Using Amazon Comprehend Medical and LLMs for healthcare and life sciences

AWS Prescriptive Guidance

AWS Prescriptive Guidance: Using Amazon Comprehend Medical and LLMs for healthcare and life sciences

Table of Contents

Using Amazon Comprehend Medical and LLMs for healthcare and life sciences

Joe King, Rajesh Sitaraman, and Ross Claytor, Amazon Web Services

December 2024 (document history)

Overview

The ever-increasing volume of medical data and the need for efficient and accurate processing have driven the adoption of natural language processing (NLP) with artificial intelligence and machine learning (AI/ML) technologies. Pretrained classifier models and large language models (LLMs) have emerged as powerful tools for various medical NLP tasks, including clinical question-answering, report summarization, and insight generation. However, the healthcare and life science domain presents unique challenges due to the complexity of medical terminology, domain-specific knowledge, and regulatory requirements. Effectively using pretrained classifiers or LLMs in this domain requires a well-designed approach that combines the strengths of these models with domain-specific resources and techniques.

Industry practices in healthcare and life science have traditionally relied on rule-based systems, manual coding, and expert review processes. These systems and processes are time-consuming and error-prone. The integration of AI and NLP technologies, such as Amazon Comprehend Medical and the foundation models in Amazon Bedrock, offers efficient and scalable solutions for processing medical data while improving accuracy and consistency.

This guide explores the use of Amazon Comprehend Medical and LLMs for intelligent automation in the healthcare industry. It outlines best practices, challenges, and practical approaches to streamlining medical coding, patient information extraction, and record summarization processes. By using the capabilities of Amazon Comprehend Medical and LLMs, healthcare organizations can unlock new levels of operational efficiency, reduce costs, and potentially improve patient care.

The guide details the unique considerations of the healthcare domain, such as understanding medical terminology, using domain-specific LLMs, and addressing the limitations of AI/ML systems. It provides a comprehensive decision path for healthcare IT managers, architects, and technical leads to assess organizational readiness, evaluate implementation options, and use the appropriate AWS services and tools for successful automation.

By following the guidelines and best practices outlined in this guide, healthcare organizations can harness the power of AI/ML technologies while navigating the complexities of the medical domain. This approach supports compliance with ethical and regulatory guidelines and promotes the responsible use of AI systems in healthcare. It is designed to generate insights that are accurate and private.

Intended audience

This guide is intended for technology stakeholders, architects, technical leads, and decision makers who want to implement AI-powered natural language processing solutions for medical data analysis and automation.

Objectives

Healthcare and life science organizations can meet multiple business goals by using Amazon Comprehend Medical and LLMs. These outcomes commonly include increasing operational efficiency, reducing costs, and improving patient care. This section outlines key business objectives and the associated benefits of implementing the strategies and best practices outlined in this guide.

The following are the some of the objectives that organizations can achieve by implementing the guidelines and best practices in this guide:

- **Reduce the development time** – This guide's ultimate goal is to reduce development time with cost, decrease technical debt, and mitigate potential project failure from POC. By understanding key AI/ML services, such as Amazon Comprehend Medical, and the advantages and limitations of LLM usage for healthcare tasks, businesses can achieve faster time to market and increase their velocity in meeting business objectives.
- **Extract information to automate medical coding tasks** – After patient visits, coding specialist and providers can extract insights from medical text, such as subjective, objective, assessment, and plan (SOAP) notes. This can reduce manual documentation efforts and help the provider focus on the patient's needs. By combining the entity recognition capabilities of Amazon Comprehend Medical with LLMs, organizations can extract relevant medical information from patient records, clinical notes, and other healthcare data sources. This can minimize human errors and promote consistent practices.
- **Summarize patient records and clinical documentation** – Automated summarization of patient history, treatment plans, and medical results can save valuable time for healthcare providers.

LLMs can help generate comprehensive and structured clinical documentation. You can get additional context with Amazon Comprehend Medical, use a medical domain LLM, or fine-tune an LLM with medical data. These approaches can help provide accurate summaries and make sure that documentation adheres to compliance requirements and standards.

- **Support clinical decisions and patient care –** By using ontology linking in Amazon Comprehend Medical and by using LLMs, providers can answer medical questions or seek recommendations addressing patient care. This empowers healthcare professionals to make informed decisions that improve patient outcomes and reduce the risk of medical errors.

Generative AI and NLP approaches for healthcare and life sciences

Natural language processing (NLP) is a machine learning technology that gives computers the ability to interpret, manipulate, and comprehend human language. Healthcare and life science organizations have large volumes of data from patient records. They can use NLP software to automatically process this data. For example, they can combine NLP with generative AI to streamline medical coding, extract patient information, and summarize records.

Depending on the NLP task that you want to perform, different architectures might be best suited for your use case. This guide addresses the following generative AI and NLP options for healthcare and life science applications on AWS:

- Using Amazon Comprehend Medical – Learn about how to use Amazon Comprehend Medical independently, without integrating it with a large language model (LLM).
- Combining Amazon Comprehend Medical with LLMs – Learn about how to combine Amazon Comprehend Medical with an LLM in a Retrieval Augment Generation (RAG) architecture.
- Using LLMs – Learn about how to use an LLM for healthcare and life science applications, either by using a fine-tuned LLM or a RAG architecture.

Using Amazon Comprehend Medical

Amazon Comprehend Medical is an AWS service that detects and returns useful information in unstructured clinical text such as physician's notes, discharge summaries, test results, and case notes. It uses natural language processing (NLP) models to detect entities. *Entities* are textual references to medical information, such as medical conditions, medications, or protected health information (PHI).

Important

Amazon Comprehend Medical is not a substitute for professional medical advice, diagnosis, or treatment. Amazon Comprehend Medical provides confidence scores that indicate the level of confidence in the accuracy of the detected entities. Identify the right confidence threshold for your use case, and use high confidence thresholds in situations that require high accuracy. In certain use cases, results should be reviewed and verified by appropriately trained human reviewers. For example, Amazon Comprehend Medical should only be used in patient care scenarios after review for accuracy and sound medical judgment by trained medical professionals.

You can access Amazon Comprehend Medical through the AWS Management Console, the AWS Command Line Interface (AWS CLI), or through the AWS SDKs. The AWS SDKs are available for various programming languages and platforms, such as Java, Python, Ruby, .NET, iOS, and Android. You can use the SDKs to programmatically access Amazon Comprehend Medical from your client application.

This section reviews the main capabilities of Amazon Comprehend Medical. It also discusses the advantages of using this service compared to a large language model (LLM).

Amazon Comprehend Medical capabilities

Amazon Comprehend Medical offers APIs for near real-time and batch inference. These APIs can ingest medical text and provide results for medical NLP tasks by using medical entity recognition and identifying entity relationships. You can perform analysis both on single files or as a batch analysis on multiple files stored in an Amazon Simple Storage Service (Amazon S3) bucket. Amazon Comprehend Medical offers the following text analysis API operations for synchronous entity detection:

- Detect entities – Detects general medical categories such as anatomy, medical condition, PHI category, procedures, and time expressions.
- Detect PHI – Detects specific entities such as age, date, name, and similar personal information.

Amazon Comprehend Medical also includes multiple API operations that you can use to perform batch text analysis on clinical documents. To learn more about how to use these API operations, see Text analysis batch APIs.

Use Amazon Comprehend Medical to detect entities in clinical text and link those entities to concepts in standardized medical ontologies, including the RxNorm, ICD-10-CM, and SNOMED CT knowledge bases. You can perform analysis both on single files or as a batch analysis on large documents or multiple files stored in an Amazon S3 bucket. Amazon Comprehend Medical offers the following ontology linking API operations:

- InferICD10CM – The **InferICD10CM** operation detects potential medical conditions and links them to codes from the 2019 version of the International Classification of Diseases, 10th Revision, Clinical Modification (ICD-10-CM). For each potential medical condition detected, Amazon Comprehend Medical lists the matching ICD-10-CM codes and descriptions. Listed medical conditions in the results include a confidence score, which indicates the confidence that Amazon Comprehend Medical has in the accuracy of the entities to the matched concepts in the results.
- InferRxNorm – The **InferRxNorm** operation identifies medications that are listed in a patient record as entities. It links entities to concept identifiers (RxCUI) from the RxNorm database from the National Library of Medicine. Each RxCUI is unique for different strengths and dose forms. Listed medications in the results include a confidence score, which indicates the confidence that Amazon Comprehend Medical has in the accuracy of the entities matched to the concepts from the RxNorm knowledge base. Amazon Comprehend Medical lists the top RxCUIs that potentially match for each medication that it detects in descending order based on confidence score.
- InferSNOMEDCT – The **InferSNOMEDCT** operation identifies possible medical concepts as entities and links them to codes from the 2021-03 version of the Systematized Nomenclature of Medicine, Clinical Terms (SNOMED CT). SNOMED CT provides a comprehensive vocabulary of medical concepts, including medical conditions and anatomy, as well as medical tests, treatments, and procedures. For each matched concept ID, Amazon Comprehend Medical returns the top five medical concepts, each with a confidence score and contextual information such as traits and attributes. The SNOMED CT concept IDs can then be used to structure patient clinical data for medical coding, reporting, or clinical analytics when used with the SNOMED CT polyhierarchy.

For more information, see Text analysis APIs and Ontology Linking APIs in the Amazon Comprehend Medical documentation.

Use cases for Amazon Comprehend Medical

As a standalone service, Amazon Comprehend Medical might address your organization's use case. Amazon Comprehend Medical can perform tasks such as the following:

- Help with medical coding in patient records
- Detect protected health information (PHI) data
- Validating medication, including attributes such as dosage, frequency, and form

Amazon Comprehend Medical results are digestible for the majority of medical practices. However, you might need to consider alternatives if you have limitations such as the following:

- **Different entity definitions –** For example, your definition of `FREQUENCY` of a medication entity might differ. For frequency, Amazon Comprehend Medical predicts *as needed*, but your organization might use the term *pro re nata (PRN)*.
- **Overwhelming quantity of results –** For example, patient notes frequently contain multiple symptoms and keywords that map to multiple ICD-10-CM codes. However, several of the keywords are not applicable for diagnosis. In this case, the provider has to evaluate numerous ICD-10-CM entities and their confidence scores, which requires manual processing time.
- **Custom entities or NLP tasks –** For example, providers might want to extract PRN evidence, such as *take as needed for pain*. Because this isn't available through Amazon Comprehend Medical, a different AI/ML model is warranted. A different AI/ML solution is required if the NLP task is outside of entity recognition, such as summarization, question-answering, and sentiment analysis.

Combining Amazon Comprehend Medical with large language models

A 2024 study by NEJM AI showed that using an LLM, with zero-shot prompting, for medical coding tasks generally leads to poor performance. Using Amazon Comprehend Medical with an LLM can help mitigate these performance issues. Amazon Comprehend Medical results are helpful context for an LLM that is performing NLP tasks. For example, providing context from Amazon Comprehend Medical to the large language model can help you:

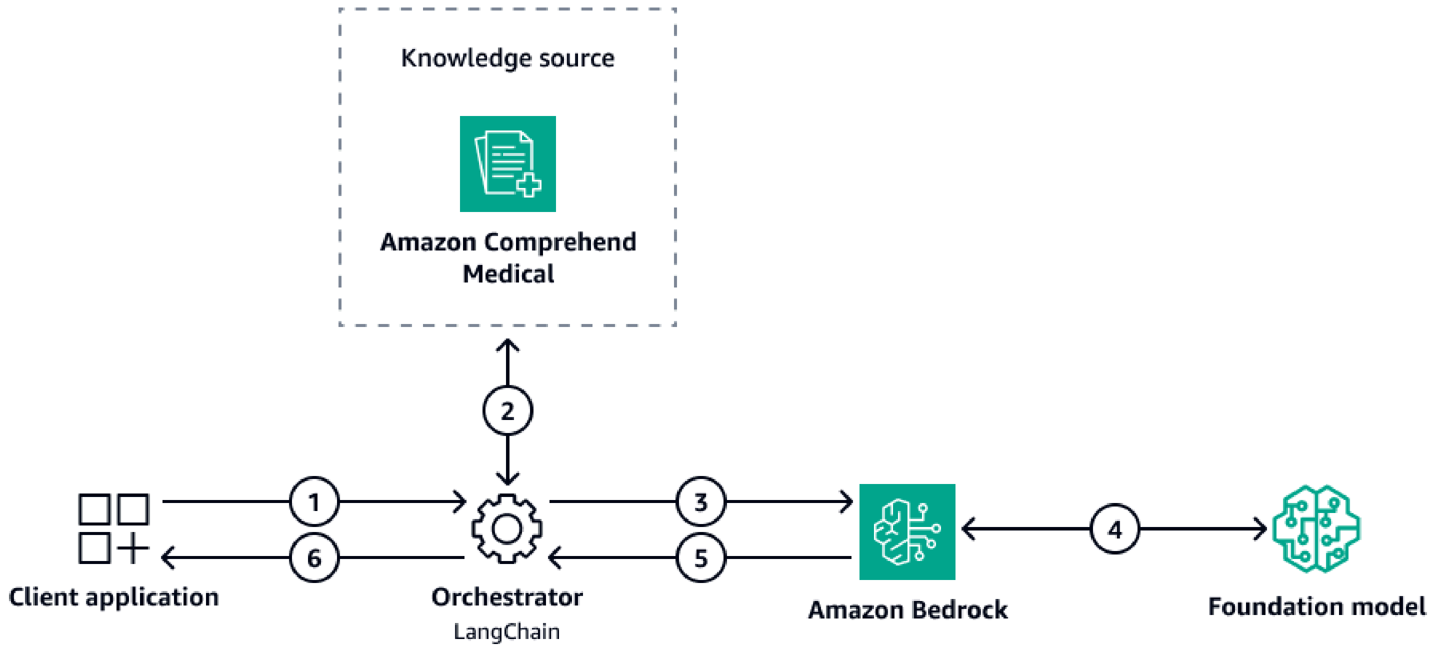
- Enhance the accuracy of entity selections by using the initial results from Amazon Comprehend Medical as context for the LLM
- Implement custom entity recognition, summarization, question-answering, and additional use cases

This section describes how you can combine Amazon Comprehend Medical with an LLM by using a Retrieval Augmented Generation (RAG) approach. *Retrieval Augmented Generation (RAG)* is a generative AI technology in which an LLM references an authoritative data source that is outside of its training data sources before generating a response. For more information, see What is RAG.

