Amazon Comprehend Medical

Developer Guide

Amazon Comprehend Medical: Developer Guide

Table of Contents

What is Amazon Comprehend Medical?

Amazon Comprehend Medical detects and returns useful information in unstructured clinical text such as physician's notes, discharge summaries, test results, and case notes. Amazon Comprehend Medical uses natural language processing (NLP) models to detect entities, which are textual references to medical information such as medical conditions, medications, or Protected Health Information(PHI). For a full list of detected entities, see Detect Entities (Version 2) (p. 16). Amazon Comprehend Medical also enables users to link these detected entities to standardized medical knowledge bases such as RxNorm and ICD10-CM through ontology linking operations.

The information in this developer guide is intended for application developers. This guide includes information about using Amazon Comprehend Medical programmatically through either the AWS CLI or the Amazon Comprehend Medical APIs.

Pricing for Amazon Comprehend Medical differs from Amazon Comprehend pricing. For more information, see Amazon Comprehend Medical Pricing.

Supported Languages

Amazon Comprehend Medical only detects medical entities in English language (US-EN) texts.

Important notice

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.

Amazon Comprehend Medical use cases

You can use Amazon Comprehend Medical for the following healthcare applications:

- **Patient case management and outcome**— Doctors and healthcare providers can manage and easily access medical information that doesn't fit into traditional forms. It allows patients to report their health concerns in a narrative that can provide more information than standard formats. By analyzing case notes, providers can identify candidates for early screening of medical conditions before the condition becomes more difficult and expensive to treat.
- **Clinical research**—Life sciences and research organizations can optimize the matching process for enrolling patients into clinical trials. By using Amazon Comprehend Medical to detect pertinent information in clinical text, researchers can improve pharmacovigilance, perform post-market surveillance to monitor adverse drug events, and assess therapeutic effectiveness by easily detecting vital information in follow-up notes and other clinical texts. For instance, it can be easier and more effective to monitor how patients respond to certain therapies by analyzing their narratives.

- **Medical billing and healthcare revenue cycle management**—Payors can expand their analytics to include unstructured documents such as clinical notes. More information about a diagnosis can be analyzed and used to help determine appropriate billing codes from unstructured documents. Natural language processing (NLP) is the most critical component of computer-assisted coding (CAC). Amazon Comprehend Medical uses the latest advances in NLP to analyze clinical text, helping to decrease time to revenue and improve reimbursement accuracy.
- **Ontology linking** —Use the ontology linking features to detect entities from clinical text and link those entities to standardized concepts in common medical ontologies. **InferICD10CM** identifies possible medical conditions as entities. **InferICD10CM** links those entities to unique codes from the 2019 version of the International Classification of Diseases, 10th Revision, Clinical Modification (ICD-10-CM). **InferRxNorm** identifies medications listed in clinical text as entities and links those entities to normalized concept identifiers from the RxNorm database from the US National Library of Medicine.

Benefits of Amazon Comprehend Medical

Some of the benefits of using Amazon Comprehend Medical include:

- **Easy, powerful natural language processing integration into your applications**—Amazon Comprehend Medical uses a simple API to build text analysis capabilities into your applications for powerful and accurate natural language processing.
- **Accuracy**—Amazon Comprehend Medical uses deep learning technology to accurately analyze text. Our models are constantly trained with new data across multiple domains to improve accuracy.
- **Scalability**—Amazon Comprehend Medical enables you to detect information from multiple documents, making rapid insights into patient health and care possible.
- **Integrate with other AWS services**—Amazon Comprehend Medical is designed to work seamlessly with other AWS services like Amazon S3 and AWS Lambda. Store your documents in Amazon S3, analyze real-time data with Kinesis Data Firehose, or use Amazon Transcribe to transcribe patient narratives into text that can be analyzed by Amazon Comprehend Medical. Support for AWS Identity and Access Management (IAM) makes it easy to securely control access to Amazon Comprehend Medical operations. Using IAM, you can create and manage AWS users and groups to grant the appropriate access to your developers and end users.
- **Low cost**—With Amazon Comprehend Medical, you only pay for the documents that you analyze. There are no minimum fees or upfront commitments.

HIPAA compliance

This is a HIPAA Eligible Service. For more information about AWS, U.S. Health Insurance Portability and Accountability Act of 1996 (HIPAA), and using AWS services to process, store, and transmit protected health information (PHI), see HIPAA Overview.

Connections to Amazon Comprehend Medical containing PHI must be encrypted. By default, all connections to Amazon Comprehend Medical use HTTPS over TLS. Amazon Comprehend Medical does not persistently store customer content. Therefore, you do not need to configure encryption at-rest within the service.

Accessing Amazon Comprehend Medical

1. AWS Management Console– Provides a web interface that you can use to access Amazon Comprehend Medical.

2. AWS Command Line Interface (AWS CLI) – Provides commands for a broad set of AWS services, including Amazon Comprehend Medical, and is supported on Windows, macOS, and Linux. For more information about installing the AWS CLI, see AWS Command Line Interface.
3. AWS SDKs – AWS provides SDKs (software development kits) that consist of libraries and sample code for various programming languages and platforms (Java, Python, Ruby, .NET, iOS, Android, etc.). The SDKs provide a convenient way to create programmatic access to Amazon Comprehend Medical and AWS. For more information, see AWS SDKs.

How to get started with Amazon Comprehend Medical

If you are a first-time user of Amazon Comprehend Medical, we recommend that you read the following sections in order:

1. How Amazon Comprehend Medical works (p. 4) – This section introduces Amazon Comprehend Medical concepts.
2. Getting started with Amazon Comprehend Medical (p. 6) – This section explains how to set up your account and test Amazon Comprehend Medical.

How Amazon Comprehend Medical works

Amazon Comprehend Medical uses a pretrained natural language processing (NLP) model to analyze unstructured clinical text through entity detection. An entity is a textual reference to medical information such as medical conditions, medications, or Protected Health Information (PHI). Some operations go one step further by detecting entities and then linking those entities to standardized ontologies. The model is continuously trained on a large body of medical texts, so you don't need to provide training data. All results include a confidence score, which indicates the confidence that Amazon Comprehend Medical has in the accuracy of the detected entities.

Both entity detection and ontology linking can be performed as either synchronous or asynchronous operations:

- Synchronous operations— Enables analysis on single documents which return the results of the analysis directly to your applications. Use the single-document operations when you are creating an interactive application that works on one document at a time.
- Asynchronous operations — Enables analysis on a collection or batch of documents stored in an Amazon S3 bucket. The results of the analysis are returned in an S3 bucket.

Note
Amazon Comprehend Medical can analyze only text in English (US-EN).

Synchronous entity detection

The **DetectEntitiesV2** and **DetectPHI** operations detect entities in unstructured clinical text from individual documents. You send a document to the Amazon Comprehend Medical service and receive the results of the analysis in the response.

Asynchronous batch analysis

The **StartEntitiesDetectionV2Job** and **StartPHIDetectionJob** operations start asynchronous jobs to detect references to medical information such as medical condition, treatment, tests and results or protected health information that is stored in an Amazon S3 bucket. The output of the detection job is written to a separate Amazon S3 bucket from which it can be used for further processing or downstream analysis.

The **StartICD10CMInferenceJob**, and **StartRxNormInferenceJob** operations start ontology linking batch operations which detect entities and link those entities to standardized codes in the RxNorm and ICD-10-CM knowledge bases.

Ontology linking

The **InferICD10CM** and **InferRxNorm** operations detect potential medical conditions and medications and link them to codes in the ICD-10-CM and RxNorm knowledge bases, respectively. You can use ontology linking batch analysis to analyze either a collection of documents or a single large document with up to 20,000 characters. By using either the console or the InferRxNorm and InferIC10CM ontology linking batch APIs, you can perform operations to start, stop, list, and describe ongoing batch analysis jobs.

Linking to concepts in the ICD-10-CM knowledge base of medical conditions

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.

Linking to concepts in the RxNorm knowledge base of medications

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.

Getting started with Amazon Comprehend Medical

To get started using Amazon Comprehend Medical, set up an AWS account and create an AWS Identity and Access Management (IAM) user. To use the Amazon Comprehend Medical CLI, download and configure it.

Topics

Step 1: Set up an AWS account and create an administrator user

Before you use Amazon Comprehend Medical for the first time, complete the following tasks:

1. Sign up for AWS (p. 6)
2. Create an IAM User (p. 7)

Sign up for AWS

When you sign up for Amazon Web Services (AWS), your AWS account is automatically signed up for all AWS services, including Amazon Comprehend Medical. You are charged only for the services that you use.

With Amazon Comprehend Medical, you pay only for the resources that you use. If you are a new AWS customer, you can get started with Amazon Comprehend Medical for free. For more information, see AWS Free Usage Tier.

If you already have an AWS account, skip to the next section.

To create an AWS account

1. Open https://portal.aws.amazon.com/billing/signup.
2. Follow the online instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.

Record your AWS account ID because you'll need it for the next task.

Create an IAM User

Services in AWS, such as Amazon Comprehend Medical, require that you provide credentials when you access them. This allows the service to determine whether you have permissions to access the service's resources.

We strongly recommend that you access AWS using AWS Identity and Access Management (IAM), not the credentials for your AWS account. To use IAM to access AWS, create an IAM user, add the user to an IAM group with administrative permissions, and then grant administrative permissions to the IAM user. You can then access AWS using a special URL and the IAM user's credentials.

The Getting Started exercises in this guide assume that you have a user with administrator privileges, `adminuser`.

To create an administrator and sign in to the console

1. Create a user named `adminuser` in your AWS account. For instructions, see Creating Your First IAM User and Administrators Group in the *IAM User Guide*.
2. Sign in to the AWS Management Console using a special URL. For more information, see How Users Sign In to Your Account in the *IAM User Guide*.

