuário do IAM

Antes de usar o Amazon Textract pela primeira vez, execute as seguintes tarefas:

1. Cadastre-se na AWS.
2. Criar um usuário do IAM.

Cadastre-se na AWS

Quando você se cadastra na Amazon Web Services (AWS), a conta da AWS é cadastrada automaticamente em todos os serviços lançados na AWS. Você será cobrado apenas pelos serviços que usar.

Com o Amazon Textract, você paga apenas pelos recursos que usa. Para obter mais informações sobre as tarifas de uso do Amazon Textract, consulteDefinição de preço do Amazon Textract. Se você for um cliente novo da AWS, poderá começar a usar gratuitamente o Amazon Textract. Para obter mais informações, consulte Nível de uso gratuito da AWS.

Caso você já tenha uma conta da AWS, passe à próxima tarefa. Caso você ainda não tenha uma conta da AWS, realize as etapas no procedimento a seguir para criar uma.

Para criar uma conta da AWS

1. Abra https://portal.aws.amazon.com/billing/signup.
2. Siga as instruções online.

   Parte do procedimento de cadastro envolve uma chamada telefônica e a digitação de um código de verificação usando o teclado do telefone.

Observe o ID da conta da AWS porque você precisará dele na próxima tarefa.

Criar um usuário do IAM

Os serviços da AWS, como o Amazon Textract, exigem o fornecimento de credenciais quando acessados. Dessa maneira, o serviço pode determinar se você tem permissões para acessar os recursos próprios desse serviço. O console requer sua senha. Você pode criar chaves de acesso para a conta da AWS a fim de acessar a AWS CLI ou a API. No entanto, não recomendamos que você acesse a AWS usando as credenciais da conta da AWS. Em vez disso, recomendamos o uso de:

- Usar oAWS Identity and Access Management(IAM) para criar um usuário do IAM.
- Adicione o usuário a um grupo do IAM com permissões de administrador.

Em seguida, você pode acessar a AWS usando uma URL especial e as credenciais desse usuário do IAM.

Se tiver se cadastrado na AWS, mas não tiver criado um usuário do IAM para você mesmo, poderá criar um usando o console do IAM. Siga o procedimento para criar um usuário do IAM na conta.

Para criar um usuário do IAM e fazer login no console

1. Crie um usuário do IAM com permissões do administrador na conta da AWS. Para obter instruções, consulteCriação do primeiro usuário do IAM e do grupo de administradoresnoManual do usuário do IAM.
2. Assim como o usuário do IAM, faça login no AWS Management Console usando uma URL especial. Para obter mais informações, consulteComo usuários fazem login em sua contanoManual do usuário do IAM.

> **Note**
>
> Um usuário do IAM com permissões de administrador tem acesso irrestrito aoAWSserviços em sua conta. Os exemplos de código neste guia pressupõem que você tenha um usuário do com a`AmazonTextractFullAccess`permissões.`AmazonS3ReadOnlyAccess`é obrigatório para exemplos que acessam documentos armazenados em um bucket do Amazon S3. Dependendo dos requisitos de segurança, convém usar um grupo do IAM que seja limitado a essas permissões. Para obter mais informações, consulteCriação de grupos do IAM.

Para obter mais informações sobre IAM, consulte o seguinte:

- AWS Identity and Access Management (IAM)
- Conceitos básicos
- Guia do usuário do IAM

Próxima etapa

Etapa 2: Configurar aAWS CLIeAWSSDKs da

Etapa 2: Configurar aAWS CLIeAWSSDKs da

As etapas a seguir mostram como instalar a AWS Command Line Interface (AWS CLI) e os SDKs da AWS usados pelos exemplos nesta documentação.

Existem várias maneiras diferentes de autenticar chamadas de SDK da AWS. Os exemplos neste guia presumem que você esteja usando um perfil de credenciais padrão para chamar comandos da AWS CLI e operações de API do SDK da AWS. Suas credenciais padrão funcionarão em todos os serviços, portanto, se você já configurou suas credenciais, não precisará fazê-lo novamente. No entanto, se você quiser criar outro conjunto de credenciais para esse serviço, você pode criar um perfil de nome. Para obter mais informações sobre como criar perfis,consulte Perfis nomeados.

Para ver uma lista dos disponíveisAWSRegiões, consulteRegiões e endpoints donoReferência geral do Amazon Web Services.

Para configurar a AWS CLI e os SDKs da AWS

1. Faça download e instale a AWS CLI e os AWS SDKs que você deseja usar. Este guia apresenta exemplos daAWS CLI, Java e Python. Para obter mais informações sobre outros AWS SDKs, consulte Ferramentas para a Amazon Web Services.

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

2. Crie uma chave de acesso para o usuário criado por você emCriar um usuário do IAM.

   a. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.

   b. No painel de navegação, escolha Usuários.

   c. Selecione o nome do usuário criado por você emCriar um usuário do IAM.

   d. Selecione a guia Credenciais de segurança.

   e. Selecione Create access key (Criar chave de acesso). Em seguida, selecione o arquivo Download.csv para salvar o ID de chave de acesso e a chave de acesso secreta em um arquivo CSV em seu computador. Armazene o arquivo em um lugar seguro. Você não terá mais acesso à chave de acesso secreta depois que essa caixa de diálogo for fechada. Depois de baixar do arquivo CSV, selecioneFechar.

3. Defina credenciais no arquivo de perfil de credenciais da AWS no sistema local, localizado em:

   - `~/.aws/credentials`no Linux, macOS ou Unix.
   - `C:\Users\USERNAME\.aws\credentials`no Windows.

   O `.aws`a pasta não existe antes da primeira configuração inicial da instância da AWS. Na primeira vez que você configurar suas credenciais com a CLI, essa pasta será criada. Para obter mais informações sobre credenciais da AWS, consulteConfigurações de arquivos de configuração e credenciais.

   Esse arquivo deve conter linhas no seguinte formato:

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

Substitua o ID da chave de acesso e a chave de acesso secretayour_access_key_ideyour_secret_access_key.

4. Defina a região da AWS padrão na AWS`config`no sistema local, localizado em:

   - `~/.aws/config`no Linux, macOS ou Unix.
   - `C:\Users\USERNAME\.aws\config`no Windows.

   O`.aws`a pasta não existe antes da primeira configuração inicial da instância da AWS. Na primeira vez que você configurar suas credenciais com a CLI, essa pasta será criada. Para obter mais informações sobre credenciais da AWS, consulteConfigurações de arquivos de configuração e credenciais.

   Esse arquivo deve conter as seguintes linhas:

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

   Substitua a região da AWS desejada (por exemplo, “us-west-2") poryour_aws_region.

   > **Note**
   > Se você não escolher uma região, us-east-1 será usada por padrão.

Próxima etapa

Etapa 3: Conceitos básicos do uso daAWS CLIeAWSAPI SDK

Etapa 3: Conceitos básicos do uso daAWS CLIeAWSAPI SDK

Depois de configurar oAWS CLIeAWSOs SDKs que deseja usar, você pode compilar aplicativos que usam o Amazon Textract. Os tópicos a seguir mostram como começar a usar o Amazon Textract.

- Analisando texto do documento com o Amazon Textract

Formatação dos exemplos de AWS CLI

Os exemplos de AWS CLI neste guia são formatados para o sistema operacional Linux. Para usar as amostras com o Microsoft Windows, você precisa alterar a formatação JSON do parâmetro `--document` e mudar as quebras de linha de barras invertidas (\) para acentos circunflexos (^). Para obter mais informações sobre a formatação JSON, consulte Especificação dos valores de parâmetro para a interface da linha de comando da AWS.

Processando documentos com operações síncronas

O Amazon Textract pode detectar e analisar texto em documentos de página única que são fornecidos como imagens nos formatos JPEG, PNG, PDF e TIFF. As operações são síncronas e retornam resultados quase em tempo real. Para obter mais informações sobre documentos, consulte Objetos de resposta de detecção de texto e análise de.

Esta seção aborda como você pode usar o Amazon Textract para detectar e analisar texto em um documento de página única de forma síncrona. Para detectar e analisar texto em documentos de várias páginas ou para detectar documentos JPEG e PNG de forma assíncrona, consulteProcessando documentos com operações assíncronas.

É possível usar operações síncronas do Amazon Textract para as seguintes finalidades:

- Detecção de texto — Você pode detectar linhas e palavras em uma imagem de documento de página única usando oDetectDocumentTextoperação. Para obter mais informações, consulte Detectar texto.
- Análise de texto — Você pode identificar relacionamentos entre o texto detectado em um documento de página única usando oAnalyzeDocumentoperação. Para obter mais informações, consulte Analisar documentos.
- Análise de NFF e Recebimento — Você pode identificar relações financeiras entre o texto detectado em uma NFF ou recebimento de uma única página usando a operação AnalyzeExpense. Para obter mais informações, consulteAnalisando faturas e recibos
- Análise de documentos de identidade — Você pode analisar documentos de identidade emitidos pelo governo dos EUA e extrair informações junto com tipos comuns de informações encontradas em documentos de identidade. Para obter mais informações, consulteAnalisar documentos de identidade.

Tópicos

- Chamando operações síncronas do Amazon Textract
- Detectando texto do documento com o Amazon Textract
- Analisando texto do documento com o Amazon Textract
- Analisando faturas e recibos com o Amazon Textract
- Analisando a documentação de identidade com o Amazon Textract

Chamando operações síncronas do Amazon Textract

As operações Amazon Textract processam imagens de documentos armazenadas em um sistema de arquivos local ou imagens de documento armazenadas em um bucket do Amazon S3. Você especifica onde o documento de entrada está localizado usando oDocumentParâmetros de entrada. A imagem do documento pode estar no formato PNG, JPEG, PDF ou TIFF. Os resultados das operações síncronas são retornados imediatamente e não são armazenados para recuperação.

Para obter um exemplo completo, consulteDetectando texto do documento com o Amazon Textract.

Solicitação

A seguir descreve como as solicitações funcionam no Amazon Textract.

Documentos passados como bytes de imagem

Você pode passar uma imagem de documento para uma operação Amazon Textract passando a imagem como uma matriz de bytes codificada em base64. Um exemplo é uma imagem de documento carregada de um sistema de arquivos local. Seu código pode não precisar codificar bytes de arquivo de documento se você estiver usando umAWSSDK para chamar as operações da API Amazon Textract.

Os bytes de imagem são especificados na`Bytes`campo do`Document`Parâmetros de entrada. O exemplo a seguir mostra o JSON de entrada para uma operação do Amazon Textract que passa os bytes de imagem na`Bytes`Parâmetros de entrada.

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

> **Note**
>
> Se você estiver usando oAWS CLI, você não pode passar bytes de imagem para operações Amazon Textract. Em vez disso, é necessário fazer referência a uma imagem armazenada em um bucket do Amazon S3.

O código Java a seguir mostra como carregar uma imagem de um sistema de arquivos local e chamar uma operação Amazon Textract.

```
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);
```

Documentos armazenados em um bucket do Amazon S3

O Amazon Textract pode analisar imagens de documentos armazenadas em um bucket do Amazon S3. Especifique o bucket e o nome do arquivo usando oS3Objectcampo do`Document`Parâmetros de entrada. O exemplo a seguir mostra o JSON de entrada para uma operação Amazon Textract que processa um documento armazenado em um bucket do Amazon S3.

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

O exemplo a seguir mostra como chamar uma operação Amazon Textract usando uma imagem armazenada em um bucket do Amazon S3.

```
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);
```

Resposta

O exemplo a seguir é a resposta JSON de uma chamada para o`DetectDocumentText`. Para obter mais informações, consulte Detectar texto.

```
{
{
  "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
  }
}
}
```


Detectando texto do documento com o Amazon Textract

Para detectar texto em um documento, você usa o`DetectDocumentText`e passe um arquivo de documento como entrada.`DetectDocumentText`retorna uma estrutura JSON que contém linhas e palavras do texto detectado, a localização do texto no documento e as relações entre o texto detectado. Para obter mais informações, consulte Detectar texto.

Você pode fornecer um documento de entrada como uma matriz de bytes de imagem (bytes de imagem codificados em base64) ou um objeto do Amazon S3. Neste procedimento, você carrega um arquivo de imagem no bucket do S3 e especifica o nome do arquivo.

Para detectar texto em um documento (API do)

1. Se ainda não tiver feito isso:

   a. Criar ou atualizar um usuário do IAM com`AmazonTextractFullAccess`e`AmazonS3ReadOnlyAccess`permissões. Para obter mais informações, consulte Etapa 1: Configurar uma conta da AWS e criar um usuário do IAM.

   b. Instale e configure a AWS CLI e os SDKs da AWS. Para obter mais informações, consulte Etapa 2: Configurar aAWS CLIeAWSSDKs da.

2. Carregue um documento para o bucket do S3.

   Para obter instruções, consulteCarregar objetos no Amazon S3noGuia do usuário do Amazon Simple Storage Service.

3. Use os exemplos a seguir para chamar a operação `DetectDocumentText`.

   Java

   O código de exemplo a seguir exibe o documento e as caixas em torno de linhas de texto detectado.

   Na função`main`, substitua os valores de`bucket`e`document`Com os nomes do bucket do Amazon S3 e do documento usado na etapa 2.

```
//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

Esse comando da AWS CLI exibe a saída JSON da operação da CLI `detect-document-text`.

Substitua os valores de`BucketeNameCom` os nomes do bucket do Amazon S3 e do documento usado na etapa 2.

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

Python

O código de exemplo a seguir exibe o documento e as caixas ao redor de linhas de texto detectadas.

Na função`main`, substitua os valores de`bucket`e`document`Com os nomes do bucket do Amazon S3 e do documento usado na etapa 2.

```
#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

O código de exemplo Node.js a seguir exibe o documento e as caixas em torno de linhas de texto detectadas, gerando uma imagem dos resultados para o diretório do qual você executa o código. Ele faz uso do`image-size`e`images`Pacotes.

Na função`main`, substitua os valores de`bucket`e`document`Com os nomes do bucket do Amazon S3 e do documento usado na etapa 2. Substitua o valor de`regionConfig`com o nome da região em que sua conta está.

```
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. Execute o exemplo. Os exemplos Python e Java exibem a imagem do documento. Uma caixa preta envolve cada linha do texto detectado. Uma linha vertical verde é o início de uma palavra

detectada. Uma linha vertical vermelha é o fim de uma palavra detectada. OAWS CLIO exemplo da exibe somente a saída JSON do`DetectDocumentText`operação.

Analisando texto do documento com o Amazon Textract

Para analisar texto em um documento, você usa oAnalyzeDocumente passe um arquivo de documento como entrada.`AnalyzeDocument`Retorna uma estrutura JSON que contenha o texto analisado. Para obter mais informações, consulte Analisar documentos.

Você pode fornecer um documento de entrada como uma matriz de bytes de imagem (bytes de imagem codificados em base64) ou um objeto do Amazon S3. Neste procedimento, você carrega um arquivo de imagem no bucket do S3 e especifica o nome do arquivo.

Para analisar texto em um documento (API)

1. Se ainda não tiver feito isso:

   a. Criar ou atualizar um usuário do IAM com`AmazonTextractFullAccess`e`AmazonS3ReadOnlyAccess`permissões. Para obter mais informações, consulte Etapa 1: Configurar uma conta da AWS e criar um usuário do IAM.

   b. Instale e configure a AWS CLI e os SDKs da AWS. Para obter mais informações, consulte Etapa 2: Configurar aAWS CLIeAWSSDKs da.

2. Carregue uma imagem que contenha um documento no bucket do S3.

   Para obter instruções, consulteCarregar objetos no Amazon S3noGuia do usuário do Amazon Simple Storage Service.

3. Use os exemplos a seguir para chamar a operação `AnalyzeDocument`.

   Java

   O código de exemplo a seguir exibe o documento e as caixas ao redor de itens detectados.

   Na função`main`, substitua os valores de`bucket`e`document`Com os nomes do bucket do Amazon S3 e da imagem usada na etapa 2.

```
//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

Esse comando da AWS CLI exibe a saída JSON da operação da CLI `detect-document-text`.

Substitua os valores de`BucketeNameCom` os nomes do bucket do Amazon S3 e do documento usado na etapa 2.

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

Python

O código de exemplo a seguir exibe o documento e as caixas ao redor de itens detectados.

Na função`main`, substitua os valores de`bucket`e`document`Com os nomes do bucket do Amazon S3 e do documento usado na etapa 2.

```
#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

O código de exemplo a seguir exibe o documento e as caixas ao redor de itens detectados.

No código abaixo, substitua os valores de`bucket`e`photo`Com os nomes do bucket do Amazon S3 e do documento usado na etapa 2. Substitua o valor de`region`Com a região associada à sua conta do.

```
// 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. Execute o exemplo. Os exemplos Python e Java exibem a imagem do documento com as seguintes caixas delimitadoras coloridas:

   - Vermelho — Objetos KEY Block
   - Verde — VALUE Block objetos
   - Azul — Objetos TABLE Block
   - Amarelo — Objetos CELL Block

   Os elementos de seleção selecionados são preenchidos com azul.

   OAWS CLIO exemplo da exibe somente a saída JSON do`AnalyzeDocument`operação.

Analisando faturas e recibos com o Amazon Textract

Para analisar documentos de fatura e recebimento, use a API AnalyzeExpense e passa um arquivo de documento como entrada.`AnalyzeExpense`é uma operação síncrona que retorna uma estrutura JSON que contém o texto analisado. Para obter mais informações, consulte Analisando faturas e recibos.

Para analisar faturas e recebimentos de forma assíncrona, use`StartExpenseAnalysis`para começar a processar um arquivo de documento de entrada e usar`GetExpenseAnalysis`para obter os resultados.

Você pode fornecer um documento de entrada como uma matriz de bytes de imagem (bytes de imagem codificados em base64) ou um objeto do Amazon S3. Neste procedimento, você carrega um arquivo de imagem no bucket do S3 e especifica o nome do arquivo.

Para analisar uma fatura ou recibo (API)

1. Se ainda não tiver feito isso:

   a. Criar ou atualizar um usuário do IAM com`AmazonTextractFullAccess`e`AmazonS3ReadOnlyAccess`permissões. Para obter mais informações, consulte Etapa 1: Configurar uma conta da AWS e criar um usuário do IAM.

   b. Instale e configure a AWS CLI e os SDKs da AWS. Para obter mais informações, consulte Etapa 2: Configurar aAWS CLIeAWSSDKs da.

2. Carregue uma imagem que contenha um documento no bucket do S3.

   Para obter instruções, consulteCarregar objetos no Amazon S3noGuia do usuário do Amazon Simple Storage Service.

3. Use os exemplos a seguir para chamar a operação `AnalyzeExpense`.

   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. Isso fornecerá a você a saída do JSON para o`AnalyzeExpense`operação.

Analisando a documentação de identidade com o Amazon Textract

Para analisar documentos de identidade, use a API AnalyzeID e passa um arquivo de documento como entrada.`AnalyzeID`Retorna uma estrutura JSON que contenha o texto analisado. Para obter mais informações, consulte Analisar documentos de identidade.

Você pode fornecer um documento de entrada como uma matriz de bytes de imagem (bytes de imagem codificados em base64) ou um objeto do Amazon S3. Neste procedimento, você carrega um arquivo de imagem no bucket do S3 e especifica o nome do arquivo.

Para analisar um documento de identidade (API)

1. Se ainda não tiver feito isso:

   a. Criar ou atualizar um usuário do IAM com`AmazonTextractFullAccess`e`AmazonS3ReadOnlyAccess`permissões. Para obter mais informações, consulte Etapa 1: Configurar uma conta da AWS e criar um usuário do IAM.

   b. Instale e configure a AWS CLI e os SDKs da AWS. Para obter mais informações, consulte Etapa 2: Configurar aAWS CLIeAWSSDKs da.

2. Carregue uma imagem que contenha um documento no bucket do S3.

   Para obter instruções, consulteCarregar objetos no Amazon S3noGuia do usuário do Amazon Simple Storage Service.

3. Use os exemplos a seguir para chamar a operação `AnalyzeID`.

   CLI

   O exemplo a seguir recebe um arquivo de entrada de um bucket do S3 e executa a`AnalyzeID`Operação nele. No código abaixo, substitua o valor de*`balde`*Com o nome do seu bucket do S3, o valor de*`arquivo`*Com o nome do arquivo no bucket e o valor de*`região`*Com o nome do`region`Associado à sua conta do.

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

   Você também pode chamar a API com a frente e a parte de trás de uma carteira de motorista adicionando outro objeto S3 à entrada.

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

Se você estiver acessando a CLI em um dispositivo Windows, use aspas duplas em vez de aspas simples e escape das aspas duplas internas por barra invertida (ou seja,\) para resolver quaisquer erros de analisador que você possa encontrar. Para obter um exemplo, veja a seguir:

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

Python

O exemplo a seguir recebe um arquivo de entrada de um bucket do S3 e executa a`AnalyzeID`operação sobre ela, retornando os pares de chave/valor detectados. No código abaixo, substitua o valor de*`bucket_name`*Com o nome do seu bucket do S3, o valor de*`file_name`*Com o nome do arquivo no bucket e o valor de*`região`*Com o nome do`region`Associado à sua conta do.

```
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

O exemplo a seguir recebe um arquivo de entrada de um bucket do S3 e executa a`AnalyzeID`operação nele, retornando os dados detectados. Na função principal, substitua os valores des`3bucket`e`sourceDoc`Com os nomes do bucket do Amazon S3 e da imagem usada na etapa 2.

```
/*
   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. Isso fornecerá a você a saída do JSON para o`AnalyzeID`operação.

Processando documentos com operações assíncronas

Amazon Textract pode detectar e analisar texto em documentos de várias páginas que estão em formato PDF ou TIFF. Isso inclui faturas e recibos. O processamento de documentos de várias páginas é uma operação assíncrona. O processamento assíncrono de documentos é útil para processar documentos grandes de várias páginas. Por exemplo, um arquivo PDF com mais de 1.000 páginas leva algum tempo para ser processado. O processamento do arquivo PDF de forma assíncrona permite que seu aplicativo conclua outras tarefas enquanto aguarda a conclusão do processo.

Esta seção aborda como você pode usar o Amazon Textract para detectar e analisar texto de forma assíncrona em um documento de várias páginas ou de uma única página. Documentos com várias páginas devem estar em formato PDF ou TIFF. Documentos de página única processados com operações assíncronas podem estar no formato JPEG, PNG, TIFF ou PDF.

Você pode usar as operações assíncronas do Amazon Textract para as seguintes finalidades:

- Detecção de texto — Você pode detectar linhas e palavras em um documento de várias páginas. As operações assíncronas sãoStartDocumentTextDetectioneGetDocumentTextDetection. Para obter mais informações, consulte Detectar texto.
- Análise de texto — Você pode identificar relacionamentos entre o texto detectado em um documento de várias páginas. As operações assíncronas sãoStartDocumentAnalysiseGetDocumentAnalysis. Para obter mais informações, consulte Analisar documentos.
- Análise de despesas — Você pode identificar relacionamentos de dados em faturas e recebimentos de várias páginas. Amazon Textract trata cada fatura ou uma página de recibo de um documento de várias páginas como um recibo individual ou uma fatura. Ele não retém o contexto de uma página para outra de um documento de várias páginas. As operações assíncronas sãoStartExpenseAnalysiseGetExpenseAnalysis. Para obter mais informações, consulte Analisando faturas e recibos.

Tópicos

- Chamando operações assíncronas do Amazon Textract
- Configurando o Amazon Textract para operações assíncronas
- Detectando ou analisando texto em um documento de várias páginas
- Notificação de resultados do Amazon Textract

Chamando operações assíncronas do Amazon Textract

O Amazon Textract fornece uma API assíncrona que você pode usar para processar documentos de várias páginas em formato PDF ou TIFF. Você também pode usar operações assíncronas para processar documentos de página única que estão no formato JPEG, PNG, TIFF ou PDF.

As informações neste tópico usam operações de detecção de texto para mostrar como usar operações assíncronas do Amazon Textract. A mesma abordagem funciona com as operações de análise de texto dothe section called “StartDocumentAnalysis”ethe section called “GetDocumentAnalysis”. Também funciona da mesma maneira comthe section called “StartExpenseAnalysis”ethe section called “GetExpenseAnalysis”.

Para ver um exemplo, consulte Detectando ou analisando texto em um documento de várias páginas.

Amazon Textract processa assíncrona um documento armazenado em um bucket do Amazon S3. Você começa a processar chamando um`Start`operação, comoStartDocumentTextDetection. O status de conclusão da solicitação é publicada em um tópico do Amazon Simple Notification Service (Amazon SNS). Para obter o status de conclusão do tópico do Amazon SNS, use uma fila do Amazon Simple Queue Service (Amazon SQS) ou umaAWS Lambdafunção. Ao receber o status de conclusão, chame uma operação `Get`, como a GetDocumentTextDetection para obter os resultados da solicitação.

Os resultados de chamadas assíncronas são criptografados e armazenados por 7 dias em um bucket de propriedade do Amazon Textract por padrão, a menos que você especifique um bucket do Amazon S3 usando uma operação`OutputConfig`Argumento.

A tabela a seguir mostra as operações Iniciar e Get correspondentes para os diferentes tipos de processamento assíncrono suportados pelo Amazon Textract:

Iniciar/obter operações de API para operações assíncronas do Amazon Textract

Tipo de processamento	API Iniciar	Obter API
Detecção de texto	StartDocumentTextDetection	GetDocumentTextDetection
Análise de texto	StartDocumentAnalysis	GetDocumentAnalysis
Análise de despesas	StarTExpenseAnalysis	GetExpenseAnalysis

Para um exemplo que usaAWS Lambdafunções, consulteProcessamento de documentos em grande escala com o Amazon Textract.

O diagrama a seguir mostra o processo para detectar o texto do documento em uma imagem de documento armazenada em um bucket do Amazon S3. No diagrama, uma fila do Amazon SQS obtém o status de conclusão do tópico do Amazon SNS.

O processo exibido pelo diagrama anterior é o mesmo para analisar texto e faturas/recibos. Você começa a analisar o texto chamandothe section called “StartDocumentAnalysis”e comece a analisar faturas/recibos ligandothe section called “StartExpenseAnalysis”Você obtém os resultados ligandothe section called “GetDocumentAnalysis”outhe section called “GetExpenseAnalysis”respectivamente.

Iniciar a detecção de texto

Você pode iniciar uma solicitação de detecção de texto do Amazon Textract chamandoStartDocumentTextDetection. Veja a seguir um exemplo de uma solicitação JSON que é transmitida por `StartDocumentTextDetection`.

```
{
    "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"
}
```

O parâmetro de entrada`DocumentLocation`fornece o nome do arquivo do documento e o bucket do Amazon S3 do qual recuperá-lo.`NotificationChannel`contém o Nome de recurso da Amazon (ARN) do tópico do Amazon SNS que o Amazon Textract notifica quando a solicitação de detecção de texto é concluída. O tópico do Amazon SNS deve estar na mesma região da AWS que o endpoint do Amazon Textract que você está chamando.`NotificationChannel`também contém o ARN para uma função que permite que o Amazon Textract publique no tópico do Amazon SNS. Você concede

permissões de publicação Amazon Textract para os tópicos do Amazon SNS criando uma função de serviço do IAM. Para obter mais informações, consulte Configurando o Amazon Textract para operações assíncronas.

Você também pode especificar um parâmetro de entrada opcional,`JobTag`, que permite identificar o trabalho ou grupos de trabalhos no status de conclusão que é publicada no tópico do Amazon SNS. Por exemplo, você pode usar`JobTag`para identificar o tipo de documento que está sendo processado, como um formulário fiscal ou recibo.

Para evitar a duplicação acidental de trabalhos de análise, você pode opcionalmente fornecer um token de idempotência, `ClientRequestToken`. Se você fornecer um valor para`ClientRequestToken`, o`Start`A operação retorna o mesmo`JobId`para várias chamadas idênticas ao`Start`operação, como`StartDocumentTextDetection`. Um token `ClientRequestToken` tem uma vida útil de 7 dias. Depois de 7 dias, você pode reutilizá-lo. Se você reutilizar o token durante a vida útil, o seguinte acontecerá:

- Se você reutilizar o token com a mesma operação `Start` e os mesmos parâmetros de entrada, o mesmo `JobId` será retornado. O trabalho não é realizado novamente, e o Amazon Textract não envia um status de conclusão ao tópico do Amazon SNS registrado.
- Se você reutilizar o token com a mesma operação `Start` e uma pequena alteração no parâmetro de entrada, você receberá uma exceção `idempotentparametermismatchexception` (código de status HTTP: 400).
- Se você reutilizar o token com uma operação `Start` diferente, a operação será bem-sucedida.

Outro parâmetro opcional disponível é`OutputConfig`, que permite ajustar onde sua saída será colocada. Por padrão, o Amazon Textract armazenará os resultados internamente e só poderá ser acessado pelas operações Get API. com`OutputConfig`habilitado, você pode definir o nome do bucket para o qual a saída será enviada e o prefixo do arquivo dos resultados, onde você pode baixar seus resultados. Além disso, você pode definir o`KMSKeyID`parâmetro para uma chave gerenciada pelo cliente para criptografar sua saída. Sem esse conjunto de parâmetros, o Amazon Textract criptografará o lado do servidor usando oChave gerenciada pela AWSpara Amazon S3

> **Note**
>
> Antes de usar esse parâmetro, certifique-se de ter a permissão PutObject para o bucket de saída. Além disso, certifique-se de ter as permissões Decrypt, ReEnCrypt, GenerateDataKey e DescribeKey para oAWS KMSchave se você decidir usá-lo.

A resposta à operação `StartDocumentTextDetection` é um identificador de trabalho (`JobId`). Usar o`JobId`Para rastrear solicitações e obter os resultados da análise após o Amazon Textract publicar o status de conclusão no tópico do Amazon SNS. Veja um exemplo a seguir:

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

Se você iniciar muitos trabalhos simultaneamente, ligue para`StartDocumentTextDetection`RAISE`LimitExceededException`Exceção (código de status HTTP: 400) até que o número de trabalhos simultâneos fique abaixo do limite de serviço do Amazon Textract.

Se você acredita que exceções LimitExceededException estão sendo geradas com intermitências de atividade, considere a possibilidade de usar uma fila do Amazon SQS para gerenciar as solicitações recebidas. ContatoAWSSupport se você acredita que o número médio de solicitações simultâneas não pode ser gerenciado por uma fila do Amazon SQS e você ainda está recebendo`LimitExceededException`exceções.