To illustrate this approach, this section uses the example of medical (diagnosis) coding related to ICD-10-CM. It includes a sample architecture and prompt engineering templates to help accelerate your innovation. It also includes best practices for using Amazon Comprehend Medical within a RAG workflow.

RAG-based architecture with Amazon Comprehend Medical

The following diagram illustrates a RAG approach for identifying ICD-10-CM diagnosis codes from patient notes. It uses Amazon Comprehend Medical as a knowledge source. In a RAG approach, the retrieval method commonly retrieves information from a vector database containing applicable knowledge. Instead of a vector database, this architecture uses Amazon Comprehend Medical for the retrieval task. The orchestrator sends the patient note information to Amazon Comprehend Medical and retrieves the ICD-10-CM code information. The orchestrator sends this context to the downstream foundation model (LLM), through Amazon Bedrock. The LLM generates a response by using the ICD-10-CM code information, and that response is sent back to the client application.



The diagram shows the following RAG workflow:

1. The client application sends the patient notes as a query to the orchestrator. An example of these patient notes might be "The patient is a 71-year-old female patient of Dr. X. The patient presented to the emergency room last evening with approximately 7-day to 8-day history of abdominal pain, which has been persistent. She has had no definite fevers or chills and no history of jaundice. The patient denies any significant recent weight loss."

2. The orchestrator uses Amazon Comprehend Medical to retrieve ICD-10-CM codes relevant to the medical information in the query. It uses the **InferICD10CM** API to extract and infer the ICD-10-CM codes from the patient notes.
3. The orchestrator constructs a prompt that includes the prompt template, the original query, and the ICD-10-CM codes retrieved from Amazon Comprehend Medical. It sends this enhanced context to Amazon Bedrock.
4. Amazon Bedrock processes the input and uses a foundation model to generate a response that includes the ICD-10-CM codes and their corresponding evidence from the query. The generated response includes the identified ICD-10-CM codes and evidence from the patient notes that supports each code. The following is a sample response:

```
<response>
<icd10>
<code>R10.9</code>
<evidence>history of abdominal pain</evidence>
</icd10>
<icd10>
<code>R10.30</code>
<evidence>history of abdominal pain</evidence>
</icd10>
</response>
```

5. Amazon Bedrock sends the generated response to the orchestrator.
6. The orchestrator sends the response back to the client application, where the user can review the response.

Use cases for using Amazon Comprehend Medical in a RAG workflow

Amazon Comprehend Medical can perform specific NLP tasks. For more information, see Use cases for Amazon Comprehend Medical.

You might want to integrate Amazon Comprehend Medical into a RAG workflow for advanced use cases, such as the following:

- Generate detailed clinical summaries by combining extracted medical entities with contextual information from patient records
- Automate medical coding for complex cases by using extracted entities with ontology-linked information for code assignment

- Automate the creation of structured clinical notes from unstructured text by using extracted medical entities
- Analyze medication side effects based on extracted medication names and attributes
- Develop intelligent clinical support systems that combine extracted medical information with up-to-date research and guidelines

Best practices for using Amazon Comprehend Medical in a RAG workflow

When integrating Amazon Comprehend Medical results into a prompt for an LLM, it's essential to follow best practices. This can improve performance and accuracy. The following are key recommendations:

- **Understand Amazon Comprehend Medical confidence scores –** Amazon Comprehend Medical provides confidence scores for each detected entity and ontology linking. It's crucial to understand the meaning of these scores and establish appropriate thresholds for your specific use case. Confidence scores help filter out low-confidence entities, reducing noise and improving the quality of the LLM's input.
- **Use confidence scores in prompt engineering –** When crafting prompts for the LLM, consider incorporating Amazon Comprehend Medical confidence scores as additional context. This helps the LLM prioritize or weigh entities based on their confidence levels, potentially improving the quality of the output.
- **Evaluate Amazon Comprehend Medical results with ground truth data –** *Ground truth data* is information that is known to be true. It can be used to validate that an AI/ML application is producing accurate results. Before integrating Amazon Comprehend Medical results into your LLM workflow, evaluate the service's performance on a representative sample of your data. Compare the results with ground truth annotations to identify potential discrepancies or areas for improvement. This evaluation helps you understand the strengths and limitations of Amazon Comprehend Medical for your use case.
- **Strategically select relevant information –** Amazon Comprehend Medical can provide a large amount of information, but not all of it may be relevant to your task. Carefully select the entities, attributes, and metadata that are most relevant to your use case. Providing too much irrelevant information to the LLM can introduce noise and potentially decrease performance.
- **Align entity definitions –** Ensure that the definitions of entities and attributes used by Amazon Comprehend Medical align with your interpretation. If there are discrepancies, consider

providing additional context or clarification to the LLM to bridge the gap between the Amazon Comprehend Medical output and your requirements. If Amazon Comprehend Medical entity doesn't meet your expectations, you can implement custom entity detection by including additional instructions (and possible examples) within the prompt.

- **Provide domain-specific knowledge –** While Amazon Comprehend Medical provides valuable medical information, it might not capture all the nuances of your specific domain. Consider supplementing Amazon Comprehend Medical results with additional domain-specific knowledge sources, such as ontologies, terminologies, or expert-curated datasets. This provides more comprehensive context to the LLM.
- **Adhere to ethical and regulatory guidelines –** When dealing with medical data, it's important to adhere to ethical principles and regulatory guidelines, such as those related to data privacy, security, and responsible use of AI systems in healthcare. Make sure that your implementation complies with relevant laws and industry best practices.

By following these best practices, AI/ML practitioners can effectively use the strengths of both Amazon Comprehend Medical and LLMs. For medical NLP tasks, these best practices help mitigate potential risks and can improve performance.

Prompt engineering for Amazon Comprehend Medical context

Prompt engineering is the process of designing and refining prompts to guide a generative AI solution to generate desired outputs. You choose the most appropriate formats, phrases, words, and symbols that guide the AI to interact with your users more meaningfully.

Depending on the API operation you perform, Amazon Comprehend Medical returns the detected entities, ontology codes and descriptions, and confidence scores. These results become context within the prompt when your solution invokes the target LLM. You must engineer the prompt to present the context within the prompt template.

Note

The example prompts in this section follow Anthropic guidance. If you're using a different LLM provider, follow the recommendations from that provider.

In general, you insert both the original medical text and the Amazon Comprehend Medical results into the prompt. The following is a common prompt structure:

```
<medical_text>
medical text
</medical_text>

<comprehend_medical_text_results>
comprehend medical text results
</comprehend_medical_text_results>

<prompt_instructions>
prompt instructions
</prompt_instructions>
```

This section provides strategies for including Amazon Comprehend Medical results as prompt context for the following common medical NLP tasks:

- Filter Amazon Comprehend Medical results
- Extend medical NLP tasks with Amazon Comprehend Medical
- Apply guardrails with Amazon Comprehend Medical

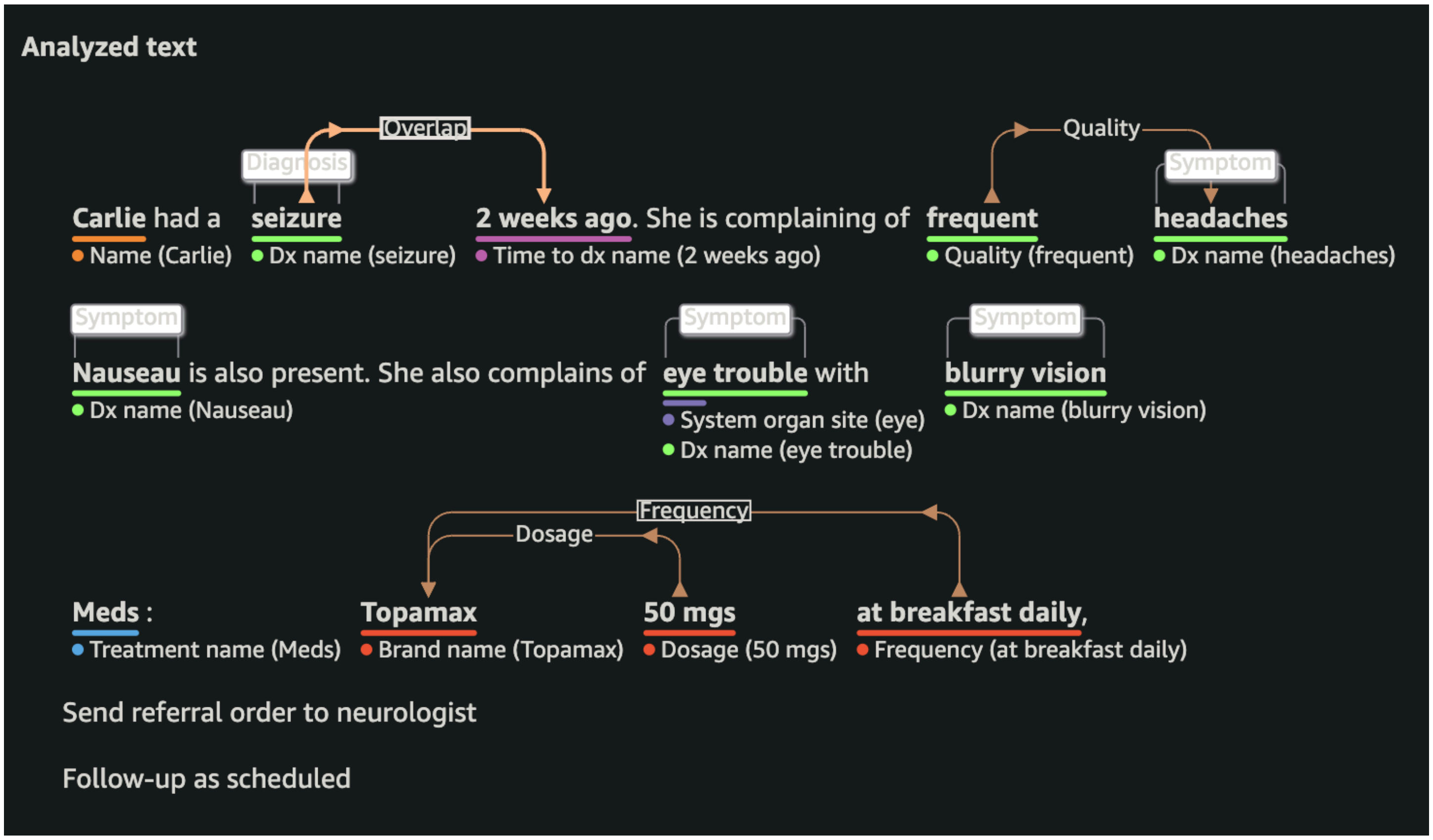
Filter Amazon Comprehend Medical results

Amazon Comprehend Medical typically provides a large amount of information. You might want to reduce the number of results that the medical professional must review. In this case, you can use an LLM to filter these results. Amazon Comprehend Medical entities include a confidence score that you can use as a filtering mechanism when designing the prompt.

The following is an example patient note:

```
Carlie had a seizure 2 weeks ago. She is complaining of frequent headaches
Nausea is also present. She also complains of eye trouble with blurry vision
Meds : Topamax 50 mgs at breakfast daily,
Send referral order to neurologist
Follow-up as scheduled
```

In this patient note, Amazon Comprehend Medical detects the following entities.



Carlie had a seizure 2 weeks ago. She is complaining of frequent headaches

● Name (Carlie) ● Dx name (seizure) ● Time to dx name (2 weeks ago) ● Quality (frequent) ● Dx name (headaches)

Nauseau is also present. She also complains of eye trouble with blurry vision

● Dx name (Nauseau) ● System organ site (eye) ● Dx name (eye trouble) ● Dx name (blurry vision)

Meds : Topamax 50 mgs at breakfast daily,

● Treatment name (Meds) ● Brand name (Topamax) ● Dosage (50 mgs) ● Frequency (at breakfast daily)

Send referral order to neurologist

Follow-up as scheduled

The entities link to the following ICD-10-CM codes for seizure and headaches.