For more information about IAM, see the following:

- AWS Identity and Access Management (IAM)
- Getting started
- IAM User Guide

Next step

Step 2: Set up the AWS Command Line Interface (AWS CLI) (p. 7)

Step 2: Set up the AWS Command Line Interface (AWS CLI)

You don't need the AWS CLI to perform the steps in the Getting Started exercises. However, some of the other exercises in this guide do require it. If you prefer, you can skip this step and go to Step 3: Getting started using the Amazon Comprehend Medical console (p. 8), and set up the AWS CLI later.

To set up the AWS CLI

1. Download and configure the AWS CLI. For instructions, see the following topics in the *AWS Command Line Interface User Guide*:
   - Getting Set Up with the AWS Command Line Interface
   - Configuring the AWS Command Line Interface
2. In the AWS CLI config file, add a named profile for the administrator:

```
[profile adminuser]
aws_access_key_id = adminuser access key ID
```

```
aws_secret_access_key = adminuser secret access key
region = aws-region
```

You use this profile when executing the AWS CLI commands. For more information about named profiles, see Named Profiles in the *AWS Command Line Interface User Guide*. For a list of AWS Regions, see Regions and Endpoints in the *Amazon Web Services General Reference*.

3. Verify the setup by typing the following help command at the command prompt:

```
aws help
```

Next step

Step 3: Getting started using the Amazon Comprehend Medical console (p. 8)

Step 3: Getting started using the Amazon Comprehend Medical console

The easiest way to get started using the Comprehend Medical console is to analyze a short text file. If you haven't reviewed the concepts and terminology in How Amazon Comprehend Medical works (p. 4), we recommend that you do that before proceeding.

Topics

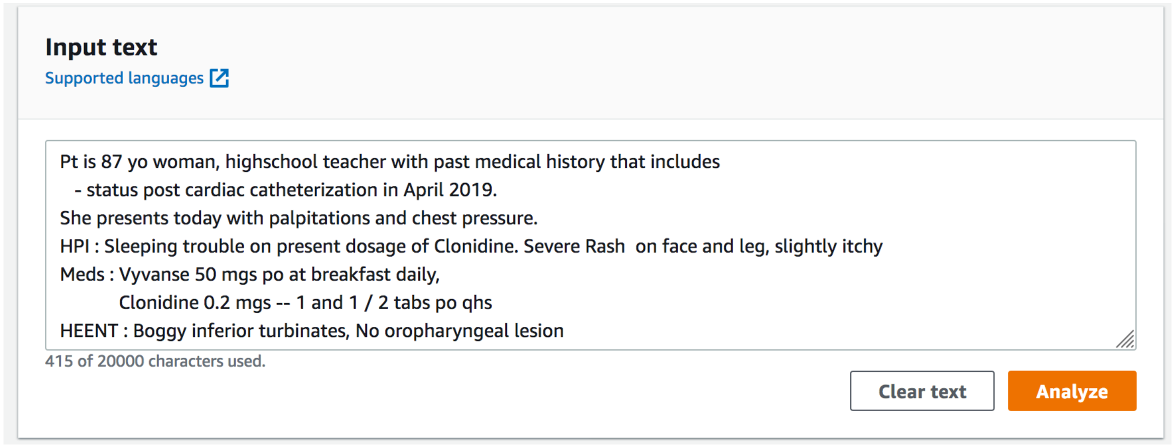
Analyzing clinical text using the console

The Comprehend Medical console enables you to analyze the contents of clinical text, up to 20,000 characters long. The results are shown in the console so that you can review the analysis.

To start analyzing documents, sign in to the AWS Management Console and open the Comprehend Medical console.

Under **Comprehend Medical**, choose **Real-time analysis**.

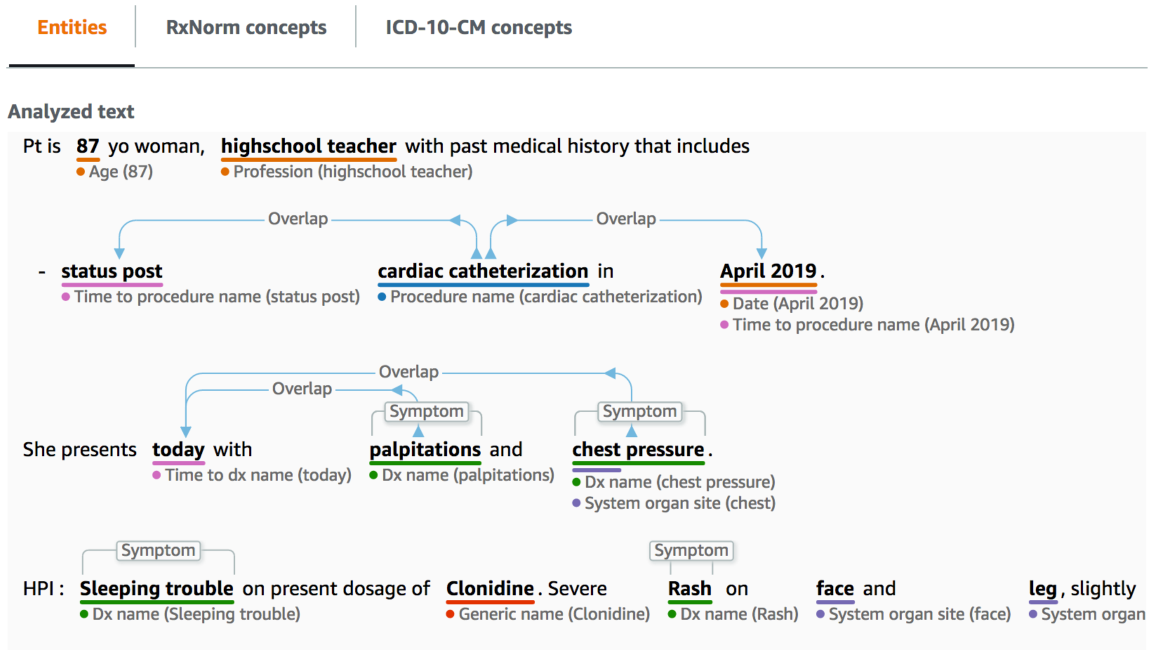
The console displays sample text and the analysis of that text:



```
Pt is 87 yo woman, highschool teacher with past medical history that includes
  - status post cardiac catheterization in April 2019.
She presents today with palpitations and chest pressure.
HPI : Sleeping trouble on present dosage of Clonidine. Severe Rash  on face and leg, slightly itchy
Meds : Vyvanse 50 mgs po at breakfast daily,
        Clonidine 0.2 mgs -- 1 and 1 / 2 tabs po qhs
HEENT : Boggy inferior turbinates, No oropharyngeal lesion
```

415 of 20000 characters used.

You can replace the sample text with your own text in English and then choose **Analyze** to get an analysis of your text.


Entities
RxNorm concepts
ICD-10-CM concepts
Analyzed text
Overlap
Overlap
Overlap
Overlap
Symptom
Symptom
Symptom
Symptom
● Age (87)
● Profession (highschool teacher)
● Time to procedure name (status post)
● Procedure name (cardiac catheterization)
● Date (April 2019)
● Time to procedure name (April 2019)
● Time to dx name (today)
● Dx name (palpitations)
● Dx name (chest pressure)
● System organ site (chest)
● Dx name (Sleeping trouble)
● Generic name (Clonidine)
● Dx name (Rash)
● System organ site (face)
● System organ

Pt is 87 yo woman, highschool teacher with past medical history that includes

- status post cardiac catheterization in April 2019.

She presents today with palpitations and chest pressure.

HPI : Sleeping trouble on present dosage of Clonidine. Severe Rash on face and leg, slightly

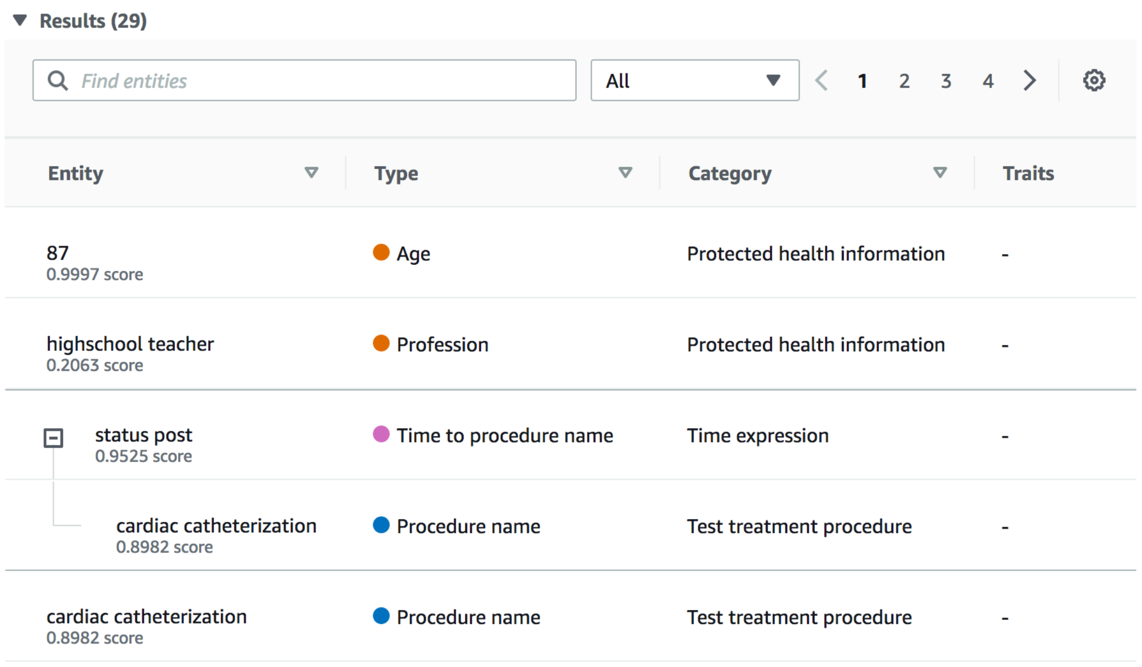
Below the input text, the analyzed text is color-coded to indicate the entity category:

- Orange tags identify PHI data.
- Red tags identify Medication.
- Green tags identify Medical Condition.
- Blue tags identify Test, Treatment, or Procedure (TTP).
- Purple tags identify Anatomy.
- Pink tags identify Time Expressions.