Obter o status de conclusão de uma solicitação de análise do Amazon Textract

Amazon Textract envia uma notificação de conclusão de análise para o tópico do Amazon SNS registrado. A notificação inclui o identificador do trabalho e o status de conclusão da operação em uma string JSON. Uma solicitação de detecção de texto bem-sucedida tem um`SUCCEEDED`Status. Por exemplo, o resultado a seguir mostra o processamento bem-sucedido de um trabalho de detecção de texto.

```
{
    "JobId": "642492aea78a86a40665555dc375ee97bc963f342b29cd05030f19bd8fd1bc5f",
    "Status": "SUCCEEDED",
    "API": "StartDocumentTextDetection",
    "JobTag": "Receipt",
    "Timestamp": 1543599965969,
    "DocumentLocation": {
        "S3ObjectName": "document",
        "S3Bucket": "bucket"
    }
}
```

Para obter mais informações, consulte Notificação de resultados do Amazon Textract.

Para obter as informações de status publicadas no tópico do Amazon SNS pelo Amazon Textract, use uma das seguintes opções:

- AWS Lambda— Você pode assinar umAWS LambdaFunção do que você grava em um tópico do Amazon SNS. A função é chamada quando o Amazon Textract notifica o tópico do Amazon SNS que a solicitação foi concluída. Use uma função do Lambda se desejar que o código do lado do servidor processe os resultados de uma solicitação de detecção de texto. Por exemplo, você pode usar o código do lado do servidor para anotar a imagem ou criar um relatório no texto detectado antes de retornar as informações para um aplicativo cliente.
- Amazon SQS— Você pode inscrever uma fila do Amazon SQS em um tópico do Amazon SNS. Em seguida, pesquise a fila do Amazon SQS para recuperar o status de conclusão publicado pelo Amazon Textract quando uma solicitação de detecção de texto é concluída. Para obter mais informações, consulte Detectando ou analisando texto em um documento de várias páginas. Use uma fila do Amazon SQS se desejar chamar as operações Amazon Textract somente a partir de um aplicativo cliente.

> **Important**
>
> Nós não recomendamos obter o status de conclusão da solicitação chamando repetidamente o Amazon Textract`Get`operação. Isso ocorre porque o Amazon Textract acelera o`GetS`e muitas solicitações forem realizadas. Se você estiver processando vários documentos ao mesmo tempo, é mais simples e mais eficiente monitorar uma fila do SQS para a notificação de conclusão do que pesquisar no Amazon Textract para o status de cada trabalho individualmente.

Obtendo resultados de detecção de texto Amazon Textract

Para obter os resultados de uma solicitação de detecção de texto, primeiro verifique se o status de conclusão que é recuperado do tópico do Amazon SNS é`SUCCEEDED`. Em seguida, chame `GetDocumentTextDetection`, que transmite o valor `JobId` que é retornado de `StartDocumentTextDetection`. O JSON solicitado é semelhante ao exemplo a seguir:

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

```
}
```

`JobId`é o identificador da operação de detecção de texto. Como a detecção de texto pode gerar grandes quantidades de dados, use`MaxResults`Para especificar o número máximo de resultados a serem retornados em um único`Get`operação. O valor padrão para`MaxResults`É 1.000. Se você especificar um valor maior que 1.000, somente 1.000 resultados serão retornados. Se a operação não retornar todos os resultados, um token de paginação para a próxima página será retornado. Para obter a próxima página de resultados, especifique o token no`NextToken`parâmetro .

> **Note**
>
> O Amazon Textract retém os resultados de operações assíncronas por 7 dias. Você não pode recuperar os resultados após esse período.

O`GetDocumentTextDetection`O JSON solicitado da operação é semelhante ao seguinte. O número total de páginas detectadas é retornado em`DocumentMetadata`. O texto detectado é retornado no`Blocks`matriz. Para obter mais informações sobre o`Block`objetos, consulte[Objetos de resposta de detecção de texto e análise de](#).

```
{
    "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
        }
    ]
}
```

Configurando o Amazon Textract para operações assíncronas

Os procedimentos a seguir mostram como configurar o Amazon Textract para usar com um tópico do Amazon Simple Notification Service (Amazon SNS) e em uma fila do Amazon Simple Queue Service (Amazon SQS).

> **Note**
>
> Se você estiver usando essas instruções para configurar oDetectando ou analisando texto em um documento de várias páginasPor exemplo, você não precisa executar as etapas 3 — 6. O exemplo inclui código para criar e configurar o tópico do Amazon SNS e a fila do Amazon SQS.

Para configurar o Amazon Textract

1. Configurar umAWSconta para acessar o Amazon Textract. Para obter mais informações, consulte Etapa 1: Configurar uma conta da AWS e criar um usuário do IAM.

   Certifique-se de que o usuário tenha pelo menos as seguintes permissões:

   - AmazonTextractFullAccess
   - AmazonS3ReadOnlyAccess
   - AmazonSNSFullAccess
   - AmazonSQSFullAccess
2. Instale e configure o SDK da AWS necessário. Para obter mais informações, consulte Etapa 2: Configurar aAWS CLIeAWSSDKs da.
3. Criar um tópico do Amazon SNS. Anteca o nome do tópico comAmazonTextact. Anote o nome de recurso da Amazon (ARN) do tópico. Verifique se o tópico do está na mesma região do que oAWSO endpoint do que você está usando com sua conta da AWS.

4. Criar uma fila padrão do Amazon SQSusando oConsole do Amazon SQS. Anotar o ARN da fila.
5. Inscreva a fila no tópico criado na etapa 3.
6. Conceda permissão ao tópico do Amazon SNS para enviar mensagens à fila do Amazon SQS.
7. Crie uma função de serviço do IAM para dar ao Amazon Textract acesso aos tópicos do Amazon SNS. Observe o nome de recurso da Amazon (ARN) da função de serviço. Para obter mais informações, consulte Dando acesso ao Amazon Textract ao seu tópico do Amazon SNS.
8. Adicione a seguinte política em linhaPara o usuário do IAM que você criou na etapa 1.

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

   Dê um nome à política em linha.
9. Agora, você pode executar os exemplos noDetectando ou analisando texto em um documento de várias páginas.

Dando acesso ao Amazon Textract ao seu tópico do Amazon SNS

Amazon Textract precisa de permissão para enviar uma mensagem ao tópico do Amazon SNS quando uma operação assíncrona for concluída. Use uma função de serviço do IAM para dar ao Amazon Textract acesso ao tópico do Amazon SNS.

Ao criar o tópico do Amazon SNS, você deve anexar o nome do tópico com**`AmazonTextract`**—por exemplo,**`AmazonTextractMyTopicName`**.

1. Faça login no console do IAM (https://console.aws.amazon.com/iam).
2. No painel de navegação, selecione Roles.
3. Selecione Create role.

4. Em Select type of trusted entity (Selecionar tipo de entidade confiável), escolha AWS service (Serviço da AWS).
5. para oEscolha o serviço que usará esta função, escolhaTextract.
6. Selecione Next (Próximo): Permissions
7. Verificar se oAmazonTextractServiceRolea política foi incluída na lista de políticas anexadas. Para exibir a política na lista, insira parte do nome da política noPolíticas de filtros.
8. Selecione Next (Próximo): Tags.
9. Você não precisa adicionar tags, então selecionePróximo: Análise.
10. Na seção Review (Revisar), em Role Name (Nome da função), insira um nome para a função (por exemplo, `TextractRole`). DentroDescrição de função, atualize a descrição da função e, em seguida, escolhaCriar função do.
11. Escolha a nova função para abrir a página de detalhes da função.
12. Em Summary (Resumo), copie o valor do Role ARN (ARN da função) e salve-o.
13. Escolha Trust relationships (Relações de confiança).
14. SelecioneEditar relação de confiançae garanta que a política de confiança seja a seguinte.

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

15. Escolha Update Trust Policy.

Detectando ou analisando texto em um documento de várias páginas

Este procedimento mostra como detectar ou analisar texto em um documento de várias páginas usando operações de detecção do Amazon Textract, um documento armazenado em um bucket

do Amazon S3, um tópico do Amazon SNS e uma fila do Amazon SQS. O processamento de documentos de várias páginas é uma operação assíncrona. Para obter mais informações, consulte Chamando operações assíncronas do Amazon Textract.

Você pode escolher o tipo de processamento que deseja que o código faça: detecção de texto, análise de texto ou análise de despesas.

Os resultados do processamento são retornados em uma matriz dethe section called “Block”Objetos, que diferem dependendo do tipo de processamento usado.

Para detectar texto em documentos de várias páginas ou analisar, faça o seguinte:

1. Crie o tópico do Amazon SNS e a fila do Amazon SQS.
2. Assinar a fila do tópico do.
3. Conceda permissão ao tópico do para enviar mensagens à fila do.
4. Comece a processar o documento. Use a operação apropriada para o tipo de análise escolhido:
   - StartDocumentTextDetectionpara tarefas de detecção de texto.
   - StartDocumentAnalysispara tarefas de análise de texto.
   - StartExpenseAnalysispara tarefas de análise de despesas.
5. Obtenha o status de conclusão a partir da fila do Amazon SQS. O código de exemplo rastreia o identificador da tarefa (`JobId`) que é retornado pelo`Start`operação. Ele somente obtém os resultados para identificadores de trabalho correspondentes que são lidos a partir do status de conclusão. Isso é importante se outros aplicativos estiverem usando a mesma fila do e tópico do. Para simplificar, o exemplo exclui trabalhos não correspondentes. Considere adicionar as tarefas excluídas a uma fila de mensagens mortas do Amazon SQS para mais investigações.
6. Obtenha e exiba os resultados do processamento chamando a operação apropriada para o tipo de análise escolhido:
   - GetDocumentTextDetectionpara tarefas de detecção de texto.
   - GetDocumentAnalysispara tarefas de análise de texto.
   - GetExpenseAnalysispara tarefas de análise de despesas.
7. Exclua o tópico do Amazon SNS e a fila do Amazon SQS.

Executando operações assíncronas

O código de exemplo para este procedimento é fornecido em Java, Python e oAWS CLI. Antes de começar, instale o apropriadoAWSSDK. Para obter mais informações, consulte Etapa 2: Configurar aAWS CLIeAWSSDKs da.

Para detectar ou analisar texto em um documento de várias páginas

1. Configure o acesso de usuário ao Amazon Textract e configure o acesso do Amazon Textract ao Amazon SNS. Para obter mais informações, consulte Configurando o Amazon Textract para operações assíncronas. Para concluir este procedimento, você precisa de um documento de várias páginas em formato PDF. Ignorar as etapas 3 — 6 porque o código de exemplo cria e configura o tópico do Amazon SNS e a fila do Amazon SQS. Se completNo exemplo da CLI, você não precisa configurar uma fila SQS.
2. Faça upload de um arquivo de documento de várias páginas em formato PDF ou TIFF para o seu bucket do Amazon S3. (Documentos de página única em formato JPEG, PNG, TIFF ou PDF também podem ser processados).

   Para obter instruções, consulteFazer upload de objetos no Amazon S3noGuia do usuário do Amazon Simple Storage Service.
3. Use o seguinteAWS SDK para Java, SDK for Python (Boto3) ouAWS CLIcódigo para detectar texto ou analisar texto em um documento de várias páginas. No`main`função:

   - Substitua o valor de`roleArn`Com o ARN da função do IAM no qual você salvouDando acesso ao Amazon Textract ao seu tópico do Amazon SNS.
   - Substitua os valores de`bucket`e`document`Com o bucket e o nome do arquivo do documento especificado na etapa 2.
   - Substitua o valor do`type`parâmetro de entrada do`ProcessDocument`Função com o tipo de processamento que você deseja fazer. Usar o`ProcessType.DETECTION`para detectar texto. Usar o`ProcessType.ANALYSIS`para analisar texto.
   - Para o exemplo Python, substitua o valor de`region_name`com a região em que seu cliente está operando.

   Para oAWS CLIPor exemplo, faça o seguinte:

   - Ao ligarStartDocumentTextDetection, substitua o valor de`bucket-name`Com o nome do seu bucket do S3 e substitua`file-name`Com o nome do arquivo especificado na etapa

2. Especifique a região do bucket substituindo`region-name`Com o nome da sua região. Observe que o exemplo da CLI não faz uso do SQS.

- Ao ligarGetDocumentTextDetectionsubstituir`job-id-number`com `o`job-idRetornado porStartDocumentTextDetection. Especifique a região do bucket substituindo`region-name`Com o nome da sua região.

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

EsseAWS CLIO comando inicia a detecção assíncrona de texto em um documento especificado. Ele retorna um`job-id`que pode ser usado para retreviver os resultados da detecção.

```
aws textract start-document-text-detection --document-location
```

```
"{\"S3Object\":{\"Bucket\":\"bucket-name\",\"Name\":\"file-name\"}}" --
region region-name
```

EsseAWS CLIcomando retorna os resultados de uma operação assíncrona do Amazon Textract quando fornecido com um`job-id`.

```
aws textract get-document-text-detection --region region-name --job-id job-id-
number
```

Se você estiver acessando a CLI em um dispositivo Windows, use aspas duplas em vez de aspas simples e escape das aspas duplas internas por barra invertida (ou seja,\) para resolver quaisquer erros de analisador que você possa encontrar. Para obter um exemplo, veja abaixo

```
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

Neste exemplo, substitua o valor de`roleArn`Com o ARN da função do IAM no qual você salvou[Dando acesso ao Amazon Textract ao seu tópico do Amazon SNS](). Substitua os valores de`bucket`e`document`Com o bucket e o nome do arquivo do documento especificado na etapa 2 acima. Substitua o valor de`processType`com o tipo de processamento que você gostaria de usar no documento de entrada. Finalmente, substitua o valor de`REGION`com a região em que seu cliente está operando.

```
 // 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. Execute o código. A operação pode demorar um pouco para ser concluída. Depois de ser concluída, uma lista de blocos para texto detectado ou analisado é exibida.

Notificação de resultados do Amazon Textract

O Amazon Textract publica os resultados de uma solicitação de análise do Amazon Textract, incluindo o status de conclusão, em um tópico do Amazon Simple Notification Service (Amazon SNS). Para obter a notificação de um tópico do Amazon SNS, use uma fila do Amazon SQS ou umaAWS Lambdafunção. Para obter mais informações, consulte Chamando operações assíncronas do Amazon Textract. Para ver um exemplo, consulte Detectando ou analisando texto em um documento de várias páginas.

Os resultados possuem o seguinte formato JSON:

```
{
  "JobId": "String",
  "Status": "String",
  "API": "String",
  "JobTag": "String",
  "Timestamp": Number,
  "DocumentLocation": {
    "S3ObjectName": "String",
    "S3Bucket": "String"
  }
}
```

Esta tabela descreve os diferentes parâmetros dentro de uma resposta do Amazon Textract.

Parâmetro	Descrição
JobId	O identificador exclusivo que o Amazon Textract atribui ao trabalho. Ele corresponde a um identificador de trabalho que é retornado de um`Start`operação, como StartDocumentTextDetection.
Status	O status do trabalho. Os valores válidos são bem-sucedidos, com falha ou erro.
API	A operação do Amazon Textract usada para analisar o documento de entrada,

Parâmetro	Descrição
	comoStartDocumentTextDetectionouStartDocumentAnalysis.
JobTag	O identificador especificado pelo usuário para a tarefa. Você especifica`JobTag`em uma chamada para o`Start`operação, comoStartDocumentTextDetection.
Timestamp	O carimbo de data/hora Unix que indica quando o trabalho foi concluído, retornado em milissegundos.
DocumentLocation	Detalhes sobre o documento que foi processado. Inclui o nome do arquivo e o bucket do Amazon S3 no qual o arquivo está armazenado.

Lidando com chamadas limitadas e conexões descartadas

Uma operação do Amazon Textract pode falhar se você exceder o número máximo de transações por segundo (TPS), fazendo com que o serviço acelere seu aplicativo ou quando sua conexão cair. Por exemplo, se você fizer muitas chamadas para operações do Amazon Textract em um curto período de tempo, ele acelerará suas chamadas e enviará um`ProvisionedThroughputExceededException`erro na resposta da operação. Para obter informações sobre cotas do Amazon Textract TPS, consulteCotas do Amazon Textract.

Você pode gerenciar a limitação e as conexões descartadas ao tentar automaticamente a operação. Você pode especificar o número de novas tentativas, incluindo o`Config`parâmetro quando você cria o cliente do Amazon Textract. Recomendamos uma contagem de novas tentativas de 5. OAWSO SDK executa uma operação pelo número especificado de novas tentativas para uma operação antes que ocorra falha e seja lançada uma exceção. Para obter mais informações, consulte Novas tentativas e recuo exponencial na AWS.

> **Note**
>
> As tentativas automáticas funcionam para operações síncronas e assíncronas. Antes de especificar novas tentativas automáticas, certifique-se de que você tenha a versão mais recente do AWS SDK. Para obter mais informações, consulte Etapa 2: Configurar aAWS CLIeAWSSDKs da.

O exemplo a seguir mostra como realizar novas tentativas de novas tentativas para as operações do Amazon Textract quando você estiver processando vários documentos.

Pré-requisitos

- Se ainda não tiver feito isso:
  a. Criar ou atualizar um usuário do IAM com`AmazonTextractFullAccess`e`AmazonS3ReadOnlyAccess`Permissões Para obter mais informações, consulte Etapa 1: Configurar uma conta da AWS e criar um usuário do IAM.
  b. Instale e configure a AWS CLI e os SDKs da AWS. Para obter mais informações, consulte Etapa 2: Configurar aAWS CLIeAWSSDKs da.

Para repetir operações automaticamente

1. Carregue várias imagens de documento para o bucket do S3 para executar o exemplo síncrono. Carregue um documento de várias páginas para o bucket do S3 e execute`StartDocumentTextDetection`nele para executar o exemplo assíncrono.

   Para obter instruções, consulte[Carregar objetos no Amazon S3]noGuia do usuário do Amazon Simple Storage Service.

2. Os exemplos a seguir demonstram como usar o`Config`parâmetro para tentar novamente uma operação automaticamente. O exemplo síncrono chama o`DetectDocumentText`operação, enquanto o exemplo assíncrono chama o`GetDocumentTextDetection`operação.

   Sync Example

   Use os exemplos a seguir para chamar o`DetectDocumentText`operação nos documentos no bucket do Amazon S3. Dentro`main`, altere o valor de`bucket`no bucket do S3. Altere o valor de`documents`para os nomes das imagens do documento que você carregou na etapa 2.

```
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

Use os exemplos a seguir para chamar a operação `GetDocumentTextDetection`. Ele pressupõe que você já ligou`StartDocumentTextDetection`nos documentos no bucket do Amazon S3 e obteve um`JobId`. Dentro`main`, altere o valor de`bucket`no bucket do S3 e o valor de`roleArn`para o Arn atribuído à sua função Textract. Você também precisará alterar o valor de`document`no nome do documento de várias páginas no bucket do Amazon S3. Finalmente, substitua o valor de`region_name`Fornece o nome da sua região e forneça o`GetResults`função com o nome do seu`jobId`.

```
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()
```

Práticas recomendadas para o Amazon Textract

Amazon Textract usa aprendizado de máquina para ler documentos como uma pessoa faria. Ele extrai texto, tabelas e formulários de documentos. Use as melhores práticas a seguir para obter os melhores resultados de seus documentos.

Forneça um documento de entrada ideal

Veja a seguir uma lista de algumas maneiras pelas quais você pode otimizar seus documentos de entrada para obter melhores resultados.

- Certifique-se de que o texto do documento esteja em um idioma compatível com o Amazon Textract. Atualmente, o Amazon Textract é compatível com inglês, espanhol, alemão, italiano, francês e português.
- Forneça uma imagem de alta qualidade, idealmente pelo menos 150 DPI.
- Se o documento já estiver em um dos formatos de arquivo compatíveis com o Amazon Textract (PDF, TIFF, JPEG e PNG), não converta ou diminua a amostra do documento antes de enviá-lo para o Amazon Textract.

Para obter os melhores resultados ao extrair texto de tabelas em documentos, certifique-se de que:

- As tabelas em seu documento são separadas visualmente dos elementos ao redor na página. Por exemplo, a tabela não é sobreposta em uma imagem ou padrão complexo.
- O texto dentro da tabela está na posição vertical. Por exemplo, o texto não é girado em relação a outro texto na página.

Ao extrair texto de tabelas, você pode ver resultados inconsistentes quando:

- Células de tabela mescladas que abrangem várias colunas.
- Tabelas com células, linhas ou colunas diferentes de outras partes da mesma tabela.

Recomendamos usarDetecção de textoComo solução.

Usar escores de confiança

Você deve levar em consideração as pontuações de confiança retornadas pelas operações da API Amazon Textract e a sensibilidade de seu caso de uso. Um escore de confiança é um número entre 0 e 100 que indica a probabilidade de que uma determinada previsão esteja correta. Isso ajuda você a tomar decisões informadas sobre como você usa os resultados.

Em aplicativos sensíveis a erros de detecção (falsos positivos), imponha um limite mínimo de pontuação de confiança. O aplicativo deve descartar resultados abaixo desse limite ou sinalizar situações como exigindo um nível mais alto de escrutínio humano.

O limite ideal depende do aplicativo. Para fins de arquivamento, como documentar notas manuscritas, pode ser tão baixo quanto 50%. Processos de negócios envolvendo decisões financeiras podem exigir limites de 90% ou mais.

Considere usar a análise humana

Considere também incorporar a revisão humana em seus fluxos de trabalho. Isso é especialmente importante para aplicativos confidenciais, como processos de negócios que envolvem decisões financeiras.

Tutoriais

the section called “Block”objetos retornados das operações do Amazon Textract contêm os resultados das operações de detecção de texto e análise de texto, comothe section called “AnalyzeDocument”. Os tutoriais do Python a seguir mostram algumas maneiras diferentes de usar Objetos de Bloquear. Por exemplo, você pode exportar as informações da tabela para um arquivo de valores separados por vírgula (CSV).

Os tutoriais usam operações síncronas Amazon Textract que retornam todos os resultados. Se você quiser usar operações assíncronas, comothe section called “StartDocumentAnalysis”, você precisa alterar o código de exemplo para acomodar vários lotes de retornados`Block`objetos. Para fazer uso do exemplo de operações assíncronas, certifique-se de ter seguido as instruções dadas emConfigurando o Amazon Textract para operações assíncronas.

Para obter exemplos que mostram outras maneiras de usar o Amazon Textract, consulteExemplos de código adicionais.

Tópicos

- Pré-requisitos
- Extraindo pares de valores-chave de um documento de formulário
- Exportando tabelas para um arquivo CSV
- Como criar umAWS LambdaFunção
- Exemplos de código adicionais

Pré-requisitos

Antes de executar os exemplos desta seção, você precisa configurar o ambiente.

Para configurar o ambiente

1. Criar ou atualizar um usuário do IAM com`AmazonTextractFullAccess`permissões. Para obter mais informações, consulte Etapa 1: Configurar uma conta da AWS e criar um usuário do IAM.
2. Instale e configure a AWS CLI e os SDKs da AWS. Para obter mais informações, consulte Etapa 2: Configurar aAWS CLIeAWSSDKs da.

Extraindo pares de valores-chave de um documento de formulário

O exemplo Python a seguir mostra como extrair pares de valores-chave em documentos de formulário dethe section called “Block”Objetos armazenados em um mapa. Objetos de bloco são retornados de uma chamada parathe section called “AnalyzeDocument”. Para obter mais informações, consulte Dados do formulário (pares de chave-valor).

Você usa as seguintes funções:

- `get_kv_map`— Chamar oAnalyzeDocumente armazena os objetos KEY e VALUE BLOCK em um mapa.
- `get_kv_relationshipefind_value_block`— Constrói as relações chave-valor a partir do mapa.

Para extrair pares de chave/valor de um documento de formulário

1. Configure o ambiente. Para obter mais informações, consulte Pré-requisitos.
2. Salve o seguinte código de exemplo em um arquivo chamadoArquivo textract_python_kv_parser.py.

```
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. No prompt de comando, digite o seguinte comando. Substituir`file`pelo arquivo de imagem do documento que você deseja analisar.

```
textract_python_kv_parser.py file
```

4. Quando você for solicitado, insira uma chave que esteja no documento de entrada. Se o código detectar a chave, ele exibirá o valor da chave.

Exportando tabelas para um arquivo CSV

Esses exemplos Python mostram como exportar tabelas de uma imagem de um documento para um arquivo de valores separados por vírgula (CSV).

O exemplo para análise de documentos síncronos coleta informações da tabela de uma chamada parathe section called “AnalyzeDocument”. O exemplo para análise de documentos assíncronos faz uma chamada parathe section called “StartDocumentAnalysis”Em seguida, recupera os resultados dothe section called “GetDocumentAnalysis”como`Block`objetos.

As informações da tabela são retornadas comothe section called “Block”objetos de uma chamada parathe section called “AnalyzeDocument”. Para obter mais informações, consulte Tabelas. O`Block`os objetos são armazenados em uma estrutura de mapa usada para exportar os dados da tabela para um arquivo CSV.

Synchronous

Neste exemplo, você usará as funções:

- `get_table_csv_results`— Chamar oAnalyzeDocumente constrói um mapa de tabelas detectadas no documento. Cria uma representação CSV de todas as tabelas detectadas.
- `generate_table_csv`— Gera o arquivo CSV para uma tabela individual.
- `get_rows_columns_map`— Obtém as linhas e colunas do mapa.
- `get_text`— Obtém o texto de uma célula.

Para exportar tabelas para um arquivo CSV

1. Configure o ambiente. Para obter mais informações, consulte Pré-requisitos.
2. Salve o seguinte código de exemplo em um arquivo chamadoArquivo textract_python_table_parser.py.

```
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. No prompt de comando, digite o seguinte comando. Substituir`file`Com o nome do arquivo de imagem do documento que você deseja analisar.

```
python textract_python_table_parser.py file
```

Quando você executa o exemplo, a saída CSV é salva em um arquivo chamado`output.csv`.

Asynchronous

Neste exemplo, você usará fazer uso de dois scripts diferentes. O primeiro script inicia o processo de análise assíncrona de documentos com`StartDocumentAnalysis`Obter o`Block`informações retornadas por`GetDocumentAnalysis`. O segundo script leva o retornado`Block`informações para cada página, formata os dados como uma tabela e salva as tabelas em um arquivo CSV.

Para exportar tabelas para um arquivo CSV

1. Configure o ambiente. Para obter mais informações, consulte Pré-requisitos.
2. Certifique-se de ter seguido as instruções dadas em verConfigurando o Amazon Textract para operações assíncronas. O processo documentado nessa página permite enviar e receber mensagens sobre o status de conclusão de trabalhos assíncronos.
3. No exemplo de código a seguir, substitua o valor de`roleArn`Com o Arn atribuído à função criada na Etapa 2. Substitua o valor de`bucket`Com o nome do bucket do S3 que contém o

documento. Substitua o valor de`document`pelo nome do documento no seu bucket do S3. Substitua o valor de`region_name`pelo nome da região do seu bucket.

Salve o seguinte código de exemplo em um arquivo chamado`start_doc_analysis_for_table_extraction.py`..

```
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. Execute o código. O código imprimirá um JobId. Copie este JobId para baixo.
5. Aguarde até que o trabalho termine o processamento e, depois de terminar, copie o seguinte código para um arquivo chamadoArquivo get_doc_analysis_for_table_extraction.py. Substitua o valor de`jobId`com o ID do Job que você copiou anteriormente. Substitua o valor de`region_name`com o nome da região associada à sua função Textract. Substitua o valor de`file_name`pelo nome que você deseja fornecer ao CSV de saída.

```
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. Execute o código.

   Depois de obter os resultados, certifique-se de excluir os recursos SNS e SQS associados, ou então você poderá acumular cobranças por eles.

Como criar umAWS LambdaFunção

Você pode chamar as operações da API Amazon Textract de dentro de umAWS Lambdafunção. As instruções a seguir mostram como criar uma função Lambda no Python que chamathe section called "DetectDocumentText". Ele retorna uma lista dethe section called "Block"objetos. Para executar este exemplo, você precisa de um bucket do Amazon S3 que contenha um documento no formato PNG ou JPEG. Para criar uma função, use o console.

Para obter um exemplo que usa funções do Lambda para processar documentos em grande escala, consulteProcessamento de documentos em grande escala com o Amazon Textract.

Para chamar a operação DetectDocumentText a partir de uma função do Lambda:

Etapa 1: Criar um pacote de implantação do Lambda

1. Abra uma janela de comando.
2. Insira os comandos a seguir para criar um pacote de implantação com a versão mais recente doAWSSDK.

```
pip install boto3 --target python/.
zip boto3-layer.zip -r python/
```

Etapa 2: Criar uma função do Lambda

1. Faça login no AWS Management Console e abra o console AWS Lambda em https://console.aws.amazon.com/lambda/.
2. Escolha Create function (Criar função).

3. Especifique o seguinte.
   - Escolha Author from scratch.
   - Em Function name (Nome da função), insira um nome.
   - para oTempo de execução, escolhaPython 3.7ouPython 3.6.
   - para oEscolha ou crie uma função de execução, escolhaCriar uma nova função com permissões básicas do Lambda.
4. SelecioneCriar funçãoPara criar a função do Lambda.
5. Abra o console do IAM em https://console.aws.amazon.com/iam/.
6. No painel de navegação, escolhaFunções do.
7. Na lista de recursos, escolha a função do IAM criada pelo Lambda para você. O nome da função começa com o nome da função do Lambda.
8. Selecione oPermissõese, em seguida, escolhaAnexar políticas.
9. Selecione as políticas AmazonTextractFullAccess e AmazonS3ReadOnlyAccess.
10. SelectAnexar política.

Para obter mais informações, consulteCriar uma função do Lambda com o console

Etapa 3: Criar e adicionar uma camada

1. Abra o console do AWS Lambda em https://console.aws.amazon.com/lambda/.
2. No painel de navegação, escolha Layers (Camadas).
3. Escolha Create layer (Criar camada).
4. para oName (Nome)Insira um nome.
5. Em Description (Descrição), insira uma descrição.
6. para oTipo de entrada de código, escolhaFazer upload do arquivo .zipSelecione e selecioneCarregar.
7. Na caixa de diálogo, selecione o arquivo zip (boto3-layer.zip), o zip que você criou emEtapa 1: Criar um pacote de implantação do Lambda.
8. para oTempos de execução compatíveis, escolha a versão do tempo de execução que você escolheuEtapa 2: Criar uma função do Lambda.
9. SelecioneCriarPara criar a camada.
10. Escolha o ícone do menu do painel de navegação.

11. Selecione Functions (Funções) no painel de navegação.
12. Na lista de recursos, selecione a função criada emEtapa 2: Criar uma função do Lambda.
13. SelecioneConfiguraçãoe noDesignerEscolha, escolhaCamadas do(sob o nome da função do Lambda).
14. NoCamadas doEscolha, escolhaAdicionar uma camada.
15. SelecioneSelecione na lista de camadas compatíveis com tempo de execução.
16. DentroCamadas compatíveis, selecione oName (Nome)eVersãoda camada criada na etapa 3.
17. Escolha Add (Adicionar).

Etapa 4: Adicionar código python à função

1. DentroDesignerEscolha a função do.
2. No editor de código da função, adicione o seguinte ao arquivolambda_function.py. Altere os valores de`bucket`e`document`para seu bucket e documento.