Category	ICD-10-CM code	ICD-10-CM description	Confidence score
Seizure	R56.9	Unspecified convulsions	0.8348
Seizure	G40.909	Epilepsy, unspecified, not intractable, without status epilepticus	0.5424
Seizure	R56.00	Simple febrile convulsions	0.4937
Seizure	G40.09	Other seizures	0.4397

Category	ICD-10-CM code	ICD-10-CM description	Confidence score
Seizure	G40.409	Other generalized epilepsy and epileptic syndromes, not intractable, without status epilepticus	0.4138
Headaches	R51	Headache	0.4067
Headaches	R51.9	Headache, unspecified	0.3844
Headaches	G44.52	New daily persistent headache (NDPH)	0.3005
Headaches	G44	Other headache syndrome	0.2670
Headaches	G44.8	Other specified headache syndromes	0.2542

You can pass ICD-10-CM codes into the prompt to increase LLM precision. To reduce noise, you can filter the ICD-10-CM codes by using the confidence score included in the Amazon Comprehend Medical results. The following is an example prompt that includes only ICD-10-CM codes that have a confidence score higher than 0.4:

```
<patient_note>
Carlie had a seizure 2 weeks ago. She is complaining of frequent headaches
Nausea is also present. She also complains of eye trouble with blurry vision
Meds : Topamax 50 mgs at breakfast daily,
Send referral order to neurologist
Follow-up as scheduled
</patient_note>

<comprehend_medical_results>
<icd-10>
  <entity>
```

```
    <text>seizure</text>
    <code>
      <description>Unspecified convulsions</description>
      <code_value>R56.9</code_value>
      <score>0.8347607851028442</score>
    </code>
    <code>
      <description>Epilepsy, unspecified, not intractable, without status epilepticus</
description>
      <code_value>G40.909</code_value>
      <score>0.542376697063446</score>
    </code>
    <code>
      <description>Other seizures</description>
      <code_value>G40.89</code_value>
      <score>0.43966275453567505</score>
    </code>
    <code>
      <description>Other generalized epilepsy and epileptic syndromes, not intractable,
 without status epilepticus</description>
      <code_value>G40.409</code_value>
      <score>0.41382506489753723</score>
    </code>
  </entity>
  <entity>
    <text>headaches</text>
    <code>
      <description>Headache</description>
      <code_value>R51</code_value>
      <score>0.4066613018512726</score>
    </code>
  </entity>
  <entity>
    <text>Nausea</text>
    <code>
      <description>Nausea</description>
      <code_value>R11.0</code_value>
      <score>0.6460834741592407</score>
    </code>
  </entity>
  <entity>
    <text>eye trouble</text>
    <code>
```

```
      <description>Unspecified disorder of eye and adnexa</description>
      <code_value>H57.9</code_value>
      <score>0.6780954599380493</score>
    </code>
    <code>
      <description>Unspecified visual disturbance</description>
      <code_value>H53.9</code_value>
      <score>0.5871203541755676</score>
    </code>
    <code>
      <description>Unspecified disorder of binocular vision</description>
      <code_value>H53.30</code_value>
      <score>0.5539672374725342</score>
    </code>
  </entity>
  <entity>
    <text>blurry vision</text>
    <code>
      <description>Other visual disturbances</description>
      <code_value>H53.8</code_value>
      <score>0.9001834392547607</score>
    </code>
  </entity>
</icd-10>
</comprehend_medical_results>

<prompt>
Given the patient note and Amazon Comprehend Medical ICD-10-CM code results above,
 please select the most relevant ICD-10-CM diagnosis codes for the patient.
For each selected code, provide a brief explanation of why it is relevant based on the
 information in the patient note.
</prompt>
```

Extend medical NLP tasks with Amazon Comprehend Medical

When processing medical text, context from Amazon Comprehend Medical can help the LLM to select better tokens. In this example, you want to match diagnosis symptoms to medications. You also want to find text that relates to medical tests, such as terms that relate to a blood panel test. You can use Amazon Comprehend Medical to detect the entities and the medication names. In this case, you would use the DetectEntitiesV2 and InferRxNorm APIs for Amazon Comprehend Medical.

The following is an example patient note:

```
Carlie had a seizure 2 weeks ago. She is complaining of increased frequent headaches
Given lyme disease symptoms such as muscle ache and stiff neck will order prescription.
Meds : Topamax 50 mgs at breakfast daily. Amoxicillan 25 mg by mouth twice a day
Place MRI radiology order at RadNet
```

To focus on the diagnosis code, only the entities related to `MEDICAL_CONDITION` with type `DX_NAME` are used in the prompt. Other metadata is excluded due to irrelevance. For medication entities, the medication name along with extracted attributes is included. Other medication entity metadata from Amazon Comprehend Medical is excluded due to irrelevance. The following is an example prompt that uses filtered Amazon Comprehend Medical results. The prompt focuses on `MEDICAL_CONDITION` entities that have the `DX_NAME` type. This prompt is designed to more precisely link diagnosis codes with medication and more precisely extract medical order tests:

```
<patient_note>
Carlie had a seizure 2 weeks ago. She is complaining of increased freqeunt headaches
Given lyme disease symptoms such as muscle ache and stiff neck will order
 prescription.
Meds : Topamax 50 mgs at breakfast daily. Amoxicillan 25 mg by mouth twice a day
Place MRI radiology order at RadNet
</patient_note>

<detect_entity_results>
<entity>
    <text>seizure</text>
    <category>MEDICAL_CONDITION</category>
    <type>DX_NAME</type>
</entity>
<entity>
    <text>headaches</text>
    <category>MEDICAL_CONDITION</category>
    <type>DX_NAME</type>
</entity>
<entity>
    <text>lyme disease</text>
    <category>MEDICAL_CONDITION</category>
    <type>DX_NAME</type>
</entity>
<entity>
    <text>muscle ache</text>
    <category>MEDICAL_CONDITION</category>
    <type>DX_NAME</type>
```

```
</entity>
<entity>
    <text>stiff neck</text>
    <category>MEDICAL_CONDITION</category>
    <type>DX_NAME</type>
</entity>
</detect_entity_results>

<rx_results>
<entity>
    <text>Topamax</text>
    <category>MEDICATION</category>
    <type>BRAND_NAME</type>
    <attributes>
        <attribute>
            <type>FREQUENCY</type>
            <text>at breakfast daily</text>
        </attribute>
        <attribute>
            <type>DOSAGE</type>
            <text>50 mgs</text>
        </attribute>
        <attribute>
            <type>ROUTE_OR_MODE</type>
            <text>by mouth</text>
        </attribute>
    </attributes>
</entity>
<entity>
    <text>Amoxicillan</text>
    <category>MEDICATION</category>
    <type>GENERIC_NAME</type>
    <attributes>
        <attribute>
            <type>ROUTE_OR_MODE</type>
            <text>by mouth</text>
        </attribute>
        <attribute>
            <type>DOSAGE</type>
            <text>25 mg</text>
        </attribute>
        <attribute>
            <type>FREQUENCY</type>
            <text>twice a day</text>
        </attribute>
    </attributes>
</entity>
</rx_results>

<prompt>
Based on the patient note and the detected entities, can you please:
1. Link the diagnosis symptoms with the medications prescribed.
Provide your reasoning for the linkages.
2. Extract any entities related to medical order tests mentioned in the note.
</prompt>
```

Apply guardrails with Amazon Comprehend Medical

You can use an LLM and Amazon Comprehend Medical to create guardrails before the generated response is used. You can run this workflow on either unmodified or post-processed medical text. Use cases include addressing protected health information (PHI), detecting hallucinations, or implementing custom policies for publishing results. For example, you can use context from Amazon Comprehend Medical to identify PHI data and then use the LLM to remove that PHI data.

The following is an example of information from a patient record that includes PHI:

```
Patient name: John Doe
Patient SSN: 123-34-5678
Patient DOB: 01/01/2024
Patient address: 123 Main St, Anytown USA
Exam details: good health. Pulse is 60 bpm. needs to work on diet with BMI of 190
```

The following is an example prompt that includes the Amazon Comprehend Medical results as context:

```
<original_text>
Patient name: John Doe
Patient SSN: 123-34-5678 Patient DOB: 01/01/2024
Patient address: 123 Main St, Anytown USA
Exam details: good health. Pulse is 60 bpm. needs to work on diet with BMI of 190
</original_text>

<comprehend_medical_phi_entities>
<entity>
  <text>John Doe</text>
```

```
    <category>PROTECTED_HEALTH_INFORMATION</category>
    <score>0.9967944025993347</score>
    <type>NAME</type>
  </entity>
  <entity>
    <text>123-34-5678</text>
    <category>PROTECTED_HEALTH_INFORMATION</category>
    <score>0.9998034834861755</score>
    <type>ID</type>
  </entity>
  <entity>
    <text>01/01/2000</text>
    <category>PROTECTED_HEALTH_INFORMATION</category>
    <score>0.9964448809623718</score>
    <type>DATE</type>
  </entity>
  </comprehend_medical_phi_entities>

  <instructions>
  Using the provided original text and the Amazon Comprehend Medical PHI entities
   detected, please analyze the text to determine if it contains any additional protected
   health information (PHI) beyond the entities already identified. If additional PHI is
   found, please list and categorize it. If no additional PHI is found, please state that
   explicitly.
  In addition if PHI is found, generate updated text with the PHI removed.
  </instructions>
```

Using large language models for healthcare and life science use cases

This describes how you can use large language models (LLMs) for healthcare and life science applications. Some use cases require the use of a large language model for generative AI capabilities. There are advantages and limitations for even the most state-of-the-art LLMs, and the recommendations in this section are designed to help you achieve your target results.

You can use the decision path to determine the appropriate LLM solution for your use case, considering factors such as domain knowledge and available training data. Additionally, this section discusses popular pretrained medical LLMs and best practices for their selection and use. It also discusses the trade-offs between complex, high-performance solutions and simpler, lower-cost approaches.

Use cases for an LLM

Amazon Comprehend Medical can perform specific NLP tasks. For more information, see Use cases for Amazon Comprehend Medical.

The logical and generative AI capabilities of an LLM might be required for the advanced healthcare and life science use cases, such as the following:

- Classifying custom medical entities or text categories
- Answering clinical questions
- Summarizing medical reports
- Generating and detecting insights from medical information

Customization approaches

It's critical to understand how LLMs are implemented. LLMs are commonly trained with billions of parameters, including training data from many domains. This training allows the LLM to address most generalized tasks. However, challenges often arise when domain-specific knowledge is required. Examples of domain knowledge in healthcare and life science are clinic codes, medical terminology, and health information that is required to generate accurate answers. Therefore, using the LLM as is (zero-shot prompting without supplementing domain knowledge) for these use cases likely results in inaccurate results. There are two approaches you can use to overcome this challenge: Retrieval Augmented Generation and fine-tuning.

Retrieval Augmented Generation

Retrieval Augmented Generation (RAG) is a generative AI technology in which an LLM references an authoritative data source that is outside of its training data sources before generating a response. A RAG system can retrieve medical ontology information (such as international classifications of diseases, national drug files, and medical subject headings) from a knowledge source. This provides additional context to the LLM to support the medical NLP task.

As discussed in the Combining Amazon Comprehend Medical with large language models section, you can use a RAG approach to retrieve context from Amazon Comprehend Medical. Other common knowledge sources include medical domain data that is stored in a database service, such as Amazon OpenSearch Service, Amazon Kendra, or Amazon Aurora. Extracting information from these knowledge sources can affect retrieval performance, especially with semantic queries that use a vector database.

Another option for storing and retrieving domain-specific knowledge is by using Amazon Q Business in your RAG workflow. Amazon Q Business can index internal document repositories or public-facing web sites (such as CMS.gov for ICD-10 data). Amazon Q Business can then extract relevant information from these sources before passing your query to the LLM.

There are multiple ways to build a custom RAG workflow. For example, there are many ways to retrieve data from a knowledge source. For simplicity, we recommend the common retrieval approach of using a vector database, such as Amazon OpenSearch Service, to store knowledge as embeddings. This requires that you use an embedding model, such as a sentence transformer, to generate embeddings for the query and for the knowledge stored in the vector database.

For more information about fully managed and custom RAG approaches, see Retrieval Augmented Generation options and architectures on AWS.

Fine-tuning

Fine-tuning an existing model involves taking an LLM, such as an Amazon Titan, Mistral, or Llama model, and then adapting the model to your custom data. There are various techniques for fine-tuning, most of which involve modifying only a few parameters instead of modifying all of the parameters in the model. This is called *parameter-efficient fine-tuning (PEFT)*. For more information, see Hugging Face PEFT on GitHub.

The following are two common use cases when you might choose to fine-tune an LLM for a medical NLP task:

- **Generative task** – Decoder-based models perform generative AI tasks. AI/ML practitioners use ground truth data to fine-tune an existing LLM. For example, you might train the LLM by using MedQuAD, a public medical question-answering dataset. When you invoke a query to the fine-tuned LLM, you don't need a RAG approach to provide the additional context to the LLM.
- **Embeddings** – Encoder-based models generate embeddings by transforming text into numerical vectors. These encoder-based models are typically called *embedding models*. A *sentence-transformer model* is a specific type of embedding model that is optimized for sentences. The objective is to generate embeddings from input text. The embeddings are then used for semantic analysis or in retrieval tasks. To fine-tune the embedding model, you must have a corpus of medical knowledge, such as documents, that you can use as training data. This is accomplished with pairs of text based on similarity or sentiment to fine-tune a sentence

transformer model. For more information, see Training and Finetuning Embedding Models with Sentence Transformers v3 on Hugging Face.

You can use Amazon SageMaker Ground Truth to build a high-quality, labeled training dataset. You can use the labeled dataset output from Ground Truth to train your own models. You can also use the output as a training dataset for an Amazon SageMaker AI model. For more information about named entity recognition, single label text classification, and multi-label text classification, see Text labeling with Ground Truth in the Amazon SageMaker AI documentation.

Choosing an LLM

Amazon Bedrock is the recommended starting point to evaluate high-performing LLMs. For more information, see Supported foundation models in Amazon Bedrock. You can use model evaluation jobs in Amazon Bedrock in order to compare the outputs from multiple outputs and then choose the model that is best suited for your use case. For more information, see Choose the best performing model using Amazon Bedrock evaluations in the Amazon Bedrock documentation.

Some LLMs have limited training on medical domain data. If your use case requires fine-tuning an LLM or an LLM that Amazon Bedrock doesn't support, consider using Amazon SageMaker AI. In SageMaker AI, you can use a fine-tuned LLM or choose a custom LLM that has been trained on medical domain data.

The following table lists popular LLMs that have been trained on medical domain data.

LLM	Tasks	Knowledge	Architecture
BioBERT	Information retrieval , text classification, and named entity recognition	Abstracts from PubMed, full-text articles from PubMedCentral, and general domain knowledge	Encoder
ClinicalBERT	Information retrieval , text classification, and named entity recognition	Large, multi-cen ter dataset along with over 3,000,000 patient records from	Encoder

LLM	Tasks	Knowledge	Architecture
		electronic health record (EHR) systems	
ClinicalGPT	Summarization, question-answering, and text generation	Extensive and diverse medical datasets, including medical records, domain-sp ecific knowledge, and multi-round dialogue consultations	Decoder
GatorTron-OG	Summarization, question-answering, text generation, and information retrieval	Clinical notes and biomedical literature	Encoder
Med-BERT	Information retrieval , text classification, and named entity recognition	Large dataset of medical texts, clinical notes, research papers, and healthcar e-related documents	Encoder
Med-PaLM	Question-answering for medical purposes	Datasets of medical and biomedical text	Decoder
medAlpaca	Question-answering and medical dialogue tasks	A variety of medical texts, encompassing resources such as medical flashcards, wikis, and dialogue datasets	Decoder
BiomedBERT	Information retrieval , text classification, and named entity recognition	Exclusively abstracts from PubMed and full-text articles from PubMedCentral	Encoder

LLM	Tasks	Knowledge	Architecture
BioMedLM	Summarization, question-answering, and text generation	Biomedical literature from PubMed knowledge sources	Decoder

The following are best practices for using pretrained medical LLMs:

- Understand the training data and its relevance to your medical NLP task.
- Identify the LLM architecture and its purpose. Encoders are appropriate for embeddings and NLP tasks. Decoders are for generation tasks.
- Evaluate the infrastructure, performance, and cost requirements for hosting the pretrained medical LLM.
- If fine-tuning is required, ensure accurate ground truth or knowledge for the training data. Make sure that you mask or redact any personally identifiable information (PII) or protected health information (PHI).