For more information, see How Amazon Comprehend Medical works (p. 4).

In the console, below the input box, the **Analyzed Text** pane shows more information about the text.

The **Entity** section displays cards for the entities found in the text:



Entity	Type	Category	Traits
87 0.9997 score	● Age	Protected health information	-
highschool teacher 0.2063 score	● Profession	Protected health information	-
status post 0.9525 score	● Time to procedure name	Time expression	-
cardiac catheterization 0.8982 score	● Procedure name	Test treatment procedure	-
cardiac catheterization 0.8982 score	● Procedure name	Test treatment procedure	-

Each card shows the text and its entity type.

Next to each of the entities, a score represents the confidence that Comprehend Medical has in the identification of the text as the type of entity shown.

To see the JSON structure of both the request and the results, choose **Application integration**. The JSON structure is the same as the structure returned by the operation.

Next Step

Step 4: Getting started using the Amazon Comprehend Medical APIs (p. 10)

Step 4: Getting started using the Amazon Comprehend Medical APIs

The following examples demonstrate how to use Amazon Comprehend Medical operations using the AWS CLI, Java, and Python. Use them to learn about Amazon Comprehend Medical operations and as building blocks for your own applications.

To run the AWS CLI and Python examples, install the AWS CLI. For more information, see Step 2: Set up the AWS Command Line Interface (AWS CLI) (p. 7).

To run the Java examples, install the AWS SDK for Java. For instructions for installing the AWS SDK for Java, see Set up the AWS SDK for Java.

Topics

Detecting medical entities using the AWS Command Line Interface

The following example demonstrates using the `DetectEntitiesV2` operation using the AWS CLI to return the medical entities detected in text. To run the example, you must install the AWS CLI. For more information, see the section called "Step 2: Set Up the AWS CLI" (p. 7).

The example is formatted for Unix, Linux, and macOS. For Windows, replace the backslash (\) Unix continuation character at the end of each line with a caret (^).

```
aws comprehendmedical detect-entities-v2 \
    --endpoint endpoint \
    --region region \
    --text "aspirin is required 20 mg po daily for 2 times as tab"
```

The response is the following:

```
{
    "Entities": [
        {
            "Category": "MEDICATION",
            "BeginOffset": 0,
            "EndOffset": 7,
            "Text": "aspirin",
            "Traits": [],
            "Score": 0.9988090991973877,
            "Attributes": [
                {
                    "BeginOffset": 20,
                    "EndOffset": 25,
                    "Text": "20 mg",
                    "Traits": [],
                    "Score": 0.9559056162834167,
                    "Type": "DOSAGE",
                    "Id": 1,
                    "RelationshipScore": 0.9981593489646912
                },
                {
                    "BeginOffset": 26,
                    "EndOffset": 28,
                    "Text": "po",
                    "Traits": [],
                    "Score": 0.9995359182357788,
                    "Type": "ROUTE_OR_MODE",
                    "Id": 2,
                    "RelationshipScore": 0.9969323873519897
                },
                {
                    "BeginOffset": 29,
                    "EndOffset": 34,
                    "Text": "daily",
                    "Traits": [],
                    "Score": 0.9803128838539124,
                    "Type": "FREQUENCY",
                    "Id": 3,
                    "RelationshipScore": 0.9990783929824829
                },
                {
                    "BeginOffset": 39,
                    "EndOffset": 46,
                    "Text": "2 times",
                    "Traits": [],
                    "Score": 0.8623972535133362,
                    "Type": "DURATION",
                    "Id": 4,
                    "RelationshipScore": 0.9996501207351685
                },
                {
                    "BeginOffset": 50,
                    "EndOffset": 53,
                    "Text": "tab",
                    "Traits": [],
                    "Score": 0.784785270690918,
                    "Type": "FORM",
                    "Id": 5,
                    "RelationshipScore": 0.9986748695373535
                }
            ],
            "Type": "GENERIC_NAME",
            "Id": 0
        }
    ],
    "UnmappedAttributes": []
}
```

Detecting medical entities using the AWS SDK for Java

The following example uses the `DetectEntitiesV2` operation with Java. To run the example, install the AWS SDK for Java. For instructions on installing the AWS SDK for Java, see Set up the AWS SDK for Java.

```
import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.AWSCredentialsProvider;
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.client.builder.AwsClientBuilder;
import com.amazonaws.services.comprehendmedical.AWSComprehendMedical;
import com.amazonaws.services.comprehendmedical.AWSComprehendMedicalClient;
import com.amazonaws.services.comprehendmedical.model.DetectEntitiesRequest;
import com.amazonaws.services.comprehendmedical.model.DetectEntitiesResult;

public class SampleAPICall {

    public static void main() {

        AWSCredentialsProvider credentials
                = new AWSStaticCredentialsProvider(new BasicAWSCredentials("YOUR AWS ACCESS
 KEY", "YOUR AWS SECRET"));

        AWSComprehendMedical client = AWSComprehendMedicalClient.builder()

 .withCredentials(credentials)
                                                               .withRegion("YOUR REGION")
                                                               .build();


        DetectEntitiesV2Request request = new DetectEntitiesV2Request();
        request.setText("cerealx 84 mg daily");

        DetectEntitiesV2Result result = client.detectEntitiesV2(request);
        result.getEntities().forEach(System.out::println);
    }
}
```

The output contains the three entities found in the input text, their location in the input text. A confidence level that the entity was correctly identified is also listed with each entity. The following output shows the `Generic_Name`, `Dosage`, and `Frequency` entities from the preceding example.

```
{Id: 0,BeginOffset: 0,EndOffset: 3,Score: 0.9940211,Text: Bob,Category:
PROTECTED_HEALTH_INFORMATION,Type: NAME,Traits: [],}
{Id: 2,BeginOffset: 23,EndOffset: 30,Score: 0.99914634,Text: aspirin,Category:
 MEDICATION,Type: GENERIC_NAME,Traits: [],Attributes:
[{Type: DOSAGE,Score: 0.9630807,RelationshipScore: 0.99969745,Id: 1,BeginOffset:
 14,EndOffset: 19,Text: 50 mg,Traits: []}]}
```

Detecting medical entities using the AWS SDK for Python (Boto)

The following example uses the `DetectEntitiesV2` operation with Python. To run the sample, install the AWS CLI. For more information, see the section called “Step 2: Set Up the AWS CLI” (p. 7).

```
import boto3
client = boto3.client(service_name='comprehendmedical', region_name='YOUR REGION')
result = client.detect_entities(Text= 'cerealx 84 mg daily')
entities = result['Entities'];
for entity in entities:
    print('Entity', entity)
```

The output contains the three entities found in the input text, their location in the input text. A confidence level that the entity was correctly identified is also listed with each entity. The following output shows the `Generic_Name`, `Dosage`, and `Frequency` entities from the preceding example.

```
('Entity', {u'Category': u'MEDICATION', u'BeginOffset': 0, u'EndOffset': 7,
            u'Text': u'cerealx', u'Traits': [], u'Score': 0.8877691626548767,
 u'Attributes': [{u'BeginOffset': 8, u'EndOffset': 13,
            u'Text': u'84 mg', u'Traits': [], u'Score': 0.9337134957313538, u'Type':
 u'DOSAGE', u'Id': 1, u'RelationshipScore': 0.9995118379592896},
            {u'BeginOffset': 14, u'EndOffset': 19, u'Text': u'daily', u'Traits': [],
 u'Score': 0.990627646446228, u'Type': u'FREQUENCY',
            u'Id': 2, u'RelationshipScore': 0.9987651109695435}], u'Type': u'BRAND_NAME',
 u'Id': 0})
```

Amazon Comprehend Medical and interface VPC endpoints (AWS PrivateLink)

You can establish a private connection between your VPC and Amazon Comprehend Medical by creating an *interface VPC endpoint*. Interface VPC endpoints are powered by AWS PrivateLink, a technology that you can use to privately access Amazon Comprehend Medical APIs without an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC don't need public IP addresses to communicate with Amazon Comprehend Medical APIs. Traffic between your VPC and Amazon Comprehend Medical does not leave the Amazon network.

Each interface endpoint is represented by one or more Elastic Network Interfaces in your subnets.

For more information, see Interface VPC endpoints (AWS PrivateLink) in the *Amazon VPC User Guide*.

Considerations for Amazon Comprehend Medical VPC endpoints

Before you set up an interface VPC endpoint for Amazon Comprehend Medical, ensure that you review Interface endpoint properties and limitations in the *Amazon VPC User Guide*.

Amazon Comprehend Medical supports making calls to all of its API actions from your VPC.

Creating an interface VPC endpoint for Amazon Comprehend Medical

You can create a VPC endpoint for the Amazon Comprehend Medical service using either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see Creating an interface endpoint in the *Amazon VPC User Guide*.