```
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. SelecioneSalvarPara salvar a função do Lambda.

Etapa 5: Testar o Lambda

1. SelectTeste.
2. Insira um valor para oEvent name (Nome do evento).
3. Escolha Create (Criar).
4. A saída, uma lista dethe section called “Block”objetos, aparece no painel Resultados da execução.

Se oAWS Lambdafunção retorna um erro de tempo limite, uma chamada de operação da API Amazon Textract pode ser a causa. Para obter mais informações sobre como estender o período de tempo limite para umAWS LambdaConsulte a função doConfiguração da função do AWS Lambda.

Para obter mais informações sobre como invocar uma função Lambda do seu código, consulteInvocarAWS LambdaFunções.

Exemplos de código adicionais

A tabela a seguir disponibiliza links para mais exemplos de código do Amazon Textract.

Exemplo	Descrição
Exemplos de código do Amazon Textract	Mostre várias maneiras pelas quais você pode usar o Amazon Textract.
Processamento de documentos em grande escala com o Amazon Textract	Mostra uma arquitetura de referência sem servidor que processa documentos em grande escala.
Analisador Amazon Textract	Mostra como analisar othe section called “Block”objetos retornados pelas operações do Amazon Textract.
Exemplos de código de documentação Amazon Textract	Exemplos de código usados neste guia.

Exemplo	Descrição
Textractor	Mostra como converter a saída do Amazon Textract em vários formatos.
Gerar documentos PDF pesquisáveis com o Amazon Textract	Mostra como criar um documento PDF pesquisável a partir de diferentes tipos de documentos de entrada, como imagens no formato JPG/PNG e documentos PDF digitaliz ados.

Exemplos de código para o Amazon Textract

Os exemplos de código a seguir mostram como usar o Amazon Textract com umAWSkit de desenvolvimento de software (SDK).

Os exemplos são divididos nas seguintes categorias:

Ações

Trechos de código que mostram como chamar funções de serviço individuais.

Exemplos entre serviços

Aplicativos de amostra que funcionam em váriosAWSServiços da .

Para obter uma lista completa deAWSGuias do desenvolvedor do SDK e exemplos de código, consulteUsando o Amazon Textract com umAWSSDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Exemplos de código

- Ações para o Amazon Textract
  - Analise um documento usando o Amazon Textract e umAWSSDK
  - Detectar texto em um documento usando o Amazon Textract e umAWSSDK
  - Obter dados sobre um trabalho de análise de documentos Amazon Textract usando umAWSSDK
  - Inicie a análise assíncrona de um documento usando o Amazon Textract e umAWSSDK
  - Inicie a detecção de texto assíncrona usando o Amazon Textract e umAWSSDK
- Exemplos entre serviços do Amazon Textract
  - Criar uma aplicação de exploração do Amazon Textract
  - Detectar entidades em texto extraído de uma imagem usando umAWSSDK

Ações para o Amazon Textract

Os exemplos de código a seguir demonstram como executar ações individuais do Amazon Textract com oAWSSDKs. Esses trechos chamam a API Amazon Textract e não devem ser executados

isoladamente. Cada exemplo inclui um link para o GitHub, onde você pode encontrar instruções sobre como configurar e executar o código no contexto.

Os exemplos a seguir incluem apenas as ações mais usadas. Para obter uma lista completa, consulte oReferência da API do Amazon Textract.

Exemplos

- Analise um documento usando o Amazon Textract e umAWSSDK
- Detectar texto em um documento usando o Amazon Textract e umAWSSDK
- Obter dados sobre um trabalho de análise de documentos Amazon Textract usando umAWSSDK
- Inicie a análise assíncrona de um documento usando o Amazon Textract e umAWSSDK
- Inicie a detecção de texto assíncrona usando o Amazon Textract e umAWSSDK

Analise um documento usando o Amazon Textract e umAWSSDK

Os exemplos de código a seguir mostram como analisar um documento usando o Amazon Textract.

Java

SDK para 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);
        }
    }
```

- Encontre instruções e mais código no GitHub.
- Para obter mais detalhes da API, consulte.AnalyzeDocumentemAWS SDK for Java 2.xReferência de API do.

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
```

- Encontre instruções e mais código no GitHub.
- Para obter mais detalhes da API, consulte.AnalyzeDocumentemAWSReferência da API do SDK for Python (Boto3).

Para obter uma lista completa deAWSGuias do desenvolvedor do SDK e exemplos de código, consulteUsando o Amazon Textract com umAWSSDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Detectar texto em um documento usando o Amazon Textract e umAWSSDK

Os exemplos de código a seguir mostram como detectar texto em um documento usando o Amazon Textract.

Java

SDK para Java 2.x

Detecte texto de um documento de entrada.

```
   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);
        }
    }
```

Detecte texto de um documento localizado em um bucket do Amazon S3.

```
    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);
        }
    }
```

- Encontre instruções e mais código no GitHub.
- Para obter mais detalhes da API, consulte.DetectDocumentTextemAWS SDK for Java 2.xReferência de API do.

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
```

- Encontre instruções e mais código no GitHub.
- Para obter mais detalhes da API, consulte.DetectDocumentTextemAWSReferência da API do SDK for Python (Boto3).

Para obter uma lista completa deAWSGuias do desenvolvedor do SDK e exemplos de código, consulteUsando o Amazon Textract com umAWSSDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Obter dados sobre um trabalho de análise de documentos Amazon Textract usando umAWSSDK

O exemplo de código a seguir mostra como obter dados sobre uma tarefa de análise de documentos do Amazon Textract.

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
```

- Encontre instruções e mais código no GitHub.
- Para obter mais detalhes da API, consulte.GetDocumentAnalysisemAWSReferência da API do SDK for Python (Boto3).

Para obter uma lista completa deAWSGuias do desenvolvedor do SDK e exemplos de código, consulteUsando o Amazon Textract com umAWSSDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Inicie a análise assíncrona de um documento usando o Amazon Textract e umAWSSDK

Os exemplos de código a seguir mostram como iniciar uma análise assíncrona de um documento usando o Amazon Textract.

Java

SDK para 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 "";
    }
```

- Encontre instruções e mais código no GitHub.
- Para obter mais detalhes da API, consulte.StartDocumentAnalysisemAWS SDK for Java 2.xReferência de API do.

Python

SDK for Python (Boto3).

Inicie um trabalho assíncrono para analisar um documento.

```
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
```

- Encontre instruções e mais código no GitHub.
- Para obter mais detalhes da API, consulte.StartDocumentAnalysisemAWSReferência da API do SDK for Python (Boto3).

Para obter uma lista completa deAWSGuias do desenvolvedor do SDK e exemplos de código, consulteUsando o Amazon Textract com umAWSSDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Inicie a detecção de texto assíncrona usando o Amazon Textract e umAWSSDK

O exemplo de código a seguir mostra como iniciar a detecção de texto assíncrona em um documento usando o Amazon Textract.

Python

SDK for Python (Boto3).

Inicie uma tarefa assíncrona para detectar texto em um documento.

```
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
```

- Encontre instruções e mais código no GitHub.
- Para obter mais detalhes da API, consulte.StartDocumentTextDetectionemAWSReferência da API do SDK for Python (Boto3).

Para obter uma lista completa deAWSGuias do desenvolvedor do SDK e exemplos de código, consulteUsando o Amazon Textract com umAWSSDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Exemplos entre serviços do Amazon Textract

Os aplicativos de amostra a seguir usamAWSSDKs para combinar o Amazon Textract com outrosAWSServiços da . Cada exemplo inclui um link para o GitHub, onde você pode encontrar instruções sobre como configurar e executar o aplicativo.

Exemplos

- Criar uma aplicação de exploração do Amazon Textract
- Detectar entidades em texto extraído de uma imagem usando umAWSSDK

Criar uma aplicação de exploração do Amazon Textract

Os exemplos de código a seguir mostram como explorar a saída do Amazon Textract por meio de uma aplicação interativa.

JavaScript

SDK para JavaScript V3

Mostra como usar o AWS SDK para JavaScript para criar uma aplicação React que usa o Amazon Textract para extrair dados de uma imagem de documento e exibi-los em uma página da Web interativa. Este exemplo é executado em um navegador da Web e requer uma identidade autenticada do Amazon Cognito como credenciais. Ele usa o Amazon Simple Storage Service (Amazon S3) para armazenamento e, para notificações, pesquisa uma fila do Amazon Simple Queue Service (Amazon SQS) que está inscrita em um tópico do Amazon Simple Notification Service (Amazon SNS).

Para obter o código-fonte completo e instruções sobre como configurar e executar o exemplo, consulte o exemplo completo no GitHub.

Serviços usados neste exemplo

- Amazon Cognito Identity
- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Python

SDK for Python (Boto3).

Mostra como usar o AWS SDK para Python (Boto3) com o Amazon Textract para detectar elementos de texto, formulários e tabelas em uma imagem de documento. A imagem de entrada e a saída do Amazon Textract são mostradas em uma aplicaçãoTkinter que permite explorar os elementos detectados.

- Envie uma imagem de documento para o Amazon Textract e explore a saída dos elementos detectados.
- Envie imagens diretamente para o Amazon Textract ou por meio de um bucket do Amazon Simple Storage Service (Amazon S3).
- Use APIs assíncronas para iniciar um trabalho que publica uma notificação em um tópico do Amazon Simple Notification Service (Amazon SNS) quando o trabalho for concluído.

- Faça uma pesquisa em uma fila do Amazon Simple Queue Service (Amazon SQS) para obter uma mensagem de conclusão do trabalho e exiba os resultados.

Para obter o código-fonte completo e instruções sobre como configurar e executar o exemplo, consulte o exemplo completo no GitHub.

Serviços usados neste exemplo

- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Para obter uma lista completa deAWSGuias do desenvolvedor do SDK e exemplos de código, consulteUsando o Amazon Textract com umAWSSDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Detectar entidades em texto extraído de uma imagem usando umAWSSDK

O exemplo de código a seguir mostra como usar o Amazon Comprehend para detectar entidades em texto extraído pelo Amazon Textract de uma imagem armazenada no Amazon S3.

Python

SDK for Python (Boto3).

Mostra como usar oAWS SDK para Python (Boto3)em um notebook Jupyter para detectar entidades em texto extraído de uma imagem. Este exemplo usa o Amazon Textract para extrair texto de uma imagem armazenada no Amazon Simple Storage Service (Amazon S3) e no Amazon Comprehend para detectar entidades no texto extraído.

Este exemplo é um notebook Jupyter e deve ser executado em um ambiente que possa hospedar notebooks. Para obter instruções sobre como executar o exemplo usando o Amazon SageMaker, consulte as instruções noTextactAndComprehendNotebook.IPYNB.

Para obter o código-fonte completo e instruções sobre como configurar e executar o exemplo, consulte o exemplo completo no GitHub.

Serviços usados neste exemplo

- Amazon Comprehend

- Amazon S3
- Amazon Textract

Para obter uma lista completa deAWSGuias do desenvolvedor do SDK e exemplos de código, consulteUsando o Amazon Textract com umAWSSDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usando a Augmented AI da Amazon para adicionar análise humana à saída do Amazon Textract

O Amazon Augmented AI (Amazon A2I) é um serviço de machine Learning (ML) que facilita a criação de fluxos de trabalho para a análise humana das análises de ML.

Amazon Textract é integrado ao Amazon A2I. Você pode usá-lo para rotear resultados de análise de documentos que tenham uma baixa pontuação de confiança para revisores humanos.

Você pode usar o Amazon Textract`AnalyzeDocument`API para extrair dados de formulários e do console Amazon A2I. Você pode especificar as condições sob as quais a Amazon A2I encaminha previsões para revisores. Você define condições com base no limite de confiança de chaves de formulário importantes. Por exemplo, você pode enviar um documento para um revisor humano se a chaveName (Nome)ou seu valor associadoJane Doefoi detectado com baixa confiança.

Tópicos

- Conceitos básicos do Amazon A2I
- Comece a usar o Amazon A2I

Conceitos básicos do Amazon A2I

Revise os seguintes termos para se familiarizar com os principais conceitos do Amazon A2I.

Condições de ativação da revisão humana

Você pode usar o Amazon A2Icondições de ativaçãopara especificar quando um documento é enviado a humanos para revisão e o conteúdo do formulário que os trabalhadores são solicitados a revisar.

Por exemplo, você pode definir uma condição de ativação para que o Amazon Textract roteie formulários para o Amazon A2I quando uma chave importante for detectada com baixa confiança, comoPhone Number (Número de telefone). Neste exemplo, os revisores humanos são solicitados a rever oPhone Number (Número de telefone)campo e valor associados detectados pelo Amazon Textract.

Você pode usar as seguintes condições de ativação para especificar quando os formulários são enviados para humanos para revisão:

- Acionar uma análise humana para chaves de formulário específicas com base na pontuação de confiança da chave do formulário. Os revisores humanos devem revisar essas chaves de formulário e valores associados.
- Acionar uma análise humana quando chaves de formulário específicas estão faltando. Os revisores humanos são solicitados a identificar essas chaves de formulário e valores associados.
- Acionar a análise humana para todas as chaves de formulário identificadas pelo Amazon Textract com pontuações de confiança em um intervalo especificado.
- Enviar aleatoriamente uma amostra dos formulários a humanos para análise. Os revisores humanos devem revisar todas as chaves de formulário e valores detectados pelo Amazon Textract.

Quando sua condição de ativação depender das pontuações de confiança da chave do formulário, você pode usar dois tipos de limites de confiança de previsão para acionar a análise humana:

- Confiança na— A pontuação de confiança para os pares chave-valor dentro de um formulário.
- Confiança na qualificação— A pontuação de confiança para o texto contido em um par chave-valor em um formulário.

Se você especificar um limite de confiança, o Amazon A2I encaminha somente as previsões que estão dentro do limite para revisores humanos. Você pode ajustar esses limites a qualquer momento para alcançar o equilíbrio certo entre precisão e custo-benefício. Isso pode ajudá-lo a implementar auditorias para monitorar regularmente a precisão da previsão.

> **Note**
>
> Você pode personalizar ainda mais as condições sob as quais os documentos são enviados para humanos para revisão usando o tipo de tarefa personalizada do Amazon A2I. Com esse tipo de tarefa, você especifica condições para uma revisão humana diretamente em seu aplicativo. Para obter mais informações, consulteUsar a Augmented AI da Amazon com tipos de tarefa personalizadosno Guia do desenvolvedor do Amazon SageMaker.

Fluxo de trabalho de revisão humana (definição de fluxo)

Você usa umfluxo de trabalho de revisão humana, também conhecido como umDefinição de fluxo, para especificar recursos usados para criar seu fluxo de trabalho de revisão humana e especificar suas condições de ativação.

Os recursos especificados são:

- Uma função do IAM com permissão para chamar as operações da API do Amazon A2I
- Um bucket do Amazon S3 no qual você deseja armazenar a saída da análise humana
- Seu humanoequipe de trabalho
- UMAModelo de tarefa de operadorque inclui instruções e exemplos para ajudar os trabalhadores a concluir a tarefa de revisão

Você também usa o fluxo de trabalho de revisão humana para especificar condições de ativação. Para obter mais informações, consulte Condições de ativação da revisão humana.

É possível usar um único fluxo de trabalho de revisão humana para criar váriosLoops humanos.

Você pode criar um fluxo de trabalho de revisão humana no console do SageMaker ou com a API do SageMaker. Para obter mais informações, consulte Criar um fluxo de trabalho de revisão humana.

Modelo de tarefa do trabalhador

Você usa umModelo de tarefa de operadorpara criar uma interface de usuário usada para suas tarefas de revisão humana.

A interface do usuário do trabalhador exibe seus documentos e instruções do trabalhador. Ele também fornece ferramentas que os trabalhadores usam para concluir suas tarefas.

Você pode usar o console do SageMaker para configurar o modelo de tarefa de operador ao criar um fluxo de trabalho de revisão humana. Para obter mais informações, consulte Criar um fluxo de trabalho de revisão humana.

Equipe de trabalho

UMAequipe de trabalhoé um grupo de trabalhadores humanos para os quais você envia suas tarefas de revisão humana.

Quando você cria um fluxo de trabalho de revisão humana, você especifica uma única equipe de trabalho.

Com o Amazon A2I, você pode usar um conjunto de revisores dentro da sua própria organização. Você também pode acessar a força de trabalho composta por mais de 500.000 contratados independentes que já estão realizando tarefas de aprendizado de máquina por meio do Amazon Mechanical Turk. Outra opção é usar fornecedores de força de trabalho que são selecionados pela AWS para qualidade e conformidade com os procedimentos de segurança.

O Amazon A2I também fornece aos revisores uma interface da Web que consiste em todas as instruções e ferramentas necessárias para concluir suas tarefas de revisão.

Para cada tipo de força de trabalho (privada, fornecedor e Mechanical Turk), você pode criar várias equipes de trabalho. Você pode usar cada equipe de trabalho em vários fluxos de trabalho de revisão humana. Para aprender a criar uma força de trabalho e equipes de trabalho, consulteCriar e gerenciar forças de trabalhono Guia do desenvolvedor do Amazon SageMaker.

> **Important**
>
> Clique emaquipara ver os programas de conformidade que cobrem a Augmented AI da Amazon no momento. Se você usar o Amazon Augmented AI em conjunto com outros serviços da AWS (como o Amazon Rekognition e o Amazon Textract), observe que o Amazon Augmented AI pode não estar no escopo para os mesmos programas de conformidade que os outros serviços. Você é responsável por como usar o Amazon Augmented AI, incluindo o entendimento de como o serviço processará ou armazenará os dados de clientes e qualquer impacto na conformidade do ambiente de dados. Você deve discutir seus objetivos de carga de trabalho com a equipe de contas da AWS. Eles podem ajudá-lo a avaliar se o serviço é adequado para o caso de uso propostos e arquitetura. Atualmente, a Augmented AI da Amazon é compatível com PCI, exceto para casos de força de trabalho pública e de fornecedores.
> Para obter informações sobre a conformidade com a HIPAA com Augmented AI da Amazon, clique emaqui.

Loops humanos

Usar um loop humano para criar uma tarefa de revisão humana.

Você atribui uma tarefa de revisão humana a um trabalhador em sua equipe de trabalhadores humanos. O trabalhador é solicitado a revisar pares de valores-chave detectados pelo Amazon Textract no documento de entrada que você especificou em suas condições de ativação.

Por exemplo, digamos que uma imagem caia entre cinquenta e sessenta por cento de confiança de que contém uma maçã. Isso está dentro de seu limite de confiança para revisão humana e é enviado para um trabalhador. O trabalhador verifica o documento em busca de uma maçã, marcando-o como contendo ou não contendo uma maçã. Em seguida, o Amazon A2I envia o documento de volta para o fluxo de trabalho.

Quando você chama o Amazon Textract`AnalyzeDocumente` especifique um fluxo de trabalho de revisão humana (definição de fluxo) e um nome de loop humano, uma tarefa de revisão humana é criada quando as condições de ativação especificadas em seu fluxo de trabalho de revisão humana são atendidas. Essas tarefas são criadas usando os recursos especificados em seu fluxo de trabalho de revisão humana.

Comece a usar o Amazon A2I

As etapas a seguir ajudam você a integrar o Amazon A2I em uma tarefa de análise de documento de página única do Amazon Textract. Faça o seguinte:

1. Crie um fluxo de trabalho de revisão humana usando o console do Amazon A2I (recomendado para novos usuários) ou a API do Amazon A2I.
2. Para analisar um formulário e incluir revisão humana quando necessário, use o`AnalyzeDocument`Operação e especifique o nome de recurso da Amazon (ARN) do fluxo de trabalho de revisão humana. A resposta informa se a revisão humana é necessária.
3. Monitore seu loop humano usando o console e a API do Amazon A2I.
4. Revise os resultados da revisão humana em um bucket do Amazon S3 para onde os resultados são enviados.

Para configurar uma instância de notebook SageMaker e usar um bloco de notas de exemplo, consulteDemonstração completa usando o Amazon Textract e a Augmented AInoGuia do desenvolvedor do Amazon SageMaker

> **Note**
>
> Esta seção explica como criar um fluxo de trabalho de revisão humana para o tipo de tarefa Amazon A2I, Amazon Textract. Para personalizar ainda mais a integração com o Amazon A2I e o Amazon Textract, você pode usar o tipo de tarefa personalizada do Amazon A2I. Com essa opção, você fornece um modelo de tarefa de trabalhador personalizado

e especifica as condições sob as quais um documento é enviado para revisão humana diretamente no aplicativo. Para obter mais informações, consulteUsar a Augmented AI da Amazon com tipos de tarefa personalizadosnoGuia do desenvolvedor do Amazon SageMaker.

Tópicos

- Criar um fluxo de trabalho de revisão humana
- Analisar o documento
- Monitorar o loop humano
- Exibir dados de saída e métricas do trabalhador

Criar um fluxo de trabalho de revisão humana

Você pode criar um fluxo de trabalho de revisão humana usando o console Amazon A2I (recomendado para novos usuários) ou o Amazon A2I`CreateFlowDefinition`operação.

Tópicos

- Criar um fluxo de trabalho de revisão humana (console)
- Criar um fluxo de trabalho de revisão humana (API)

Criar um fluxo de trabalho de revisão humana (console)

Você pode concluir este exemplo usando seu próprio documento no Amazon S3 ou fazer o downloadeste exemplo de documentoe coloque-o no seu bucket do Amazon S3.

Verifique se seu bucket do S3 está no mesmoAWSRegião que você está usando o Amazon Textract. Para criar um bucket, consulteCrie um bucketnoGuia do usuário do console do Amazon Simple Storage Service.

> **Note**
>
> O console Amazon A2I está incorporado no console do SageMaker. Para usar o console, é necessário permissões para acessar o console do SageMaker e criar uma equipe de trabalho. Para começar, você pode usar oAmazonSageMakerFullAccessPolítica gerenciada

do IAM que inclui todas as permissões necessárias para executar a maioria das ações no SageMaker. Para obter mais informações, consulteIdentity and Access Management para o Amazon SageMakernoGuia do desenvolvedor do Amazon SageMaker.

Tópicos

- Etapa 1: Criar uma equipe de trabalho (Console)
- Etapa 2: Criar um fluxo de trabalho de revisão humana (console)

Etapa 1: Criar uma equipe de trabalho (Console)

Primeiro, crie uma equipe de trabalho no console do Amazon A2I e adicione-se como trabalhador para que você possa visualizar a tarefa de revisão humana no portal do trabalhador, os membros da equipe de trabalho podem examinar diferentes tarefas e documentos atribuídos a eles.

Para criar uma força de trabalho privada usando e-mails de operador (Console)

1. Abra o console do SageMaker emhttps://console.aws.amazon.com/sagemaker/.
2. No painel de navegação, emGround Truth, escolhaForças de trabalho de rotulação.
3. Selecione Private (Privado) e escolha Create private team (Criar equipe privada).
4. Selecione Invite new workers by email (Convidar novos operadores por e-mail).
5. Para este exemplo, insira seu endereço de e-mail e o endereço de e-mail de qualquer outro que você deseja visualizar o portal do trabalhador. Você pode colar ou digitar uma lista de até 50 endereços de e-mail, separados por vírgulas, noEndereços de e-mail(Criar snapshot final?).
6. Insira o nome de uma organização e um e-mail de contato.
7. Selecione Create private team (Criar equipe privada).

Se você se adicionar a uma equipe de trabalho particular, receberá um e-mail de`no-reply@verificationemail.com`com informações de login. Use o link neste e-mail para redefinir sua senha e fazer login no portal do trabalhador. É aqui que suas tarefas de revisão humana aparecerão depois que você ligar`AnalyzeDocument`.

Etapa 2: Criar um fluxo de trabalho de revisão humana (console)

Nesta etapa, você cria um fluxo de trabalho de revisão humana Amazon Textract.

Para criar um fluxo de trabalho de revisão humana (console)

1. Abra o console do Amazon A2I emhttps://console.aws.amazon.com/a2iPara acessar ofluxos de trabalho de revisão humana.
2. SelecioneCriar fluxo de trabalho de revisão humana.
3. para oName (Nome), insira um nome de fluxo de trabalho.
4. para oS3 bucket (Bucket do S3)Escolha o bucket no qual você deseja que o Amazon A2I armazene os resultados de suas tarefas de análise humana. Se você não escolher um bucket, altere-o para inserir o nome do bucket.
5. UNDERIAM role (Função do IAM), selecioneCriar uma nova função. Uma janela aparece com o títuloCriar uma função do IAM. Use essa janela para especificar os buckets do Amazon S3 aos quais você deseja que essa função tenha acesso. Se você não selecionarQualquer bucket do S3, especifique o intervalo de saída especificado na etapa 4 e o bucket que contém o documento de entrada.
6. para oTipo de tarefa, escolhaTextract par de chave/valor.
7. DentroExtração de formulários Amazon Textract - Condições para invocar revisão humana, especifique as condições de ativação. Recomendamos que você defina um limite de pontuação de confiança alto para pelo menos uma chave no documento para acionar uma revisão humana para que você possa visualizar uma tarefa do trabalhador no portal do trabalhador.

   Se você usou o documento de amostra fornecido neste passo a passo, especifique as condições de ativação da seguinte maneira:

   a. SelecioneAcionar uma análise humana para chaves de formulário específicas com base na pontuação de confiança da chave do formulário ou quando chaves de formulário específicas estiverem ausentes.
   b. para oNome da chave, insira**`Mail Address`**.
   c. DefinaIdentificação de confiançaLimite entre0e99.
   d. DefinaConfiança qualificaçãoLimite entre0e99.
   e. SelecioneAcionar uma análise humana para todas as chaves de formulário identificadas pelo Amazon Textract com pontuações de confiança em um intervalo específico.
   f. para o**`identification confidence`**, escolha qualquer número entre 0 e 90.
   g. para o**`qualification confidence`**, escolha qualquer número entre 0 e 90.

Isso aciona uma avaliação humana se o Amazon Textract retornar uma pontuação de confiança menor que 99 paraEndereço de correioe seu valor, ou se retornar uma pontuação de confiança menor que 90 para qualquer par de valores-chave detectado no documento.

8. UNDERCriação de modelo de tarefa de, selecioneCriar a partir de um modelo padrão.
9. para oNome do modelo, insira um nome descritivo.
10. para oTask description, adicione algo semelhante ao seguinte:

    ```
    Read the instructions and review the document.
    ```

11. para oOperadoresescolherPrivate.
12. No menu, escolha a equipe privada que você criou.
13. Escolha Create (Criar).

Depois que seu fluxo de trabalho de revisão humana for criado, ele aparece na tabela nafluxos de trabalho de revisão humana. Quando oStatuséAtivo, copie e salve o ARN de fluxo de trabalho.

Criar um fluxo de trabalho de revisão humana (API)

É possível criar um fluxo de trabalho de revisão humana ou umDefinição de fluxo, usando o Amazon A2I,`CreateFlowDefinition`operação.

Para este exemplo, você pode usar seu próprio documento no Amazon S3 ou fazer o downloadeste exemplo de documentoe armazene-o no seu bucket do S3.

Verifique se seu bucket do Amazon S3 está no mesmoAWSRegião que você planeja usar para ligar`AnalyzeDocument`. Para criar um bucket, siga as instruções em Criar um bucket no Guia do usuário do console do Amazon Simple Storage Service.

Pré-requisitos

Para usar a API do Amazon A2I para criar um fluxo de trabalho de revisão humana, você deve preencher os seguintes pré-requisitos:

- Configure uma função do IAM com permissão para chamar as operações de API do Amazon A2I e Amazon Textract. Para começar, você pode anexar as políticas da AWS, AmazonAugmentedAIFullAccess e AmazonTextractFullAccess a uma função do IAM. Registre a função do IAM Amazon Resources Name (ARN) porque você precisará dela mais tarde.

Para obter permissões mais granulares ao usar o Amazon Textract, consulteExemplos de políticas baseadas em identidade do Amazon Textract. Para o Amazon A2I, consultePermissões e segurança na Augmented AI da AmazonnoGuia do desenvolvedor do Amazon SageMaker.

- Crie uma equipe de trabalho particular e registre o ARN da equipe de trabalho. Se você for um novo usuário do Amazon A2I, siga as instruções emEtapa 1: Criar uma equipe de trabalho (Console).
- Crie um modelo de tarefa de operador. Siga as instruções emCriar um modelo de tarefa de operadorpara criar um modelo usando o console Amazon A2I. Quando você estiver criando o modelo, escolhaExtração em forma de textopeloTipo de modelo. No modelo, substituas`3_arn`com o ARN do Amazon S3 do documento. Adicione outras instruções do trabalhador em`<full-instructions header="Instructions"></full-instructions>`.

  Se você quiser visualizar o modelo, verifique se a sua função do IAM tem as permissões descritas emHabilitar visualizações do modelo de tarefa de operador.

  Depois de criar seu modelo, registre o ARN do modelo de tarefa do trabalhador.

Usar recursos criados noPré-requisitosPara configurar o`CreateFlowDefinition`solicitação. Nesta solicitação, você também especifica condições de ativação no formato JSON. Para saber como configurar suas condições de ativação, consulteUsar o esquema JSON de condições de ativação de loop humano com o Amazon Textract.

Criando um fluxo de trabalho de revisão humana (AWS SDK for Python (Boto3))

Para usar esse exemplo, substitua a*`vermelho`*Texto com suas especificações e recursos.

Primeiro, codifique suas condições de ativação em um objeto JSON usando o código a seguir. Isso aciona uma avaliação humana se o Amazon Textract retornar uma pontuação de confiança menor que 99 paraEndereço de correioe seu valor, ou se retornar uma pontuação de confiança menor que 90 para qualquer par de valores-chave detectado no documento. Se você estiver usando o documento de amostra fornecido neste exemplo, essas condições de ativação criarão uma tarefa de revisão humana.