Real-world medical NLP tasks might differ from pretrained LLMs in terms of knowledge or intended use cases. If a domain-specific LLM does not meet your evaluation benchmarks, you can fine-tune an LLM with your own dataset or you can train a new foundation model. Training a new foundation model is an ambitious, and often expensive, undertaking. For most use cases, we recommend fine-tuning an existing model.

When you use or fine-tune a pretrained medical LLM, it's important to address infrastructure, security, and guardrails.

Infrastructure

Compared to using Amazon Bedrock for on-demand or batch inference, hosting pretrained medical LLMs (commonly from Hugging Face) requires significant resources. To host pretrained medical LLMs, it's common to use an Amazon SageMaker AI image that runs on an Amazon Elastic Compute Cloud (Amazon EC2) instance with one or more GPUs, such as ml.g5 instances for accelerated computing or ml.inf2 instances for AWS Inferentia. This is because LLMs consume a large amount of memory and disk space.

Security and guardrails

Depending on your business compliance requirements, consider using Amazon Comprehend and Amazon Comprehend Medical to mask or redact personally identifiable information (PII) and protected health information (PHI) from training data. This helps prevent the LLM from using confidential data when it generates responses.

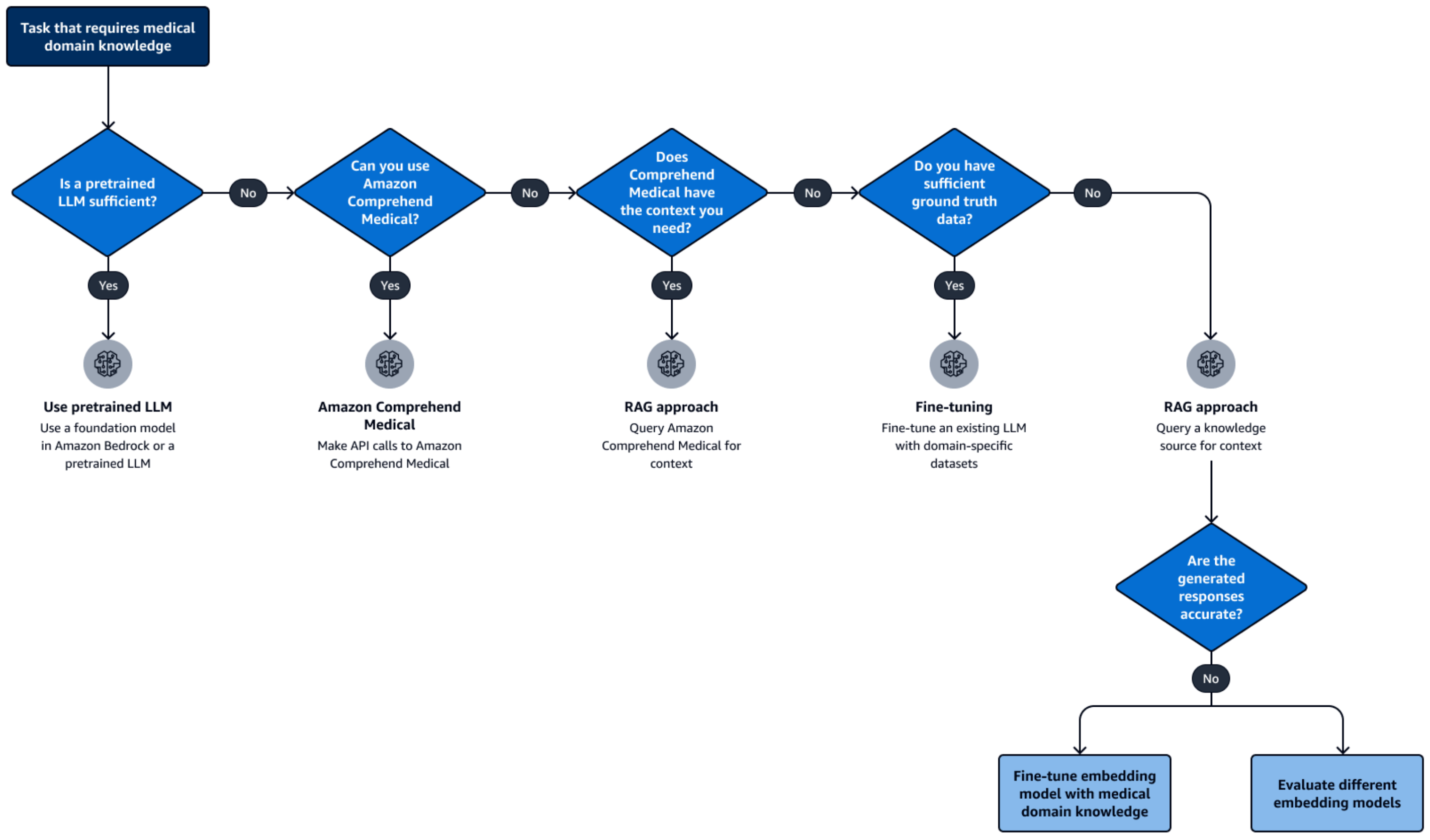
We recommend that you consider and evaluate bias, fairness, and hallucinations in your generative AI applications. Whether you are using a preexisting LLM or fine-tuning one, implement guardrails to prevent harmful responses. *Guardrails* are safeguards that you customize to your generative AI application requirements and responsible AI policies. For example, you can use Amazon Bedrock Guardrails.

Choosing an NLP approach for healthcare and life sciences

The Generative AI and NLP approaches for healthcare and life sciences section describes the following approaches for addressing natural language processing (NLP) tasks for healthcare and life science applications:

- Using Amazon Comprehend Medical
- Combining Amazon Comprehend Medical with an LLM in a Retrieval Augment Generation (RAG) workflow
- Using a fine-tuned LLM
- Using a RAG workflow

By evaluating the known limitations of LLMs for medical domain tasks and your use case, you can choose which approach will work best for your task. The following decision tree can help you choose an LLM approach for your medical NLP task:


Task that requires medical domain knowledge
Is a pretrained LLM sufficient?
No
Can you use Amazon Comprehend Medical?
No
Does Comprehend Medical have the context you need?
No
Do you have sufficient ground truth data?
No
Yes
Yes
Yes
Yes
Use pretrained LLM
Use a foundation model in Amazon Bedrock or a pretrained LLM
Amazon Comprehend Medical
Make API calls to Amazon Comprehend Medical
RAG approach
Query Amazon Comprehend Medical for context
Fine-tuning
Fine-tune an existing LLM with domain-specific datasets
RAG approach
Query a knowledge source for context
Are the generated responses accurate?
No
Fine-tune embedding model with medical domain knowledge
Evaluate different embedding models

The diagram shows the following workflow:

1. For healthcare and life science use cases, identify whether the NLP task requires specific domain knowledge. As needed, coordinate with subject matter experts (SMEs).
2. If you can use a general LLM or a model that has been trained on medical datasets, then use an available foundation model in Amazon Bedrock or the pretrained LLM. For more information, see Choosing an LLM in this guide.
3. If the entity detection and ontology linking capabilities of Amazon Comprehend Medical address your use case, then use the Amazon Comprehend Medical APIs. For more information, see Using Amazon Comprehend Medical in this guide.
4. Sometimes, Amazon Comprehend Medical has the required context but doesn't support your use case. For example, you might need different entity definitions, receive an overwhelming number of results, need custom entities, or need a custom NLP task. If this is the case, use a RAG approach to query Amazon Comprehend Medical for context. For more information, see Combining Amazon Comprehend Medical with large language models in this guide.
5. If you have a sufficient amount of ground truth data, fine-tune an existing LLM. For more information, see Customization approaches in this guide.
6. If the other approaches do not satisfy medical your NLP task objectives, implement a RAG solution. For more information, see Customization approaches in this guide.
7. After implementing the RAG solution, evaluate whether the generated responses are accurate. For more information, see Evaluating LLMs for healthcare and life science applications in this guide. It's common to start with an Amazon Titan Text Embeddings model or a general sentence transformer model, such as all-MiniLM-L6-v2. However, due to a lack of domain context, these models might not capture the medical terminology of the text. If necessary, consider the following adjustments:
   a. Evaluate other embedding models
   b. Fine-tune the embedding model with domain-specific datasets

Business maturity considerations

Business maturity is critical when adapting LLM solutions for healthcare and life science applications. These organizations face varying levels of complexity when implementing LLMs, depending on their acceptance criteria. Frequently, organizations that lack AI/ML resources invest in contractor support to build LLM solutions. In these situations, it's important to understand the following trade-offs:

- **High performance for high cost and maintenance –** You might require a complex solution that involves fine-tuned or custom LLMs to meet stringent performance standards. However, this comes with higher costs and maintenance requirements. You might need to hire specialized resources or partner with contractors to maintain these sophisticated solutions. This can potentially slow development.
- **Good performance for low cost and maintenance –** Alternatively, you might find that services such as Amazon Bedrock or Amazon Comprehend Medical provide acceptable performance. Although these LLMs or approaches might provide perfect results, these solutions can often provide consistent, high-quality results. These solutions are lower cost and reduce the maintenance burden. This can accelerate development.

If a simpler, lower-cost approach consistently delivers high-quality results that meet your acceptance criteria, consider whether the increasing the performance is worth the cost, maintenance, and time tradeoffs. However, if the simpler solution falls significantly short of the target performance, and if your organization lacks the investment capacity for complex solutions and their maintenance requirements, consider postponing AI/ML development until more resources or alternative solutions are available.

In addition, for any medical NLP solution that relies on an LLM, we recommend that you perform continuous monitoring and evaluation. Assess feedback from users over time, and implement periodic assessments to make sure that the solution continues to meet your business objectives.

Evaluating LLMs for healthcare and life science applications

This section provides a comprehensive overview of the requirements and considerations for evaluating large language models (LLMs) in healthcare and life science use cases.

It is important to use ground truth data and SME feedback to mitigate bias and validate the accuracy of the LLM-generated response. This section describes best practices for collecting and curating training and test data. It also helps you implement guardrails and measure data bias and fairness. It also discusses the common medical natural language processing (NLP) tasks, such as text classification, named entity recognition, and text generation, and their associated evaluation metrics.

It also presents workflows for performing LLM evaluation during the training experimentation phase and post-production phase. Model monitoring and LLM operations are important elements of this evaluation process.

Training and test data for medical NLP tasks

Medical NLP tasks commonly use medical corpora (such as PubMed) or patient information (such as clinic patient visit notes) to classify, summarize, and generate insights. Medical personnel, such physicians, health care administrators, or technicians, vary in expertise and viewpoints. Due to subjectivity between these medical personnel, smaller training and test data sets pose a risk of bias. To mitigate this risk, we recommend the following best practices:

- When using a pretrained LLM solution, make sure that you have an adequate amount of test data. The test data should be an exact match or closely resemble the actual medical data. Depending on the task, this can range from 20 to more than 100 records.
- When fine-tuning an LLM, collect a sufficient number of labeled (ground truth) records from a variety of SMEs of the targeted medical domain. A general starting point is at least 100 high-quality records, and we recommend no more than 20 records from each SME. However, given the complexity of the task and your accuracy acceptance criteria, more records might be required.
- If required for your medical use case, implement guardrails and measure data bias and fairness. For example, make sure that the LLM prevents misdiagnosis due to racial profiles of patients. For more information, see the Security and guardrails section in this guide.

Many AI research and development companies, such as Anthropic, have already implemented guardrails in their foundation models to avoid toxicity. You can use toxicity detection to check input prompts and the output responses from LLMs. For more information, see Toxicity detection in the Amazon Comprehend documentation.

In any generative AI task, there is a risk of hallucination. You can mitigate this risk by performing NLP tasks, such as classification. You can also use more advanced techniques, such as text similarity metrics.BertScore is a commonly adopted text similarity metric. For more information about techniques that you can use to mitigate hallucination, see A Comprehensive Survey of Hallucination Mitigation Techniques in Large Language Models.

Metrics for medical NLP tasks

You can create quantifiable metrics after you establish ground truth data and SME-provided labels for training and testing. Checking quality through qualitative processes, such as stress testing and reviewing LLM results, is helpful for quick development. However, metrics act as quantitative benchmarks that support future LLM operations and act as performance benchmarks for each production release.

Understanding the medical task is critical. Metrics typically map to one of the following general NLP tasks:

- **Text classification** – The LLM categorizes the text into one or more predefined categories, based on the input prompt and provided context. An example is classifying a pain category by using a pain scale. Examples of text classification metrics include:
  - Accuracy
  - Precision, also known as *macro precision*
  - Recall, also known as *macro recall*
  - F1 score, also known as *macro F1 score*
  - Hamming loss
- **Named entity recognition (NER)** – Also known as *text extraction*, named entity recognition is the process of locating and classifying named entities that are mentioned in unstructured text into predefined categories. An example is extracting the names of medications from patient records. Examples of NER metrics include:
  - Accuracy
  - Precision

  - Recall
  - F1 score
  - Hamming loss
- **Generation –** The LLM is generates new text by processing the prompt and provided context. Generation includes summarization tasks or question-answering tasks. Examples of generation metrics include:
  - Recall-Oriented Understudy for Gisting Evaluation (ROUGE)
  - Metric for Evaluation of Translation with Explicit ORdering (METEOR)
  - Bilingual evaluation understudy (BLEU) (for translations)
  - String distance, also known as *cosine similarity*

FAQ about healthcare and life science use cases

The following are frequently asked questions related to using Amazon Comprehend Medical or LLMs for medical NLP tasks.