Create a VPC endpoint for Amazon Comprehend Medical using the following service name:

- com.amazonaws.*`region`*.comprehendmedical

If you turn on private DNS for the endpoint, you can make API requests to Amazon Comprehend Medical using its default DNS name for the Region. For example, *`comprehendmedical.us-east-1.amazonaws.com`*.

For more information, see Accessing a service through an interface endpoint in the *Amazon VPC User Guide*.

Creating a VPC endpoint policy for Amazon Comprehend Medical

You can attach an endpoint policy to your VPC endpoint that controls access to Amazon Comprehend Medical. The policy specifies the following information:

- The principal that can perform actions.
- The actions that can be performed.
- The resources on which actions can be performed.

For more information, see Controlling access to services with VPC endpoints in the *Amazon VPC User Guide*.

Example: VPC endpoint policy for Amazon Comprehend Medical actions

The following is an example of an endpoint policy for Amazon Comprehend Medical. When attached to an endpoint, this policy grants access to the Amazon Comprehend Medical `DetectEntitiesV2` action for all principals on all resources.

```
{
   "Statement":[
      {
         "Principal":"*",
         "Effect":"Allow",
         "Action":[
            "comprehendmedical:DetectEntitiesV2"
         ],
         "Resource":"*"
      }
   ]
}
```

Text analysis APIs

Amazon Comprehend Medical enables you to examine clinical documents to gain various insights about their content using pre-trained Natural Language Processing (NLP) models. Analysis can be performed both on single files or as a batch analysis on multiple files stored in an Amazon S3 bucket.

With Amazon Comprehend Medical, you can perform the following on your documents:

- Detect Entities (Version 2) (p. 16) —Examine unstructured clinical text to detect textual references to medical information such as medical condition, treatment, tests and results, and medications. This version uses a new model and changes the way some entities are returned in the output. For more information, see .
- Detect PHI (p. 29)—Examine unstructured clinical text to detect textual references to protected health information (PHI) such as names and addresses.
- Detect entities (p. 20)— Examine unstructured clinical text to detect textual references to medical information such as medical condition, treatment, tests and results, and medications. Use `Detect Entities Version 2` for all new applications.

Topics

Detect Entities (Version 2)

Use the **DetectEntitiesV2** to detect entities in single files or **StartEntitiesDetectionV2Job** for batch analysis on multiple files. You can detect entities in the following categories:

- `ANATOMY:` Detects references to the parts of the body or body systems and the locations of those parts or systems.
- `MEDICAL_CONDITION:` Detects the signs, symptoms, and diagnosis of medical conditions.
- `MEDICATION:` Detects medication and dosage information for the patient.
- `PROTECTED_HEALTH_INFORMATION:` Detects the patient's personal information.
- `TEST_TREATMENT_PROCEDURE:` Detects the procedures that are used to determine a medical condition.
- `TIME_EXPRESSION:` Detects entities related to time when they are associated with a detected entity.

All six categories are detected by the **DetectEntitiesV2** operation. For analysis specific to detecting PHI, use **DetectPHI** on single files and **StartPHIDetectionJob** for batch analysis.

Important
Amazon Comprehend Medical provides confidence scores that indicate the level of confidence in the accuracy of detected entities. When you are identifying protected health information (PHI), evaluate these scores and identify the right confidence threshold for your use case.

> Use high-confidence thresholds in situations that require high accuracy. For certain use cases, results should be reviewed and verified by appropriately trained human reviewers. Use Amazon Comprehend Medical in patient care scenarios only after review by trained medical professionals for accuracy and exercising medical judgment.

Amazon Comprehend Medical detects information in the following classes:

- *Entity:* A text reference to the name of relevant objects, such as people, treatments, medications, and medical conditions. For example, `ibuprofen`.
- *Category:* The generalized grouping to which an entity belongs. For example, ibuprofen is part of the `MEDICATION` category.
- *Type:* The type of entity detected within a single category. For example, ibuprofen is in the `GENERIC_NAME` type in the `MEDICATION` category.
- *Attribute:* Information related to an entity, such as the dosage of a medication. For example, `200 mg` is an attribute of the ibuprofen entity.
- *Trait:* Something that Amazon Comprehend Medical understands about an entity, based on context. For example, a medication has the `NEGATION` trait if a patient is not taking it.
- *Relationship Type:* The relationship between an entity and an attribute.

Amazon Comprehend Medical provides the location of an entity in the input text. In the Amazon Comprehend console, it shows the location graphically. When you use the API, it shows the location by numerical offset.

Each entity and attribute includes a score that indicates the confidence level that Amazon Comprehend Medical has in the accuracy of the detection. Each attribute also has a relationship score. The score indicates the confidence level that Amazon Comprehend Medical has in the accuracy of the relationship between the attribute and its parent entity. Identify the right confidence threshold for your use case. Use high-confidence thresholds in situations that require great accuracy. Filter out data that doesn't meet the threshold.

Anatomy Category

The `ANATOMY` category detects references to the parts of the body or body systems and the locations of those parts or systems.

Types

- `SYSTEM_ORGAN_SITE`: Body systems, anatomic locations or regions, and body sites.

Attributes

- `DIRECTION`: Directional terms. For example, left, right medial, lateral, upper, lower, posterior, anterior, distal, proximal, contralateral, bilateral, ipsilateral, dorsal, ventral, and so on.

Medical condition category

The `MEDICAL_CONDITION` category detects the signs, symptoms, and diagnosis of medical conditions. The category has one entity type, one attribute, and four traits. One or more traits can be associated with a type. Contextual information about attributes and their relationship to the diagnosis is detected and mapped to the `DX_NAME` through `RELATIONSHIP_EXTRACTION.` For instance, from the text "chronic pain in left leg", "chronic" is detected as the attribute `ACUITY`, "left" is detected as the attribute

`DIRECTION`, and "leg" is detected as the attribute `SYSTEM_ORGAN_SITE`. The relationships of each of these attributes are mapped to the medical condition entity "pain", along with a confidence score.

Types

- `DX_NAME`: All medical conditions listed. The `DX_NAME` type includes present illness, reason for visit, and medical history.

Attributes

- `ACUITY`: Determination of disease instance, such as chronic, acute, sudden, persistent, or gradual.
- `DIRECTION`: Directional terms. For example, left, right medial, lateral, upper, lower, posterior, anterior, distal, proximal, contralateral, bilateral, ipsilateral, dorsal, or ventral.
- `SYSTEM_ORGAN_SITE`: anatomical location.

Traits

- `DIAGNOSIS`: A medical condition that is determined as the cause or result of the symptoms. Symptoms can be found through physical findings, laboratory or radiological reports, or any other means. Applies only to the `DX_NAME` type.
- `NEGATION`: An indication that a result or action is negative or not being performed.
- `SIGN`: A medical condition that the physician reported. Applies only to the `DX_NAME` type.
- `SYMPTOM`: A medical condition reported by the patient. Applies only to the `DX_NAME` type.

Medication Category

The `MEDICATION` category detects medication and dosage information for the patient. One or more attributes can apply to a type.

Types

- `BRAND_NAME`: The copyrighted brand name of the medication or therapeutic agent.
- `GENERIC_NAME`: The non-brand name, ingredient name, or formula mixture of the medication or therapeutic agent.

Attributes

- `DOSAGE`: The amount of medication ordered.
- `DURATION`: How long the medication should be administered.
- `FORM`: The form of the medication.
- `FREQUENCY`: How often to administer the medication.
- `RATE`: The administration rate of the medication (Primarily for medication infusions or IVs).
- `ROUTE_OR_MODE`: The administration method of a medication.
- `STRENGTH`: The medication strength.

Traits

- `NEGATION`: Any indication that the patient is not taking a medication.

Protected health information category

The PROTECTED_HEALTH_INFORMATION category detects the patient's personal information. See Detect PHI (p. 29) to learn more about this operation.

Types

- ADDRESS: All geographical subdivisions of an address of any facility, units, or wards within a facility.
- AGE: All components of age, spans of age, or any age mentioned. This includes those of a patient, family members, or others. The default is years unless otherwise noted.
- EMAIL: Any email address.
- ID: Social security number, medical record number, facility identification number, clinical trial number, certificate or license number, vehicle or device number, the place of care, or provider. This also includes any biometric number of the patient, such as height, weight, or a lab value.
- NAME: All names. Typically, names of the patient, family, or provider.
- PHONE_OR_FAX: Any phone, fax, or pager number. Excludes named phone numbers, such as 1-800-QUIT-NOW and 911.
- PROFESSION: Any profession or employer that pertains to the patient or the patient's family. It does not include the profession of the clinician mentioned in the note.

Test, treatment, procedure Category

The TEST_TREATMENT_PROCEDURE category detects the procedures that are used to determine a medical condition. One or more attributes can be related to an entity of the TEST_NAME type.

Types

- PROCEDURE_NAME: Interventions as a one-time action performed on the patient to treat a medical condition or to provide patient care.
- TEST_NAME: Procedures performed on a patient for diagnostic, measurement, screening, or rating that might have a resulting value. This includes any procedure, process, evaluation, or rating to determine a diagnosis, to rule out or find a condition, or to scale or score a patient.
- TREATMENT_NAME: Interventions performed over a span of time for combating a disease or disorder. This includes groupings of medications, such as antivirals and vaccinations.

Attributes

- TEST_VALUE: The result of a test. Applies only to the TEST_NAME entity type.
- TEST_UNIT: The unit of measure that might accompany the value of the test. Applies only to the TEST_NAME entity type.

Time expression category

The TIME_EXPRESSION category detects entities related to time. This includes entities such as dates and time expressions such as "three days ago," "today," "currently," "day of admission," "last month," or "16 days." Results in this category are only returned if they are associated with an entity. For example, "Yesterday, the patient took 200 mg of ibuprofen" would return Yesterday as a TIME_EXPRESSION entity that overlaps with GENERIC_NAME entity "ibuprofen." However, it would not be recognized as an entity in "yesterday, the patient walked their dog."