```
import json

humanLoopActivationConditions = json.dumps("{
                "Conditions": [
                    {
```

```
                        "ConditionType": "ImportantFormKeyConfidenceCheck",
                        "ConditionParameters": {
                            "ImportantFormKey": "Mail Address",
                            "KeyValueBlockConfidenceLessThan": 99,
                            "WordBlockConfidenceLessThan": 99
                        }
                    },
                    {
                        "ConditionType": "ImportantFormKeyConfidenceCheck",
                        "ConditionParameters": {
                            "ImportantFormKey": "*",
                            "KeyValueBlockConfidenceLessThan": 90,
                            "WordBlockConfidenceLessThan": 90
                        }
                    }
                ]
            }"
)
```

Usar o`humanLoopActivationConditions`para configurar o`create_flow_definition`solicitação. O exemplo a seguir usa o SDK for Python (Boto3) para chamar`create_flow_definition`na região us-west-2 da AWS. Ele especifica o uso de uma equipe de trabalho privada.

```
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"
)
```

Analisar o documento

Para incorporar o Amazon A2I em um fluxo de trabalho de análise de documentos Amazon Textract, você configura`HumanLoopConfig`no`AnalyzeDocument`operação.

Dentro`HumanLoopConfig`Você especifica seu fluxo de trabalho de revisão humana (definição de fluxo) ARN no`FlowDefinitionArn`, e dê um nome ao seu loop humano em`HumanLoopName`.

Analyze the Document (AWS SDK for Python (Boto3))

O exemplo a seguir usa o SDK for Python (Boto3) para chamar`analyze_document`em us-west-2. Substitua o*`vermelho, itálico`*texto com seus recursos. Para obter mais informações, consulte analyze_documentnoAWSReferência da API do SDK for Python (Boto).

```
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)

O exemplo a seguir usa aAWSCLI para ligar`analyze_document`. Esses exemplos são compatíveis comAWSCLI versão 2. A primeira é a sintaxe abreviada, a segunda na sintaxe JSON. Para obter mais informações, consulteAnalisar documentonoAWS CLIReferência de comando.

```
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**
>
> Evite espaços brancos em seu parâmetro —human-loop-config, pois isso pode causar problemas de processamento para seu código.

A resposta a essa solicitação contémHumanLoopActivationOutput, o que indica se um loop humano foi criado, e se foi, por quê. Se um loop humano foi criado, esse objeto também conterá o`HumanLoopArn`.

Para obter mais informações sobre o e exemplos usando o`AnalyzeDocument`operação, consulteAnalisando texto do documento com o Amazon Textract.

Monitorar o loop humano

Você pode visualizar detalhes sobre seu loop humano e interromper um loop humano ativo em caso de erro usando o console e a API do Amazon A2I.

Ver detalhes do loop humano

Você pode visualizar o status do loop humano no console do Amazon A2I e usando oAPI de tempo de execução do Amazon A2I.

Para encontrar detalhes sobre seu loop humano (console)

1. Abra o console do Amazon A2I emhttps://console.aws.amazon.com/a2iPara acessar ofluxos de trabalho de revisão humana.
2. Escolha o fluxo de trabalho de revisão humana que você usou também para configurar`HumanLoopConfig`em`AnalyzeDocument`.
3. NoLoops humanosSeção, escolha o loop humano cujos detalhes você deseja visualizar.

Para encontrar detalhes sobre seu loop humano (API):

Use o Amazon A2I`DescribeHumanLoop`operação. Especifique o nome do loop humano usado para chamar`AnalyzeDocument`.

Chamadas de exemplo do SDK para Python (Boto3)`describe_human_loop`.

```
response = client.describe_human_loop(HumanLoopName="human-loop-name")
```

Pare um loop humano

Depois que um loop humano for iniciado, é possível interrompê-lo usando o console e a API do Amazon A2I.

Para parar seu loop humano (console)

1. Abra o console do Amazon A2I emhttps://console.aws.amazon.com/a2iPara acessar ofluxos de trabalho de revisão humana.
2. Escolha o fluxo de trabalho de revisão humana usado para configurar`HumanLoopConfig`no`AnalyzeDocument`operação.
3. NoLoops humanosSeção, escolha o loop humano que você deseja interromper.
4. Escolha Stop (Parar).

Para interromper seu loop humano (API)

Use o Amazon A2I`StopHumanLoop`operação. Especifique o nome do loop humano usado para chamar`AnalyzeDocument`.

Chamadas de exemplo do SDK for Python (Boto3)`stop_human_loop`.

```
response = client.stop_human_loop(HumanLoopName="human-loop-name")
```

Exibir dados de saída e métricas do trabalhador

Quando uma tarefa de revisão humana é concluída por um trabalhador, o Amazon A2I armazena seus dados de saída no bucket do Amazon S3 especificado no fluxo de trabalho de revisão humana.

Se você usar uma força de trabalho privada, seus dados de saída conterão metadados de trabalho que você pode usar para rastrear a atividade do trabalhador individual.

Encontre dados de saída no Amazon S3

O Amazon A2I usa o nome do fluxo de trabalho de revisão humana como prefixo para o nome do arquivo que armazena os dados de saída de loops humanos criados usando esse fluxo de trabalho de revisão humana.

O caminho para uma saída de loop humano usa o seguinte padrão no qual*YYYY*/*MM*/*DD*/*hh*/*mm*/*ss*representa a data de criação do loop humano com o ano (`YYYY`), mês (`MM`) e dia (`DD`) e o tempo de criação com hora (`hh`), minuto (`mm`), e segundo (`ss`).

```
s3://output-bucket-specified-in-flow-definition/flow-definition-
name/YYYY/MM/DD/hh/mm/ss/human-loop-name/output.json
```

Para ver a saída de um loop humano, use o console do Amazon A2I.

Para ver a saída de loop humano

1. Abra o console do Amazon A2I emhttps://console.aws.amazon.com/a2iPara acessar ofluxos de trabalho de revisão humana.
2. Escolha o fluxo de trabalho de revisão humana usado para configurar`HumanLoopConfig`em`AnalyzeDocument`.
3. NoLoops humanos, escolha o loop humano cuja saída você deseja revisar.
4. UNDERLocal de saída, escolha o link para os dados de saída.

Acompanhe a atividade dos trabalhadores

Quando você usa uma força de trabalho privada para tarefas de revisão humana, os dados de saída incluem as seguintes informações sobre o trabalhador que concluiu a revisão:

- A ferramenta `workerId`.
- Em `workerMetadata`:
  - `identityProviderType`— O serviço usado para gerenciar a força de trabalho privada.
  - `issuer`— O grupo de usuários do Amazon Cognito ou o emissor do OIDC Identity Provider (IdP) associado à equipe de trabalho atribuída a esta tarefa de análise humana.
  - `sub`— Um identificador exclusivo que se refere ao operador. Se você criou uma força de trabalho usando o Amazon Cognito, é possível recuperar detalhes sobre esse operador (como o nome ou nome de usuário) usando esse ID usando o Amazon Cognito. Para saber como, consulteGerenciamento e pesquisa de contas de usuáriosemGuia do desenvolvedor do Amazon Cognito.

Veja a seguir um exemplo da saída que você pode ver se você usou o Amazon Cognito para criar uma força de trabalho privada.

```
"workerId": "a12b3cdefg4h5i67",
            "workerMetadata": {
                "identityData": {
                    "identityProviderType": "Cognito",
                    "issuer": "https://cognito-idp.aws-region.amazonaws.com/aws-
region_123456789",
                    "sub": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee"
```

Veja a seguir um exemplo da saída que você pode ver se você usou seu próprio OIDC IdP para criar uma força de trabalho privada:

```
"workerId": "a12b3cdefg4h5i67",
            "workerMetadata": {
                "identityData": {
                    "identityProviderType": "Oidc",
                    "issuer": "https://example-oidc-ipd.com/adfs",
                    "sub": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee"
```

Para saber mais sobre como usar forças de trabalho privadas, consulteUsar uma força de trabalho privadanoGuia do desenvolvedor do Amazon SageMaker.

Segurança no Amazon Textract

A segurança da nuvem na AWS é a nossa maior prioridade. Como cliente da AWS, você se beneficiará de um datacenter e de uma arquitetura de rede criados para atender aos requisitos das empresas com as maiores exigências de segurança.

Use os tópicos a seguir para saber como proteger seus recursos Amazon Textract.

Tópicos

- Proteção de dados no Amazon Textract
- Identity and Access Management para o Amazon Textract
- Registro em log e monitoramento
- Registro de chamadas de API do Amazon Textract comAWS CloudTrail
- Validação de conformidade para o Amazon Textract
- Resiliência no Amazon Textract
- Segurança da infraestrutura no Amazon Textract
- Análise de vulnerabilidade e configuração no Amazon Textract
- Amazon Textract e VPC endpoints de interface (AWS PrivateLink)

Proteção de dados no Amazon Textract

Amazon Textract está em conformidade com oAWS Modelo de responsabilidade compartilhada, que inclui normas e diretrizes para proteção de dados.AWSÉ responsável pela proteção da infraestrutura global que executa todas asAWSServiços da .AWSA mantém o controle sobre os dados hospedados nessa infraestrutura, incluindo os controles de configuração de segurança para lidar com conteúdos e dados pessoais dos clientes.AWSclientes eAPNOs parceiros, atuando como controladores de dados ou processadores de dados, são responsáveis por todos os dados pessoais que colocam naAWSNuvem.

Para fins de proteção de dados, recomendamos que você proteja as credenciais da conta da AWS e configure as contas de usuário individuais com o AWS Identity and Access Management . Isso garante que cada usuário receberá apenas as permissões necessárias para cumprir suas obrigações de trabalho. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA) com cada conta.
- Use SSL/TLS para se comunicar com os recursos da AWS.
- Configure o registro em log das atividades da API e do usuário com o AWS CloudTrail.
- Use as soluções de criptografia da AWS, juntamente com todos os controles de segurança padrão nos serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados pessoais armazenados no Amazon S3.

É altamente recomendável que você nunca coloque informações de identificação confidenciais, como números de conta dos seus clientes, em campos de formato livre, como um campo Name (Nome). Isso inclui quando você trabalha com o Amazon Textract ou outrosAWSserviços usando o console, a API,AWS CLI, ouAWSSDKs. Todos os dados inseridos no Amazon Textract ou em outros serviços poderão ser selecionados para inclusão em logs de diagnóstico. Ao fornecer um URL para um servidor externo, não inclua informações de credenciais no URL para validar a solicitação a esse servidor.

Para obter mais informações sobre proteção de dados, consulte a publicação Modelo de responsabilidade compartilhada da AWS e do GDPR no Blog de segurança da AWS.

Criptografia no Amazon Textract

A criptografia de dados refere-se à proteção de dados em trânsito e em repouso. Você pode proteger seus dados usando chaves gerenciadas do Amazon S3 ouAWS KMS keyem repouso, juntamente com a Segurança da Camada de Transporte padrão durante o trânsito.

Criptografia em repouso

O principal método de criptografia de dados no Amazon Textract é a criptografia do lado do servidor. Os documentos de entrada passados de buckets do Amazon S3 são criptografados pelo Amazon S3 e descriptografados quando você os acessa. Contanto que você autentique sua solicitação e tenha permissões de acesso, não há diferença na forma de acesso aos objetos criptografados ou não criptografados. Por exemplo, se você compartilhar seus objetos usando um pre-signed URL, esse URL funcionará da mesma forma para objetos criptografados e não criptografados. Além disso, quando você lista objetos no bucket do, `oListA` API retorna uma lista de todos os objetos, independentemente de estarem ou não criptografados.

O Amazon Textract usa dois métodos mutuamente exclusivos de criptografia no lado do servidor.

Criptografia do lado do servidor com chaves gerenciadas pelo Amazon S3 (SSE-S3)

Quando você usa criptografia no lado do servidor com as chaves gerenciadas pelo Amazon S3 (SSE-S3), cada objeto é criptografado com uma chave exclusiva. Como uma proteção adicional, esse método criptografa a a própria chave utilizando uma chave-mestra que é alterada regularmente. A criptografia no lado do servidor do Amazon S3 usa uma das cifras de bloco mais fortes disponíveis, o padrão de criptografia avançada de 256 bits (AES-256), para criptografar seus dados. Para obter mais informações, consulte Proteção de dados usando criptografia no lado do servidor com chaves de criptografia gerenciadas do Amazon S3 (SSE-S3).

Criptografia no lado do servidor com chaves do KMS armazenadas no AWS Key Management Service (SSE-KMS)

A criptografia no lado do servidor com chaves KMS armazenadas no AWS Key Management Service (SSE-KMS) é semelhante a SSE-S3, mas com alguns benefícios adicionais e cobranças para usar esse serviço. Há outras permissões para uso de uma chave do KMS que fornece maior proteção contra acesso não autorizado de seus objetos no Amazon S3. O SSE-KMS também fornece uma trilha de auditoria que mostra quando a chave do KMS foi usada e por quem. Além disso, você pode criar e gerenciar chaves KMS ou usarChaves gerenciadas pela AWSSão exclusivas para você, para o seu serviço e para a sua região. Para obter mais informações, consulteProteção de dados usando criptografia no lado do servidorCom chaves KMS armazenadas no AWS Key Management Service (SSE-KMS).

Criptografia em trânsito

Para dados em trânsito, o Amazon Textract usa Transport Layer Security (TLS) para criptografar dados enviados entre o serviço e o agente. Além disso, o Amazon Textract usa endpoints VPC para enviar dados entre os vários microsserviços usados quando o Amazon Textract processa um documento.

Privacidade do tráfego entre redes

O Amazon Textract se comunica exclusivamente por meio de endpoints HTTPS, que são compatíveis em todas as regiões suportadas pelo Amazon Textract

Identity and Access Management para o Amazon Textract

O AWS Identity and Access Management (IAM) é um serviço da AWS que ajuda a controlar o acesso aos recursos da AWS de forma segura. Os administradores do IAM controlam quem pode

serautenticada(assinado) eautorizada(tem permissões) para usar os recursos do Amazon Textract. O IAM é um serviço da AWS que pode ser usado sem custo adicional.

Tópicos

- Público
- Autenticação com identidades
- Gerenciamento do acesso usando políticas
- Como o Amazon Textract funciona com o IAM
- Exemplos de políticas baseadas em identidade do Amazon Textract
- Solução de Amazon Textract e acesso

Público

Como você usaAWS Identity and Access ManagementO (IAM) varia em função do trabalho realizado no Amazon Textract.

Usuário do serviço— se você usar o serviço Amazon Textract para fazer sua tarefa, o administrador fornecerá as credenciais e as permissões de que você precisa. À medida que mais recursos Amazon Textract forem usados para realizar o trabalho, talvez sejam necessárias permissões adicionais. Entender como o acesso é gerenciado pode ajudar você a solicitar as permissões corretas ao seu administrador. Se não for possível acessar um recurso no Amazon Textract, consulteSolução de Amazon Textract e acesso.

Administrador de serviçosSe você for o responsável pelos recursos do Amazon Textract em sua empresa, você provavelmente terá acesso total ao Amazon Textract. É seu trabalho determinar quais recursos do Amazon Textract os funcionários devem acessar. Assim, é necessário enviar solicitações ao administrador do IAM para alterar as permissões dos usuários de seu serviço. Revise as informações nesta página para entender os conceitos básicos do IAM. Para saber mais sobre como a empresa pode usar o IAM com o Amazon Textract, consulteComo o Amazon Textract funciona com o IAM.

Administrador do IAM— se você é um administrador do IAM, talvez queira saber detalhes sobre como pode escrever políticas para gerenciar o acesso ao Amazon Textract. Para visualizar exemplos de políticas baseadas em identidade do Amazon Textract que podem ser usadas no IAM, consulteExemplos de políticas baseadas em identidade do Amazon Textract.

Autenticação com identidades

A autenticação é a forma como você faz login na AWS usando suas credenciais de identidade. Para obter mais informações sobre como fazer login usando o AWS Management Console, consulte Login no AWS Management Console como usuário do IAM ou usuário root no Manual do usuário do IAM.

É necessário estar autenticado (conectado à AWS) como o usuário root da Conta da AWS ou um usuário do IAM, ou ainda assumindo uma função do IAM. Também é possível usar a autenticação de logon único da sua empresa ou até mesmo fazer login usando o Google ou o Facebook. Nesses casos, o administrador configurou anteriormente federação de identidades usando funções do IAM. Ao acessar a AWS usando credenciais de outra empresa, você estará assumindo uma função indiretamente.

Para fazer login diretamente no AWS Management Console, use sua senha com o e-mail do usuário root ou seu nome de usuário do IAM. É possível acessar a AWS de maneira programática usando chaves de acesso do seu usuário root ou dos usuários do IAM. AWS fornece ferramentas SDK e de linha de comando para assinar de forma criptográfica a sua solicitação usando suas credenciais. Se você não utilizar as ferramentas AWS, você deverá assinar a solicitação por conta própria. Faça isso usando o Signature versão 4, um protocolo para autenticação de solicitações de API de entrada. Para obter mais informações sobre solicitações de autenticação, consulte Processo de assinatura do Signature Version 4 na Referência geral da AWS.

Independentemente do método de autenticação usado, também pode ser exigido que você forneça informações adicionais de segurança. Por exemplo, a AWS recomenda o uso da autenticação multifator (MFA) para aumentar a segurança de sua conta. Para saber mais, consulte Uso da autenticação multifator (MFA) na AWS no Manual do usuário do IAM.

Usuário root da Conta da AWS

Ao criar uma Conta da AWS, você começa com uma única identidade de login que tenha acesso total a todos os recursos e serviços da AWS na conta. Essa identidade é denominada usuário root da Conta da AWS e é acessada pelo login com o endereço de e-mail e a senha que você usou para criar a conta. Recomendamos que não use o usuário raiz para suas tarefas do dia a dia, nem mesmo as administrativas. Em vez disso, siga as práticas recomendadas para o uso do usuário root somente a fim de criar seu primeiro usuário do IAM. Depois, armazene as credenciais do usuário raiz com segurança e use-as para executar somente algumas tarefas de gerenciamento de contas e de serviços.

Grupos e usuários do IAM

Um usuário do IAM é uma identidade dentro da Conta da AWS que tem permissões específicas para uma única pessoa ou aplicação. Um usuário do IAM pode ter credenciais de longo prazo, como um nome de usuário e uma senha ou um conjunto de chaves de acesso. Para saber como gerar chaves de acesso, consulte Gerenciar chaves de acesso para usuários do IAM no Manual do usuário do IAM. Ao gerar chaves de acesso para um usuário do IAM, visualize e salve o par de chaves de maneira segura. Não será possível recuperar a chave de acesso secreta futuramente. Em vez disso, você deverá gerar outro par de chaves de acesso.

Um grupo do IAM é uma identidade que especifica uma coleção de usuários do IAM. Não é possível fazer login como um grupo. É possível usar grupos para especificar permissões para vários usuários de uma vez. Os grupos facilitam o gerenciamento de permissões para grandes conjuntos de usuários. Por exemplo, você pode ter um grupo chamado IAMAdmins e atribuir a esse grupo permissões para administrar recursos do IAM.

Usuários são diferentes de funções. Um usuário é exclusivamente associado a uma pessoa ou a uma aplicação, mas uma função pode ser assumida por qualquer pessoa que precisar dela. Os usuários têm credenciais permanentes de longo prazo, mas as funções fornecem credenciais temporárias. Para saber mais, consulte Quando criar um usuário do IAM (em vez de uma função) no Manual do usuário do IAM.

Funções do IAM

Uma função do IAM é uma identidade dentro da Conta da AWS que tem permissões específicas. Ela é semelhante a um usuário do IAM, mas não está associada a uma pessoa específica. É possível assumir temporariamente uma função do IAM no AWS Management Console alternando funções. É possível assumir uma função chamando uma operação de API da AWS CLI ou da AWS, ou usando um URL personalizado. Para mais informações sobre métodos para o uso de funções, consulte Usar funções do IAM no Manual do usuário do IAM.

As funções do IAM com credenciais temporárias são úteis nas seguintes situações:

- Permissões temporárias para usuários do IAM: um usuário do IAM pode assumir uma função do IAM para obter temporariamente permissões diferentes para uma tarefa específica.
- Acesso de usuário federado: em vez de criar um usuário do IAM, você poderá usar identidades de usuários existentes no AWS Directory Service, em seu diretório de usuários corporativos ou em um provedor de identidades da Web. Estes são conhecidos como usuários federados. A AWS atribui uma função a um usuário federado quando o acesso é solicitado por meio de um provedor de

identidades. Para obter mais informações sobre usuários federados, consulte Usuários federados e funções no Manual do usuário do IAM.

- Acesso entre contas: é possível usar uma função do IAM para permitir que alguém (um principal confiável) em outra conta acesse recursos em sua conta. As funções são a principal forma de conceder acesso entre contas. No entanto, alguns serviços da AWS permitem que você anexe uma política diretamente a um recurso (em vez de usar uma função como proxy). Para saber a diferença entre funções e políticas baseadas em recurso para acesso entre contas, consulte Como as funções do IAM diferem das políticas baseadas em recurso no Manual do usuário do IAM.
- Acesso entre serviços: alguns serviços da AWS usam recursos em outros serviços da AWS. Por exemplo, quando você faz uma chamada em um serviço, é comum que esse serviço execute aplicações no Amazon EC2 ou armazene objetos no Amazon S3. Um serviço pode fazer isso usando as permissões do principal de chamada, usando uma função de serviço ou uma função vinculada ao serviço.
  - Permissões de principal: ao usar um usuário ou uma função do IAM para executar ações na AWS, você é considerado um principal. As políticas concedem permissões a uma entidade principal. Quando você usa alguns serviços, pode executar uma ação que, em seguida, aciona outra ação em outro serviço. Nesse caso, você deve ter permissões para executar ambas as ações. Para ver se uma ação requer ações dependentes adicionais em uma política, consulteAções, recursos e chaves de condição do Amazon TextractnoReferência de autorização do serviço.
  - Função de serviço: uma função de serviço é uma função do IAM que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir uma função de serviço do IAM. Para obter mais informações, consulte Criação de uma função para delegar permissões a um serviço da AWS no Manual do usuário do IAM.
  - Função vinculada a serviço: uma função vinculada a serviço é um tipo de função de serviço vinculada a um serviço da AWS. O serviço pode assumir a função de executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em sua conta do IAM e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode editar as permissões para funções vinculadas ao serviço.
- Aplicações em execução no Amazon EC2: é possível usar uma função do IAM para gerenciar credenciais temporárias para aplicações em execução em uma instância do EC2 e fazer solicitações da AWS CLI ou da AWS API. É preferível fazer isso do que armazenar chaves de acesso na instância do EC2. Para atribuir uma função da AWS a uma instância do EC2 e disponibilizá-la para todas as suas aplicações, crie um perfil de instância que esteja anexado à instância. Um perfil de instância contém a função e permite que programas que estão em

execução na instância do EC2 obtenham credenciais temporárias. Para mais informações, consulte Usar uma função do IAM para conceder permissões a aplicações em execução nas instâncias do Amazon EC2 no Manual do usuário do IAM.

Para saber se deseja usar as funções do IAM, consulte Quando criar uma função do IAM (em vez de um usuário) no Manual do usuário do IAM.

Gerenciamento do acesso usando políticas

Você controla o acesso na AWS criando e anexando políticas às identidades do IAM ou aos recursos da AWS. Uma política é um objeto na AWS que, quando associado a uma identidade ou recurso, define suas permissões. Você pode fazer login como o usuário raiz ou um usuário do IAM ou assumir uma função do IAM. Quando você faz uma solicitação, a AWS avalia as políticas relacionadas baseadas em identidade ou baseadas em recursos. As permissões nas políticas determinam se a solicitação será permitida ou negada. A maioria das políticas são armazenadas na AWS como documentos JSON. Para obter mais informações sobre a estrutura e o conteúdo de documentos de políticas JSON, consulte Visão geral das políticas JSON no Manual do usuário do IAM.

Os administradores podem usar as políticas JSON da AWS para especificar quem tem acesso a quê. Ou seja, qual principal pode executar ações em quais recursos, e em que condições.

Cada entidade do IAM (usuário ou função) começa sem permissões. Em outras palavras, por padrão, os usuários não podem fazer nada, nem mesmo alterar sua própria senha. Para dar permissão a um usuário para fazer algo, um administrador deve anexar uma política de permissões ao usuário. Ou o administrador pode adicionar o usuário a um grupo que tenha as permissões pretendidas. Quando um administrador concede permissões a um grupo, todos os usuários desse grupo recebem essas permissões.

As políticas do IAM definem permissões para uma ação, independentemente do método usado para executar a operação. Por exemplo, suponha que você tenha uma política que permite a ação `iam:GetRole`. Um usuário com essa política pode obter informações de funções do AWS Management Console, da AWS CLI ou da API da AWS.

Políticas baseadas em identidade

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário, grupo de usuários ou função do IAM. Essas políticas controlam quais ações os usuários e funções podem realizar, em quais recursos e em que

condições. Para saber como criar uma política baseada em identidade, consulte Criar políticas do IAM no Manual do usuário do IAM.

As políticas baseadas em identidade podem ser categorizadas ainda mais como políticas em linha ou políticas gerenciadas. As políticas em linha são anexadas diretamente a um único usuário, grupo ou função. As políticas gerenciadas são políticas independentes que podem ser anexadas a vários usuários, grupos e funções na Conta da AWS. As políticas gerenciadas incluem políticas gerenciadas pela AWS e políticas gerenciadas pelo cliente. Para saber como escolher entre uma política gerenciada ou uma política em linha, consulte Escolher entre políticas gerenciadas e políticas em linha no Manual do usuário do IAM.

Políticas com base em recurso

Políticas baseadas em recurso são documentos de políticas JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de função do IAM e as políticas de bucket do Amazon S3. Em serviços compatíveis com políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o recurso ao qual a política está anexada, a política define quais ações um principal especificado pode executar nesse recurso e em que condições. Você deve especificar um principal em uma política baseada em recursos. Os principais podem incluir contas, usuários, funções, usuários federados ou serviços da AWS.

Políticas baseadas em recursos são políticas em linha que estão localizadas nesse serviço. Não é possível usar as políticas gerenciadas da AWS do IAM em uma política baseada em recursos.

Listas de controle de acesso (ACLs)

As listas de controle de acesso (ACLs) controlam quais principais (membros, usuários ou funções da conta) têm permissões para acessar um recurso. As ACLs são semelhantes às políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

Amazon S3, AWS WAF e Amazon VPC são exemplos de serviços que oferecem suporte a ACLs. Para saber mais sobre ACLs, consulte Visão geral da lista de controle de acesso (ACL) no Guia do desenvolvedor do Amazon Simple Storage Service.

Outros tipos de política

A AWS oferece suporte a tipos de política menos comuns. Esses tipos de política podem definir o máximo de permissões concedidas a você pelos tipos de política mais comuns.

- Limites de permissões: um limite de permissões é um recurso avançado no qual você define o máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM (usuário ou função do IAM). É possível definir um limite de permissões para uma entidade. As permissões resultantes são a interseção das políticas baseadas em identidade da entidade e seus limites de permissões. As políticas baseadas em recurso que especificam o usuário ou a função no campo `Principal` não são limitadas pelo limite de permissões. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações sobre limites de permissões, consulte Limites de permissões para identidades do IAM no Manual do usuário do IAM.
- Políticas de controle de serviço (SCPs): SCPs são políticas JSON que especificam as permissões máximas para uma organização ou unidade organizacional (UO) no AWS Organizations. O AWS Organizations é um serviço para agrupar e gerenciar centralmente várias Contas da AWS pertencentes à sua empresa. Se você habilitar todos os recursos em uma organização, poderá aplicar políticas de controle de serviço (SCPs) a qualquer uma ou a todas as contas. O SCP limita as permissões para entidades em contas-membro, incluindo cadaConta da AWSusuário raiz. Para obter mais informações sobre o Organizations e SCPs, consulte Como os SCPs funcionam no Manual do usuário do AWS Organizations.
- Políticas de sessão: são políticas avançadas que você transmite como um parâmetro quando cria de forma programática uma sessão temporária para uma função ou um usuário federado. As permissões da sessão resultante são a interseção das políticas baseadas em identidade do usuário ou da função e das políticas de sessão. As permissões também podem ser provenientes de uma política baseada em recurso. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações, consulte Políticas de sessão no Manual do usuário do IAM.

Vários tipos de política

Quando vários tipos de política são aplicáveis a uma solicitação, é mais complicado compreender as permissões resultantes. Para saber como a AWS determina se deve permitir uma solicitação quando há vários tipos de política envolvidos, consulte Lógica da avaliação de políticas no Manual do usuário do IAM.

Como o Amazon Textract funciona com o IAM

Antes de usar o IAM para gerenciar o acesso ao Amazon Textract, você deve entender quais recursos do IAM estão disponíveis para uso com o Amazon Textract. Para obter uma visão detalhada

de como o Amazon Textract e outrosAWSOs serviços funcionam com o IAM, consulteAWSServiços compatíveis com o IAMnoManual do usuário do IAM.

Tópicos

- Políticas baseadas em identidade do Amazon Textract
- Políticas baseadas em recursos do Amazon Textract
- Autorização baseada em tags do Amazon Textract
- Funções do IAM do Amazon Textract

Políticas baseadas em identidade do Amazon Textract

Com as políticas baseadas em identidade do IAM, é possível especificar ações ou recursos permitidos ou negados, além das condições sob as quais as ações são permitidas ou negadas. O Amazon Textract oferece suporte a ações, chaves de condição e recursos específicos. Para conhecer todos os elementos usados em uma política JSON, consulte Referência de elementos de política JSON do IAM no Manual do usuário do IAM.

Ações

Os administradores podem usar as políticas JSON da AWS para especificar quem tem acesso a quê. Ou seja, qual principal pode executar ações em quais recursos, e em que condições.

O elemento `Action` de uma política JSON descreve as ações que você pode usar para permitir ou negar acesso em uma política. As ações de política geralmente têm o mesmo nome que a operação de API da AWS associada. Existem algumas exceções, como ações somente de permissão, que não têm uma operação de API correspondente. Há também algumas operações que exigem várias ações em uma política. Essas ações adicionais são chamadas de ações dependentes.

Inclua ações em uma política para conceder permissões para executar a operação associada.

Ações assíncronas no Amazon Textract exigem duas permissões de ação a serem dadas, uma para ações Iniciar e outra para ações Get. Além disso, se você estiver usando um bucket do Amazon S3 para passar documentos, precisará conceder acesso de leitura à sua conta.

No Amazon Textract, todas as ações de política começam com: `textract:`. Por exemplo, para conceder a alguém permissão para executar uma operação do Amazon Textract com o Amazon Textract`AnalyzeDocument`operação, você inclui `otextract:AnalyzeDocumentação` em sua política. As instruções de política devem incluir um elemento `Action` ou `NotAction`. Amazon

Textract define seu próprio conjunto de ações que descrevem as tarefas que você pode executar com esse serviço.

Para especificar várias ações em uma única declaração, separe-as com vírgulas, conforme o seguinte.