How do I choose between Amazon Comprehend Medical and an LLM?

If your task is to detect medical entities within your medical text, then review the Amazon Comprehend Medical documentation to understand which medical entities can be extracted and if any of the ontologies address your use case. If not, consider using an LLM. For more information, see Use cases for Amazon Comprehend Medical and Use cases for an LLM in this guide.

How can I provide Amazon Comprehend Medical results to an LLM?

You can incorporate Amazon Comprehend Medical results as context within your LLM prompts. This provides additional medical knowledge and terminology to the LLM. The provided context can improve the LLM's performance on tasks such as entity recognition, summarization, or question-answering. The guide provides several examples of how to structure prompts with Amazon Comprehend Medical results. For more information, see Combining Amazon Comprehend Medical with large language models in this guide.

What are some best practices when using Amazon Comprehend Medical with LLMs?

We recommend using the Amazon Comprehend Medical confidence scores to filter or prioritize entities within your prompts. It's also important to evaluate its performance on your specific data and validate that the entity definitions align with your requirements. Combining Amazon Comprehend Medical with domain-specific knowledge sources can further enhance the LLM's performance. For more information, see Best practices for using Amazon Comprehend Medical in a RAG workflow in this guide.

Should I use a pretrained medical LLM or fine-tune a general LLM for my healthcare use case?

The decision depends on your specific requirements and the availability of high-quality training data. Pretrained medical LLMs can provide a good starting point. However, you might still need to fine-tune them with your domain-specific data. If you have sufficient labeled data, fine-tuning a general LLM can be a viable option. For more information, see Choosing an LLM and Choosing an NLP approach for healthcare and life sciences in this guide.

How do I evaluate the performance of LLMs for medical NLP tasks?

We recommend using quantitative metrics, such as accuracy, precision, recall, and F1 score for text classification and named entity recognition tasks. You can use ROUGE and METEOR for text generation tasks. It's important to have reliable ground truth data labeled by subject matter experts and to implement processes for monitoring model performance over time. For more information, see Evaluating LLMs for healthcare and life science applications in this guide.

What are the trade-offs between high-complexity and low-complexity LLM solutions?

Fine-tuning an LLM or building a custom LLM are highly complex solutions. These approaches can improve performance but come with higher costs and maintenance requirements. Simpler solutions, such as using pretrained LLMs or Amazon Comprehend Medical, might provide acceptable performance with lower costs and faster development cycles. However, these approaches might not meet stringent accuracy requirements for some use cases. For more information, see Business maturity considerations in this guide.

Next steps and resources

This guide helps you use AWS services to automate medical NLP and generative AI tasks for real-world applications in production environments. It describes how you can use Amazon Comprehend Medical, supported LLMs in Amazon Bedrock, pretrained medical LLMs, or fine-tuned LLMs to achieve your healthcare and life science business objectives. This guide describes the advantages and limitations for the following approaches:

- Using Amazon Comprehend Medical independently
- Providing Amazon Comprehend Medical results to an LLM
- Using a pretrained general LLM or a medical LLM in a Retrieval Augmented Generation (RAG) approach
- Fine-tuning a general LLM or medical LLM

Use the decision tree and the business maturity considerations in this guide to choose between these approaches based on your organization's AI/ML maturity level. Although Amazon Comprehend Medical and Amazon Bedrock LLMs provide powerful capabilities, they are only successful if you properly implement and evaluate them. Use the evaluation information and metrics described in this guide to validate the performance of your solution.

For next steps, we recommend that healthcare IT managers, architects, and technical leads work with AI/ML practitioners to identify their NLP medical task. Use this guide to choose a development path, and then use the appropriate AWS services and features to successfully implement an automated solution on AWS.

AWS resources

- Amazon Comprehend Medical documentation:
  - Developer Guide
  - API Reference
- Amazon Bedrock model evaluation
- Amazon SageMaker Ground Truth
- Amazon Comprehend toxicity detection
- AWS Healthcare Competency Partners

Other resources

- Open Medical-LLM Leaderboard
- A Survey of Large Language Models for Healthcare: from Data, Technology, and Applications to Accountability and Ethics
- Large Language Models Are Poor Medical Coders — Benchmarking of Medical Code Querying
- From Beginner to Expert: Modeling Medical Knowledge into General LLMs

Document history

The following table describes significant changes to this guide. If you want to be notified about future updates, you can subscribe to an RSS feed.

Change	Description	Date
Initial publication	—	December 16, 2024

AWS Prescriptive Guidance glossary

The following are commonly used terms in strategies, guides, and patterns provided by AWS Prescriptive Guidance. To suggest entries, please use the **Provide feedback** link at the end of the glossary.

Numbers

7 Rs

Seven common migration strategies for moving applications to the cloud. These strategies build upon the 5 Rs that Gartner identified in 2011 and consist of the following:

- Refactor/re-architect – Move an application and modify its architecture by taking full advantage of cloud-native features to improve agility, performance, and scalability. This typically involves porting the operating system and database. Example: Migrate your on-premises Oracle database to the Amazon Aurora PostgreSQL-Compatible Edition.
- Replatform (lift and reshape) – Move an application to the cloud, and introduce some level of optimization to take advantage of cloud capabilities. Example: Migrate your on-premises Oracle database to Amazon Relational Database Service (Amazon RDS) for Oracle in the AWS Cloud.
- Repurchase (drop and shop) – Switch to a different product, typically by moving from a traditional license to a SaaS model. Example: Migrate your customer relationship management (CRM) system to Salesforce.com.
- Rehost (lift and shift) – Move an application to the cloud without making any changes to take advantage of cloud capabilities. Example: Migrate your on-premises Oracle database to Oracle on an EC2 instance in the AWS Cloud.
- Relocate (hypervisor-level lift and shift) – Move infrastructure to the cloud without purchasing new hardware, rewriting applications, or modifying your existing operations. You migrate servers from an on-premises platform to a cloud service for the same platform. Example: Migrate a Microsoft Hyper-V application to AWS.
- Retain (revisit) – Keep applications in your source environment. These might include applications that require major refactoring, and you want to postpone that work until a later time, and legacy applications that you want to retain, because there’s no business justification for migrating them.

- Retire – Decommission or remove applications that are no longer needed in your source environment.

A

ABAC

See attribute-based access control.

abstracted services

See managed services.

ACID

See atomicity, consistency, isolation, durability.

active-active migration

A database migration method in which the source and target databases are kept in sync (by using a bidirectional replication tool or dual write operations), and both databases handle transactions from connecting applications during migration. This method supports migration in small, controlled batches instead of requiring a one-time cutover. It's more flexible but requires more work than active-passive migration.

active-passive migration

A database migration method in which in which the source and target databases are kept in sync, but only the source database handles transactions from connecting applications while data is replicated to the target database. The target database doesn't accept any transactions during migration.

aggregate function

A SQL function that operates on a group of rows and calculates a single return value for the group. Examples of aggregate functions include `SUM` and `MAX`.

AI

See artificial intelligence.

AIOps

See artificial intelligence operations.

anonymization

> The process of permanently deleting personal information in a dataset. Anonymization can help protect personal privacy. Anonymized data is no longer considered to be personal data.

anti-pattern

> A frequently used solution for a recurring issue where the solution is counter-productive, ineffective, or less effective than an alternative.

application control

> A security approach that allows the use of only approved applications in order to help protect a system from malware.

application portfolio

> A collection of detailed information about each application used by an organization, including the cost to build and maintain the application, and its business value. This information is key to the portfolio discovery and analysis process and helps identify and prioritize the applications to be migrated, modernized, and optimized.

artificial intelligence (AI)

> The field of computer science that is dedicated to using computing technologies to perform cognitive functions that are typically associated with humans, such as learning, solving problems, and recognizing patterns. For more information, see What is Artificial Intelligence?

artificial intelligence operations (AIOps)

> The process of using machine learning techniques to solve operational problems, reduce operational incidents and human intervention, and increase service quality. For more information about how AIOps is used in the AWS migration strategy, see the operations integration guide.

asymmetric encryption

> An encryption algorithm that uses a pair of keys, a public key for encryption and a private key for decryption. You can share the public key because it isn't used for decryption, but access to the private key should be highly restricted.

atomicity, consistency, isolation, durability (ACID)

> A set of software properties that guarantee the data validity and operational reliability of a database, even in the case of errors, power failures, or other problems.

attribute-based access control (ABAC)

The practice of creating fine-grained permissions based on user attributes, such as department, job role, and team name. For more information, see ABAC for AWS in the AWS Identity and Access Management (IAM) documentation.

authoritative data source

A location where you store the primary version of data, which is considered to be the most reliable source of information. You can copy data from the authoritative data source to other locations for the purposes of processing or modifying the data, such as anonymizing, redacting, or pseudonymizing it.

Availability Zone

A distinct location within an AWS Region that is insulated from failures in other Availability Zones and provides inexpensive, low-latency network connectivity to other Availability Zones in the same Region.

AWS Cloud Adoption Framework (AWS CAF)

A framework of guidelines and best practices from AWS to help organizations develop an efficient and effective plan to move successfully to the cloud. AWS CAF organizes guidance into six focus areas called perspectives: business, people, governance, platform, security, and operations. The business, people, and governance perspectives focus on business skills and processes; the platform, security, and operations perspectives focus on technical skills and processes. For example, the people perspective targets stakeholders who handle human resources (HR), staffing functions, and people management. For this perspective, AWS CAF provides guidance for people development, training, and communications to help ready the organization for successful cloud adoption. For more information, see the AWS CAF website and the AWS CAF whitepaper.

AWS Workload Qualification Framework (AWS WQF)

A tool that evaluates database migration workloads, recommends migration strategies, and provides work estimates. AWS WQF is included with AWS Schema Conversion Tool (AWS SCT). It analyzes database schemas and code objects, application code, dependencies, and performance characteristics, and provides assessment reports.

B

bad bot

A bot that is intended to disrupt or cause harm to individuals or organizations.

BCP

See business continuity planning.

behavior graph

A unified, interactive view of resource behavior and interactions over time. You can use a behavior graph with Amazon Detective to examine failed logon attempts, suspicious API calls, and similar actions. For more information, see Data in a behavior graph in the Detective documentation.

big-endian system

A system that stores the most significant byte first. See also endianness.

binary classification

A process that predicts a binary outcome (one of two possible classes). For example, your ML model might need to predict problems such as “Is this email spam or not spam?" or "Is this product a book or a car?"

bloom filter

A probabilistic, memory-efficient data structure that is used to test whether an element is a member of a set.

blue/green deployment

A deployment strategy where you create two separate but identical environments. You run the current application version in one environment (blue) and the new application version in the other environment (green). This strategy helps you quickly roll back with minimal impact.

bot

A software application that runs automated tasks over the internet and simulates human activity or interaction. Some bots are useful or beneficial, such as web crawlers that index information on the internet. Some other bots, known as *bad bots*, are intended to disrupt or cause harm to individuals or organizations.

botnet

Networks of bots that are infected by malware and are under the control of a single party, known as a *bot herder* or *bot operator*. Botnets are the best-known mechanism to scale bots and their impact.

branch

A contained area of a code repository. The first branch created in a repository is the *main branch*. You can create a new branch from an existing branch, and you can then develop features or fix bugs in the new branch. A branch you create to build a feature is commonly referred to as a *feature branch*. When the feature is ready for release, you merge the feature branch back into the main branch. For more information, see About branches (GitHub documentation).

break-glass access

In exceptional circumstances and through an approved process, a quick means for a user to gain access to an AWS account that they don't typically have permissions to access. For more information, see the Implement break-glass procedures indicator in the AWS Well-Architected guidance.

brownfield strategy

The existing infrastructure in your environment. When adopting a brownfield strategy for a system architecture, you design the architecture around the constraints of the current systems and infrastructure. If you are expanding the existing infrastructure, you might blend brownfield and greenfield strategies.

buffer cache

The memory area where the most frequently accessed data is stored.

business capability

What a business does to generate value (for example, sales, customer service, or marketing). Microservices architectures and development decisions can be driven by business capabilities. For more information, see the Organized around business capabilities section of the Running containerized microservices on AWS whitepaper.

business continuity planning (BCP)

A plan that addresses the potential impact of a disruptive event, such as a large-scale migration, on operations and enables a business to resume operations quickly.

C

CAF

See AWS Cloud Adoption Framework.

canary deployment

The slow and incremental release of a version to end users. When you are confident, you deploy the new version and replace the current version in its entirety.

CCoE

See Cloud Center of Excellence.

CDC

See change data capture.

change data capture (CDC)

The process of tracking changes to a data source, such as a database table, and recording metadata about the change. You can use CDC for various purposes, such as auditing or replicating changes in a target system to maintain synchronization.

chaos engineering

Intentionally introducing failures or disruptive events to test a system's resilience. You can use AWS Fault Injection Service (AWS FIS) to perform experiments that stress your AWS workloads and evaluate their response.

CI/CD

See continuous integration and continuous delivery.

classification

A categorization process that helps generate predictions. ML models for classification problems predict a discrete value. Discrete values are always distinct from one another. For example, a model might need to evaluate whether or not there is a car in an image.

client-side encryption

Encryption of data locally, before the target AWS service receives it.

Cloud Center of Excellence (CCoE)

A multi-disciplinary team that drives cloud adoption efforts across an organization, including developing cloud best practices, mobilizing resources, establishing migration timelines, and leading the organization through large-scale transformations. For more information, see the CCoE posts on the AWS Cloud Enterprise Strategy Blog.

cloud computing

The cloud technology that is typically used for remote data storage and IoT device management. Cloud computing is commonly connected to edge computing technology.

cloud operating model

In an IT organization, the operating model that is used to build, mature, and optimize one or more cloud environments. For more information, see Building your Cloud Operating Model.

cloud stages of adoption

The four phases that organizations typically go through when they migrate to the AWS Cloud:

- Project – Running a few cloud-related projects for proof of concept and learning purposes
- Foundation – Making foundational investments to scale your cloud adoption (e.g., creating a landing zone, defining a CCoE, establishing an operations model)
- Migration – Migrating individual applications
- Re-invention – Optimizing products and services, and innovating in the cloud

These stages were defined by Stephen Orban in the blog post The Journey Toward Cloud-First & the Stages of Adoption on the AWS Cloud Enterprise Strategy blog. For information about how they relate to the AWS migration strategy, see the migration readiness guide.