Types

- `TIME_TO_MEDICATION_NAME`: The date a medication was taken. The attributes specific to this type are `BRAND_NAME` and `GENERIC_NAME`.
- `TIME_TO_DX_NAME`: The date a medical condition occurred. The attribute for this type is `DX_NAME`.
- `TIME_TO_TEST_NAME`: The date a test was performed. The attribute for this type is `TEST_NAME`.
- `TIME_TO_PROCEDURE_NAME`: The date a procedure was performed. The attribute for this type is `PROCEDURE_NAME`.
- `TIME_TO_TREATMENT_NAME`: The date a treatment was administered. The attribute for this type is `TREATMENT_NAME`.

Relationship Type

- The relationship between an entity and an attribute. The recognized `Relationship_type` is:

  `Overlap` – The `TIME_EXPRESSION` concurs with the entity detected.

Detect entities

> **Note**
> This version of the **DetectEntities** operation should not be used for new applications. You should use version 2 of the operation instead. All new iterations and enhancements of features will be specific to Detect Entities Version 2. For more information, see Detect Entities (Version 2) (p. 16).

Use the **DetectEntities** operation to detect the medical entities in your text. It detects entities in the following categories:

- `ANATOMY`
- `MEDICAL_CONDITION`
- `MEDICATION`
- `PROTECTED_HEALTH_INFORMATION`
- `TEST_TREATMENT_PROCEDURE`

All five categories are detected by the `DetectEntities` operation. The operation detects entities only in the `PROTECTED_HEALTH_INFORMATION` category. Use it when only PHI (protected health information) is required. For information about this operation, see Detect PHI (p. 29).

> **Important**
> Amazon Comprehend Medical provides confidence scores that indicate the level of confidence in the accuracy of detected entities. When you are identifying protected health information (PHI), evaluate these scores and identify the right confidence threshold for your use case. Use high confidence thresholds in situations that require high accuracy. For certain use cases, results should be reviewed and verified by appropriately trained human reviewers. Results from Amazon Comprehend Medical in patient care scenarios should only be used after review by trained medical professionals reviewing results for accuracy and sound medical judgment.

Amazon Comprehend Medical detects information in the following classes:

- *Entity:* A textual reference to the name of relevant objects, such as people, treatments, medications, and medical conditions. For example, "Ibuprofen."

- *Category:* The generalized grouping to which a detected entity belongs. For example, "Ibuprofen" is part of the `MEDICATION` category.
- *Type:* The type of entity detected, scoped to a category. For example, "Ibuprofen" is in the `GENERIC_NAME` type in the `MEDICATION` category.
- *Attribute:* Information related to a detected entity, such as the dosage of a medication. For example, "200 mg" is an attribute of the "Ibuprofen" entity.
- *Trait:* Something that Amazon Comprehend Medical understands about an entity, based on context. For example, a medication has the `NEGATION` trait if a patient is not taking it.

Amazon Comprehend Medical provides the location of an entity in the input text. In the Amazon Comprehend console, it shows the location graphically. When you use the API, it shows the location by numerical offset.

Each entity and attribute includes a score that indicates the level of confidence that Amazon Comprehend Medical has in the accuracy of the detection. Each attribute also has a relationship score. This score indicates the level of confidence Amazon Comprehend Medical has in the accuracy of the relationship between the attribute and its parent entity. Identify the right confidence threshold for your use case. Use high confidence thresholds in situations that require great accuracy, and filter out data that doesn't meet the threshold.

Anatomy category

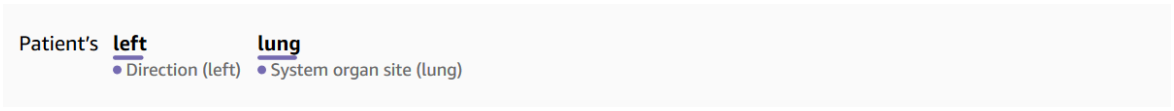
The `ANATOMY` category detects references to the parts of the body or body systems and the locations of those parts or systems. It contains two entity types.

Types

- `DIRECTION`: Directional terms. For example, left, right medial, lateral, upper, lower, posterior, anterior, distal, proximal, contralateral, bilateral, ipsilateral, dorsal, ventral, and so on.
- `SYSTEM_ORGAN_SITE`: Body systems, anatomic locations or regions, and body sites.

Example

The text "Patient's left lung" returns:


Patient's **left** **lung**
● Direction (left) ● System organ site (lung)

- "left" is a `DIRECTION` *type*.
- "lung" is a `SYSTEM_ORGAN_SITE` *type*.

The operation returns the following JSON structure:

```
{
    "Entities": [
        {
            "Id": 0,
            "BeginOffset": 10,
            "EndOffset": 14,
            "Score": 0.9876197576522827,
            "Text": "left",
            "Category": "ANATOMY",
            "Type": "DIRECTION",
            "Traits": []
        },
        {
            "Id": 1,
            "BeginOffset": 15,
            "EndOffset": 19,
            "Score": 0.9820258021354675,
            "Text": "lung",
            "Category": "ANATOMY",
            "Type": "SYSTEM_ORGAN_SITE",
            "Traits": []
        }
    ],
    "UnmappedAttributes": []
}
```

Medical condition category

The `MEDICAL_CONDITION` category detects the symptoms and diagnosis of medical conditions. It contains two entity types and four traits. One or more traits can be associated with a type.

Types

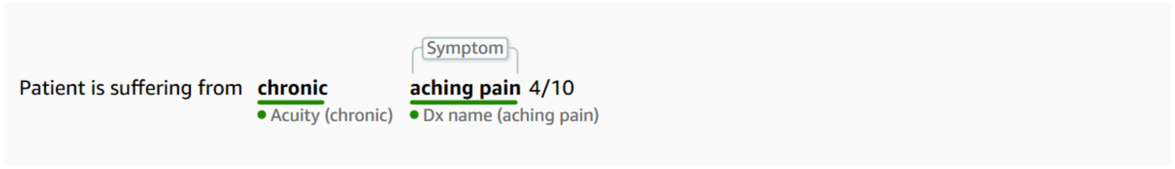
- `ACUITY`: Determination of disease instance, such as chronic, acute, sudden, persistent, or gradual.
- `DX_NAME`: All medical conditions listed. The `DX_NAME` type includes present illness, reason for visit, and medical history.

Traits

- `DIAGNOSIS`: An identification of a medical condition that is determined by evalutation of the symptoms. This evaluation comes from physical findings, laboratory or radiological reports, or the patient narrative. Applies only to the `DX_NAME` type.
- `NEGATION`: An indication that a result or action is negative or not being performed.
- `SIGN`: A medical condition that the physician reported. Applies only to the `DX_NAME` type.
- `SYMPTOM`: A medical condition reported by the patient. Applies only to the `DX_NAME` type.

Example

The text "Patient is suffering from chronic aching pain 4/10" returns:



- "aching pain" is the `DX_NAME` *type*.
- `SYMPTOM` is a *trait* of the "aching pain" *type*.
- "chronic" is the `ACUITY` *type*.

The operation returns the following JSON structure:

```
{
    "Entities": [
        {
            "Id": 0,
            "BeginOffset": 26,
            "EndOffset": 33,
            "Score": 0.9961825013160706,
            "Text": "chronic",
            "Category": "MEDICAL_CONDITION",
            "Type": "ACUITY",
            "Traits": []
        },
        {
            "Id": 1,
            "BeginOffset": 34,
            "EndOffset": 45,
            "Score": 0.8380221724510193,
            "Text": "aching pain",
            "Category": "MEDICAL_CONDITION",
            "Type": "DX_NAME",
            "Traits": [
                {
                    "Name": "SYMPTOM",
                    "Score": 0.6004688739776611
                }
            ]
        }
    ],
    "UnmappedAttributes": []
}
```

Medication category

The `MEDICATION` category detects medication and dosage information for the patient. It contains two entity types, seven attributes, and one trait. One or more attributes can apply to a type.

Types

- `BRAND_NAME`: The copyrighted brand name of the medication or therapeutic agent.
- `GENERIC_NAME`: Non-brand name, ingredient name, or formula mixture of the medication or therapeutic agent.

Attributes

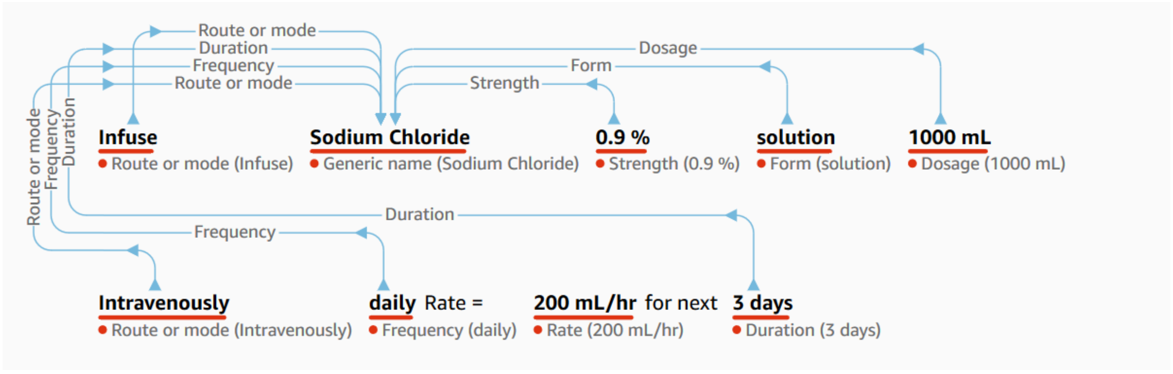
- `DOSAGE`: The amount of medication ordered.
- `DURATION`: How long the medication should be administered.
- `FORM`: The form of the medication.
- `FREQUENCY`: How often to administer the medication.
- `RATE`: Primarily for medication infusions or IVs, the administration rate of the medication.
- `ROUTE_OR_MODE`: The administration method of a medication.
- `STRENGTH`: The medication strength.