```
"Action": [
      "textract:action1",
      "textract:action2"
```

Você também pode especificar várias ações usando caracteres curinga (*). Por exemplo, para especificar todas as ações que começam com a palavra `Describe`, inclua a ação a seguir.

```
"Action": "textract:Describe*"
```

Para obter uma lista das ações do Amazon Textract, consulteAções definidas pelo Amazon Textractno Manual do usuário do IAM.

Recursos

Os administradores podem usar as políticas JSON da AWS para especificar quem tem acesso a quê. Ou seja, qual principal pode executar ações em quais recursos, e em que condições.

O elemento `Resource` de política JSON especifica o objeto ou os objetos aos quais a ação se aplica. As instruções devem incluir um elemento `Resource` ou um elemento `NotResource`. Como prática recomendada, especifique um recurso usando seu Nome de recurso da Amazon (ARN). Isso pode ser feito para ações que oferecem suporte a um tipo de recurso específico, conhecido como permissões em nível de recurso.

Para ações que não oferecem suporte a permissões em nível de recurso, como operações de listagem, use um curinga (*) para indicar que a instrução se aplica a todos os recursos.

```
"Resource": "*"
```

O Amazon Textract não oferece suporte à especificação de ARNs de recursos em uma política.

Chaves de condição

Os administradores podem usar as políticas JSON da AWS para especificar quem tem acesso a quê. Ou seja, qual principal pode executar ações em quais recursos, e em que condições.

O elemento `Condition` (ou bloco de `Condition`) permite que você especifique condições nas quais uma instrução está em vigor. O elemento `Condition` é opcional. É possível criar expressões condicionais que usam agentes de condição, como “igual a” ou “menor que”, para fazer a condição da política corresponder aos valores na solicitação.

Se você especificar vários elementos `Condition` em uma instrução ou várias chaves em um único elemento `Condition`, a AWS os avaliará usando uma operação lógica `AND`. Se você especificar vários valores para uma única chave de condição, a AWS avaliará a condição usando uma operação lógica `OR`. Todas as condições devem ser atendidas para que as permissões da instrução sejam concedidas.

Você também pode usar variáveis de espaço reservado ao especificar as condições. Por exemplo, é possível conceder a um usuário do IAM permissão para acessar um recurso somente se ele estiver marcado com seu nome de usuário do IAM. Para obter mais informações, consulte Elementos de política do IAM: variáveis e tags no Manual do usuário do IAM.

A AWS oferece suporte a chaves de condição globais e chaves de condição específicas do serviço. Para ver todas as chaves de condição globais da AWS, consulte Chaves de contexto de condição globais da AWS no Manual do usuário do IAM.

Amazon Textract não fornece nenhuma chave de condição específica ao serviço, mas oferece suporte ao uso de algumas chaves de condição globais. Para obter uma lista de todosAWSChaves de condição globais, consulteAWSChaves de contexto de condição da globaisnoManual do usuário do IAM.

Exemplos

Para visualizar exemplos de políticas baseadas em identidade do Amazon Textract, consulteExemplos de políticas baseadas em identidade do Amazon Textract.

Políticas baseadas em recursos do Amazon Textract

O Amazon Textract não oferece suporte a políticas baseadas em recursos.

Autorização baseada em tags do Amazon Textract

Amazon Textract não oferece suporte à marcação de recursos nem ao controle de acesso com base em tags.

Funções do IAM do Amazon Textract

Uma função do IAM é uma entidade dentro da sua conta da AWS que tem permissões específicas.

Usar credenciais temporárias com o Amazon Textract

É possível usar credenciais temporárias para fazer login com federação, assumir uma função do IAM ou assumir uma função entre contas. As credenciais de segurança temporárias são obtidas chamando operações da API do AWS STS, como AssumeRole ou GetFederationToken.

O Amazon Textract oferece suporte ao uso de credenciais temporárias.

Funções vinculadas ao serviço

Funções vinculadas ao serviço permitem que os serviços da AWS acessem recursos em outros serviços para concluir uma ação em seu nome. As funções vinculadas ao serviço aparecem em sua conta do IAM e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode editar as permissões para funções vinculadas ao serviço.

O Amazon Textract não oferece suporte a funções vinculadas ao serviço.

> **Note**
>
> Como o Amazon Textract não oferece suporte a funções vinculadas ao serviço, ele não oferece suporte a entidades principais de serviços da AWS. Para obter mais informações sobre os principais de serviços do, consulteEntividades de serviço da AWSnoManual do usuário do IAM

Funções de serviço

Esse recurso permite que um serviço assuma uma função de serviço em seu nome. A função permite que o serviço acesse recursos em outros serviços para concluir uma ação em seu nome. As funções de serviço aparecem em sua conta do IAM e são de propriedade da conta. Isso significa que um administrador do IAM pode alterar as permissões para essa função. Porém, fazer isso pode alterar a funcionalidade do serviço.

O Amazon Textract oferece suporte às funções de serviço.

Exemplos de políticas baseadas em identidade do Amazon Textract

Por padrão, os usuários e as funções do IAM não têm permissão para criar ou modificar recursos Amazon Textract. Eles também não podem executar tarefas usando o AWS Management Console, a AWS CLI ou uma API da AWS. Um administrador do IAM deve criar políticas do IAM que concedam aos usuários e funções permissão para executarem operações de API específicas nos recursos especificados de que precisam. O administrador deve anexar essas políticas aos usuários ou grupos do IAM que exigem essas permissões.

Para saber como criar uma política baseada em identidade do IAM usando esses exemplos de documentos de política JSON, consulte Criar políticas na guia JSON no Manual do usuário do IAM.

Tópicos

- Melhores práticas de políticas
- Permitir que os usuários visualizem suas próprias permissões
- Dando acesso a operações síncronas no Amazon Textract
- Dando acesso a operações assíncronas no Amazon Textract

Melhores práticas de políticas

As políticas baseadas em identidade são muito eficientes. Elas determinam se alguém pode criar, acessar ou excluir recursos Amazon Textract em sua conta. Essas ações podem incorrer em custos para a Conta da AWS. Ao criar ou editar políticas baseadas em identidade, siga estas diretrizes e recomendações:

- Comece a usar oAWSpolíticas gerenciadas— Para começar a usar o Amazon Textract rapidamente, useAWSpolíticas gerenciadas pela para conceder a seus funcionários as permissões de que precisam. Essas políticas já estão disponíveis em sua conta e são mantidas e atualizadas pela AWS. Para obter mais informações, consulte Começar a usar permissões com políticas gerenciadas da AWS no Manual do usuário do IAM.
- Conceder privilégio mínimo: ao criar políticas personalizadas, conceda apenas as permissões necessárias para executar uma tarefa. Comece com um conjunto mínimo de permissões e conceda permissões adicionais conforme necessário. Fazer isso é mais seguro do que começar com permissões que são muito lenientes e tentar restringi-las superiormente. Para obter mais informações, consulte Conceder privilégio mínimo no Manual do usuário do IAM.
- Habilitar MFA para operações confidenciais: para aumentar a segurança, exija que os usuários do IAM usem Multi-Factor Authentication (MFA) para acessar recursos ou operações de API

confidenciais. Para obter mais informações, consulte Usar autenticação multifator (MFA) na AWS no Manual do Usuário do IAM.

- Usar condições de política para segurança adicional: na medida do possível, defina as condições sob as quais suas políticas baseadas em identidade permitem o acesso a um recurso. Por exemplo, você pode gravar condições para especificar um intervalo de endereços IP permitidos do qual a solicitação deve partir. Você também pode escrever condições para permitir somente solicitações em uma data especificada ou período ou para exigir o uso de SSL ou MFA. Para obter mais informações, consulteElementos de política JSON do IAM: CondiçãonoManual do usuário do IAM.

Permitir que os usuários visualizem suas próprias permissões

Este exemplo mostra como você pode criar uma política que permite que os usuários do IAM visualizem as políticas gerenciadas e em linha anexadas a sua identidade de usuário. Essa política inclui permissões para concluir essa ação no console ou de forma programática usando a AWS CLI ou a API da AWS.

```
{
    "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": "*"
        }
    ]
}
```

Dando acesso a operações síncronas no Amazon Textract

Este exemplo de política concede acesso às ações síncronas no Amazon Textract a um usuário do IAM em seuAWSconta.

```
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "textract:DetectDocumentText",
                "textract:AnalyzeDocument"
            ],
            "Resource": "*"
        }
    ]
```

Dando acesso a operações assíncronas no Amazon Textract

O exemplo de política a seguir fornece a um usuário do IAM em seuAWSacesso à conta a todas as operações assíncronas usadas no Amazon Textract.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "textract:StartDocumentTextDetection",
                "textract:StartDocumentAnalysis",
                "textract:GetDocumentTextDetection",
```

```
                "textract:GetDocumentAnalysis"
            ],
            "Resource": "*"
        }
    ]
}
```

Solução de Amazon Textract e acesso

Use as informações a seguir para ajudar a diagnosticar e corrigir problemas comuns que você possa encontrar ao trabalhar com a Amazon Textract e o IAM.

Tópicos

- Não tenho autorização para executar uma ação no Amazon Textract
- Não estou autorizado a executar iam:PassRole
- Quero visualizar minhas chaves de acesso
- Sou administrador e desejo conceder acesso ao Amazon Textract
- Quero permitir que as pessoas fora da minhaAWSConta para acessar meus recursos Amazon Textract

Não tenho autorização para executar uma ação no Amazon Textract

Se o AWS Management Console informar que você não está autorizado a executar uma ação, você deverá entrar em contato com o administrador para obter assistência. O administrador é a pessoa que forneceu a você o seu nome de usuário e senha.

O exemplo a seguir ocorre quando o`mateojackson`O usuário do IAM tenta executar`DetectDocumentText`em uma imagem de teste, mas não tem`textract:DetectDocumentText`permissões.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 textract:DetectDocumentText on resource: textimage.png
```

Neste caso, Mateo pede ao administrador para atualizar suas políticas para permitir a ele o acesso ao recurso `textimage.`png usando a ação `textract:DetectDocumentText`.

Não estou autorizado a executar iam:PassRole

Se você receber uma mensagem de erro informando que você não está autorizado a executar a ação `iam:PassRole`, entre em contato com o administrador para obter assistência. O administrador é a pessoa que forneceu a você o seu nome de usuário e senha. Peça a essa pessoa para atualizar suas políticas para permitir que você transmita uma função ao Amazon Textract.

Alguns serviços da AWS permitem que você passe uma função existente para o serviço, em vez de criar uma nova função de serviço ou função vinculada ao serviço. Para fazer isso, um usuário deve ter permissões para passar a função para o serviço.

O erro de exemplo a seguir ocorre quando uma usuária do IAM chamada`marymajor`Tenta usar o console para executar uma ação no Amazon Textract. No entanto, a ação exige que o serviço tenha permissões concedidas por uma função de serviço. Mary não tem permissões para passar a função para o serviço.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Neste caso, Mary pede ao administrador para atualizar suas políticas para permitir que ela execute a ação `iam:PassRole`.

Quero visualizar minhas chaves de acesso

Depois de criar suas chaves de acesso de usuário do IAM, é possível visualizar seu ID da chave de acesso a qualquer momento. No entanto, você não pode visualizar sua chave de acesso secreta novamente. Se você perder sua chave secreta, crie um novo par de chaves de acesso.

As chaves de acesso consistem em duas partes: um ID de chave de acesso (por exemplo, `AKIAIOSFODNN7EXAMPLE`) e uma chave de acesso secreta (por exemplo, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Como um nome de usuário e uma senha, você deve usar o ID da chave de acesso e a chave de acesso secreta em conjunto para autenticar suas solicitações. Gerencie suas chaves de acesso de forma tão segura quanto você gerencia seu nome de usuário e sua senha.

> **Important**
>
> Não forneça as chaves de acesso a terceiros, mesmo que seja para ajudar a encontrar o ID de usuário canônico. Ao fazer isso, você pode dar a alguém acesso permanente à sua conta.

Ao criar um par de chaves de acesso, você é solicitado a guardar o ID da chave de acesso e a chave de acesso secreta em um local seguro. A chave de acesso secreta só está disponível no momento em que é criada. Se você perder sua chave de acesso secreta, será necessário adicionar novas chaves de acesso para seu usuário do IAM. Você pode ter no máximo duas chaves de acesso. Se você já tiver duas, você deverá excluir um par de chaves para poder criar um novo. Para visualizar as instruções, consulte Gerenciar chaves de acesso no Manual do usuário do IAM.

Sou administrador e desejo conceder acesso ao Amazon Textract

Para permitir que outros usuários acessem a Amazon Textract, é necessário criar uma entidade do IAM (usuário ou função) para a pessoa ou a aplicação que precisa do acesso. Elas usarão as credenciais dessa entidade para acessar a AWS. Você deve anexar uma política à entidade que concede a eles as permissões corretas no Amazon Textract.

Para começar a usar imediatamente, consulte Criar os primeiros usuário e grupo delegados do IAM no Manual do usuário do IAM.

Quero permitir que as pessoas fora da minhaAWSConta para acessar meus recursos Amazon Textract

Você pode criar uma função que os usuários de outras contas ou pessoas fora da sua organização podem usar para acessar seus recursos. Você pode especificar quem é confiável para assumir a função. Para serviços que oferecem suporte a políticas baseadas em recursos ou listas de controle de acesso (ACLs), você pode usar essas políticas para conceder às pessoas acesso aos seus recursos.

Para saber mais, consulte o seguinte:

- Para saber se o Amazon Textract oferece suporte a esses recursos, consulteComo o Amazon Textract funciona com o IAM.
- Para saber como conceder acesso a seus recursos em todas as Contas da AWS pertencentes a você, consulte Fornecimento de acesso a um usuário do IAM em outra Conta da AWS pertencente a você no Guia de usuário do IAM.
- Para saber como conceder acesso a seus recursos para Contas da AWS de terceiros, consulte Fornecimento de acesso a Contas da AWS pertencentes a terceiros no Manual do usuário do IAM.
- Para saber como conceder acesso por meio da federação de identidades, consulte Conceder acesso a usuários autenticados externamente (federação de identidades) no Manual do usuário do IAM.

- Para saber a diferença entre usar funções e políticas baseadas em recursos para acesso entre contas, consulte Como as funções do IAM diferem de políticas baseadas em recursos no Manual do usuário do IAM.

Registro em log e monitoramento

Para monitorar o Amazon Textract, use o Amazon CloudWatch. Esta seção fornece informações sobre como configurar o monitoramento do Amazon Textract. Ele também fornece conteúdo de referência para métricas do Amazon Textract.

Tópicos

- Monitorar o Amazon Textract
- Métricas do CloudWatch para o Amazon Textract

Monitorar o Amazon Textract

Com o CloudWatch, você pode obter métricas para operações individuais do Amazon Textract ou métricas globais do Amazon Textract para a conta. É possível usar métricas para rastrear a integridade da sua solução baseada em Amazon Textact e configurar alarmes para notificação quando uma ou mais métricas estiverem fora de um limite definido. Por exemplo, você pode ver métricas do número de erros de servidor que ocorreram. Você também pode ver métricas para o número de vezes em que uma operação Amazon Textract específica foi bem-sucedida. Para ver métricas, você pode usarAmazon CloudWatch, oAWS CLI, ou oAPI do CloudWatch.

Como usar métricas do CloudWatch para o Amazon Textract

Para usar métricas, você deve especificar as seguintes informações:

- A dimensão da métrica ou nenhuma dimensão. Uma dimensão é um par nome/valor, que ajuda a identificar com exclusividade uma métrica. O Amazon Textract tem uma dimensão chamadaOperação. Ele fornece métricas para uma operação específica. Se você não especificar uma dimensão, a métrica terá escopo em todas as operações Amazon Textract dentro da conta.
- O nome da métrica, como `UserErrorCount`.

Você pode obter dados de monitoramento para o Amazon Textract usando oAWS Management Console, oAWS CLIou a API do CloudWatch. Além disso, é possível usar a API do CloudWatch por

meio de um dos Kits de desenvolvimento de software (SDKs) Amazon AWS ou das ferramentas da API do CloudWatch. O console exibe uma série de gráficos com base nos dados brutos da API do CloudWatch. Dependendo das necessidades, você pode preferir usar os gráficos exibidos no console ou recuperados da API.

A lista a seguir mostra alguns usos comuns para as métricas. Essas são sugestões para você começar, e não uma lista abrangente.

Como eu faço para...	Métricas relevantes
Como sei se meu aplicativo atingiu o número máximo de solicitações por segundo?	Monitore a estatística `Sum` da métrica `ThrottledCount` .
Como posso monitorar os erros de solicitação?	Use a estatística `Sum` da métrica `UserErrorCount` .
Como posso encontrar o número total de solicitações?	Use a estatística `SampleCount` da métrica `ResponseTime` . Isso inclui qualquer solicitação que resulte em um erro. Se você quiser ver apenas as chamadas de operação bem-sucedidas, use a métrica `SuccessfulRequestCount` .
Como posso monitorar a latência das chamadas de operação do Amazon Textract?	Use a métrica `ResponseTime` .

Você deve ter as permissões do CloudWatch do apropriadas para monitorar o Amazon Textract com o CloudWatch. Para obter mais informações, consulte Autenticação e controle de acesso para o Amazon CloudWatch.

Acesse métricas do Amazon Textract

Os exemplos a seguir mostram como acessar métricas do Amazon Textract usando o console do CloudWatch, oAWS CLIe a API do CloudWatch.

Para exibir métricas (console)

1. Abra o console do CloudWatch em https://console.aws.amazon.com/cloudwatch/.

2. SelecioneMétricas, escolha oTodas as métricas doguia e, em seguida, selecioneAmazon Textract.
3. SelecionePor operaçãoe, em seguida, selecione uma métrica.

   Por exemplo, escolha oStartDocumentAnalysismétrica para medir quantas vezes a análise de documentos assíncrona foi iniciada.
4. Escolha um valor para o intervalo de datas. A contagem de métricas exibidas no gráfico.

Para visualizar métricas para obter sucesso**`StartDocumentAnalysis`**Chamadas de operação feitas durante um período (CLI)

- Abra a AWS CLI e digite o comando a seguir:

```
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
```

  Este exemplo mostra as chamadas de operação `StartDocumentAnalysis` bem-sucedidas feitas ao longo de um período. Para obter mais informações, consulte get-metric-statistics.

Para acessar métricas (API do CloudWatch)

- Chame `GetMetricStatistics`. Para obter mais informações, consulte o .Referência da API do Amazon CloudWatch.

Criar um alarme

Você pode criar um alarme do CloudWatch que envia uma mensagem do Amazon Simple Notification Service (Amazon SNS) quando o alarme muda de estado. Um alarme observa uma única métrica por um período tempo que você especifica. Ele executa uma ou mais ações com base no valor da métrica em relação a um limite especificado ao longo de vários períodos. A ação é uma notificação enviada a um tópico do Amazon SNS ou a uma política de Auto Scaling.

Os alertas invocam ações apenas para alterações de estado mantidas. Os alarmes do CloudWatch não invocam ações somente porque estão em um determinado estado. O estado deve ter sido alterado e mantido por um período especificado.

Para definir um alarme (console)

1. Faça login no AWS Management Console e abra o console do CloudWatch em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, selecioneAlarmes, e escolhaCriar alarme. Isso abre oCriar assistente de alarme do.
3. Escolha Select metric (Selecionar métrica).
4. NoTodas as métricasguia, selecioneTextract.
5. SelecionePor operaçãoe, em seguida, selecione uma métrica.

   Por exemplo, escolhaStartDocumentAnalysisPara definir um alarme para um número máximo de operações de análise de documentos assíncronas.
6. Escolha a guia Graphed metrics (Métricas em gráfico).
7. Em Statistic (Estatística), selecione Sum (Soma).
8. Escolha Select metric (Selecionar métrica).
9. Preencha o Name e a Description. Em Whenever, escolha >= e digite um valor máximo de sua escolha.
10. Se você quiser que o CloudWatch envie um e-mail quando o estado do alarme for atingido, paraSempre que este alarme:, escolhaO estado é ALARME. Para enviar alarmes para um tópico do Amazon SNS existente, paraEnviar notificação para:, escolha um tópico do SNS existente. Para definir o nome e os endereços de e-mail para uma nova lista de assinaturas de e-mail, escolhaNova lista. O CloudWatch salva a lista e a exibe no campo, de maneira que você possa usá-la a fim de definir alarmes futuros.

> **Note**
>
> Se você usarNova listaPara criar um novo tópico do Amazon SNS, os endereços de e-mail devem ser verificados para os destinatários desejados receberem notificações. O Amazon SNS envia e-mails somente quando o alarme entra em um estado de alarme. Se essa alteração no status de alarme ocorrer antes que os endereços de e-mail sejam verificados, os destinatários desejados não receberão notificação.

11. Escolha Create Alarm.

Para definir um alarme (AWS CLI)

- Abra a AWS CLI e digite o comando a seguir. Altere o valor do`alarm-actions`Para fazer referência a um tópico do Amazon SNS criado anteriormente por você.

```
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
```

  Este exemplo mostra como criar um alarme para quando mais de 10 erros de usuário ocorrem em 5 minutos para as chamadas para`StartDocumentAnalysis`. Para obter mais informações, consulte put-metric-alarm.

Para definir um alerta (API do CloudWatch)

- Chame `PutMetricAlarm`. Para obter mais informações, consulteReferência da API do Amazon CloudWatch.

Métricas do CloudWatch para o Amazon Textract

Esta seção contém informações sobre as métricas do Amazon CloudWatch e oOperaçãodimensão que estão disponíveis para o Amazon Textract.

Você também pode ver uma exibição agregada das métricas do Amazon Textract no console do Amazon Textract.

Métricas do CloudWatch para o Amazon Textract

A tabela a seguir resume as métricas do Amazon Textract.

Métrica	Descrição
SuccessfulRequestCount	O número de solicitações bem-sucedidas. O intervalo de códigos de resposta para uma solicitação bem-sucedida vai de 200 até 299. Unidade: contagem Estatística válida: `Sum,Average`
ThrottledCount	O número de solicitações limitadas. O Amazon Textract limita uma solicitação ao receber mais solicitações do que o limite de transações por segundo definido para a conta. Se o limite definido para a conta for frequentemente excedido, você poderá solicitar um aumento no limite. Para solicitar um aumento, consulte Limites de serviço da AWS. Unidade: contagem Estatística válida: `Sum,Average`
ResponseTime	O tempo em milissegundos para o Amazon Textract computar a resposta. Unidades: 1. Contagem de estatísticas `Data Samples` 2. Milissegundos para estatísticas `Average` Estatística válida: `Data Samples,Average` **Note** O`ResponseTime` a métrica não está incluída no painel de métricas Amazon Textract.
ServerErrorCount	O número de erros do servidor. O intervalo de códigos de resposta para um erro de servidor vai de 500 até 599.

Métrica	Descrição
	Unidade: contagem Estatística válida: `Sum`,`Average`
UserErrorCount	O número de erros de usuário (parâmetros inválidos, imagem inválida, sem permissão e assim por diante). O intervalo de códigos de resposta para um erro de usuário vai de 400 até 499. Unidade: contagem Estatística válida: `Sum`,`Average`

Dimensão do CloudWatch para Amazon Textract

Para recuperar métricas específicas da operação, use o namespace `AWS/Textract` e forneça uma dimensão de operação. Para obter mais informações sobre dimensões, consulteDimensõesnoGuia do usuário do Amazon CloudWatch.

Registro de chamadas de API do Amazon Textract comAWS CloudTrail

O Amazon Textract está integrado comAWS CloudTrail, um serviço que fornece um registro das ações executadas por um usuário, uma função ou umAWSserviço no Amazon Textract. O CloudTrail captura todas as chamadas de API para o Amazon Textract como eventos. As chamadas capturadas incluem as chamadas do console do Amazon Textract e as chamadas de código para as operações da API do Amazon Textract.

Se você criar uma trilha, poderá habilitar a entrega contínua de eventos do CloudTrail para um bucket do Amazon S3, incluindo eventos para o Amazon Textract. Se você não configurar uma trilha, ainda poderá visualizar os eventos mais recentes no console do CloudTrail em Event history (Histórico de eventos). Usando as informações coletadas pelo CloudTrail, é possível determinar a solicitação feita para o Amazon Textract, o endereço IP no qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita, além de detalhes adicionais.

Para saber mais sobre o CloudTrail, consulte o Guia do usuário do AWS CloudTrail.

Informações do Amazon Textract no CloudTrail

O CloudTrail é habilitado em sua conta da AWS quando ela é criada. Quando ocorre atividade no Amazon Textract, essa atividade é registrada em um evento do CloudTrail junto com outrosAWSeventos de serviço emHistórico do evento. Você pode visualizar, pesquisar e baixar os eventos recentes em sua conta da AWS. Para obter mais informações, consulte Como visualizar eventos com o histórico de eventos do CloudTrail.

Para obter um registro contínuo de eventos noAWSConta, incluindo eventos do Amazon Textract, crie uma trilha. Uma trilha permite que o CloudTrail entregue arquivos de log a um bucket do Amazon S3. Por padrão, quando você cria uma trilha no console, ela é aplicada a todas as regiões da AWS. A trilha registra em log eventos de todas as regiões na partição da AWS e entrega os arquivos de log para o bucket do Amazon S3 especificado por você. Além disso, você pode configurar outrosAWSServiços para analisar e atuar mais profundamente sobre os dados de eventos coletados nos logs do CloudTrail. Para obter mais informações, consulte:

- Visão geral da criação de uma trilha
- CloudTrail Serviços compatíveis e integrações do
- Configuração de notificações do Amazon SNS para o CloudTrail
- Receber arquivos de log do CloudTrail de várias regiões e receber arquivos de log do CloudTrail de várias contas

Todas as operações do Amazon Textract são registradas pelo CloudTrail e documentadas naReferência de API do. Por exemplo, as chamadas para as APIs `DetectDocumentText`, `AnalyzeDocument` e `GetDocumentText` geram entradas nos arquivos de log do CloudTrail.

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar:

- Se a solicitação foi feita com credenciais de usuário raiz ou do AWS Identity and Access Management (IAM).
- Se a solicitação foi feita com credenciais de segurança temporárias de uma função ou de um usuário federado.
- Se a solicitação foi feita por outro serviço da AWS.

Para obter mais informações, consulte o Elemento userIdentity do CloudTrail.

Parâmetros de solicitação e campos de resposta que não estão registrados

Para fins de privacidade, determinados parâmetros de solicitação e campos de resposta não são registrados — por exemplo, bytes de imagem de solicitação ou informações da caixa delimitadora de resposta. Os nomes de bucket e os nomes de arquivos do Amazon S3 fornecidos nos parâmetros de solicitação são fornecidos nas entradas de log do CloudTrail. Nenhuma informação sobre bytes de imagem transmitida em uma solicitação é fornecida em um log do CloudTrail. A tabela a seguir mostra os parâmetros de entrada e os parâmetros de resposta que não são registrados para cada operação Amazon Textract.

Operação	Parâmetros de solicitação	Campos de resposta
AnalyzeDocument	`Bytes`	Tudo
DetectDocumentText	`Bytes`	Tudo
StartDocumentAnalysis	Nenhum	Nenhum
GetDocumentAnalysis	Nenhum	Tudo
StartDocumentTextDetection	Nenhum	Nenhum
GetDocumentTextDetection	Nenhum	Tudo

Noções básicas das entradas dos arquivos de log do Amazon Textract T

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log a um bucket do Amazon S3 especificado. Os arquivos de log do CloudTrail contêm uma ou mais entradas de log. Um evento representa uma única solicitação de qualquer origem e inclui informações sobre a operação solicitada, a data e a hora da operação, os parâmetros de solicitação etc. Os arquivos de log do CloudTrail não são um rastreamento de pilha ordenada de chamadas de API pública. Dessa forma, eles não são exibidos em uma ordem específica.

O exemplo a seguir mostra uma entrada de log do CloudTrail que demonstra a operação `AnalyzeDocument`. Os bytes de imagem para a entrada`document`Os resultados da análise (`responseElements`) não estão registrados.

```
{
```

```
    "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"
}
```

O exemplo a seguir mostra uma entrada de log do CloudTrail para o`StartDocumentAnalysis`operação. A entrada de log inclui o nome do bucket do Amazon S3 e o nome do arquivo de imagem no`documentLocation`. O log também inclui a resposta da operação.