CMDB

See configuration management database.

code repository

A location where source code and other assets, such as documentation, samples, and scripts, are stored and updated through version control processes. Common cloud repositories include GitHub or Bitbucket Cloud. Each version of the code is called a *branch*. In a microservice structure, each repository is devoted to a single piece of functionality. A single CI/CD pipeline can use multiple repositories.

cold cache

A buffer cache that is empty, not well populated, or contains stale or irrelevant data. This affects performance because the database instance must read from the main memory or disk, which is slower than reading from the buffer cache.

cold data

Data that is rarely accessed and is typically historical. When querying this kind of data, slow queries are typically acceptable. Moving this data to lower-performing and less expensive storage tiers or classes can reduce costs.

computer vision (CV)

A field of AI that uses machine learning to analyze and extract information from visual formats such as digital images and videos. For example, Amazon SageMaker AI provides image processing algorithms for CV.

configuration drift

For a workload, a configuration change from the expected state. It might cause the workload to become noncompliant, and it's typically gradual and unintentional.

configuration management database (CMDB)

A repository that stores and manages information about a database and its IT environment, including both hardware and software components and their configurations. You typically use data from a CMDB in the portfolio discovery and analysis stage of migration.

conformance pack

A collection of AWS Config rules and remediation actions that you can assemble to customize your compliance and security checks. You can deploy a conformance pack as a single entity in an AWS account and Region, or across an organization, by using a YAML template. For more information, see Conformance packs in the AWS Config documentation.

continuous integration and continuous delivery (CI/CD)

The process of automating the source, build, test, staging, and production stages of the software release process. CI/CD is commonly described as a pipeline. CI/CD can help you automate processes, improve productivity, improve code quality, and deliver faster. For more information, see Benefits of continuous delivery. CD can also stand for *continuous deployment*. For more information, see Continuous Delivery vs. Continuous Deployment.

CV

See computer vision.

D

data at rest

Data that is stationary in your network, such as data that is in storage.

data classification

A process for identifying and categorizing the data in your network based on its criticality and sensitivity. It is a critical component of any cybersecurity risk management strategy because it helps you determine the appropriate protection and retention controls for the data. Data classification is a component of the security pillar in the AWS Well-Architected Framework. For more information, see Data classification.

data drift

A meaningful variation between the production data and the data that was used to train an ML model, or a meaningful change in the input data over time. Data drift can reduce the overall quality, accuracy, and fairness in ML model predictions.

data in transit

Data that is actively moving through your network, such as between network resources.

data mesh

An architectural framework that provides distributed, decentralized data ownership with centralized management and governance.

data minimization

The principle of collecting and processing only the data that is strictly necessary. Practicing data minimization in the AWS Cloud can reduce privacy risks, costs, and your analytics carbon footprint.

data perimeter

A set of preventive guardrails in your AWS environment that help make sure that only trusted identities are accessing trusted resources from expected networks. For more information, see Building a data perimeter on AWS.

data preprocessing

To transform raw data into a format that is easily parsed by your ML model. Preprocessing data can mean removing certain columns or rows and addressing missing, inconsistent, or duplicate values.

data provenance

The process of tracking the origin and history of data throughout its lifecycle, such as how the data was generated, transmitted, and stored.

data subject

An individual whose data is being collected and processed.

data warehouse

A data management system that supports business intelligence, such as analytics. Data warehouses commonly contain large amounts of historical data, and they are typically used for queries and analysis.

database definition language (DDL)

Statements or commands for creating or modifying the structure of tables and objects in a database.

database manipulation language (DML)

Statements or commands for modifying (inserting, updating, and deleting) information in a database.

DDL

See database definition language.

deep ensemble

To combine multiple deep learning models for prediction. You can use deep ensembles to obtain a more accurate prediction or for estimating uncertainty in predictions.

deep learning

An ML subfield that uses multiple layers of artificial neural networks to identify mapping between input data and target variables of interest.

defense-in-depth

An information security approach in which a series of security mechanisms and controls are thoughtfully layered throughout a computer network to protect the confidentiality, integrity, and availability of the network and the data within. When you adopt this strategy on AWS, you add multiple controls at different layers of the AWS Organizations structure to help secure resources. For example, a defense-in-depth approach might combine multi-factor authentication, network segmentation, and encryption.

delegated administrator

In AWS Organizations, a compatible service can register an AWS member account to administer the organization's accounts and manage permissions for that service. This account is called the *delegated administrator* for that service. For more information and a list of compatible services, see Services that work with AWS Organizations in the AWS Organizations documentation.

deployment

The process of making an application, new features, or code fixes available in the target environment. Deployment involves implementing changes in a code base and then building and running that code base in the application's environments.

development environment

See environment.

detective control

A security control that is designed to detect, log, and alert after an event has occurred. These controls are a second line of defense, alerting you to security events that bypassed the preventative controls in place. For more information, see Detective controls in *Implementing security controls on AWS*.

development value stream mapping (DVSM)

A process used to identify and prioritize constraints that adversely affect speed and quality in a software development lifecycle. DVSM extends the value stream mapping process originally designed for lean manufacturing practices. It focuses on the steps and teams required to create and move value through the software development process.

digital twin

A virtual representation of a real-world system, such as a building, factory, industrial equipment, or production line. Digital twins support predictive maintenance, remote monitoring, and production optimization.

dimension table

In a star schema, a smaller table that contains data attributes about quantitative data in a fact table. Dimension table attributes are typically text fields or discrete numbers that behave like text. These attributes are commonly used for query constraining, filtering, and result set labeling.

disaster

An event that prevents a workload or system from fulfilling its business objectives in its primary deployed location. These events can be natural disasters, technical failures, or the result of human actions, such as unintentional misconfiguration or a malware attack.

disaster recovery (DR)

The strategy and process you use to minimize downtime and data loss caused by a disaster. For more information, see Disaster Recovery of Workloads on AWS: Recovery in the Cloud in the AWS Well-Architected Framework.

DML

See database manipulation language.

domain-driven design

An approach to developing a complex software system by connecting its components to evolving domains, or core business goals, that each component serves. This concept was introduced by Eric Evans in his book, *Domain-Driven Design: Tackling Complexity in the Heart of Software* (Boston: Addison-Wesley Professional, 2003). For information about how you can use domain-driven design with the strangler fig pattern, see Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

DR

See disaster recovery.

drift detection

Tracking deviations from a baselined configuration. For example, you can use AWS CloudFormation to detect drift in system resources, or you can use AWS Control Tower to detect changes in your landing zone that might affect compliance with governance requirements.

DVSM

See development value stream mapping.

E

EDA

See exploratory data analysis.

EDI

See electronic data interchange.

edge computing

The technology that increases the computing power for smart devices at the edges of an IoT network. When compared with cloud computing, edge computing can reduce communication latency and improve response time.

electronic data interchange (EDI)

The automated exchange of business documents between organizations. For more information, see What is Electronic Data Interchange.

encryption

A computing process that transforms plaintext data, which is human-readable, into ciphertext.

encryption key

A cryptographic string of randomized bits that is generated by an encryption algorithm. Keys can vary in length, and each key is designed to be unpredictable and unique.

endianness

The order in which bytes are stored in computer memory. Big-endian systems store the most significant byte first. Little-endian systems store the least significant byte first.

endpoint

See service endpoint.

endpoint service

A service that you can host in a virtual private cloud (VPC) to share with other users. You can create an endpoint service with AWS PrivateLink and grant permissions to other AWS accounts or to AWS Identity and Access Management (IAM) principals. These accounts or principals can connect to your endpoint service privately by creating interface VPC endpoints. For more information, see Create an endpoint service in the Amazon Virtual Private Cloud (Amazon VPC) documentation.

enterprise resource planning (ERP)

A system that automates and manages key business processes (such as accounting, MES, and project management) for an enterprise.

envelope encryption

The process of encrypting an encryption key with another encryption key. For more information, see Envelope encryption in the AWS Key Management Service (AWS KMS) documentation.

environment

An instance of a running application. The following are common types of environments in cloud computing:

- development environment – An instance of a running application that is available only to the core team responsible for maintaining the application. Development environments are used to test changes before promoting them to upper environments. This type of environment is sometimes referred to as a *test environment*.
- lower environments – All development environments for an application, such as those used for initial builds and tests.
- production environment – An instance of a running application that end users can access. In a CI/CD pipeline, the production environment is the last deployment environment.
- upper environments – All environments that can be accessed by users other than the core development team. This can include a production environment, preproduction environments, and environments for user acceptance testing.

epic

In agile methodologies, functional categories that help organize and prioritize your work. Epics provide a high-level description of requirements and implementation tasks. For example, AWS CAF security epics include identity and access management, detective controls, infrastructure security, data protection, and incident response. For more information about epics in the AWS migration strategy, see the program implementation guide.

ERP

See enterprise resource planning.

exploratory data analysis (EDA)

The process of analyzing a dataset to understand its main characteristics. You collect or aggregate data and then perform initial investigations to find patterns, detect anomalies, and check assumptions. EDA is performed by calculating summary statistics and creating data visualizations.

F

fact table

The central table in a star schema. It stores quantitative data about business operations. Typically, a fact table contains two types of columns: those that contain measures and those that contain a foreign key to a dimension table.

fail fast

A philosophy that uses frequent and incremental testing to reduce the development lifecycle. It is a critical part of an agile approach.

fault isolation boundary

In the AWS Cloud, a boundary such as an Availability Zone, AWS Region, control plane, or data plane that limits the effect of a failure and helps improve the resilience of workloads. For more information, see AWS Fault Isolation Boundaries.

feature branch

See branch.

features

The input data that you use to make a prediction. For example, in a manufacturing context, features could be images that are periodically captured from the manufacturing line.

feature importance

How significant a feature is for a model's predictions. This is usually expressed as a numerical score that can be calculated through various techniques, such as Shapley Additive Explanations (SHAP) and integrated gradients. For more information, see Machine learning model interpretability with AWS.

feature transformation

To optimize data for the ML process, including enriching data with additional sources, scaling values, or extracting multiple sets of information from a single data field. This enables the ML model to benefit from the data. For example, if you break down the "2021-05-27 00:15:37" date into "2021", "May", "Thu", and "15", you can help the learning algorithm learn nuanced patterns associated with different data components.

few-shot prompting

Providing an LLM with a small number of examples that demonstrate the task and desired output before asking it to perform a similar task. This technique is an application of in-context learning, where models learn from examples (*shots*) that are embedded in prompts. Few-shot prompting can be effective for tasks that require specific formatting, reasoning, or domain knowledge. See also zero-shot prompting.

FGAC

See fine-grained access control.

fine-grained access control (FGAC)

The use of multiple conditions to allow or deny an access request.

flash-cut migration

A database migration method that uses continuous data replication through change data capture to migrate data in the shortest time possible, instead of using a phased approach. The objective is to keep downtime to a minimum.

FM

See foundation model.

foundation model (FM)

A large deep-learning neural network that has been training on massive datasets of generalized and unlabeled data. FMs are capable of performing a wide variety of general tasks, such as understanding language, generating text and images, and conversing in natural language. For more information, see What are Foundation Models.

G

generative AI

A subset of AI models that have been trained on large amounts of data and that can use a simple text prompt to create new content and artifacts, such as images, videos, text, and audio. For more information, see What is Generative AI.

geo blocking

See geographic restrictions.

geographic restrictions (geo blocking)

In Amazon CloudFront, an option to prevent users in specific countries from accessing content distributions. You can use an allow list or block list to specify approved and banned countries. For more information, see Restricting the geographic distribution of your content in the CloudFront documentation.

Gitflow workflow

An approach in which lower and upper environments use different branches in a source code repository. The Gitflow workflow is considered legacy, and the trunk-based workflow is the modern, preferred approach.

golden image

A snapshot of a system or software that is used as a template to deploy new instances of that system or software. For example, in manufacturing, a golden image can be used to provision software on multiple devices and helps improve speed, scalability, and productivity in device manufacturing operations.

greenfield strategy

The absence of existing infrastructure in a new environment. When adopting a greenfield strategy for a system architecture, you can select all new technologies without the restriction of compatibility with existing infrastructure, also known as brownfield. If you are expanding the existing infrastructure, you might blend brownfield and greenfield strategies.

guardrail

A high-level rule that helps govern resources, policies, and compliance across organizational units (OUs). *Preventive guardrails* enforce policies to ensure alignment to compliance standards. They are implemented by using service control policies and IAM permissions boundaries. *Detective guardrails* detect policy violations and compliance issues, and generate alerts for remediation. They are implemented by using AWS Config, AWS Security Hub, Amazon GuardDuty, AWS Trusted Advisor, Amazon Inspector, and custom AWS Lambda checks.

H

HA

See high availability.

heterogeneous database migration

Migrating your source database to a target database that uses a different database engine (for example, Oracle to Amazon Aurora). Heterogeneous migration is typically part of a re-architecting effort, and converting the schema can be a complex task. AWS provides AWS SCT that helps with schema conversions.

high availability (HA)

The ability of a workload to operate continuously, without intervention, in the event of challenges or disasters. HA systems are designed to automatically fail over, consistently deliver high-quality performance, and handle different loads and failures with minimal performance impact.

historian modernization

An approach used to modernize and upgrade operational technology (OT) systems to better serve the needs of the manufacturing industry. A *historian* is a type of database that is used to collect and store data from various sources in a factory.

holdout data

A portion of historical, labeled data that is withheld from a dataset that is used to train a machine learning model. You can use holdout data to evaluate the model performance by comparing the model predictions against the holdout data.

homogeneous database migration

Migrating your source database to a target database that shares the same database engine (for example, Microsoft SQL Server to Amazon RDS for SQL Server). Homogeneous migration is typically part of a rehosting or replatforming effort. You can use native database utilities to migrate the schema.

hot data

Data that is frequently accessed, such as real-time data or recent translational data. This data typically requires a high-performance storage tier or class to provide fast query responses.

hotfix

An urgent fix for a critical issue in a production environment. Due to its urgency, a hotfix is usually made outside of the typical DevOps release workflow.

hypercare period

Immediately following cutover, the period of time when a migration team manages and monitors the migrated applications in the cloud in order to address any issues. Typically, this period is 1–4 days in length. At the end of the hypercare period, the migration team typically transfers responsibility for the applications to the cloud operations team.