Traits

- `NEGATION`: Any indication that the patient is not taking a medication.

Example

The text "Infuse Sodium Chloride 0.9% solution 1000 mL intravenously daily Rate - 200 mL/hr for next 3 days" returns:



- "Infuse" as a "ROUTE_OR_MODE" *attribute* related to the "Sodium Chloride" *type*.
- "Sodium Chloride" as a `GENERIC_NAME` *type*.
- "0.9%" as a `STRENGTH` *attribute* related to the "Sodium Chloride" *type*.
- "solution" as a `FORM` *attribute* related to the "Sodium Chloride" *type*.
- "100 mL as a `DOSAGE` *attribute* related to the "Sodium Chloride" *type*.
- "intravenously" as a `ROUTE_OR_MODE` *attribute* related to the "Sodium Chloride" *type*.
- "daily" as a `FREQUENCY` *attribute* related to the "Sodium Chloride" *type*.
- "200 ml/hr" as a `RATE` *attribute* related to the "Sodium Chloride" *type*.
- "next 3 days" as a `DURATION` *attribute* related to the "Sodium Chloride" *type*.

The operation returns the following JSON structure:

```
{
    "Entities": [
        {
            "Id": 1,
            "BeginOffset": 7,
            "EndOffset": 22,
            "Score": 0.9998517036437988,
            "Text": "Sodium Chloride",
            "Category": "MEDICATION",
            "Type": "GENERIC_NAME",
            "Traits": [],
            "Attributes": [
                {
                    "Type": "ROUTE_OR_MODE",
                    "Score": 0.32359644770622253,
                    "RelationshipScore": 0.9719992280006409,
                    "Id": 0,
                    "BeginOffset": 0,
                    "EndOffset": 6,
                    "Text": "Infuse",
                    "Traits": []
                },
                {
                    "Type": "STRENGTH",
                    "Score": 0.9976715445518494,
                    "RelationshipScore": 0.7892051339149475,
                    "Id": 2,
                    "BeginOffset": 23,
                    "EndOffset": 27,
                    "Text": "0.9%",
                    "Traits": []
                },
                {
                    "Type": "FORM",
                    "Score": 0.9930835962295532,
                    "RelationshipScore": 0.9956902861595154,
                    "Id": 3,
                    "BeginOffset": 28,
                    "EndOffset": 36,
                    "Text": "solution",
                    "Traits": []
                },
                {
                    "Type": "ROUTE_OR_MODE",
                    "Score": 0.9990690350532532,
                    "RelationshipScore": 0.9801701903343201,
                    "Id": 5,
                    "BeginOffset": 45,
                    "EndOffset": 58,
                    "Text": "intravenously",
                    "Traits": []
                },
                {
                    "Type": "FREQUENCY",
                    "Score": 0.9539222121238708,
                    "RelationshipScore": 0.9864235520362854,
                    "Id": 6,
                    "BeginOffset": 59,
                    "EndOffset": 64,
                    "Text": "daily",
                    "Traits": []
                },
                {
                    "Type": "DURATION",
                    "Score": 0.9392423033714294,
                    "RelationshipScore": 0.9961885809898376,
                    "Id": 8,
                    "BeginOffset": 91,
                    "EndOffset": 97,
                    "Text": "3 days",
                    "Traits": []
                }
            ]
        }
    ],
    "UnmappedAttributes": [
        {
            "Type": "MEDICATION",
            "Attribute": {
                "Type": "DOSAGE",
                "Score": 0.9922149777412415,
                "Id": 4,
                "BeginOffset": 37,
                "EndOffset": 44,
                "Text": "1000 mL",
                "Traits": []
            }
        },
        {
            "Type": "MEDICATION",
            "Attribute": {
                "Type": "RATE",
                "Score": 0.9728594422340393,
                "Id": 7,
                "BeginOffset": 72,
                "EndOffset": 81,
                "Text": "200 mL/hr",
                "Traits": []
            }
        }
    ]
}
```

Protected Health Information category

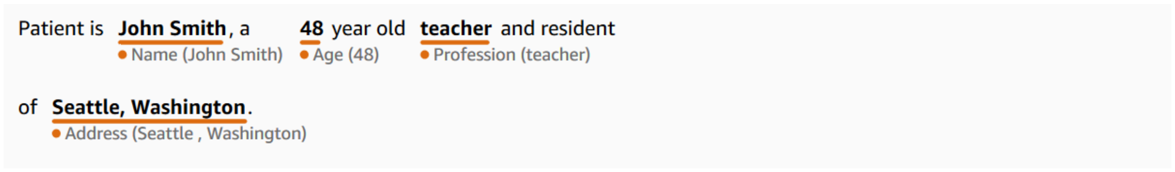
The `PROTECTED_HEALTH_INFORMATION` category detects the patient's personal information. It contains eight entity types. For complete information about the `PROTECTED_HEALTH_INFORMATION` category and how it is detected, see Detect PHI (p. 29).

Types

- `ADDRESS`: All geographical subdivisions of an address of any facility, named medical facilities, or wards within a facility.
- `AGE`: All components of age, spans of age, or any age mentioned in the clinical note of a patient or others. The default is in years unless otherwise noted.
- `EMAIL`: Any email address.
- `ID`: Any and all identification numbers that are tied to the patient. This includes patient specific numbers like the Social Security number, medical record number, certificate or license number, vehicle or device number, or any biometric numbers. It also includes the facility identification number, clinical trial number, place of care, or provider.
- `DATE`: Any date related to the patient or patient care.
- `NAME`: All names mentioned in the clinical note. Typically, names belonging to the patient, family, or provider.
- `PHONE_OR_FAX`: Any phone, fax, or pager number. Excludes named phone numbers, such as 1-800-QUIT-NOW and 911.
- `PROFESSION`: Any profession or employer mentioned in the clinical note that pertains to the patient or the patient's family. This does not refer to the profession of the clinician mentioned in the note.

Example

The text "Patient is John Smith, a 48-year old teacher and resident of Seattle, Washington." returns:


Patient is John Smith, a 48 year old teacher and resident
● Name (John Smith)
● Age (48)
● Profession (teacher)
of Seattle, Washington.
● Address (Seattle , Washington)

- "John Smith" is a `NAME` *type*.
- "48" is an `AGE` *type*.
- "teacher" is a `PROFESSION` *type*.
- "Seattle, Washington" is an `ADDRESS` *type*.

The operation returns the following JSON structure:

```
{
    "Entities": [
        {
            "Id": 0,
            "BeginOffset": 11,
            "EndOffset": 21,
            "Score": 0.9967977404594421,
            "Text": "John Smith",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "NAME",
            "Traits": []
        },
        {
            "Id": 1,
            "BeginOffset": 25,
            "EndOffset": 27,
            "Score": 0.9998422861099243,
            "Text": "48",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "AGE",
            "Traits": []
        },
        {
            "Id": 2,
            "BeginOffset": 37,
            "EndOffset": 44,
            "Score": 0.9079490900039673,
            "Text": "teacher",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "PROFESSION",
            "Traits": []
        },
        {
            "Id": 3,
            "BeginOffset": 61,
            "EndOffset": 80,
            "Score": 0.986108124256134,
            "Text": "Seattle, Washington",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "ADDRESS",
            "Traits": []
        }
    ],
    "UnmappedAttributes": []
}
```

Test, treatment, procedure category

The TEST_TREATMENT_PROCEDURE category detects the procedures used to determine a medical condition. It contains three entity types and two attributes. One or more attributes can be related to an entity of the TEST_NAME type.

Types

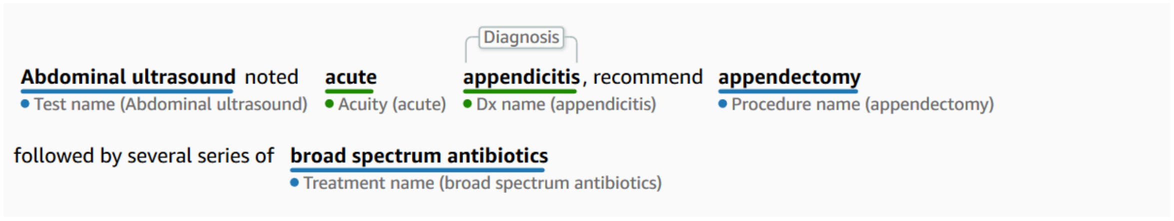
- PROCEDURE_NAME: Interventions as a one-time action performed on the patient to treat a medical condition or to provide patient care.
- TEST_NAME: Procedures performed on a patient for diagnostic, measurement, screening, or rating that might have a resulting value. This includes any procedure, evaluation, or rating to determine a diagnosis, to rule out a condition, or to scale or score a patient.
- TREATMENT_NAME: Interventions performed over a span of time for combating a disease or disorder. This includes groupings of medications, such as antivirals and vaccinations.

Attributes

- `TEST_VALUE`: The result of a test. Applies only to the `TEST_NAME` entity type.
- `TEST_UNIT`: The unit of measure that might accompany the value of the test. Applies only to the `TEST_NAME` entity type.