```
{
    "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"
        }

    ]
}
```


Validação de conformidade para o Amazon Textract

Auditores de terceiros avaliam a segurança e a conformidade do Amazon Textract como parte de váriosAWSProgramas de conformidade. Isso inclui HIPAA, SOC, ISO e PCI.

> **Note**
>
> Se você estiver processando dados por meio do serviço Textract que está sujeito à conformidade com o PCI DSS, você deverá cancelar sua conta entrando em contato com o AWS Support e seguindo o processo fornecido a você.

Para obter uma lista de serviços da AWS no escopo de programas de conformidade específicos, consulte Serviços da AWS no escopo pelo programa de conformidade. Para obter informações gerais, consulte Programas de conformidade da AWS.

Você pode fazer download de relatórios de auditoria de terceiros usando o AWS Artifact. Para obter mais informações, consulte Fazer download de relatórios no AWS Artifact.

Sua responsabilidade de conformidade ao usar a Amazon Textract é determinada pela confidencialidade dos dados, pelos objetivos de conformidade da sua empresa e pelos regulamentos e leis aplicáveis.AWSFornece os seguintes recursos para ajudar com a conformidade:

- Guias de início rápido de segurança e compatibilidade: estes guias de implantação abordam as considerações de arquitetura e fornecem etapas para implantação de ambientes de linha de base focados em compatibilidade e segurança na AWS.
- Whitepaper Architecting for HIPAA Security and Compliance: este whitepaper descreve como as empresas podem usar a AWS para criar aplicações em conformidade com a HIPAA.
- Recursos de conformidade da AWS: essa coleção de manuais e guias pode ser aplicada a seu setor e local.
- Avaliar recursos com regras no AWS Config Developer Guide (Guia do desenvolvedor do CCI): o serviço AWS Config avalia como as configurações de recursos estão em conformidade com práticas internas, diretrizes do setor e regulamentos.
- AWS Security Hub— IssoAWSO serviço da fornece uma visão abrangente do seu estado de segurança noAWS. O hub de segurança ajuda você a verificar a sua conformidade com os padrões e as melhores práticas do setor de segurança.

Resiliência no Amazon Textract

A infraestrutura global da AWS é criada com base em regiões da AWS e zonas de disponibilidade. As regiões da AWS fornecem várias zonas de disponibilidade separadas e isoladas fisicamente, que são conectadas com baixa latência, altas taxas de transferência e redes altamente redundantes. Com as zonas de disponibilidade, você pode projetar e operar aplicações e bancos de dados que automaticamente executam o failover entre as zonas sem interrupção. As zonas de disponibilidade são mais altamente disponíveis, tolerantes a falhas e escaláveis que uma ou várias infraestruturas de data center tradicionais.

Para obter mais informações sobre regiões e zonas de disponibilidade da AWS, consulte Infraestrutura global da AWS.

> **Note**
>
> A transferência de dados entre regiões não é permitida devido ao Regulamento Geral de Proteção de Dados (GDPR).

Segurança da infraestrutura no Amazon Textract

Como um serviço gerenciado, o Amazon Textract é protegido peloAWSProcedimentos de segurança de rede global descritos noAmazon Web Services: Visão geral dos processos de segurançaWhitepaper.

Você usaAWSChamadas de API publicadas pela para acessar o Amazon Textract pela rede. Os clientes devem oferecer suporte a Transport Layer Security (TLS) 1.0 ou posterior. Recomendamos TLS 1.2 ou posterior. Os clientes também devem ter suporte a conjuntos de criptografia com perfect forward secrecy (PFS) como Ephemeral Diffie-Hellman (DHE) ou Ephemeral Elliptic Curve Diffie-Hellman (ECDHE). A maioria dos sistemas modernos como Java 7 e versões posteriores oferece suporte a esses modos.

Além disso, as solicitações devem ser assinadas usando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou você pode usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

Análise de vulnerabilidade e configuração no Amazon Textract

A configuração e os controles de TI são uma responsabilidade compartilhada entre a AWS e você, nosso cliente. Para obter mais informações, consulte o modelo de responsabilidade compartilhada da AWS.

Amazon Textract e VPC endpoints de interface (AWS PrivateLink)

É possível estabelecer uma conexão privada entre a VPC e o Amazon Textract criando umVPC endpoint de interface. Os valores-limite da interface são desenvolvidos peloAWS PrivateLink, uma tecnologia que permite acessar de forma privada as APIs Amazon Textract sem um gateway da Internet, um dispositivo NAT, uma conexão VPN ou uma conexão do AWS Direct Connect. As instâncias na VPC não precisam de endereços IP públicos para a comunicação com APIs do Amazon Textract. O tráfego de rede entre a VPC e o Amazon Textract não deixa a rede da AWS.

Cada endpoint de interface é representado por uma ou mais interfaces de rede elástica nas sub-redes.

Para obter mais informações, consulte VPC endpoints de interface (AWS PrivateLink) no Guia do usuário da Amazon VPC.

Considerações endpoints da VPC do Amazon Textract

Antes de configurar um VPC endpoint de interface para o Amazon Textract, revisePropriedades e limitações do endpoint de interfacenoManual do usuário da Amazon VPC.

O Amazon Textract oferece suporte a chamadas para todas as ações de API da VPC.

Criar um VPC endpoint de interface para o Amazon Textract

É possível criar um endpoint da VPC para o serviço Amazon Textract usando o console da Amazon VPC ou aAWS Command Line Interface(AWS CLI). Para obter mais informações, consulte Criar um endpoint de interface no Guia do usuário da Amazon VPC.

Crie um VPC endpoint para o Amazon Textract usando o seguinte nome de serviço:

- com.amazonaws.*região*.textract - Para criar um endpoint para a maioria das operações Amazon Textract.
- com.amazonaws.*região*.textract-fips — Para criar um endpoint para o Amazon Textract que esteja em conformidade com a publicação 140-2 do Federal Information Processing Standard (FIPS — Padrão Federal de Processamento de Informações) do padrão do governo dos Estados Unidos.

Se habilitar o DNS privado para o endpoint, poderá fazer solicitações de API para o Amazon Textract usando seu nome DNS padrão para a região, por exemplo,`textract.us-east-1.amazonaws.com`.

Para obter mais informações, consulte Acessar um serviço por um endpoint de interface no Manual do usuário da Amazon VPC.

Criar uma política de endpoint da VPC para o Amazon Textract

É possível anexar uma política de endpoint ao VPC endpoint que controla o acesso ao Amazon Textract. Essa política especifica as seguintes informações:

- A entidade principal que pode executar ações.
- As ações que podem ser executadas.
- Os recursos sobre os quais as ações podem ser realizadas.

Para obter mais informações, consulte Controlar o acesso a serviços com VPC endpoints no Manual do usuário da Amazon VPC.

Exemplo: Política de endpoint da VPC para ações do Amazon Textract

Veja a seguir um exemplo de política de endpoint para o Amazon Textract. Quando anexada a um endpoint, essa política concede acesso às ações indicadas do Amazon Textract para todos os principais em todos os recursos.

Esse exemplo de política permite acesso a somente as operações`DetectDocumentTexte``AnalyzeDocument`. Os usuários ainda podem chamar operações do Amazon Textract de fora do VPC Endpoint.

```
   "Statement":[
      {
         "Principal":"*",
         "Effect":"Allow",
         "Action":[
            "textract:DetectDocumentText",
            "textract:AnalyzeDocument",
         ],
         "Resource":"*"
      }
   ]
}
```

Referência da API

Esta seção fornece a documentação das operações da API do Amazon Textract.

Tópicos

- Ações
- Tipos de dados

Ações

As ações a seguir são compatíveis:

- AnalyzeDocument
- AnalyzeExpense
- AnalyzeID
- DetectDocumentText
- GetDocumentAnalysis
- GetDocumentTextDetection
- GetExpenseAnalysis
- StartDocumentAnalysis
- StartDocumentTextDetection
- StartExpenseAnalysis

AnalyzeDocument

Analisa um documento de entrada para os relacionamentos entre itens detectados em um documento

Os tipos de informações retornadas são os seguintes:

- Dados do formulário (pares de chave-valor). As informações relacionadas são retornadas em doisBlockobjetos, cada um dos tiposKEY_VALUE_SET: uma CHAVEBlockobjeto e um VALORBlockobjeto. Por exemplo,Name (Nome): Ana Silva Carolinacontém uma chave e um valor. Name (Nome):é a chave. Ana Silva CarolinaÉ o valor de.
- Dados de células de tabela e tabela. UMA TABELABlockobjeto contém informações sobre uma tabela detectada. UMA CÉLULABlockobjeto é retornado para cada célula em uma tabela.
- Linhas e palavras de texto. UMA LINHABlockO objeto contém um ou mais WORDBlockobjetos. Todas as linhas e palavras detectadas no documento são retornadas (incluindo texto que não tem relação com o valor deFeatureTypes).

Elementos de seleção, como caixas de seleção e botões de opção (botões de opção), podem ser detectados em dados de formulário e em tabelas. UM SELECTION_ELEMENTBlockobjeto contém informações sobre um elemento de seleção, incluindo o status da seleção.

Você pode escolher qual tipo de análise executar especificando oFeatureTypesLista.

A saída é retornada em uma lista deBlockobjetos.

AnalyzeDocumentÉ uma operação síncrona. Para analisar documentos de forma assíncrona, useStartDocumentAnalysis.

Para obter mais informações, consulteAnálise de texto de documentos.

Sintaxe da solicitação

```
{
   "Document": {
      "Bytes": blob,
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
```

```
    },
    "FeatureTypes": [ "string" ],
    "HumanLoopConfig": {
       "DataAttributes": {
          "ContentClassifiers": [ "string" ]
       },
       "FlowDefinitionArn": "string",
       "HumanLoopName": "string"
    }
}
```

Parâmetros de solicitação

A solicitação aceita os dados a seguir no formato JSON.

Document

O documento de entrada como bytes codificados em base64 ou um objeto do Amazon S3. Se você usar a AWS CLI para chamar as operações do Amazon Textract, não poderá passar bytes de imagem. O documento deve ser uma imagem no formato JPEG, PNG, PDF ou TIFF.

Se você estiver usando um AWS SDK para chamar o Amazon Textract, talvez não seja necessário codificar bytes de imagem que são passados usando o`Bytes`campo.

Tipo: objeto Document

: obrigatório Sim

FeatureTypes

Uma lista dos tipos de análise a serem executadas. Adicione TABLES à lista para retornar informações sobre as tabelas detectadas no documento de entrada. Adicione FORMS para retornar dados de formulário detectados. Para executar os dois tipos de análise, adicione TABLES e FORMS ao`FeatureTypes`. Todas as linhas e palavras detectadas no documento estão incluídas na resposta (incluindo texto que não está relacionado ao valor de`FeatureTypes`).

Type: Matriz de strings

Valores válidos: `TABLES | FORMS`

: obrigatório Sim

HumanLoopConfig

Define a configuração para o humano no fluxo de trabalho de loop para analisar documentos.

Tipo: objeto HumanLoopConfig

: obrigatório Não

Sintaxe da resposta

```
{
   "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"
    }
}
```

Elementos de resposta

Se a ação for bem-sucedida, o serviço reenviará uma resposta HTTP 200.

Os seguintes dados são retornados no formato JSON pelo serviço.

AnalyzeDocumentModelVersion

A versão do modelo usado para analisar o documento.

Type: String

Blocks

Os itens que são detectados e analisados por`AnalyzeDocument`.

Type: Matriz de Block objetos

DocumentMetadata

Metadados sobre o documento analisado. Um exemplo é o número de páginas.

Tipo: objeto DocumentMetadata

HumanLoopActivationOutput

Mostra os resultados do humano na avaliação de loop.

Tipo: objeto HumanLoopActivationOutput

Erros

AccessDeniedException

Você não está autorizado a executar a ação. Use o nome de recurso da Amazon (ARN) de um usuário autorizado ou a função do IAM para executar a operação.

Código de status HTTP: 400

BadDocumentException

O Amazon Textract não consegue ler o documento. Para obter mais informações sobre os limites de documentos no Amazon Textract, consulteLimites rígidos no Amazon Textract.

Código de status HTTP: 400

DocumentTooLargeException

O documento não pode ser processado porque é muito grande. O tamanho máximo do documento para operações síncronas 10 MB. O tamanho máximo do documento para operações assíncronas é de 500 MB para arquivos PDF.

Código de status HTTP: 400

HumanLoopQuotaExceededException

Indica que você excedeu o número máximo de humanos ativos nos fluxos de trabalho de loop disponíveis

Código de status HTTP: 400

InternalServerError

O Amazon Textract teve um problema de serviço. Tente fazer a chamada novamente.

Código de status HTTP: 500

InvalidParameterException

Um parâmetro de entrada violou uma restrição. Por exemplo, em operações síncronas, um`InvalidParameterException`exceção ocorre quando nenhum dos`S3Object`ou`Bytes`valores são fornecidos na`Document`Parâmetro de solicitação. Valide seu parâmetro antes de chamar a operação de API novamente.

Código de status HTTP: 400

InvalidS3ObjectException

O Amazon Textract não pode acessar o objeto do S3 especificado na solicitação. para obter mais informações,Configuração de acesso ao Amazon S3Para obter informações sobre a solução de problemas, consulteSolução de problemas do Amazon S3

Código de status HTTP: 400

ProvisionedThroughputExceededException

O número de solicitações excedeu o limite da taxa de transferência. Se precisar aumentar esse limite, entre em contato com o Amazon Textract.

Código de status HTTP: 400

ThrottlingException

O Amazon Textract está temporariamente indisponível para processar a solicitação. Tente fazer a chamada novamente.

Código de status HTTP: 500

UnsupportedDocumentException

O formato do documento de entrada não é compatível. Os documentos para operações podem estar no formato PNG, JPEG, PDF ou TIFF.

Código de status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for JavaScript
- AWS SDK para PHP V3

- AWS SDK para Python
- AWS SDK for Ruby V3

AnalyzeExpense

`AnalyzeExpense`analisa de forma síncrona um documento de entrada para relacionamentos financeiramente relacionados entre o texto.

As informações são retornadas como`ExpenseDocuments`e separado da seguinte forma.

- `LineItemGroups`- Um conjunto de dados contendo`LineItems`que armazenam informações sobre as linhas de texto, como um item comprado e seu preço em um recibo.
- `SummaryFields`- Contém todas as outras informações de um recibo, como informações de cabeçalho ou o nome dos fornecedores.

Sintaxe da solicitação

```
{
   "Document": {
      "Bytes": blob,
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   }
}
```

Parâmetros de solicitação

A solicitação aceita os dados a seguir no formato JSON.

Document

O documento de entrada, seja como bytes ou como um objeto S3.

Passe bytes de imagem para a operação de uma API Amazon Textract usando a`Bytes`propriedade. Por exemplo, você usaria a`Bytes`propriedade para passar um documento carregado de um sistema de arquivos local. Bytes de imagem passados usando o`Bytes`A propriedade deve ser codificada em base64. Talvez seu código não precise codificar bytes de arquivos de documentos se você estiver usando um AWS SDK para chamar as operações da API do Amazon Textract.

Passe imagens armazenadas em um bucket do S3 para uma operação de API Amazon Textract usando `aS3Objectpropriedade`. Os documentos armazenados em um bucket do S3 não precisam ser codificados em base64.

A Região da AWS do bucket do S3 que contém o objeto S3 deve corresponder à região da AWS que você usa para operações Amazon Textract.

Se você usar a AWS CLI para chamar operações do Amazon Textract, não haverá suporte para a passagem dos bytes da imagem da propriedade. Você deve primeiramente carregar o documento em um bucket do Amazon S3 e, em seguida, chamar a operação usando a propriedade S3Object.

Para que o Amazon Textract processe um objeto do S3, o usuário deve ter permissão para acessar o objeto do S3.

Tipo: objeto Document

: obrigatório Sim

Sintaxe da resposta

```
{
   "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"
                  }
               }
            ]
         }
      ]
}
```

Elementos de resposta

Se a ação for bem-sucedida, o serviço reenviará uma resposta HTTP 200.

Os seguintes dados são retornados no formato JSON pelo serviço.

DocumentMetadata

Informações sobre o documento de entrada.

Tipo: objeto DocumentMetadata

ExpenseDocuments

As despesas detectadas pelo Amazon Textract.

Type: Matriz deExpenseDocumentobjetos

Erros

AccessDeniedException

Você não está autorizado a executar a ação. Use o nome de recurso da Amazon (ARN) de um usuário autorizado ou a função do IAM para executar a operação.

Código de status HTTP: 400

BadDocumentException

O Amazon Textract não consegue ler o documento. Para obter mais informações sobre os limites de documentos no Amazon Textract, consulteLimites rígidos no Amazon Textract.

Código de status HTTP: 400

DocumentTooLargeException

O documento não pode ser processado porque é muito grande. O tamanho máximo do documento para operações síncronas 10 MB. O tamanho máximo do documento para operações assíncronas é de 500 MB para arquivos PDF.

Código de status HTTP: 400

InternalServerError

Amazon Textract teve um problema de serviço. Tente fazer a chamada novamente.

Código de status HTTP: 500

InvalidParameterException

Um parâmetro de entrada violou uma restrição. Por exemplo, em operações síncronas, um`InvalidParameterException`exceção ocorre quando nenhum dos`S3Object`ou`Bytes`valores são fornecidos no`Document`parâmetro de solicitação. Valide seu parâmetro antes de chamar a operação de API novamente.

Código de status HTTP: 400

InvalidS3ObjectException

O Amazon Textract não pode acessar o objeto do S3 especificado na solicitação. para obter mais informações,Configuração de acesso ao Amazon S3Para obter informações sobre a solução de problemas, consulteSolução de problemas do Amazon S3

Código de status HTTP: 400

ProvisionedThroughputExceededException

O número de solicitações excedeu o limite da taxa de transferência. Se precisar aumentar esse limite, entre em contato com o Amazon Textract.

Código de status HTTP: 400

ThrottlingException

Amazon Textract está temporariamente indisponível para processar a solicitação. Tente fazer a chamada novamente.

Código de status HTTP: 500

UnsupportedDocumentException

O formato do documento de entrada não é oferecido o suporte. Os documentos para operações podem estar no formato PNG, JPEG, PDF ou TIFF.

Código de status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for JavaScript
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK for Ruby V3

AnalyzeID

Analisa documentos de identidade para obter informações relevantes. Essas informações são extraídas e retornadas como`IdentityDocumentFields`, que registra o campo normalizado e o valor do texto extraído.Ao contrário de outras operações Amazon Textract,`AnalyzeID`não retorna nenhum dado de geometria.

Sintaxe da solicitação

```
{
   "DocumentPages": [
      {
         "Bytes": blob,
         "S3Object": {
            "Bucket": "string",
            "Name": "string",
            "Version": "string"
         }
      }
   ]
}
```

Parâmetros de solicitação

A solicitação aceita os dados a seguir no formato JSON.

DocumentPages

O documento que está sendo passado para o AnalyzeID.

Type: Matriz deDocumentobjetos

Membros da matriz: Número mínimo de 1 item. Número máximo de 2 itens.

Obrigatório: Sim

Sintaxe da resposta

```
{
   "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"
                    }
                }
            ]
        }
    ]
}
```

Elementos de resposta

Se a ação for bem-sucedida, o serviço reenviará uma resposta HTTP 200.

Os seguintes dados são retornados no formato JSON pelo serviço.

AnalyzeIDModelVersion

A versão da API AnalyzeIdentity que está sendo usada para processar documentos.

Type: String

DocumentMetadata

Informações sobre o documento de entrada.

Tipo: objeto DocumentMetadata

IdentityDocuments

A lista de documentos processados pelo AnalyzeID. Inclui um número que denota seu lugar na lista e a estrutura de resposta do documento.

Type: Matriz deIdentityDocumentobjetos

Erros

AccessDeniedException

Você não está autorizado a executar a ação. Use o nome de recurso da Amazon (ARN) de um usuário autorizado ou a função do IAM para executar a operação.

Código de status HTTP: 400

BadDocumentException

O Amazon Textract não consegue ler o documento. Para obter mais informações sobre os limites de documentos no Amazon Textract, consulteLimites rígidos no Amazon Textract.

Código de status HTTP: 400

DocumentTooLargeException

O documento não pode ser processado porque é muito grande. O tamanho máximo do documento para operações síncronas 10 MB. O tamanho máximo do documento para operações assíncronas é de 500 MB para arquivos PDF.

Código de status HTTP: 400

InternalServerError

O Amazon Textract teve um problema de serviço. Tente fazer a chamada novamente.

Código de status HTTP: 500

InvalidParameterException

Um parâmetro de entrada violou uma restrição. Por exemplo, em operações síncronas, um`InvalidParameterException`exceção ocorre quando nenhum dos`S3Object`ou`Bytes`valores são fornecidos no`Document`Parâmetro de solicitação. Valide seu parâmetro antes de chamar a operação de API novamente.

Código de status HTTP: 400

InvalidS3ObjectException

O Amazon Textract está indisponível para acessar o objeto do S3 especificado na solicitação. para obter mais informações,Configuração de acesso ao Amazon S3Para obter informações sobre a solução de problemas, consulteSolução de problemas do Amazon S3

Código de status HTTP: 400

ProvisionedThroughputExceededException

O número de solicitações excedeu o limite da taxa de transferência. Se precisar aumentar esse limite, entre em contato com o Amazon Textract.

Código de status HTTP: 400

ThrottlingException

O Amazon Textract está temporariamente indisponível para processar a solicitação. Tente fazer a chamada novamente.

Código de status HTTP: 500

UnsupportedDocumentException

O formato do documento de entrada não é compatível. Os documentos para operações podem estar no formato PNG, JPEG, PDF ou TIFF.

Código de status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for JavaScript

- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK for Ruby V3

DetectDocumentText

Detecta texto no documento de entrada. O Amazon Textract pode detectar linhas de texto e as palavras que compõem uma linha de texto. O documento de entrada deve ser uma imagem no formato JPEG, PNG, PDF ou TIFF.`DetectDocumentText`retorna o texto detectado em uma matriz de`Block`objetos.

Cada página do documento tem como associado`Block`do tipo PAGE. Cada PÁGINA`Block`object é o pai de LINE`Block`objetos que representam as linhas do texto detectado em uma página. UMA LINHA`Block`object é um pai para cada palavra que compõe a linha. As palavras são representadas por`Block`objetos do tipo WORD.

`DetectDocumentText`É uma operação síncrona. Para analisar documentos de forma assíncrona, use`StartDocumentTextDetection`.

Para obter mais informações, consulte`Detecção de texto de documentos`.

Sintaxe da solicitação

```
{
   "Document": {
      "Bytes": blob,
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   }
}
```

Parâmetros de solicitação

A solicitação aceita os dados a seguir no formato JSON.

Document

O documento de entrada como bytes codificados em base64 ou um objeto do Amazon S3. Se você usar a AWS CLI para chamar as operações do Amazon Textract, não poderá passar bytes de imagem. O documento deve ser uma imagem em formato JPEG ou PNG.

Se você estiver usando um AWS SDK para chamar o Amazon Textract, talvez não seja necessário codificar bytes de imagem que são passados usando o`Bytes`campo.

Tipo: objeto Document

Obrigatório: Sim

Sintaxe da resposta

```
{
   "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
    }
}
```

Elementos de resposta

Se a ação for bem-sucedida, o serviço reenviará uma resposta HTTP 200.

Os seguintes dados são retornados no formato JSON pelo serviço.

Blocks

Uma matriz de`Block`objetos que contêm o texto detectado no documento.

Type: Matriz deBlockobjetos

DetectDocumentTextModelVersion

Type: String

DocumentMetadata

Metadados sobre o documento. Ele contém o número de páginas detectadas no documento.

Tipo: objeto DocumentMetadata

Erros

AccessDeniedException

Você não está autorizado a executar a ação. Use o nome de recurso da Amazon (ARN) de um usuário autorizado ou a função do IAM para executar a operação.

Código de status HTTP: 400

BadDocumentException

O Amazon Textract não consegue ler o documento. Para obter mais informações sobre os limites de documentos no Amazon Textract, consulteLimites rígidos no Amazon Textract.

Código de status HTTP: 400

DocumentTooLargeException

O documento não pode ser processado porque é muito grande. O tamanho máximo do documento para operações síncronas 10 MB. O tamanho máximo do documento para operações assíncronas é de 500 MB para arquivos PDF.

Código de status HTTP: 400

InternalServerError

O Amazon Textract teve um problema de serviço. Tente fazer a chamada novamente.

Código de status HTTP: 500

InvalidParameterException

Um parâmetro de entrada violou uma restrição. Por exemplo, em operações síncronas, um`InvalidParameterException`exceção ocorre quando nenhum dos`S3ObjectouBytes`valores são fornecidos na`Document`Parâmetro de solicitação. Valide seu parâmetro antes de chamar a operação de API novamente.

Código de status HTTP: 400

InvalidS3ObjectException

O Amazon Textract está indisponível para acessar o objeto do S3 especificado na solicitação. para obter mais informações,Configuração de acesso ao Amazon S3Para obter informações sobre a solução de problemas, consulteSolução de problemas do Amazon S3

Código de status HTTP: 400

ProvisionedThroughputExceededException

O número de solicitações excedeu o limite da taxa de transferência. Se precisar aumentar esse limite, entre em contato com o Amazon Textract.

Código de status HTTP: 400

ThrottlingException

O Amazon Textract está temporariamente indisponível para processar a solicitação. Tente fazer a chamada novamente.

Código de status HTTP: 500

UnsupportedDocumentException

O formato do documento de entrada não tem suporte. Os documentos para operações podem estar no formato PNG, JPEG, PDF ou TIFF.

Código de status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for JavaScript
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK for Ruby V3

GetDocumentAnalysis

Obtém os resultados de uma operação assíncrona do Amazon Textract que analisa o texto em um documento.

Você inicia a análise de texto assíncrona chamandoStartDocumentAnalysis, que retorna um identificador de trabalho (`JobId`). Quando a operação de análise de texto termina, o Amazon Textract publica um status de conclusão no tópico do Amazon Simple Notification Service (Amazon SNS) registrado na chamada inicial para`StartDocumentAnalysis`. Para obter os resultados da operação de detecção de texto, primeiro verifique se o valor de status publicado no tópico do Amazon SNS é`SUCCEEDED`. Em caso afirmativo, ligue`GetDocumentAnalysis`e passar o identificador de trabalho (`JobId`) da chamada inicial para`StartDocumentAnalysis`.

`GetDocumentAnalysis`retorna uma matriz deBlockobjetos. Os seguintes tipos de informações são retornados:

- Dados de formulário (pares de chave-valor). As informações relacionadas são retornadas em doisBlockobjetos, cada um dos tipos`KEY_VALUE_SET`: uma CHAVE`Block`objeto e um VALOR`Block`objeto. Por exemplo,Name (Nome): Ana Silva Carolinacontém uma chave e um valor. Name (Nome):é a chave. Ana Silva CarolinaÉ o valor.
- Dados de células de tabela e tabela. UMA TABELA`Block`objeto contém informações sobre uma tabela detectada. UMA CÉLULA`Block`objeto é retornado para cada célula em uma tabela.
- Linhas e palavras de texto. UMA LINHA`Block`O objeto contém um ou mais WORD`Block`objetos. Todas as linhas e palavras detectadas no documento são retornadas (incluindo texto que não tem relação com o valor do`StartDocumentAnalysis` `FeatureTypes`parâmetro de entrada).

Elementos de seleção, como caixas de seleção e botões de opção (botões de opção), podem ser detectados em dados de formulário e em tabelas. UM SELECTION_ELEMENT`Block`objeto contém informações sobre um elemento de seleção, incluindo o status da seleção.

Usar a`MaxResults`Parâmetro para limitar o número de blocos retornados. Se houver mais resultados do que o especificado em`MaxResults`, o valor de`NextToken`Na resposta da operação contém um token de paginação para obter o próximo conjunto de resultados. Para obter a próxima página de resultados, ligue para`GetDocumentAnalysis`e preencha o`NextToken`parâmetro request com o valor do token retornado da chamada anterior para`GetDocumentAnalysis`.

Para obter mais informações, consulteAnálise de texto do documento.

Sintaxe da solicitação

```
{
   "JobId": "string",
   "MaxResults": number,
   "NextToken": "string"
}
```

Parâmetros de solicitação

A solicitação aceita os dados a seguir no formato JSON.

JobId

Um identificador exclusivo para o trabalho de detecção de texto. O`JobId`é retornado de`StartDocumentAnalysis`. UMA`JobId`O valor só é válido por 7 dias.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 64.

Padrão: `^[a-zA-Z0-9-_]+$`

Obrigatório Sim

MaxResults

O número máximo de resultados a serem retornados por chamada paginada. O maior valor que você pode especificar é 1.000. Se precisar um valor maior que 1.000, um máximo de 1.000 resultados será retornado. O valor padrão é 1,000.

Type: Inteiro

Intervalo válido: Valor mínimo de 1.

Obrigatório Não

NextToken

Se a resposta anterior estiver incompleta (porque há mais blocos para recuperar), o Amazon Textract retornará um token de paginação na resposta. Você pode usar esse token de paginação para recuperar o próximo conjunto de blocos.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 255.

Padrão: `.*\S.*`

Obrigatório Não

Sintaxe da resposta

```
{
   "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 ]
         }
      ]
   }
```

Elementos de resposta

Se a ação for bem-sucedida, o serviço reenviará uma resposta HTTP 200.

Os seguintes dados são retornados no formato JSON pelo serviço.

AnalyzeDocumentModelVersion

Type: String

Blocks

Os resultados da operação de análise de texto.

Type: Matriz deBlockobjetos

DocumentMetadata

Informações sobre um documento que o Amazon Textract processou.`DocumentMetadata`é retornado em todas as páginas de respostas paginadas de uma operação de vídeo do Amazon Textract.

Tipo: objeto DocumentMetadata

JobStatus

O status atual do trabalho de detecção de texto.

Type: String

Valores válidos: `IN_PROGRESS | SUCCEEDED | FAILED | PARTIAL_SUCCESS`

NextToken

Se a resposta for truncada, o Amazon Textract retornará esse token. Você pode usar esse token na solicitação subsequente para recuperar o próximo conjunto de resultados de detecção de texto.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 255.

Padrão: `.*\S.*`

StatusMessage

Retorna se o trabalho de detecção não puder ser concluído. Contém explicação para qual erro ocorreu.

Type: String

Warnings

Uma lista de avisos ocorridos durante a operação de análise de documentos.

Type: Matriz deWarningobjetos

Erros

AccessDeniedException

Você não está autorizado a executar a ação. Use o nome de recurso da Amazon (ARN) de um usuário autorizado ou a função do IAM para executar a operação.

Código de status HTTP: 400

InternalServerError

O Amazon Textract teve um problema de serviço. Tente fazer a chamada novamente.

Código de status HTTP: 500

InvalidJobIdException

Um identificador de trabalho inválido foi passado paraGetDocumentAnalysisou paraGetDocumentAnalysis.

Código de status HTTP: 400

InvalidKMSKeyException

Indica que você não tem permissões de descriptografia com a chave KMS inserida ou a chave KMS foi inserida incorretamente.

Código de status HTTP: 400

InvalidParameterException

Um parâmetro de entrada violou uma restrição. Por exemplo, em operações síncronas, um`InvalidParameterException`exceção ocorre quando nenhum dos`S3ObjectouBytes`valores são fornecidos na`Document`Parâmetro de solicitação. Valide seu parâmetro antes de chamar a operação de API novamente.

Código de status HTTP: 400

InvalidS3ObjectException

O Amazon Textract não pode acessar o objeto do S3 especificado na solicitação. para obter mais informações,Configurar o acesso ao Amazon S3Para obter informações sobre a solução de problemas, consulteSolução de problemas do Amazon S3

Código de status HTTP: 400

ProvisionedThroughputExceededException

O número de solicitações excedeu o limite da taxa de transferência. Se precisar aumentar esse limite, entre em contato com o Amazon Textract.

Código de status HTTP: 400

ThrottlingException

O Amazon Textract está temporariamente indisponível para processar a solicitação. Tente fazer a chamada novamente.

Código de status HTTP: 500

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for JavaScript
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK for Ruby V3

GetDocumentTextDetection

Obtém os resultados de uma operação assíncrona do Amazon Textract que detecta texto em um documento. O Amazon Textract pode detectar linhas de texto e as palavras que compõem uma linha de texto.

Você inicia a detecção de texto assíncrona chamandoStartDocumentTextDetection, que retorna um identificador de trabalho (`JobId`). Quando a operação de detecção de texto for concluída, o Amazon Textract publica um status de conclusão no tópico do Amazon Simple Notification Service (Amazon SNS) registrado na chamada inicial para`StartDocumentTextDetection`. Para obter os resultados da operação de detecção de texto, primeiro verifique se o valor de status publicado no tópico do Amazon SNS é`SUCCEEDED`. Em caso afirmativo, ligue`GetDocumentTextDetection`e passar o identificador de trabalho (`JobId`) da chamada inicial para`StartDocumentTextDetection`.

`GetDocumentTextDetection`retorna uma matriz deBlockobjetos.

Cada página do documento tem como associado`Block`do tipo PAGE. Cada PÁGINA`Block`object é o pai de LINE`Block`objetos que representam as linhas do texto detectado em uma página. UMA LINHA`Block`object é um pai para cada palavra que compõe a linha. As palavras são representadas por`Block`objetos do tipo WORD.

Use o parâmetro MaxResults para limitar o número de blocos retornados. Se houver mais resultados do que o especificado em`MaxResults`, o valor de`NextToken`A resposta da operação contém um token de paginação para obter o próximo conjunto de resultados. Para obter a próxima página de resultados, ligue`GetDocumentTextDetection`e preencha o`NextToken`parâmetro request com o valor do token retornado da chamada anterior para`GetDocumentTextDetection`.

Para obter mais informações, consulteDetecção de texto de documentos.

Sintaxe da solicitação