I

IaC

See infrastructure as code.

identity-based policy

A policy attached to one or more IAM principals that defines their permissions within the AWS Cloud environment.

idle application

An application that has an average CPU and memory usage between 5 and 20 percent over a period of 90 days. In a migration project, it is common to retire these applications or retain them on premises.

IIoT

See industrial Internet of Things.

immutable infrastructure

A model that deploys new infrastructure for production workloads instead of updating, patching, or modifying the existing infrastructure. Immutable infrastructures are inherently more consistent, reliable, and predictable than mutable infrastructure. For more information, see the Deploy using immutable infrastructure best practice in the AWS Well-Architected Framework.

inbound (ingress) VPC

In an AWS multi-account architecture, a VPC that accepts, inspects, and routes network connections from outside an application. The AWS Security Reference Architecture recommends setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

incremental migration

A cutover strategy in which you migrate your application in small parts instead of performing a single, full cutover. For example, you might move only a few microservices or users to the new system initially. After you verify that everything is working properly, you can incrementally move additional microservices or users until you can decommission your legacy system. This strategy reduces the risks associated with large migrations.

Industry 4.0

A term that was introduced by Klaus Schwab in 2016 to refer to the modernization of manufacturing processes through advances in connectivity, real-time data, automation, analytics, and AI/ML.

infrastructure

All of the resources and assets contained within an application’s environment.

infrastructure as code (IaC)

The process of provisioning and managing an application's infrastructure through a set of configuration files. IaC is designed to help you centralize infrastructure management, standardize resources, and scale quickly so that new environments are repeatable, reliable, and consistent.

industrial Internet of Things (IIoT)

The use of internet-connected sensors and devices in the industrial sectors, such as manufacturing, energy, automotive, healthcare, life sciences, and agriculture. For more information, see Building an industrial Internet of Things (IIoT) digital transformation strategy.

inspection VPC

In an AWS multi-account architecture, a centralized VPC that manages inspections of network traffic between VPCs (in the same or different AWS Regions), the internet, and on-premises networks. The AWS Security Reference Architecture recommends setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

Internet of Things (IoT)

The network of connected physical objects with embedded sensors or processors that communicate with other devices and systems through the internet or over a local communication network. For more information, see What is IoT?

interpretability

A characteristic of a machine learning model that describes the degree to which a human can understand how the model's predictions depend on its inputs. For more information, see Machine learning model interpretability with AWS.

IoT

See Internet of Things.

IT information library (ITIL)

A set of best practices for delivering IT services and aligning these services with business requirements. ITIL provides the foundation for ITSM.

IT service management (ITSM)

Activities associated with designing, implementing, managing, and supporting IT services for an organization. For information about integrating cloud operations with ITSM tools, see the operations integration guide.

ITIL

See IT information library.

ITSM

See IT service management.

L

label-based access control (LBAC)

An implementation of mandatory access control (MAC) where the users and the data itself are each explicitly assigned a security label value. The intersection between the user security label and data security label determines which rows and columns can be seen by the user.

landing zone

A landing zone is a well-architected, multi-account AWS environment that is scalable and secure. This is a starting point from which your organizations can quickly launch and deploy workloads and applications with confidence in their security and infrastructure environment. For more information about landing zones, see Setting up a secure and scalable multi-account AWS environment.

large language model (LLM)

A deep learning AI model that is pretrained on a vast amount of data. An LLM can perform multiple tasks, such as answering questions, summarizing documents, translating text into other languages, and completing sentences. For more information, see What are LLMs.

large migration

A migration of 300 or more servers.

LBAC

See label-based access control.

least privilege

The security best practice of granting the minimum permissions required to perform a task. For more information, see Apply least-privilege permissions in the IAM documentation.

lift and shift

See 7 Rs.

little-endian system

A system that stores the least significant byte first. See also endianness.

LLM

See large language model.

lower environments

See environment.

M

machine learning (ML)

A type of artificial intelligence that uses algorithms and techniques for pattern recognition and learning. ML analyzes and learns from recorded data, such as Internet of Things (IoT) data, to generate a statistical model based on patterns. For more information, see Machine Learning.

main branch

See branch.

malware

Software that is designed to compromise computer security or privacy. Malware might disrupt computer systems, leak sensitive information, or gain unauthorized access. Examples of malware include viruses, worms, ransomware, Trojan horses, spyware, and keyloggers.

managed services

AWS services for which AWS operates the infrastructure layer, the operating system, and platforms, and you access the endpoints to store and retrieve data. Amazon Simple Storage Service (Amazon S3) and Amazon DynamoDB are examples of managed services. These are also known as *abstracted services*.

manufacturing execution system (MES)

A software system for tracking, monitoring, documenting, and controlling production processes that convert raw materials to finished products on the shop floor.

MAP

See Migration Acceleration Program.

mechanism

A complete process in which you create a tool, drive adoption of the tool, and then inspect the results in order to make adjustments. A mechanism is a cycle that reinforces and improves itself as it operates. For more information, see Building mechanisms in the AWS Well-Architected Framework.

member account

All AWS accounts other than the management account that are part of an organization in AWS Organizations. An account can be a member of only one organization at a time.

MES

See manufacturing execution system.

Message Queuing Telemetry Transport (MQTT)

A lightweight, machine-to-machine (M2M) communication protocol, based on the publish/subscribe pattern, for resource-constrained IoT devices.

microservice

A small, independent service that communicates over well-defined APIs and is typically owned by small, self-contained teams. For example, an insurance system might include microservices that map to business capabilities, such as sales or marketing, or subdomains, such as purchasing, claims, or analytics. The benefits of microservices include agility, flexible scaling, easy deployment, reusable code, and resilience. For more information, see Integrating microservices by using AWS serverless services.

microservices architecture

An approach to building an application with independent components that run each application process as a microservice. These microservices communicate through a well-defined interface by using lightweight APIs. Each microservice in this architecture can be updated, deployed, and scaled to meet demand for specific functions of an application. For more information, see Implementing microservices on AWS.

Migration Acceleration Program (MAP)

An AWS program that provides consulting support, training, and services to help organizations build a strong operational foundation for moving to the cloud, and to help offset the initial cost of migrations. MAP includes a migration methodology for executing legacy migrations in a methodical way and a set of tools to automate and accelerate common migration scenarios.

migration at scale

The process of moving the majority of the application portfolio to the cloud in waves, with more applications moved at a faster rate in each wave. This phase uses the best practices and lessons learned from the earlier phases to implement a *migration factory* of teams, tools, and processes to streamline the migration of workloads through automation and agile delivery. This is the third phase of the AWS migration strategy.

migration factory

Cross-functional teams that streamline the migration of workloads through automated, agile approaches. Migration factory teams typically include operations, business analysts and owners, migration engineers, developers, and DevOps professionals working in sprints. Between 20 and 50 percent of an enterprise application portfolio consists of repeated patterns that can be optimized by a factory approach. For more information, see the discussion of migration factories and the Cloud Migration Factory guide in this content set.

migration metadata

The information about the application and server that is needed to complete the migration. Each migration pattern requires a different set of migration metadata. Examples of migration metadata include the target subnet, security group, and AWS account.

migration pattern

A repeatable migration task that details the migration strategy, the migration destination, and the migration application or service used. Example: Rehost migration to Amazon EC2 with AWS Application Migration Service.

Migration Portfolio Assessment (MPA)

An online tool that provides information for validating the business case for migrating to the AWS Cloud. MPA provides detailed portfolio assessment (server right-sizing, pricing, TCO comparisons, migration cost analysis) as well as migration planning (application data analysis and data collection, application grouping, migration prioritization, and wave planning). The MPA tool (requires login) is available free of charge to all AWS consultants and APN Partner consultants.

Migration Readiness Assessment (MRA)

The process of gaining insights about an organization's cloud readiness status, identifying strengths and weaknesses, and building an action plan to close identified gaps, using the AWS CAF. For more information, see the migration readiness guide. MRA is the first phase of the AWS migration strategy.

migration strategy

The approach used to migrate a workload to the AWS Cloud. For more information, see the 7 Rs entry in this glossary and see Mobilize your organization to accelerate large-scale migrations.

ML

See machine learning.

modernization

Transforming an outdated (legacy or monolithic) application and its infrastructure into an agile, elastic, and highly available system in the cloud to reduce costs, gain efficiencies, and take advantage of innovations. For more information, see Strategy for modernizing applications in the AWS Cloud.

modernization readiness assessment

An evaluation that helps determine the modernization readiness of an organization's applications; identifies benefits, risks, and dependencies; and determines how well the organization can support the future state of those applications. The outcome of the assessment is a blueprint of the target architecture, a roadmap that details development phases and milestones for the modernization process, and an action plan for addressing identified gaps. For more information, see Evaluating modernization readiness for applications in the AWS Cloud.

monolithic applications (monoliths)

Applications that run as a single service with tightly coupled processes. Monolithic applications have several drawbacks. If one application feature experiences a spike in demand, the entire architecture must be scaled. Adding or improving a monolithic application's features also becomes more complex when the code base grows. To address these issues, you can use a microservices architecture. For more information, see Decomposing monoliths into microservices.

MPA

See Migration Portfolio Assessment.

MQTT

See Message Queuing Telemetry Transport.

multiclass classification

A process that helps generate predictions for multiple classes (predicting one of more than two outcomes). For example, an ML model might ask "Is this product a book, car, or phone?" or "Which product category is most interesting to this customer?"

mutable infrastructure

A model that updates and modifies the existing infrastructure for production workloads. For improved consistency, reliability, and predictability, the AWS Well-Architected Framework recommends the use of immutable infrastructure as a best practice.

O

OAC

See origin access control.

OAI

See origin access identity.

OCM

See organizational change management.

offline migration

A migration method in which the source workload is taken down during the migration process. This method involves extended downtime and is typically used for small, non-critical workloads.

OI

See operations integration.

OLA

See operational-level agreement.

online migration

A migration method in which the source workload is copied to the target system without being taken offline. Applications that are connected to the workload can continue to function during the migration. This method involves zero to minimal downtime and is typically used for critical production workloads.

OPC-UA

See Open Process Communications - Unified Architecture.

Open Process Communications - Unified Architecture (OPC-UA)

A machine-to-machine (M2M) communication protocol for industrial automation. OPC-UA provides an interoperability standard with data encryption, authentication, and authorization schemes.

operational-level agreement (OLA)

An agreement that clarifies what functional IT groups promise to deliver to each other, to support a service-level agreement (SLA).

operational readiness review (ORR)

A checklist of questions and associated best practices that help you understand, evaluate, prevent, or reduce the scope of incidents and possible failures. For more information, see Operational Readiness Reviews (ORR) in the AWS Well-Architected Framework.

operational technology (OT)

Hardware and software systems that work with the physical environment to control industrial operations, equipment, and infrastructure. In manufacturing, the integration of OT and information technology (IT) systems is a key focus for Industry 4.0 transformations.

operations integration (OI)

The process of modernizing operations in the cloud, which involves readiness planning, automation, and integration. For more information, see the operations integration guide.

organization trail

A trail that's created by AWS CloudTrail that logs all events for all AWS accounts in an organization in AWS Organizations. This trail is created in each AWS account that's part of the organization and tracks the activity in each account. For more information, see Creating a trail for an organization in the CloudTrail documentation.

organizational change management (OCM)

A framework for managing major, disruptive business transformations from a people, culture, and leadership perspective. OCM helps organizations prepare for, and transition to, new systems and strategies by accelerating change adoption, addressing transitional issues, and driving cultural and organizational changes. In the AWS migration strategy, this framework is called *people acceleration*, because of the speed of change required in cloud adoption projects. For more information, see the OCM guide.

origin access control (OAC)

In CloudFront, an enhanced option for restricting access to secure your Amazon Simple Storage Service (Amazon S3) content. OAC supports all S3 buckets in all AWS Regions, server-side encryption with AWS KMS (SSE-KMS), and dynamic `PUT` and `DELETE` requests to the S3 bucket.

origin access identity (OAI)

In CloudFront, an option for restricting access to secure your Amazon S3 content. When you use OAI, CloudFront creates a principal that Amazon S3 can authenticate with. Authenticated principals can access content in an S3 bucket only through a specific CloudFront distribution. See also OAC, which provides more granular and enhanced access control.

ORR

See operational readiness review.

OT

See operational technology.

outbound (egress) VPC

In an AWS multi-account architecture, a VPC that handles network connections that are initiated from within an application. The AWS Security Reference Architecture recommends setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

P

permissions boundary

An IAM management policy that is attached to IAM principals to set the maximum permissions that the user or role can have. For more information, see Permissions boundaries in the IAM documentation.

personally identifiable information (PII)

Information that, when viewed directly or paired with other related data, can be used to reasonably infer the identity of an individual. Examples of PII include names, addresses, and contact information.

PII

See personally identifiable information.

playbook

A set of predefined steps that capture the work associated with migrations, such as delivering core operations functions in the cloud. A playbook can take the form of scripts, automated runbooks, or a summary of processes or steps required to operate your modernized environment.

PLC

See programmable logic controller.