Example

The text "Abdominal ultrasound noted acute appendicitis, recommend appendectomy followed by several series of broad spectrum antibiotics" returns:


Diagnosis
Abdominal ultrasound noted
Test name (Abdominal ultrasound)
acute
Acuity (acute)
appendicitis, recommend
Dx name (appendicitis)
appendectomy
Procedure name (appendectomy)
followed by several series of
broad spectrum antibiotics
Treatment name (broad spectrum antibiotics)

- "Abdominal ultrasound" is a `TEST_NAME` *type*.
- "acute" is an `ACUITY` *type*.
- "appendicitis" is a `DX_NAME` *type*.
- `DIAGNOSIS` is a *trait* of the "appendicitis" *type*.
- "appendectomy" is a `PROCEDURE_NAME` *type*.
- "broad spectrum antibiotics" is a `TREATMENT_NAME` *type*.

The operation returns the following JSON structure:

```
{
    "Entities": [
        {
            "Id": 0,
            "BeginOffset": 0,
            "EndOffset": 20,
            "Score": 0.94855135679245,
            "Text": "Abdominal ultrasound",
            "Category": "TEST_TREATMENT_PROCEDURE",
            "Type": "TEST_NAME",
            "Traits": []
        },
        {
            "Id": 3,
            "BeginOffset": 27,
            "EndOffset": 32,
            "Score": 0.9067845940589905,
            "Text": "acute",
            "Category": "MEDICAL_CONDITION",
            "Type": "ACUITY",
            "Traits": []
        },
        {
            "Id": 4,
            "BeginOffset": 33,
            "EndOffset": 45,
            "Score": 0.9954161643981934,
            "Text": "appendicitis",
            "Category": "MEDICAL_CONDITION",
            "Type": "DX_NAME",
            "Traits": [
                {
                    "Name": "DIAGNOSIS",
                    "Score": 0.9528769254684448
                }
            ]
        },
        {
            "Id": 1,
            "BeginOffset": 57,
            "EndOffset": 69,
            "Score": 0.9957893490791321,
            "Text": "appendectomy",
            "Category": "TEST_TREATMENT_PROCEDURE",
            "Type": "PROCEDURE_NAME",
            "Traits": []
        },
        {
            "Id": 2,
            "BeginOffset": 100,
            "EndOffset": 126,
            "Score": 0.9437107443809509,
            "Text": "broad spectrum antibiotics",
            "Category": "TEST_TREATMENT_PROCEDURE",
            "Type": "TREATMENT_NAME",
            "Traits": []
        }
    ],
    "UnmappedAttributes": []
}
```

Detect PHI

Use the **DetectPHI** operation to detect Protected Health Information (PHI) data in the clinical text being examined. All five categories of entity are detected using the **DetectEntities** operation, but only information in the PHI category is detected by the **DetectPHI** operation. This allows for use cases where only this specific information is required. For information about information in the non-PHI categories, see Detect entities (p. 20).

Important
Amazon Comprehend Medical provides confidence scores that indicate the level of confidence in the accuracy of the detected entities. Evaluate these confidence scores and identify the right confidence threshold for your use case. For specific compliance use cases, we recommend that you use additional human review or other methods to confirm the accuracy of detected PHI.

Under the HIPAA act, PHI that is based on a list of 18 identifiers must be treated with special care. Amazon Comprehend Medical detects entities associated with these identifiers but these entities don't map 1:1 to the list specified by the Safe Harbor method. Not all identifiers are contained in unstructured clinical text, but Amazon Comprehend Medical does cover all of the relevant identifiers. These identifiers consist of data that can be used to identify an individual patient, including the following list. For more information, see Health Information Privacy on the *U.S. Government Health and Human Services* website.

Each PHI-related entity includes a score (`Score` in the response) that indicates the level of confidence Amazon Comprehend Medical has in the accuracy of the detection. Identify the right confidence threshold for your use case and filter out entities that do not meet it. When identifying occurrences of PHI, it may be better to use a low confidence threshold for filtering to capture more potential detected entities. This is especially true when not using the values of the detected entities in compliance use cases.

The following PHI-related entities can be detected by **DetectPHI** and **DetectEntities** operations:

Detected PHI Entities

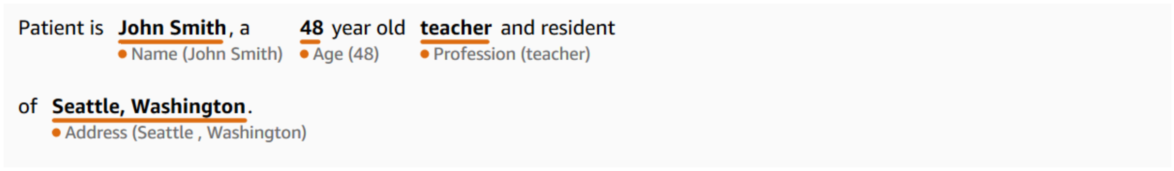
Entity	Description	HIPAA Category
AGE	All components of age, spans of age, and any age mentioned, be it patient or family member or others involved in the note. Default is in years unless otherwise noted.	3. Dates related to an individual
DATE	Any date related to patient or patient care.	3. Dates related to an individual
NAME	All names mentioned in the clinical note, typically belonging to patient, family, or provider.	1. Name
PHONE_OR_FAX	Any phone, fax, pager; excludes named phone numbers such as 1-800-QUIT-NOW as well as 911.	4. Phone number 5. FAX number
EMAIL	Any email address.	6. Email addresses
ID	Any sort of number associated with the identity of a patient. This includes their social security number, medical record number, facility identification number, clinical trial number, certificate or license number, vehicle or device number. It also includes biometric numbers, and numbers identifying the place of care or provider.	7. Social Security Number 8. Medical Record number 9. Health Plan number 10. Account numbers 11. Certificate/License numbers 12. Vehicle identifiers 13. Device numbers 16. Biometric information 18. Any other identifying characteristics
URL	Any web URL.	14. URLs
ADDRESS	This includes all geographical subdivisions of an address of any facility, named medical facilities, or wards within a facility.	2. Geographic location
PROFESSION	Includes any profession or employer mentioned in a note as it pertains to the patient or the patient's family.	18. Any other identifying characteristics

Example

The text "Patient is John Smith, a 48-year-old teacher and resident of Seattle, Washington." returns:

- "John Smith" as an *entity* of type `NAME` in the `PROTECTED_HEALTH_INFORMATION` category.
- "48" as an *entity* of type `AGE` in the `PROTECTED_HEALTH_INFORMATION` category.
- "teacher" as an *entity* of type `PROFESSION` (identifying characteristic) in the `PROTECTED_HEALTH_INFORMATION` category.
- "Seattle, Washington" as an `ADDRESS` *entity* in the `PROTECTED_HEALTH_INFORMATION` category.

In the Amazon Comprehend Medical console, this is shown like this:


Patient is **John Smith**, a **48** year old **teacher** and resident
● Name (John Smith) ● Age (48) ● Profession (teacher)

of **Seattle, Washington**.
● Address (Seattle , Washington)

When using the **DetectPHI** operation, the response appears like this. When you use the **StartPHIDetectionJob** operation, Amazon Comprehend Medical creates a file in the output location with this structure.

```
{
    "Entities": [
        {
            "Id": 0,
            "BeginOffset": 11,
            "EndOffset": 21,
            "Score": 0.997368335723877,
            "Text": "John Smith",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "NAME",
            "Traits": []
        },
        {
            "Id": 1,
            "BeginOffset": 25,
            "EndOffset": 27,
            "Score": 0.9998362064361572,
            "Text": "48",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "AGE",
            "Traits": []
        },
        {
            "Id": 2,
            "BeginOffset": 37,
            "EndOffset": 44,
            "Score": 0.8661606311798096,
            "Text": "teacher",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "PROFESSION",
            "Traits": []
        },
        {
            "Id": 3,
            "BeginOffset": 61,
            "EndOffset": 68,
            "Score": 0.9629441499710083,
            "Text": "Seattle",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "ADDRESS",
            "Traits": []
        },
        {
            "Id": 4,
            "BeginOffset": 78,
            "EndOffset": 88,
            "Score": 0.38217034935951233,
            "Text": "Washington",
            "Category": "PROTECTED_HEALTH_INFORMATION",
            "Type": "ADDRESS",
            "Traits": []
        }
    ],
    "UnmappedAttributes": []
}
```

Text analysis batch APIs

Use Amazon Comprehend Medical to analyze medical text stored in an Amazon S3 bucket. Analyze up to 10 GB of documents in one batch. You use the console to create and manage batch analysis jobs, or use batch APIs to detect medical entities, including protected health information (PHI). The APIs start, stop, list, and describe ongoing batch analysis jobs.

Pricing information for batch analysis and other Amazon Comprehend Medical operations can be found here.

Important notice

The batch analysis operations of Amazon Comprehend Medical are not a substitute for professional medical advice, diagnosis, or treatment. Identify the right confidence threshold for your use case, and use high confidence thresholds in situations that require high accuracy. For certain use cases, results should be reviewed and verified by appropriately trained human reviewers. All operations of Amazon Comprehend Medical should only be used in patient care scenarios after review for accuracy and sound medical judgment by trained medical professionals.

Performing batch analysis using the APIs

You can run a batch analysis job using either the Amazon Comprehend Medical console or the Amazon Comprehend Medical Batch APIs.

Prerequisites

When you are using the Amazon Comprehend Medical API, create an AWS Identity Access and Management (IAM) policy and attach it to an IAM role. To learn more about IAM roles and trust policies, see IAM Policies and Permissions.