```
{
   "JobId": "string",
   "MaxResults": number,
   "NextToken": "string"
}
```

Parâmetros de solicitação

A solicitação aceita os dados a seguir no formato JSON.

JobId

Um identificador exclusivo para a tarefa de detecção de texto. O`JobId`é retornado de`StartDocumentTextDetection`. UMA`JobId`O valor só é válido por 7 dias.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 64.

Padrão: `^[a-zA-Z0-9-_]+$`

Obrigatório: Sim

MaxResults

O número máximo de resultados a serem retornados por chamada paginada. O maior valor que você pode especificar é 1.000. Se você especificar um valor maior que 1.000, um máximo de 1.000 resultados será retornado. O valor padrão é 1,000.

Type: Inteiro

Intervalo válido: Valor mínimo de 1.

Obrigatório: Não

NextToken

Se a resposta anterior estiver incompleta (porque há mais blocos para recuperar), o Amazon Textract retornará um token de paginação na resposta. Você pode usar esse token de paginação para recuperação do próximo conjunto de blocos.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 255.

Padrão: `.*\S.*`

Obrigatório: Não

Sintaxe da resposta

```
{
   "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 ]
```

```
        }
    ]
}
```

Elementos de resposta

Se a ação for bem-sucedida, o serviço reenviará uma resposta HTTP 200.

Os seguintes dados são retornados no formato JSON pelo serviço.

Blocks

Os resultados da operação de detecção de texto.

Type: Matriz deBlockobjetos

DetectDocumentTextModelVersion

Type: String

DocumentMetadata

Informações sobre um documento que o Amazon Textract processou.`DocumentMetadata`é retornado em todas as páginas de respostas paginadas de uma operação de vídeo do Amazon Textract.

Tipo: objeto DocumentMetadata

JobStatus

O status atual da tarefa de detecção de texto.

Type: String

Valores válidos: `IN_PROGRESS` | `SUCCEEDED` | `FAILED` | `PARTIAL_SUCCESS`

NextToken

Se a resposta for truncada, o Amazon Textract retornará esse token. Você pode usar esse token na solicitação subsequente para recuperação do próximo conjunto de resultados de detecção de texto.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 255.

Padrão: `.*\S.*`

StatusMessage

O retorna se o trabalho de detecção não puder ser concluído. Contém explicação para qual erro ocorreu.

Type: String

Warnings

Uma lista de avisos ocorridos durante a operação de detecção de texto do documento.

Type: Matriz deWarningobjetos

Erros

AccessDeniedException

Você não está autorizado a executar a ação. Use o nome de recurso da Amazon (ARN) de um usuário autorizado ou a função do IAM para executar a operação.

Código de status HTTP: 400

InternalServerError

O Amazon Textract teve um problema de serviço. Tente fazer a chamada novamente.

Código de status HTTP: 500

InvalidJobIdException

Um identificador de trabalho inválido foi passado paraGetDocumentAnalysisou paraGetDocumentAnalysis.

Código de status HTTP: 400

InvalidKMSKeyException

Indica que você não tem permissões de descriptografia com a chave KMS inserida ou a chave KMS foi inserida incorretamente.

Código de status HTTP: 400

InvalidParameterException

Um parâmetro de entrada violou uma restrição. Por exemplo, em operações síncronas, um`InvalidParameterException`exceção ocorre quando nenhum dos`S3Object`ou`Bytes`valores são fornecidos no`Document`Parâmetro de solicitação. Valide seu parâmetro antes de chamar a operação de API novamente.

Código de status HTTP: 400

InvalidS3ObjectException

O Amazon Textract não pode acessar o objeto do S3 especificado na solicitação. para obter mais informações,Configuração de acesso ao Amazon S3Para obter informações sobre a solução de problemas, consulteSolução de problemas do Amazon S3

Código de status HTTP: 400

ProvisionedThroughputExceededException

O número de solicitações excedeu o limite da taxa de transferência. Se quiser aumentar esse limite, entre em contato com o Amazon Textract.

Código de status HTTP: 400

ThrottlingException

O Amazon Textract está temporariamente indisponível para processar a solicitação. Tente fazer a chamada novamente.

Código de status HTTP: 500

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2

- AWS SDK for JavaScript
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK for Ruby V3

GetExpenseAnalysis

Obtém os resultados de uma operação assíncrona do Amazon Textract que analisa faturas e recibos. Amazon Textract encontra informações de contato, itens comprados e nome do fornecedor, a partir de faturas e recibos de entrada.

Você inicia a análise assíncrona de fatura/recebimento ligandoStartExpenseAnalysis, que retorna um identificador de trabalho (`JobId`). Após a conclusão da análise de fatura/recebimento, o Amazon Textract publica o status de conclusão no tópico do Amazon Simple Notification Service (Amazon SNS). Este tópico deve ser registrado na chamada inicial para`StartExpenseAnalysis`. Para obter os resultados da operação de análise de fatura/recebimento, primeiro verifique se o valor de status publicado no tópico do Amazon SNS é`SUCCEEDED`. Em caso afirmativo, ligue para`GetExpenseAnalysis`e passar o identificador de trabalho (`JobId`) da chamada inicial para`StartExpenseAnalysis`.

Use o parâmetro MaxResults para limitar o número de blocos que são retornados. Se houver mais resultados do que o especificado em`MaxResults`, o valor de`NextToken`Na resposta da operação contém um token de paginação para obter o próximo conjunto de resultados. Para obter a próxima página de resultados, ligue para`GetExpenseAnalysis`e preencher as UOs`NextToken`parâmetro request com o valor do token retornado da chamada anterior para`GetExpenseAnalysis`.

Para obter mais informações, consulteAnalisando faturas e recibos.

Sintaxe da solicitação

```
{
   "JobId": "string",
   "MaxResults": number,
   "NextToken": "string"
}
```

Parâmetros de solicitação

A solicitação aceita os dados a seguir no formato JSON.

JobId

Um identificador exclusivo para o trabalho de detecção de texto. O`JobId`é retornado do`StartExpenseAnalysis`. UMA`JobId`O valor é válido apenas por 7 dias.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 64.

Padrão: `^[a-zA-Z0-9-_]+$`

: obrigatório Sim

MaxResults

O número máximo de resultados a serem retornados por chamada paginada. O maior valor que você pode especificar é 20. Se você especificar um valor maior que 20, um máximo de 20 resultados será retornado. O valor padrão é 20.

Type: Inteiro

Intervalo válido: Valor mínimo de 1.

: obrigatório Não

NextToken

Se a resposta anterior estiver incompleta (porque há mais blocos para recuperar), o Amazon Textract retornará um token de paginação na resposta. Você pode usar esse token de paginação para recuperar o próximo conjunto de blocos.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 255.

Padrão: `.*\S.*`

: obrigatório Não

Sintaxe da resposta

```
{
   "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 ]
       }
    ]
}
```

Elementos de resposta

Se a ação for bem-sucedida, o serviço reenviará uma resposta HTTP 200.

Os seguintes dados são retornados no formato JSON pelo serviço.

AnalyzeExpenseModelVersion

A versão atual do modelo do AnalyzeExpense.

Type: String

DocumentMetadata

Informações sobre um documento que o Amazon Textract processou.`DocumentMetadata`é retornado em todas as páginas de respostas paginadas de uma operação Amazon Textract.

Tipo: objeto DocumentMetadata

ExpenseDocuments

As despesas detectadas pelo Amazon Textract.

Type: Matriz deExpenseDocumentobjetos

JobStatus

O status atual do trabalho de detecção de texto.

Type: String

Valores válidos: `IN_PROGRESS | SUCCEEDED | FAILED | PARTIAL_SUCCESS`

NextToken

Se a resposta for truncada, o Amazon Textract retornará esse token. Você pode usar esse token na solicitação subsequente para recuperar o próximo conjunto de resultados de detecção de texto.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 255.

Padrão: `.*\S.*`

StatusMessage

Retorna se o trabalho de detecção não puder ser concluído. Contém explicação para qual erro ocorreu.

Type: String

Warnings

Uma lista de avisos ocorridos durante a operação de detecção de texto do documento.

Type: Matriz deWarningobjetos

Erros

AccessDeniedException

Você não está autorizado a executar a ação. Use o nome de recurso da Amazon (ARN) de um usuário autorizado ou a função do IAM para executar a operação.

Código de status HTTP: 400

InternalServerError

O Amazon Textract teve um problema de serviço. Tente fazer a chamada novamente.

Código de status HTTP: 500

InvalidJobIdException

Um identificador de trabalho inválido foi passado paraGetDocumentAnalysisou paraGetDocumentAnalysis.

Código de status HTTP: 400

InvalidKMSKeyException

Indica que você não tem permissões de descriptografia com a chave KMS inserida ou a chave KMS foi inserida incorretamente.

Código de status HTTP: 400

InvalidParameterException

Um parâmetro de entrada violou uma restrição. Por exemplo, em operações síncronas, um`InvalidParameterException`exceção ocorre quando nenhum dos`S3Object`ou`Bytes`valores são fornecidos no`Document`Parâmetro de solicitação. Valide seu parâmetro antes de chamar a operação de API novamente.

Código de status HTTP: 400

InvalidS3ObjectException

O Amazon Textract não pode acessar o objeto do S3 especificado na solicitação. para obter mais informações,Configuração de acesso ao Amazon S3Para obter informações sobre a solução de problemas, consulteSolução de problemas do Amazon S3

Código de status HTTP: 400

ProvisionedThroughputExceededException

O número de solicitações excedeu o limite da taxa de transferência. Se quiser aumentar esse limite, entre em contato com o Amazon Textract.

Código de status HTTP: 400

ThrottlingException

O Amazon Textract está temporariamente indisponível para processar a solicitação. Tente fazer a chamada novamente.

Código de status HTTP: 500

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for JavaScript
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK for Ruby V3

StartDocumentAnalysis

Inicia a análise assíncrona de um documento de entrada para pesquisar relacionamentos entre itens detectados, como pares de chave-valor, tabelas e elementos de seleção.

`StartDocumentAnalysis`pode analisar texto em documentos que estão nos formatos JPEG, PNG, TIFF e PDF. Os documentos são armazenados em um bucket do Amazon S3. Usar o[DocumentLocation](#)Para especificar o nome do bucket do e o nome do arquivo do documento.

`StartDocumentAnalysis`retorna um identificador de trabalho (`JobId`) que você usa para obter os resultados da operação. Quando a análise de texto for concluída, o Amazon Textract publica um status de conclusão no tópico do Amazon Simple Notification Service (Amazon SNS) especificado no`NotificationChannel`. Para obter os resultados da operação de análise de texto, primeiro verifique se o valor de status publicado no tópico do Amazon SNS é`SUCCEEDED`. Em caso afirmativo, ligue[GetDocumentAnalysis](#)e passar o identificador de trabalho (`JobId`) da chamada inicial para`StartDocumentAnalysis`.

Para obter mais informações, consulte[Análise de texto de documentos](#).

Sintaxe da solicitação

```
{
   "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"
   }
```

```
}
```

Parâmetros de solicitação

A solicitação aceita os dados a seguir no formato JSON.

ClientRequestToken

O token idempotente que você usa para identificar a solicitação inicial. Se você usar o mesmo token com vários`StartDocumentAnalysis`solicitações, o mesmo`JobId`O retornado. Usar o`ClientRequestToken`para evitar que o mesmo trabalho seja iniciado acidentalmente mais de uma vez. Para obter mais informações, consulte Chamando operações assíncronas do Amazon Textract.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 64.

Padrão: `^[a-zA-Z0-9-_]+$`

Obrigatório: Não

DocumentLocation

A localização do documento a ser processado.

Tipo: objeto DocumentLocation

Obrigatório: Sim

FeatureTypes

Uma lista dos tipos de análise a serem executadas. Adicione TABLES à lista para retornar informações sobre as tabelas detectadas no documento de entrada. Adicione FORMS para retornar dados de formulário detectados. Para executar os dois tipos de análise, adicione TABLES e FORMS ao`FeatureTypes`. Todas as linhas e palavras detectadas no documento estão incluídas na resposta (incluindo texto que não está relacionado ao valor de`FeatureTypes`).

Type: Matriz de strings

Valores válidos: `TABLES | FORMS`

Obrigatório: Sim

JobTag

Um identificador que você especifica que está incluído na notificação de conclusão publicada no tópico do Amazon SNS. Por exemplo, você pode usar`JobTag`para identificar o tipo de documento ao qual a notificação de conclusão corresponde (como um formulário fiscal ou um recibo).

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 64.

Padrão: `[a-zA-Z0-9_.\-:]+`

Obrigatório: Não

KMSKeyId

A chave do KMS usada para criptografar os resultados de inferência. Isso pode estar no formato Key ID ou Key Alias. Quando uma chave KMS é fornecida, a chave KMS será usada para criptografia do lado do servidor dos objetos no bucket do cliente. Quando esse parâmetro não estiver habilitado, o resultado será criptografado no lado do servidor, usando o SSE-S3.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 2048.

Padrão: `^[A-Za-z0-9][A-Za-z0-9:_/+=,@.-]{0,2048}$`

Obrigatório: Não

NotificationChannel

O tópico ARN do Amazon SNS no qual você deseja que o Amazon Textract publique o status de conclusão da operação.

Tipo: objeto NotificationChannel

Obrigatório: Não

OutputConfig

Define se a saída irá para um intervalo definido pelo cliente. Por padrão, o Amazon Textract salvará os resultados internamente para serem acessados pela operação GetDocumentAnalysis.

Tipo: objeto OutputConfig

Obrigatório: Não

Sintaxe da resposta

```
{
   "JobId": "string"
}
```

Elementos de resposta

Se a ação for bem-sucedida, o serviço reenviará uma resposta HTTP 200.

Os seguintes dados são retornados no formato JSON pelo serviço.

JobId

O identificador do trabalho de detecção de texto do documento. Usar `oJobId`para identificar o trabalho em uma chamada subsequente para`GetDocumentAnalysis`. UMA`JobId`O valor só é válido por 7 dias.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 64.

Padrão: `^[a-zA-Z0-9-_]+$`

Erros

AccessDeniedException

Você não está autorizado a executar a ação. Use o nome de recurso da Amazon (ARN) de um usuário autorizado ou a função do IAM para executar a operação.

Código de status HTTP: 400

BadDocumentException

O Amazon Textract não consegue ler o documento. Para obter mais informações sobre os limites de documentos no Amazon Textract, consulteLimites rígidos no Amazon Textract.

Código de status HTTP: 400

DocumentTooLargeException

O documento não pode ser processado porque é muito grande. O tamanho máximo do documento para operações síncronas 10 MB. O tamanho máximo do documento para operações assíncronas é de 500 MB para arquivos PDF.

Código de status HTTP: 400

IdempotentParameterMismatchException

UMA`ClientRequestToken`O parâmetro de entrada foi reutilizado com uma operação, mas pelo menos um dos outros parâmetros de entrada é diferente da chamada anterior para a operação.

Código de status HTTP: 400

InternalServerError

O Amazon Textract teve um problema de serviço. Tente fazer a chamada novamente.

Código de status HTTP: 500

InvalidKMSKeyException

Indica que você não tem permissões de descriptografia com a chave KMS inserida ou a chave KMS foi inserida incorretamente.

Código de status HTTP: 400

InvalidParameterException

Um parâmetro de entrada violou uma restrição. Por exemplo, em operações síncronas, um`InvalidParameterException`exceção ocorre quando nenhum dos`S3Object`ou`Bytes`valores são fornecidos na`Document`Parâmetro de solicitação. Valide seu parâmetro antes de chamar a operação de API novamente.

Código de status HTTP: 400

InvalidS3ObjectException

O Amazon Textract não pode acessar o objeto do S3 especificado na solicitação. para obter mais informações,Configuração de acesso ao Amazon S3Para obter informações sobre a solução de problemas, consulteSolução de problemas do Amazon S3

Código de status HTTP: 400

LimitExceededException

Um limite do serviço do Amazon Textract foi excedido. Por exemplo, se você iniciar muitos trabalhos assíncronos simultaneamente, chamadas para iniciar operações (`StartDocumentTextDetection`, por exemplo) gera uma exceção LimitExceededException (código de status HTTP: 400) até que o número de trabalhos simultâneos fique abaixo do limite de serviço do Amazon Textract.

Código de status HTTP: 400

ProvisionedThroughputExceededException

O número de solicitações excedeu o limite da taxa de transferência. Se precisar aumentar esse limite, entre em contato com o Amazon Textract.

Código de status HTTP: 400

ThrottlingException

O Amazon Textract está temporariamente indisponível para processar a solicitação. Tente fazer a chamada novamente.

Código de status HTTP: 500

UnsupportedDocumentException

O formato do documento de entrada não é suportado. Os documentos para operações podem estar no formato PNG, JPEG, PDF ou TIFF.

Código de status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2

- AWS SDK for JavaScript
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK for Ruby V3

StartDocumentTextDetection

Inicia a detecção assíncrona de texto em um documento. O Amazon Textract pode detectar linhas de texto e as palavras que compõem uma linha de texto.

`StartDocumentTextDetection`pode analisar texto em documentos que estão nos formatos JPEG, PNG, TIFF e PDF. Os documentos são armazenados em um bucket do Amazon S3. Usar oDocumentLocationPara especificar o nome do bucket do e o nome do arquivo do documento.

`StartTextDetection`retorna um identificador de trabalho (`JobId`) que você usa para obter os resultados da operação. Quando a detecção de texto for concluída, o Amazon Textract publica um status de conclusão no tópico do Amazon Simple Notification Service (Amazon SNS) especificado no`NotificationChannel`. Para obter os resultados da operação de detecção de texto, primeiro verifique se o valor de status publicado no tópico do Amazon SNS é`SUCCEEDED`. Em caso afirmativo, ligueGetDocumentTextDetectione passar o identificador de trabalho (`JobId`) da chamada inicial para`StartDocumentTextDetection`.

Para obter mais informações, consulteDetecção de texto de documentos.

Sintaxe da solicitação

```
{
   "ClientRequestToken": "string",
   "DocumentLocation": {
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   },
   "JobTag": "string",
   "KMSKeyId": "string",
   "NotificationChannel": {
      "RoleArn": "string",
      "SNSTopicArn": "string"
   },
   "OutputConfig": {
      "S3Bucket": "string",
      "S3Prefix": "string"
   }
}
```

Parâmetros de solicitação

A solicitação aceita os dados a seguir no formato JSON.

ClientRequestToken

O token idempotente usado para identificar a solicitação inicial. Se você usar o mesmo token com vários`StartDocumentTextDetection`solicitações, o mesmo`JobId`é retornado. Usar o`ClientRequestToken`para evitar que o mesmo trabalho seja iniciado acidentalmente mais de uma vez. Para obter mais informações, consulteChamando operações assíncronas do Amazon Textract.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 64.

Padrão: `^[a-zA-Z0-9-_]+$`

: obrigatório Não

DocumentLocation

A localização do documento a ser processado.

Tipo: objeto DocumentLocation

: obrigatório Sim

JobTag

Um identificador que você especifica que está incluído na notificação de conclusão publicada no tópico do Amazon SNS. Por exemplo, você pode usar`JobTag`para identificar o tipo de documento ao qual a notificação de conclusão corresponde (como um formulário fiscal ou um recibo).

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 64.

Padrão: `[a-zA-Z0-9_.\-:]+`

: obrigatório Não

KMSKeyId

A chave do KMS usada para criptografar os resultados de inferência. Isso pode estar no formato Key ID ou Key Alias. Quando uma chave KMS é fornecida, a chave KMS será usada para

criptografia do lado do servidor dos objetos no bucket do cliente. Quando esse parâmetro não estiver habilitado, o resultado será criptografado no lado do servidor, usando o SSE-S3.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 2048.

Padrão: `^[A-Za-z0-9][A-Za-z0-9:_/+=,@.-]{0,2048}$`

: obrigatório Não

NotificationChannel

O tópico ARN do Amazon SNS no qual você deseja que o Amazon Textract publique o status de conclusão da operação.

Tipo: objeto NotificationChannel

: obrigatório Não

OutputConfig

Define se a saída irá para um intervalo definido pelo cliente. Por padrão, o Amazon Textract salvará os resultados internamente para serem acessados com a operação GetDocumentTextDetection.

Tipo: objeto OutputConfig

: obrigatório Não

Sintaxe da resposta

```
{
   "JobId": "string"
}
```

Elementos de resposta

Se a ação for bem-sucedida, o serviço reenviará uma resposta HTTP 200.

Os seguintes dados são retornados no formato JSON pelo serviço.

JobId

O identificador da tarefa de detecção de texto para o documento. Usar `oJobId`para identificar o trabalho em uma chamada subsequente para`GetDocumentTextDetection`. UMA`JobId`O valor só é válido por 7 dias.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 64.

Padrão: `^[a-zA-Z0-9-_]+$`

Erros

AccessDeniedException

Você não está autorizado a executar a ação. Use o nome de recurso da Amazon (ARN) de um usuário autorizado ou a função do IAM para executar a operação.

Código de status HTTP: 400

BadDocumentException

O Amazon Textract não consegue ler o documento. Para obter mais informações sobre os limites de documentos no Amazon Textract, consulteLimites rígidos no Amazon Textract.

Código de status HTTP: 400

DocumentTooLargeException

O documento não pode ser processado porque é muito grande. O tamanho máximo do documento para operações síncronas 10 MB. O tamanho máximo do documento para operações assíncronas é de 500 MB para arquivos PDF.

Código de status HTTP: 400

IdempotentParameterMismatchException

UMA`ClientRequestToken`O parâmetro de entrada foi reutilizado com uma operação, mas pelo menos um dos outros parâmetros de entrada é diferente da chamada anterior para a operação.

Código de status HTTP: 400

InternalServerError

O Amazon Textract teve um problema de serviço. Tente fazer a chamada novamente.

Código de status HTTP: 500

InvalidKMSKeyException

Indica que você não tem permissões de descriptografia com a chave KMS inserida ou a chave KMS foi inserida incorretamente.

Código de status HTTP: 400

InvalidParameterException

Um parâmetro de entrada violou uma restrição. Por exemplo, em operações síncronas, um`InvalidParameterException`exceção ocorre quando nenhum dos`S3Object`ou`Bytes`valores são fornecidos na`Document`Parâmetro de solicitação. Valide seu parâmetro antes de chamar a operação de API novamente.

Código de status HTTP: 400

InvalidS3ObjectException

O Amazon Textract não pode acessar o objeto do S3 especificado na solicitação. para obter mais informações,Configuração de acesso ao Amazon S3Para obter informações sobre a solução de problemas, consulteSolução de problemas do Amazon S3

Código de status HTTP: 400

LimitExceededException

Um limite de serviço do Amazon Textract foi excedido. Por exemplo, se você iniciar muitos trabalhos assíncronos simultaneamente, chamadas para iniciar operações (`StartDocumentTextDetection`, por exemplo) gera uma exceção LimitExceededException (código de status HTTP: 400) até que o número de trabalhos simultâneos fique abaixo do limite de serviço Amazon Textract.

Código de status HTTP: 400

ProvisionedThroughputExceededException

O número de solicitações excedeu o limite da taxa de transferência. Se precisar aumentar esse limite, entre em contato com o Amazon Textract.

Código de status HTTP: 400

ThrottlingException

O Amazon Textract está temporariamente indisponível para processar a solicitação. Tente fazer a chamada novamente.

Código de status HTTP: 500

UnsupportedDocumentException

O formato do documento de entrada não é suportado. Os documentos para operações podem estar no formato PNG, JPEG, PDF ou TIFF.

Código de status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for JavaScript
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK for Ruby V3

StartExpenseAnalysis

Inicia a análise assíncrona de faturas ou recibos de dados como informações de contato, itens comprados e nomes de fornecedores.

`StartExpenseAnalysis`pode analisar texto em documentos que estão nos formatos JPEG, PNG e PDF. Os documentos devem ser armazenados em um bucket do Amazon S3. Usar aDocumentLocationUm parâmetro para especificar o nome do bucket do S3 e o nome do documento nesse bucket.

`StartExpenseAnalysis`retorna um identificador de trabalho (`JobId`) que você forneceráGetExpenseAnalysisPara recuperar os resultados da operação. Quando a análise das faturas/recibos de entrada for concluída, o Amazon Textract publica um status de conclusão no tópico do Amazon Simple Notification Service (Amazon SNS) que você fornece para o`NotificationChannel`. Para obter os resultados da operação de análise de recibos e fatura, certifique-se de que o valor de status publicado no tópico do Amazon SNS seja`SUCCEEDED`. Em caso afirmativo, ligueGetExpenseAnalysise passar o identificador de trabalho (`JobId`) que foi devolvido por sua chamada para`StartExpenseAnalysis`.

Para obter mais informações, consulteAnalisando faturas e recibos.

Sintaxe da solicitação

```
{
   "ClientRequestToken": "string",
   "DocumentLocation": {
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   },
   "JobTag": "string",
   "KMSKeyId": "string",
   "NotificationChannel": {
      "RoleArn": "string",
      "SNSTopicArn": "string"
   },
   "OutputConfig": {
      "S3Bucket": "string",
      "S3Prefix": "string"
```

```
    }
}
```

Parâmetros de solicitação

A solicitação aceita os dados a seguir no formato JSON.

ClientRequestToken

O token idempotente usado para identificar a solicitação inicial. Se você usar o mesmo token com vários`StartDocumentTextDetection`solicitações, o mesmo`JobId`é retornado. Usar o`ClientRequestToken`para evitar que o mesmo trabalho seja iniciado acidentalmente mais de uma vez. Para obter mais informações, consulteChamando operações assíncronas do Amazon Textract

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 64.

Padrão: `^[a-zA-Z0-9-_]+$`

: obrigatório Não

DocumentLocation

A localização do documento a ser processado.

Tipo: objeto DocumentLocation

: obrigatório Sim

JobTag

Um identificador que você especifica que está incluído na notificação de conclusão publicada no tópico do Amazon SNS. Por exemplo, você pode usar`JobTag`para identificar o tipo de documento ao qual a notificação de conclusão corresponde (como um formulário fiscal ou um recibo).

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 64.

Padrão: `[a-zA-Z0-9_.\-:]+`

: obrigatório Não

KMSKeyId

A chave do KMS usada para criptografar os resultados da inferência. Isso pode estar no formato Key ID ou Key Alias. Quando uma chave KMS é fornecida, a chave KMS será usada para criptografia do lado do servidor dos objetos no bucket do cliente. Quando esse parâmetro não estiver habilitado, o resultado será criptografado no lado do servidor, usando o SSE-S3.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 2048.

Padrão: `^[A-Za-z0-9][A-Za-z0-9:_/+=,@.-]{0,2048}$`

: obrigatório Não

NotificationChannel

O tópico ARN do Amazon SNS no qual você deseja que o Amazon Textract publique o status de conclusão da operação.

Tipo: objeto NotificationChannel

: obrigatório Não

OutputConfig

Define se a saída irá para um intervalo definido pelo cliente. Por padrão, o Amazon Textract salvará os resultados internamente para serem acessados pelo`GetExpenseAnalysis`operação.

Tipo: objeto OutputConfig

: obrigatório Não

Sintaxe da resposta