PLM

See product lifecycle management.

policy

An object that can define permissions (see identity-based policy), specify access conditions (see resource-based policy), or define the maximum permissions for all accounts in an organization in AWS Organizations (see service control policy).

polyglot persistence

Independently choosing a microservice's data storage technology based on data access patterns and other requirements. If your microservices have the same data storage technology, they can encounter implementation challenges or experience poor performance. Microservices are more easily implemented and achieve better performance and scalability if they use the data store best adapted to their requirements. For more information, see Enabling data persistence in microservices.

portfolio assessment

A process of discovering, analyzing, and prioritizing the application portfolio in order to plan the migration. For more information, see Evaluating migration readiness.

predicate

A query condition that returns `true` or `false`, commonly located in a `WHERE` clause.

predicate pushdown

A database query optimization technique that filters the data in the query before transfer. This reduces the amount of data that must be retrieved and processed from the relational database, and it improves query performance.

preventative control

A security control that is designed to prevent an event from occurring. These controls are a first line of defense to help prevent unauthorized access or unwanted changes to your network. For more information, see Preventative controls in *Implementing security controls on AWS*.

principal

An entity in AWS that can perform actions and access resources. This entity is typically a root user for an AWS account, an IAM role, or a user. For more information, see *Principal* in Roles terms and concepts in the IAM documentation.

privacy by design

A system engineering approach that takes privacy into account through the whole development process.

private hosted zones

A container that holds information about how you want Amazon Route 53 to respond to DNS queries for a domain and its subdomains within one or more VPCs. For more information, see Working with private hosted zones in the Route 53 documentation.

proactive control

A security control designed to prevent the deployment of noncompliant resources. These controls scan resources before they are provisioned. If the resource is not compliant with the control, then it isn't provisioned. For more information, see the Controls reference guide in the

AWS Control Tower documentation and see Proactive controls in *Implementing security controls on AWS*.

product lifecycle management (PLM)

The management of data and processes for a product throughout its entire lifecycle, from design, development, and launch, through growth and maturity, to decline and removal.

production environment

See environment.

programmable logic controller (PLC)

In manufacturing, a highly reliable, adaptable computer that monitors machines and automates manufacturing processes.

prompt chaining

Using the output of one LLM prompt as the input for the next prompt to generate better responses. This technique is used to break down a complex task into subtasks, or to iteratively refine or expand a preliminary response. It helps improve the accuracy and relevance of a model's responses and allows for more granular, personalized results.

pseudonymization

The process of replacing personal identifiers in a dataset with placeholder values. Pseudonymization can help protect personal privacy. Pseudonymized data is still considered to be personal data.

publish/subscribe (pub/sub)

A pattern that enables asynchronous communications among microservices to improve scalability and responsiveness. For example, in a microservices-based MES, a microservice can publish event messages to a channel that other microservices can subscribe to. The system can add new microservices without changing the publishing service.

Q

query plan

A series of steps, like instructions, that are used to access the data in a SQL relational database system.

query plan regression

When a database service optimizer chooses a less optimal plan than it did before a given change to the database environment. This can be caused by changes to statistics, constraints, environment settings, query parameter bindings, and updates to the database engine.

R

RACI matrix

See responsible, accountable, consulted, informed (RACI).

RAG

See Retrieval Augmented Generation.

ransomware

A malicious software that is designed to block access to a computer system or data until a payment is made.

RASCI matrix

See responsible, accountable, consulted, informed (RACI).

RCAC

See row and column access control.

read replica

A copy of a database that's used for read-only purposes. You can route queries to the read replica to reduce the load on your primary database.

re-architect

See 7 Rs.

recovery point objective (RPO)

The maximum acceptable amount of time since the last data recovery point. This determines what is considered an acceptable loss of data between the last recovery point and the interruption of service.

recovery time objective (RTO)

The maximum acceptable delay between the interruption of service and restoration of service.

refactor

See 7 Rs.

Region

A collection of AWS resources in a geographic area. Each AWS Region is isolated and independent of the others to provide fault tolerance, stability, and resilience. For more information, see Specify which AWS Regions your account can use.

regression

An ML technique that predicts a numeric value. For example, to solve the problem of "What price will this house sell for?" an ML model could use a linear regression model to predict a house's sale price based on known facts about the house (for example, the square footage).

rehost

See 7 Rs.

release

In a deployment process, the act of promoting changes to a production environment.

relocate

See 7 Rs.

replatform

See 7 Rs.

repurchase

See 7 Rs.

resiliency

An application's ability to resist or recover from disruptions. High availability and disaster recovery are common considerations when planning for resiliency in the AWS Cloud. For more information, see AWS Cloud Resilience.

resource-based policy

A policy attached to a resource, such as an Amazon S3 bucket, an endpoint, or an encryption key. This type of policy specifies which principals are allowed access, supported actions, and any other conditions that must be met.

responsible, accountable, consulted, informed (RACI) matrix

A matrix that defines the roles and responsibilities for all parties involved in migration activities and cloud operations. The matrix name is derived from the responsibility types defined in the matrix: responsible (R), accountable (A), consulted (C), and informed (I). The support (S) type is optional. If you include support, the matrix is called a *RASCI matrix*, and if you exclude it, it's called a *RACI matrix*.

responsive control

A security control that is designed to drive remediation of adverse events or deviations from your security baseline. For more information, see Responsive controls in *Implementing security controls on AWS*.

retain

See 7 Rs.

retire

See 7 Rs.

Retrieval Augmented Generation (RAG)

A generative AI technology in which an LLM references an authoritative data source that is outside of its training data sources before generating a response. For example, a RAG model might perform a semantic search of an organization's knowledge base or custom data. For more information, see What is RAG.

rotation

The process of periodically updating a secret to make it more difficult for an attacker to access the credentials.

row and column access control (RCAC)

The use of basic, flexible SQL expressions that have defined access rules. RCAC consists of row permissions and column masks.

RPO

See recovery point objective.

RTO

See recovery time objective.

runbook

A set of manual or automated procedures required to perform a specific task. These are typically built to streamline repetitive operations or procedures with high error rates.

S

SAML 2.0

An open standard that many identity providers (IdPs) use. This feature enables federated single sign-on (SSO), so users can log into the AWS Management Console or call the AWS API operations without you having to create user in IAM for everyone in your organization. For more information about SAML 2.0-based federation, see About SAML 2.0-based federation in the IAM documentation.

SCADA

See supervisory control and data acquisition.

SCP

See service control policy.

secret

In AWS Secrets Manager, confidential or restricted information, such as a password or user credentials, that you store in encrypted form. It consists of the secret value and its metadata. The secret value can be binary, a single string, or multiple strings. For more information, see What's in a Secrets Manager secret? in the Secrets Manager documentation.

security by design

A system engineering approach that takes security into account through the whole development process.

security control

A technical or administrative guardrail that prevents, detects, or reduces the ability of a threat actor to exploit a security vulnerability. There are four primary types of security controls: preventative, detective, responsive, and proactive.

security hardening

The process of reducing the attack surface to make it more resistant to attacks. This can include actions such as removing resources that are no longer needed, implementing the security best practice of granting least privilege, or deactivating unnecessary features in configuration files.

security information and event management (SIEM) system

Tools and services that combine security information management (SIM) and security event management (SEM) systems. A SIEM system collects, monitors, and analyzes data from servers, networks, devices, and other sources to detect threats and security breaches, and to generate alerts.

security response automation

A predefined and programmed action that is designed to automatically respond to or remediate a security event. These automations serve as detective or responsive security controls that help you implement AWS security best practices. Examples of automated response actions include modifying a VPC security group, patching an Amazon EC2 instance, or rotating credentials.

server-side encryption

Encryption of data at its destination, by the AWS service that receives it.

service control policy (SCP)

A policy that provides centralized control over permissions for all accounts in an organization in AWS Organizations. SCPs define guardrails or set limits on actions that an administrator can delegate to users or roles. You can use SCPs as allow lists or deny lists, to specify which services or actions are permitted or prohibited. For more information, see Service control policies in the AWS Organizations documentation.

service endpoint

The URL of the entry point for an AWS service. You can use the endpoint to connect programmatically to the target service. For more information, see AWS service endpoints in *AWS General Reference*.

service-level agreement (SLA)

An agreement that clarifies what an IT team promises to deliver to their customers, such as service uptime and performance.

service-level indicator (SLI)

A measurement of a performance aspect of a service, such as its error rate, availability, or throughput.

service-level objective (SLO)

A target metric that represents the health of a service, as measured by a service-level indicator.

shared responsibility model

A model describing the responsibility you share with AWS for cloud security and compliance. AWS is responsible for security *of* the cloud, whereas you are responsible for security *in* the cloud. For more information, see Shared responsibility model.

SIEM

See security information and event management system.

single point of failure (SPOF)

A failure in a single, critical component of an application that can disrupt the system.

SLA

See service-level agreement.

SLI

See service-level indicator.

SLO

See service-level objective.

split-and-seed model

A pattern for scaling and accelerating modernization projects. As new features and product releases are defined, the core team splits up to create new product teams. This helps scale your organization’s capabilities and services, improves developer productivity, and supports rapid innovation. For more information, see Phased approach to modernizing applications in the AWS Cloud.

SPOF

See single point of failure.

star schema

A database organizational structure that uses one large fact table to store transactional or measured data and uses one or more smaller dimensional tables to store data attributes. This structure is designed for use in a data warehouse or for business intelligence purposes.

strangler fig pattern

An approach to modernizing monolithic systems by incrementally rewriting and replacing system functionality until the legacy system can be decommissioned. This pattern uses the analogy of a fig vine that grows into an established tree and eventually overcomes and replaces its host. The pattern was introduced by Martin Fowler as a way to manage risk when rewriting monolithic systems. For an example of how to apply this pattern, see Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

subnet

A range of IP addresses in your VPC. A subnet must reside in a single Availability Zone.

supervisory control and data acquisition (SCADA)

In manufacturing, a system that uses hardware and software to monitor physical assets and production operations.

symmetric encryption

An encryption algorithm that uses the same key to encrypt and decrypt the data.

synthetic testing

Testing a system in a way that simulates user interactions to detect potential issues or to monitor performance. You can use Amazon CloudWatch Synthetics to create these tests.

system prompt

A technique for providing context, instructions, or guidelines to an LLM to direct its behavior. System prompts help set context and establish rules for interactions with users.

T

tags

Key-value pairs that act as metadata for organizing your AWS resources. Tags can help you manage, identify, organize, search for, and filter resources. For more information, see Tagging your AWS resources.

target variable

The value that you are trying to predict in supervised ML. This is also referred to as an *outcome variable*. For example, in a manufacturing setting the target variable could be a product defect.

task list

A tool that is used to track progress through a runbook. A task list contains an overview of the runbook and a list of general tasks to be completed. For each general task, it includes the estimated amount of time required, the owner, and the progress.

test environment

See environment.

training

To provide data for your ML model to learn from. The training data must contain the correct answer. The learning algorithm finds patterns in the training data that map the input data attributes to the target (the answer that you want to predict). It outputs an ML model that captures these patterns. You can then use the ML model to make predictions on new data for which you don’t know the target.

transit gateway

A network transit hub that you can use to interconnect your VPCs and on-premises networks. For more information, see What is a transit gateway in the AWS Transit Gateway documentation.

trunk-based workflow

An approach in which developers build and test features locally in a feature branch and then merge those changes into the main branch. The main branch is then built to the development, preproduction, and production environments, sequentially.

trusted access

Granting permissions to a service that you specify to perform tasks in your organization in AWS Organizations and in its accounts on your behalf. The trusted service creates a service-linked role in each account, when that role is needed, to perform management tasks for you. For more information, see Using AWS Organizations with other AWS services in the AWS Organizations documentation.

tuning

To change aspects of your training process to improve the ML model's accuracy. For example, you can train the ML model by generating a labeling set, adding labels, and then repeating these steps several times under different settings to optimize the model.

two-pizza team

A small DevOps team that you can feed with two pizzas. A two-pizza team size ensures the best possible opportunity for collaboration in software development.

U

uncertainty

A concept that refers to imprecise, incomplete, or unknown information that can undermine the reliability of predictive ML models. There are two types of uncertainty: *Epistemic uncertainty* is caused by limited, incomplete data, whereas *aleatoric uncertainty* is caused by the noise and randomness inherent in the data. For more information, see the Quantifying uncertainty in deep learning systems guide.

undifferentiated tasks

Also known as *heavy lifting*, work that is necessary to create and operate an application but that doesn’t provide direct value to the end user or provide competitive advantage. Examples of undifferentiated tasks include procurement, maintenance, and capacity planning.

upper environments

See environment.

V

vacuuming

A database maintenance operation that involves cleaning up after incremental updates to reclaim storage and improve performance.

version control

Processes and tools that track changes, such as changes to source code in a repository.

VPC peering

A connection between two VPCs that allows you to route traffic by using private IP addresses. For more information, see What is VPC peering in the Amazon VPC documentation.

vulnerability

A software or hardware flaw that compromises the security of the system.

W

warm cache

A buffer cache that contains current, relevant data that is frequently accessed. The database instance can read from the buffer cache, which is faster than reading from the main memory or disk.

warm data

Data that is infrequently accessed. When querying this kind of data, moderately slow queries are typically acceptable.

window function

A SQL function that performs a calculation on a group of rows that relate in some way to the current record. Window functions are useful for processing tasks, such as calculating a moving average or accessing the value of rows based on the relative position of the current row.

workload

A collection of resources and code that delivers business value, such as a customer-facing application or backend process.

workstream

Functional groups in a migration project that are responsible for a specific set of tasks. Each workstream is independent but supports the other workstreams in the project. For example, the portfolio workstream is responsible for prioritizing applications, wave planning, and collecting migration metadata. The portfolio workstream delivers these assets to the migration workstream, which then migrates the servers and applications.

WORM

See write once, read many.

WQF

See AWS Workload Qualification Framework.

write once, read many (WORM)

A storage model that writes data a single time and prevents the data from being deleted or modified. Authorized users can read the data as many times as needed, but they cannot change it. This data storage infrastructure is considered immutable.

Z

zero-day exploit

An attack, typically malware, that takes advantage of a zero-day vulnerability.

zero-day vulnerability

An unmitigated flaw or vulnerability in a production system. Threat actors can use this type of vulnerability to attack the system. Developers frequently become aware of the vulnerability as a result of the attack.

zero-shot prompting

Providing an LLM with instructions for performing a task but no examples (*shots*) that can help guide it. The LLM must use its pre-trained knowledge to handle the task. The effectiveness of zero-shot prompting depends on the complexity of the task and the quality of the prompt. See also few-shot prompting.

zombie application

An application that has an average CPU and memory usage below 5 percent. In a migration project, it is common to retire these applications.