```
{
   "JobId": "string"
}
```

Elementos de resposta

Se a ação for bem-sucedida, o serviço reenviará uma resposta HTTP 200.

Os seguintes dados são retornados no formato JSON pelo serviço.

JobId

Um identificador exclusivo para o trabalho de detecção de texto. O`JobId`é retornado de`StartExpenseAnalysis`. UMA`JobId`O valor só é válido por 7 dias.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 64.

Padrão: `^[a-zA-Z0-9-_]+$`

Erros

AccessDeniedException

Você não está autorizado a executar a ação. Use o nome de recurso da Amazon (ARN) de um usuário autorizado ou a função do IAM para executar a operação.

Código de status HTTP: 400

BadDocumentException

O Amazon Textract não consegue ler o documento. Para obter mais informações sobre os limites de documentos no Amazon Textract, consulteLimites rígidos no Amazon Textract.

Código de status HTTP: 400

DocumentTooLargeException

O documento não pode ser processado porque é muito grande. O tamanho máximo do documento para operações síncronas 10 MB. O tamanho máximo do documento para operações assíncronas é de 500 MB para arquivos PDF.

Código de status HTTP: 400

IdempotentParameterMismatchException

UMA`ClientRequestToken`Um parâmetro de entrada foi reutilizado com uma operação, mas pelo menos um dos outros parâmetros de entrada é diferente da chamada anterior para a operação.

Código de status HTTP: 400

InternalServerError

O Amazon Textract teve um problema de serviço. Tente fazer a chamada novamente.

Código de status HTTP: 500

InvalidKMSKeyException

Indica que você não tem permissões de descriptografia com a chave KMS inserida ou a chave KMS foi inserida incorretamente.

Código de status HTTP: 400

InvalidParameterException

Um parâmetro de entrada violou uma restrição. Por exemplo, em operações síncronas, um`InvalidParameterException`exceção ocorre quando nenhum dos`S3Object`ou`Bytes`valores são fornecidos na`Document`Parâmetro de solicitação. Valide seu parâmetro antes de chamar a operação de API novamente.

Código de status HTTP: 400

InvalidS3ObjectException

O Amazon Textract não pode acessar o objeto do S3 especificado na solicitação. para obter mais informações,Configuração de acesso ao Amazon S3Para obter informações sobre a solução de problemas, consulteSolução de problemas do Amazon S3

Código de status HTTP: 400

LimitExceededException

Um limite de serviço do Amazon Textract foi excedido. Por exemplo, se você iniciar muitos trabalhos assíncronos simultaneamente, chamadas para iniciar operações (`StartDocumentTextDetection`, por exemplo) geram uma exceção LimitExceededException (código de status HTTP: 400) até que o número de trabalhos simultâneos fique abaixo do limite de serviço do Amazon Textract.

Código de status HTTP: 400

ProvisionedThroughputExceededException

O número de solicitações excedeu o limite da taxa de transferência. Se precisar aumentar esse limite, entre em contato com o Amazon Textract.

Código de status HTTP: 400

ThrottlingException

O Amazon Textract está temporariamente indisponível para processar a solicitação. Tente fazer a chamada novamente.

Código de status HTTP: 500

UnsupportedDocumentException

O formato do documento de entrada não é suportado. Os documentos para operações podem estar no formato PNG, JPEG, PDF ou TIFF.

Código de status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for JavaScript
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK for Ruby V3

Tipos de dados

Os seguintes tipos de dados são compatíveis:

- 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

Usado para conter as informações detectadas por uma operação AnalyzeID.

Índice

Confidence

A pontuação de confiança do texto detectado.

Type: Float

Faixa válida: Valor mínimo de 0. Valor máximo de 100.

Obrigatório Não

NormalizedValue

Retornado somente para datas, retorna o tipo de valor detectado e a data escrita de uma maneira mais legível por máquina.

Tipo: objeto NormalizedValue

Obrigatório Não

Text

Texto do campo normalizado ou do valor associado a ele.

Type: String

Obrigatório Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

Block

UMA`Block`representa itens que são reconhecidos em um documento dentro de um grupo de pixels próximos um do outro. As informações retornadas em um`Block`O objeto depende do tipo de operação. Na detecção de texto para documentos (por exemploDetectDocumentText), você obtém informações sobre as palavras e linhas de texto detectadas. Em análise de texto (por exemploAnalyzeDocument), você também pode obter informações sobre os campos, tabelas e elementos de seleção detectados no documento.

Uma matriz de`Block`objetos são retornados por operações síncronas e assíncronas. Em operações síncronas, comoDetectDocumentText, a matriz de`Block`objetos é todo o conjunto de resultados. Em operações assíncronas, comoGetDocumentAnalysis, o array é retornado por uma ou mais respostas.

Para obter mais informações, consulteComo funciona o Amazon Textract.

Índice

BlockType

O tipo de item de texto reconhecido. Em operações para detecção de texto, os seguintes tipos são retornados:

- PÁGINA- Contém uma lista da LINHA`Block`objetos detectados em uma página de documento.
- PALAVRA- Uma palavra detectada em uma página de documento. Uma palavra é um ou mais caracteres latinos ISO básicos não separados por espaços.
- LINHA- Uma sequência de palavras contíguas delimitadas por tabulação que são detectadas em uma página de documento.

Em operações de análise de texto, os seguintes tipos são retornados:

- PÁGINA- Contém uma lista de crianças`Block`objetos detectados em uma página de documento.
- KEY_VALUE_SET- Armazena a CHAVE e O VALOR`Block`objetos para texto vinculado detectado em uma página de documento. Usar a`EntityType`campo para determinar se um objeto KEY_VALUE_SET é uma KEY`Block`objeto ou VALUE`Block`objeto.
- PALAVRA- Uma palavra detectada em uma página de documento. Uma palavra é um ou mais caracteres latinos ISO básicos não separados por espaços.
- LINHA- Uma sequência de palavras contíguas delimitadas por tabulação que são detectadas em uma página de documento.

- MESA- Uma tabela detectada em uma página de documento. Uma tabela é uma informação baseada em grade com duas ou mais linhas ou colunas, com uma extensão de célula de uma linha e uma coluna cada.
- CÉLULA- Uma célula dentro de uma tabela detectada. A célula é o pai do bloco que contém o texto na célula.
- SELECTION_ELEMENT- Um elemento de seleção, como um botão de opção (botão de opção) ou uma caixa de seleção detectada em uma página de documento. Use o valor de`SelectionStatus`para determinar o status do elemento de seleção.

Type: String

Valores válidos: `KEY_VALUE_SET | PAGE | LINE | WORD | TABLE | CELL | SELECTION_ELEMENT`

Obrigatório: Não

ColumnIndex

A coluna na qual uma célula de tabela aparece. A primeira posição da coluna é 1.`ColumnIndex`não é retornado por`DetectDocumentTexte`` GetDocumentTextDetection`.

Type: Inteiro

Intervalo válido: Valor mínimo de 0.

Obrigatório: Não

ColumnSpan

O número de colunas que uma célula de tabela abrange. Atualmente, esse valor é sempre 1, mesmo que o número de colunas estendidas seja maior que 1.`ColumnSpan`não é retornado por`DetectDocumentTexte`` GetDocumentTextDetection`.

Type: Inteiro

Intervalo válido: Valor mínimo de 0.

Obrigatório: Não

Confidence

A pontuação de confiança que o Amazon Textract tem na precisão do texto reconhecido e na precisão dos pontos de geometria em torno do texto reconhecido.

Type: Float

Intervalo válido: Valor mínimo de 0. Valor máximo de 100.

Obrigatório: Não

EntityTypes

O tipo de entidade. Pode ser retornado o seguinte:

- CHAVE- Um identificador para um campo no documento.
- VALOR- O texto do campo.

`EntityTypes`não é retornado por`DetectDocumentTexte``GetDocumentTextDetection`.

Type: Matriz de strings

Valores válidos: `KEY` | `VALUE`

Obrigatório: Não

Geometry

A localização do texto reconhecido na imagem. Inclui uma caixa delimitadora grossa alinhada ao eixo que envolve o texto e um polígono de grão mais fino para obter informações espaciais mais precisas.

Tipo: objeto Geometry

Obrigatório: Não

Id

O identificador para o texto reconhecido. O identificador é exclusivo apenas para uma única operação.

Type: String

Padrão: `.*\S.*`

Obrigatório: Não

Page

A página na qual um bloco foi detectado.`Page`é retornado por operações assíncronas. Valores de página maiores que 1 são retornados somente para documentos de várias páginas que estão

no formato PDF ou TIFF. Uma imagem digitalizada (JPEG/PNG), mesmo que contenha várias páginas de documento, é considerada um documento de página única. O valor de`Page`é sempre 1. Operações síncronas não retornam`Page`porque cada documento de entrada é considerado um documento de página única.

Type: Inteiro

Intervalo válido: Valor mínimo de 0.

Obrigatório: Não

Relationships

Uma lista de blocos filhos do bloco atual. Por exemplo, um objeto LINE tem blocos filhos para cada bloco WORD que faz parte da linha de texto. Não há objetos Relacionamento na lista para relacionamentos que não existem, como quando o bloco atual não tem blocos filhos. O tamanho da lista pode ser o seguinte:

- 0 - O bloco não tem blocos filhos.
- 1 - O bloco tem blocos filhos.

Type: Matriz de`Relationship`objetos

Obrigatório: Não

RowIndex

A linha na qual uma célula de tabela está localizada. A posição da primeira linha é 1.`RowIndex`não é retornado por`DetectDocumentTexte``GetDocumentTextDetection`.

Type: Inteiro

Intervalo válido: Valor mínimo de 0.

Obrigatório: Não

RowSpan

O número de linhas que uma célula de tabela abrange. Atualmente, esse valor é sempre 1, mesmo que o número de linhas estendidas seja maior que 1.`RowSpan`não é retornado por`DetectDocumentTexte``GetDocumentTextDetection`.

Type: Inteiro

Intervalo válido: Valor mínimo de 0.

Obrigatório: Não

SelectionStatus

O status de seleção de um elemento de seleção, como um botão de opção ou caixa de seleção.

Type: String

Valores válidos: `SELECTED | NOT_SELECTED`

Obrigatório: Não

Text

A palavra ou linha de texto reconhecida pelo Amazon Textract.

Type: String

Obrigatório: Não

TextType

O tipo de texto que o Amazon Textract detectou. Pode verificar se há texto manuscrito e texto impresso.

Type: String

Valores válidos: `HANDWRITING | PRINTED`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

BoundingBox

A caixa delimitadora ao redor da página detectada, texto, par chave-valor, tabela, célula da tabela ou elemento de seleção em uma página de documento. `Oleft`(x-coordinate) `etop`(coordenada y) são coordenadas que representam os lados superior e esquerdo da caixa delimitadora. Observe que o canto superior esquerdo da imagem é a origem (0,0).

`Otopeleft`valores retornados são proporções do tamanho geral da página do documento. Por exemplo, se a imagem de entrada tiver 700 x 200 pixels e a coordenada superior esquerda da caixa delimitadora tiver 350 x 50 pixels, a API retornará um valor `left` de 0,5 (350/700) e um valor `top` de 0,25 (50/200).

`OwidtheheightOs` valores representam as dimensões da caixa delimitadora como uma proporção da dimensão total da página do documento. Por exemplo, se o tamanho da página do documento for 700 x 200 pixels e a largura da caixa delimitadora for 70 pixels, a largura retornada será 0,1.

Índice

Height

Altura a altura da caixa delimitadora como uma proporção da altura total da página do documento.

Type: Float

Obrigatório: Não

Left

Esquerda a coordenada esquerda da caixa delimitadora como uma proporção da largura total da página do documento.

Type: Float

Obrigatório: Não

Top

Superior a coordenada superior da caixa delimitadora como uma proporção da altura total da página do documento.

Type: Float

Obrigatório: Não

Width

Largura da caixa delimitadora como uma proporção da largura total da página do documento.

Type: Float

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

Document

O documento de entrada, seja como bytes ou como um objeto S3.

Você passa bytes de imagem para a operação de uma API Amazon Textract usando a`Bytes`propriedade. Por exemplo, você usaria o`Bytes`propriedade para passar um documento carregado de um sistema de arquivos local. Bytes de imagem passados usando o`Bytes`A propriedade deve ser codificada em base64. Talvez seu código não precise codificar bytes de arquivos de documentos se você estiver usando um AWS SDK para chamar as operações da API do Amazon Textract.

Você passa imagens armazenadas em um bucket do S3 para a operação de uma API Amazon Textract usando a`S3Object`propriedade. Os documentos armazenados em um bucket do S3 não precisam ser codificados em base64.

A Região da AWS do bucket do S3 que contém o objeto S3 deve corresponder à Região da AWS que você usa para operações Amazon Textract.

Se você usar a AWS CLI para chamar operações Amazon Textract, não haverá suporte para a passagem dos bytes da imagem do usando a propriedade Bytes. Você deve primeiramente carregar o documento em um bucket do Amazon S3 e, em seguida, chamar a operação usando a propriedade S3Objeto.

Para que o Amazon Textract processe um objeto S3, o usuário deve ter permissão para acessar o objeto S3.

Índice

Bytes

Um blob de bytes de documento codificados em base64. O tamanho máximo de um documento fornecido em um blob de bytes é de 5 MB. Os bytes do documento devem estar no formato PNG ou JPEG.

Se você estiver usando um AWS SDK para chamar o Amazon Textract, talvez não seja necessário codificar bytes de imagem com base 64 passados usando o`Bytes`campo.

Type: Objeto de dados binários codificado pelo Bas

Restrições de Tamanho: Tamanho mínimo de 1. Tamanho máximo de 10485760.

: obrigatório Não

S3Object

Identifica um objeto S3 como a fonte do documento. O tamanho máximo de um documento armazenado em um bucket do S3 é de 5 MB.

Tipo: objeto S3Object

: obrigatório Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

DocumentLocation

O bucket do Amazon S3 que contém o documento a ser processado. Ele é usado por operações assíncronas, comoStartDocumentTextDetection.

O documento de entrada pode ser um arquivo de imagem em formato JPEG ou PNG. Também pode ser um arquivo em formato PDF.

Índice

S3Object

O bucket do Amazon S3 que contém o documento de entrada.

Tipo: objeto S3Object

: obrigatório Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

DocumentMetadata

Informações sobre o documento de entrada.

Índice

Pages

O número de páginas detectadas no documento.

Type: Inteiro

Intervalo válido: Valor mínimo de 0.

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

ExpenseDetection

Um objeto usado para armazenar informações sobre o valor ou rótulo detectado pelo Amazon Textract.

Índice

Confidence

A confiança na detecção, como porcentagem

Type: Float

Intervalo válido: Valor mínimo de 0. Valor máximo de 100.

Obrigatório: Não

Geometry

Informações sobre onde os seguintes itens estão localizados em uma página de documento: página detectada, texto, pares de chave-valor, tabelas, células de tabela e elementos de seleção.

Tipo: objeto Geometry

Obrigatório: Não

Text

A palavra ou linha de texto reconhecida pelo Amazon Textract

Type: String

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2

- AWS SDK for Ruby V3

ExpenseDocument

A estrutura que contém todas as informações retornadas pela AnalyzeExpense

Índice

ExpenseIndex

Denota de qual fatura ou recibo no documento as informações estão vindo. O primeiro documento será 1, o segundo 2 e assim por diante.

Type: Inteiro

Faixa válida: Valor mínimo de 0.

: obrigatório Não

LineItemGroups

Informações detectadas em cada tabela de um documento, separadas em`LineItems`.

Type: Matriz deLineItemGroupobjetos

: obrigatório Não

SummaryFields

Qualquer informação encontrada fora de uma tabela pelo Amazon Textract.

Type: Matriz deExpenseFieldobjetos

: obrigatório Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

ExpenseField

Repartição das informações detectadas, separadas nas catagorias Tipo, LabelDetection e ValueDetection

Índice

LabelDetection

O rótulo explicitamente declarado de um elemento detectado.

Tipo: objeto ExpenseDetection

Obrigatório: Não

PageNumber

O número da página em que o valor foi detectado.

Type: Inteiro

Faixa válida: Valor mínimo de 0.

Obrigatório: Não

Type

O rótulo implícito de um elemento detectado. Presente ao lado de LabelDetection para elementos explícitos.

Tipo: objeto ExpenseType

Obrigatório: Não

ValueDetection

O valor de um elemento detectado. Presente em elementos explícitos e implícitos.

Tipo: objeto ExpenseDetection

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

ExpenseType

Um objeto usado para armazenar informações sobre o Tipo detectado pelo Amazon Textract.

Índice

Confidence

A confiança da precisão, como porcentagem.

Type: Float

Faixa válida: Valor mínimo de 0. Valor máximo de 100.

Obrigatório: Não

Text

A palavra ou linha de texto detectada pelo Amazon Textract.

Type: String

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

Geometry

Informações sobre onde os seguintes itens estão localizados em uma página de documento: página detectada, texto, pares de chave-valor, tabelas, células de tabela e elementos de seleção.

Índice

BoundingBox

Uma representação grossa alinhada por eixo da localização do item reconhecido na página do documento.

Tipo: objeto BoundingBox

Obrigatório: Não

Polygon

Dentro da caixa delimitadora, um polígono de grão fino ao redor do item reconhecido.

Type: Matriz dePointobjetos

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

HumanLoopActivationOutput

Mostra os resultados do humano na avaliação de loop. Se não houver HumanLooparn, a entrada não desencadeou a revisão humana.

Índice

HumanLoopActivationConditionsEvaluationResults

Mostra o resultado de avaliações de condições, incluindo as condições que ativaram uma revisão humana.

Type: String

Restrições de tamanho: Tamanho máximo de 10240.

: obrigatório Não

HumanLoopActivationReasons

Mostra se e por que a revisão humana era necessária.

Type: Matriz de strings

Membros da matriz: Número mínimo de 1 item.

: obrigatório Não

HumanLoopArn

O nome de recurso da Amazon (ARN) do HumanLoop criado.

Type: String

Restrições de tamanho: Tamanho máximo de 256.

: obrigatório Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++

- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

HumanLoopConfig

Configura o fluxo de trabalho de revisão humana para o qual o documento será enviado se uma das condições for atendida. Você também pode definir determinados atributos da imagem antes da revisão.

Índice

DataAttributes

Define os atributos dos dados de entrada.

Tipo: objeto HumanLoopDataAttributes

: obrigatório Não

FlowDefinitionArn

O nome de recurso da Amazon (ARN) da definição de fluxo.

Type: String

Restrições de comprimento: Tamanho máximo de 256.

: obrigatório Sim

HumanLoopName

O nome do fluxo de trabalho humano usado para esta imagem. Isso deve ser mantido exclusivo em uma região.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 63.

Padrão: `^[a-z0-9](-*[a-z0-9])*`

: obrigatório Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

HumanLoopDataAttributes

Permite definir atributos da imagem. Atualmente, você pode declarar uma imagem como livre de informações pessoalmente identificáveis e conteúdo adulto.

Índice

ContentClassifiers

Define se a imagem de entrada não contém informações de identificação pessoal ou conteúdo adulto.

Type: Matriz de strings

Membros de matriz: Número máximo de 256 itens.

Valores válidos: `FreeOfPersonallyIdentifiableInformation` | `FreeOfAdultContent`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

IdentityDocument

A estrutura que lista cada documento processado em uma operação AnalyzeID.

Índice

DocumentIndex

Denota o posicionamento de um documento na lista IdentityDocument. O primeiro documento está marcado como 1, o segundo 2 e assim por diante.

Type: Inteiro

Faixa válida: Valor mínimo de 0.

Obrigatório: Não

IdentityDocumentFields

A estrutura usada para registrar informações extraídas de documentos de identidade. Contém campo normalizado e valor do texto extraído.

Type: Matriz deIdentityDocumentFieldobjetos

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

IdentityDocumentField

Estrutura contendo o tipo normalizado da informação extraída e o texto associado a ela. Eles são extraídos como Tipo e Valor, respectivamente.

Índice

Type

Usado para conter as informações detectadas por uma operação AnalyzeID.

Tipo: objeto AnalyzeIDDetections

Obrigatório: Não

ValueDetection

Usado para conter as informações detectadas por uma operação AnalyzeID.

Tipo: objeto AnalyzeIDDetections

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

LineItemFields

Uma estrutura que contém informações sobre as diferentes linhas encontradas nas tabelas de um documento.

Índice

LineItemExpenseFields

ExpenseFields usado para mostrar informações de linhas detectadas em uma tabela.

Type: Matriz deExpenseFieldobjetos

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

LineItemGroup

Um agrupamento de tabelas que contêm itens de linha, com cada tabela identificada pela tabela`LineItemGroupIndex`.

Índice

LineItemGroupIndex

O número usado para identificar uma tabela específica em um documento. A primeira tabela encontrada terá um LineItemGroupIndex de 1, o segundo 2, etc.

Type: Inteiro

Faixa válida: Valor mínimo de 0.

Obrigatório Não

LineItems

O detalhamento de informações em uma linha específica de uma tabela.

Type: Matriz deLineItemFieldsobjetos

Obrigatório Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

NormalizedValue

Contém informações relacionadas a datas em um documento, incluindo o tipo de valor e o valor.

Índice

Value

O valor da data, escrito como ano-mês-dia:minuto:segundo.

Type: String

Obrigatório: Não

ValueType

O tipo normalizado do valor detectado. Nesse caso, DATE.

Type: String

Valores válidos: `DATE`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

NotificationChannel

O tópico do Amazon Simple Notification Service (Amazon SNS) no qual o Amazon Textract publica o status de conclusão de uma operação assíncrona de documento, comoStartDocumentTextDetection.

Índice

RoleArn

O nome de recurso da Amazon (ARN) de uma função do IAM que dá permissões de publicação do Amazon Textract ao tópico do Amazon SNS.

Type: String

Restrições de comprimento: Tamanho mínimo de 20. Tamanho máximo de 2048.

Padrão: `arn:([a-z\d-]+):iam::\d{12}:role/?[a-zA-Z_0-9+=,.@\-_/]+`

Obrigatório Sim

SNSTopicArn

O tópico do Amazon SNS no qual o Amazon Textract publica o status de conclusão.

Type: String

Restrições de comprimento: Tamanho mínimo de 20. Tamanho máximo de 1024.

Padrão: `(^arn:([a-z\d-]+):sns:[a-zA-Z\d-]{1,20}:\w{12}:.+$)`

Obrigatório Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

OutputConfig

Define se a saída irá ou não para um bucket criado pelo usuário. Usado para definir o nome do bucket e o prefixo no arquivo de saída.

`OutputConfig`é um parâmetro opcional que permite ajustar onde sua saída será colocada. Por padrão, o Amazon Textract armazenará os resultados internamente e só poderá ser acessado pelas operações Get API. Com o OutputConfig ativado, você pode definir o nome do bucket para o qual a saída será enviada e o prefixo do arquivo dos resultados onde você pode baixar seus resultados. Além disso, você pode definir o`KMSKeyID`Para criptografar sua saída para uma chave mestra do cliente (CMK) para criptografar sua saída. Sem esse conjunto de parâmetros, o Amazon Textract criptografará o lado do servidor usando a CMK gerenciada pela AWS para o Amazon S3.

A descriptografia do conteúdo do cliente é necessária para o processamento dos documentos pelo Amazon Textract. Se sua conta for cancelada de acordo com uma política de exclusão de serviços de IA, todo o Conteúdo do Cliente não criptografado será excluído imediata e permanentemente após o Conteúdo do Cliente ter sido processado pelo serviço. Nenhuma cópia da saída é mantida pelo Amazon Textract. Para obter informações sobre como cancelar o, consulte[Gerenciando política de exclusão dos serviços de IA.](#)

Para obter mais informações sobre a privacidade de dados, consulte o[Perguntas frequentes sobre privacidade de dados](#).

Índice

S3Bucket

O nome do bucket para o qual a saída irá.

Type: String

Restrições de comprimento: Tamanho mínimo de 3. Tamanho máximo de 255.

Padrão: `[0-9A-Za-z\.\-_]*`

Obrigatório: Sim

S3Prefix

O prefixo da chave de objeto na qual a saída será salva. Quando não estiver ativado, o prefixo será “textract_output”.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 1024.

Padrão: `.*\S.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

Point

As coordenadas X e Y de um ponto em uma página de documento. Os valores X e Y retornados são proporções do tamanho geral da página do documento. Por exemplo, se o documento de entrada for 700 x 200 e a operação retornar X=0,5 e Y=0,25, então o ponto estará na coordenada de pixel (350,50) na página do documento.

Uma matriz de`Point`objetos,`Polygon`, O é retornado como parte daGeometryobjeto que é retornado em umBlockobjeto. UMA`Polygon`objeto representa um polígono de granulação fina em torno do texto detectado, um par de valores-chave, uma tabela, uma célula de tabela ou um elemento de seleção.

Índice

X

O valor da coordenada X para um ponto em um`Polygon`.

Type: Float

Obrigatório: Não

Y

O valor da coordenada Y para um ponto em um`Polygon`.

Type: Float

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

Relationship

Informações sobre como os blocos estão relacionados entre si. UMA`Block`O objeto contém 0 ou mais`Relation`objetos em uma lista,`Relationships`. Para obter mais informações, consulte Block.

O`Type`elemento fornece o tipo de relacionamento para todos os blocos na`IDs`matriz.

Índice

Ids

Uma matriz de IDs para blocos relacionados. Você pode obter o tipo de relacionamento no`Type`elemento.

Type: Matriz de strings

Padrão: `.*\S.*`

Obrigatório: Não

Type

O tipo de relacionamento que os blocos na matriz IDs têm com o bloco atual. O relacionamento pode ser`VALUE`ou`CHILD`. Um relacionamento do tipo VALUE é uma lista que contém o ID do bloco VALUE associado à CHAVE de um par de valores-chave. Um relacionamento do tipo CHILD é uma lista de IDs que identificam blocos WORD no caso de linhas Blocos de células no caso de Tabelas e blocos WORD no caso de Elementos de Seleção.

Type: String

Valores válidos: `VALUE | CHILD | COMPLEX_FEATURES`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go

- AWSSDK for Java V2
- AWS SDK for Ruby V3

S3Object

O nome do bucket do S3 e o nome do arquivo que identifica o documento.

A região da AWS para o bucket do S3 que contém o documento deve corresponder à região que você usa para operações do Amazon Textract.

Para que o Amazon Textract processe um arquivo em um bucket do S3, o usuário deve ter permissão para acessar o bucket e o arquivo do S3.

Índice

Bucket

O nome do bucket do S3. Observe que o caractere # não é válido no nome do arquivo.

Type: String

Restrições de comprimento: Tamanho mínimo de 3. Tamanho máximo de 255.

Padrão: `[0-9A-Za-z\.\-_]*`

: obrigatório Não

Name

O nome do arquivo do documento de entrada. As operações síncronas podem usar arquivos de imagem que estão no formato JPEG ou PNG. As operações assíncronas também suportam arquivos nos formatos PDF e TIFF.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 1024.

Padrão: `.*\S.*`

: obrigatório Não

Version

Se o bucket tem versionamento habilitado, você pode especificar a versão do objeto.

Type: String

Restrições de comprimento: Tamanho mínimo de 1. Tamanho máximo de 1024.

Padrão: `.*\S.*`

: obrigatório Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

Warning

Um aviso sobre um problema ocorrido durante a análise de texto assíncrona (StartDocumentAnalysis) ou detecção de texto de documento assíncrono (StartDocumentTextDetection).

Índice

ErrorCode

O código de erro para o aviso.

Type: String

Obrigatório Não

Pages

Uma lista das páginas às quais o aviso se aplica.

Type: Matriz de inteiros

Faixa válida: Valor mínimo de 0.

Obrigatório Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos de linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK for Go
- AWSSDK for Java V2
- AWS SDK for Ruby V3

Limites rígidos no Amazon Textract

Esta é uma lista de limites rígidos no Amazon Textract que não podem ser alterados. Para obter informações sobre limitações de localização e limites que podem ser alterados, consulteEndpoints e cotas do Amazon Textract. Para obter informações sobre os limites que podem ser alterados, consulte Limites de serviço da AWS. Para alterar um limite, consulte Criar caso.

Amazon Textract

Limite	Descrição
Formatos de arquivos aceitos	As operações suportam arquivos JPEG, PNG, PDF e TIFF. (As imagens codificadas em JPEG 2000 em PDFs são suportadas)..
Limites de tamanho de arquivo e contagem de páginas	Para operações síncronas, os arquivos JPEG, PNG, PDF e TIFF têm um limite de tamanho de 10 MB. Os arquivos PDF e TIFF também têm um limite de 1 página. Para operações assíncron as, os arquivos JPEG e PNG têm um limite de tamanho de 10 MB. Os arquivos PDF e TIFF têm um limite de 500 MB. Os arquivos PDF e TIFF têm um limite de 3.000 páginas.
Limites específicos de PDF	A altura e a largura máximas são de 40 polegadas e 2880 pontos. Os PDFs não podem ser protegidos por senha. Os PDFs podem conter imagens formatadas em JPEG 2000.
Rotação de documentos e tamanho da imagem	Amazon Textract oferece suporte a todas as rotações de documentos no plano, por exemplo, rotação no plano de 45 graus. Amazon Textract oferece suporte a imagens com uma resolução menor ou igual a 10000 pixels em todos os lados.
Alinhamento de texto	O texto pode ser alinhado horizontalmente dentro do documento . Amazon Textract não oferece suporte ao alinhamento de texto vertical dentro do documento.

Limite	Descrição
Linguagens	O Amazon Textract é compatível com detecção de texto em inglês, francês, alemão, italiano, português e espanhol. Amazon Textract não retornará o idioma detectado em sua saída.
Tamanho do caractere	A altura mínima para o texto a ser detectado é de 15 pixels. A 150 DPI, isso seria o mesmo que a fonte de 8 pontos.
Tipo de caractere	Amazon Textract oferece suporte ao reconhecimento de caracteres manuscritos e impressos.
Characteres	O Amazon Textract detecta os seguintes caracteres: • a-z • A-Z • 0-9 • ä Ä ö Ü Ü ç ç é É â ê ê ê ô ô à è è ù ë ë ï ü Ü á é í Í Í Ó ú ú ü ñ ñ ì Ì ò Ò Ã Ã õ Õ Õ Õ • ! “# $% '& () * +, -./:; =? @ [\] ^ _ `{|} ~ > <° € £ ¥ # ß ¿¡€ £ ¥ # ø Ø ⌘©®™ § ¹ ² ³ '
Limites específicos do AnalyzeID	O AnalyzeID só suporta passaportes dos EUA e Carteiras de Motorista dos EUA.

Histórico do documento do Amazon Textract

A tabela a seguir descreve as mudanças importantes em cada versão doGuia do desenvolvedor do Amazon Textract. Para receber notificações sobre atualizações dessa documentação, você pode se inscrever em um feed RSS.

- Última atualização da documentação: 29 de maio de 2019

update-history-change	update-history-description	update-history-date
Integrando exemplos de código deAWSExemplos de código do SDK de documentos GitHub repo	O guia do Amazon Textract agora contém exemplos de código adicionais. Seção de exemplos anteriores renomeada para Tutoriais.	30 de janeiro de 2022
AnalyzeExpense Adição do	O Amazon Textract agora oferece suporte à análise de documentos de fatura e recebimento usando a API AnalyzeExpense. Esta funcionalidade está disponível apenas em nossas regiões Ásia-Pacífico (Mumbai), Ásia-Pacífico (Seul), Ásia-Pacífico (Cingapura), Ásia-Pacífico (Sydney), Canadá (Central), Europa (Irlanda), Europa (Londres), Leste dos EUA (Norte da Virgínia), Leste dos EUA (Ohio) Oeste dos EUA (Norte da Califórnia) e Oeste dos EUA (Oregon).	26 de julho de 2021
Support Augmented AI	O Amazon Textract é compatível agora com	3 de dezembro de 2019

	Amazon Augmented AI para implementar revisão humana.	
Serviço e guia novos	O Amazon Textract está disponível para uso geral.	29 de maio de 2019
Support para elementos de seleção	O Amazon Textract agora pode detectar elementos de seleção (botões de opção e caixas de seleção).	24 de abril de 2019
Versão do Amazon Textract	Esta é a primeira versão da documentação do Amazon Textract.	28 de novembro de 2018

Glossário da AWS

Para obter a terminologia mais recente da AWS, consulte o glossário da AWS na Referência geral da AWS.

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.