1. Upload your data into an S3 bucket.
2. To start a new analysis job, use either the StartEntitiesDetectionV2Job operation or the StartPHIDetectionJob operation. When you start the job, tell Amazon Comprehend Medical the name of the input S3 bucket that contains the input files and designate the output S3 bucket to write the files after batch analysis.
3. Monitor the progress of the job by using the console or the DescribeEntitiesDetectionV2Job operation or the DescribePHIDetectionJob operation. Additionally, ListEntitiesDetectionV2Jobs and ListPHIDetectionJobs enable you to see the status of all ontology linking batch analysis jobs.

4. If you need to stop a job in progress, use StopEntitiesDetectionV2Job or StopPHIDetectionJob to stop analysis.
5. To view the results of your analysis job, see the output S3 bucket that you configured when you started the job.

Performing batch analysis using the console

1. Upload your data into an S3 bucket.
2. To start a new analysis job, select the type of analysis you will be performing. Then provide the name of the S3 bucket that contains the input files and the name of the S3 bucket where you want to send the output files.
3. Monitor the status of your job while it is ongoing. From the console, you are can view all batch analysis operations and their status, including when analysis was started and ended.
4. To see the results of your analysis job, see the output S3 bucket that you configured when you started the job.

IAM Policies for batch operations

The IAM role that calls the Amazon Comprehend Medical batch APIs must have a policy that grants access to the S3 buckets that contain the input and output files. It must also be assigned a trust relationship that enables the Amazon Comprehend Medical service to assume the role. To learn more about IAM roles and trust policies, see IAM Roles.

The role must have the following policy.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::input-bucket/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::input-bucket",
                "arn:aws:s3:::output-bucket",
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                " arn:aws:s3:::output-bucket/*"
            ],
            "Effect": "Allow"
        }
    ]
}
```

The role must have the following trust relationship.

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

Batch analysis output files

Amazon Comprehend Medical creates one output file for each input file in the batch. The file has the extension `.out`. Amazon Comprehend Medical first creates a directory in the output S3 bucket using the *`AwsAccountId`*-*`JobType`*-*`JobId`* as the name, and then writes all of the output files for the batch to this directory. Amazon Comprehend Medical creates this new directory so that output from one job does not overwrite the output of another.

The output from a batch operation produces the same output as a synchronous operation. For examples of the output generated by Amazon Comprehend Medical, see Detect entities (p. 20).

Each batch operation produces three manifest files that contain information about the job.

- `Manifest` – Summarizes the job. Provides information about the parameters used for the job, the total size of the job, and the number of files processed.
- `success` – Provides information about the files that were successfully processed. Includes the input and output file name and the size of the input file.
- `unprocessed` – Lists files that the batch job did not process. Typically this is because the file was added to the input directory after the batch job was started.

Amazon Comprehend Medical writes the files to the output directory that you specified for the batch job. The following sections show the structure of the manifest files.

Batch manifest file

The following is the JSON structure of the batch manifest file.

```
{
  "Summary" : {
    "Status" : "COMPLETED | FAILED | PARTIAL_SUCCESS | STOPPED",
    "JobType" : "DetectEntitiesJob | PHIDetection",
    "InputDataConfiguration" : {
      "Bucket" : "input bucket",
      "Path" : "path to files/account ID-job type-job ID"
    },
    "OutputDataConfiguration" : {
      "Bucket" : "output bucket",

      "Path" : "path to files"
    },
    "InputFileCount" : number of files in input bucket,
    "TotalMeteredCharacters" : total characters processed from all files,
    "UnprocessedFilesCount" : number of files not processed,
    "SuccessFilesCount" : total number of files processed,
    "TotalDurationSeconds" : time required for processing,
    "SuccessfulFilesListLocation" : "path to file",
    "UnprocessedFilesListLocation" : "path to file"
  }
}
```

Success manifest file

The following is the JSON structure of the file that contains information about successfully processed files.

```
{
        "Files": [{
                "Input": "input path/input file name",
                "Output": "output path/output file name",
                "InputSize": size in bytes of input file
        }, {
                "Input": "input path/input file name",
                "Output": "output path/output file name",
                "InputSize": size in bytes of input file
        }]
}
```

Unprocessed manifest file

The following is the JSON structure of the manifest file that contains information about unprocessed files.

```
{
        "Files": [
                "input path/input file name",
                "input path/input file name"
        ]
}
```

Ontology linking

Amazon Comprehend Medical enables you to detect entities in clinical text and link those entities to concepts in standardized medical ontologies, including the RxNorm and ICD-10-CM knowledge bases. Analysis can be performed both on single files or as a batch analysis on large documents or multiple files stored in an Amazon S3 bucket.

ICD-10-CM linking

Use **InferICD10CM** to detect possible medical conditions as entities and link them to codes from the 2021 version of the International Classification of Diseases, 10th Revision, Clinical Modification (ICD-10-CM). The ICD-10-CM is provided by the US Centers for Disease Control and Prevention (CDC).

For each detected potential medical condition, Amazon Comprehend Medical lists the matching ICD-10-CM codes and descriptions. They are listed in descending order of confidence along with the confidence scores. The scores indicate the confidence in the accuracy of the entities matched to the concepts. Use the ICD-10-CM codes for downstream analysis. Related information such as signs, symptoms, and negation are recognized as traits. Additional information such as anatomical designations and acuity are listed as attributes.

InferICD10CM is well suited for the following scenarios:

- Assistance for professional medical coding for patient records
- Clinical studies and trials
- Integration with a medical software system
- Early detection and diagnosis
- Population health management

Important notice

The **InferICD10CM** operation of Amazon Comprehend Medical is not a substitute for professional medical advice, diagnosis, or treatment. Identify the right confidence threshold for your use case, and use high confidence thresholds in situations that require high accuracy. All operations of Amazon Comprehend Medical should only be used in patient care scenarios after review for accuracy and sound medical judgment by trained medical professionals.

ICD-10-CM category

InferICD10CM detects entities in the `MEDICAL_CONDITION` category. Additional related information is also detected and linked as attributes or traits.

ICD-10-CM type

InferICD10CM detects entities of the types `DX_NAME` and `TIME_EXPRESSION`.

ICD-10-CM traits

InferICD10CM detects the following contextual information as traits:

- `Diagnosis:` An identification of a medical condition that is determined by evaluation of the symptoms.
- `Negation:` An indication that a medical condition is not present.
- `Sign:` A medical condition that is reported by the physician.
- `Symptom:` A medical condition reported by the patient.

ICD-10-CM attributes

InferICD10CM detects the following contextual information as attributes:

- `DIRECTION:` Directional terms. For example, left, right medial, lateral, upper, lower, posterior, anterior, distal, proximal, contralateral, bilateral, ipsilateral, dorsal, or ventral.
- `SYSTEM, ORGAN or SITE:` Anatomical location.
- `ACUITY:` Determination of disease instance, such as chronic, acute, sudden, persistent, or gradual. This only applies to the MEDICAL_CONDITION type.

Time expression category

The `TIME_EXPRESSION` category detects entities related to time. This includes entities such as dates and time expressions such as "three days ago," "today," "currently," "day of admission," "last month," or "16 days." Results in this category are only returned if they are associated with an entity. For example, "Yesterday, the patient was diagnosed with influenza " would return `Yesterday` as a `TIME_EXPRESSION` entity that overlaps with `DX_NAME` entity "influenza." However, "yesterday" would not be recognized as an entity in "yesterday, the patient walked their dog."

Types

The recognized type of `TIME_EXPRESSION` is `TIME_TO_DX_NAME`: The date a medical condition occurred. The attribute for this type is `DX_NAME`.

Relationship type

The `RELATIONSHIP_TYPE` refers to the relationship between an entity and an attribute. The recognized `RELATIONSHIP_TYPE` is: `OVERLAP` – The `TIME_EXPRESSION` concurs with the entity detected.

Input and response examples

The following example shows how the `InferICD10CM` operation works. The following is an example input text.

"The patient is a 71-year-old female patient of Dr. X. The patient presented to the emergency room last evening with approximately 7- to 8-day history of abdominal pain which has been persistent. She has had no nausea and vomiting, but has had persistent associated anorexia. She is passing flatus, but had some obstipation symptoms with the last bowel movement two days ago. She denies any bright red blood per rectum and no history of recent melena. Her last colonoscopy was approximately 5 years ago with Dr. Y. She has had no definite fevers or chills and no history of jaundice. The patient denies any significant recent weight loss."

For the input text, the **InferICD10CM** operation returns the following output (abbreviated for brevity).

```
{
    "Entities": [
        {
            "Id": 1,
            "Text": "abdominal pain",
            "Category": "MEDICAL_CONDITION",
            "Type": "DX_NAME",
            "Score": 0.9606665968894958,
            "BeginOffset": 153,
            "EndOffset": 167,
            "Attributes": [
                {
                    "Type": "ACUITY",
                    "Score": 0.764342725276947,
                    "RelationshipScore": 0.9999940395355225,
                    "Id": 2,
                    "BeginOffset": 183,
                    "EndOffset": 193,
                    "Text": "persistent",
                    "Traits": []
                }
            ],
            "Traits": [
                {
                    "Name": "SYMPTOM",
                    "Score": 0.7559975981712341
                }
            ],
            "ICD10CMConcepts": [
                {
                    "Description": "Unspecified abdominal pain",
                    "Code": "R10.9",
                    "Score": 0.7775180339813232
                },
                {
                    "Description": "Epigastric pain",
                    "Code": "R10.13",
                    "Score": 0.6876822710037231
                },
                {
                    "Description": "Lower abdominal pain, unspecified",
                    "Code": "R10.30",
                    "Score": 0.6758853197097778
                },
                {
                    "Description": "Generalized abdominal pain",
                    "Code": "R10.84",
                    "Score": 0.6746202707290649
                },
                {
                    "Description": "Upper abdominal pain, unspecified",
                    "Code": "R10.10",
                    "Score": 0.6702126860618591
                }
            ]
        }
...
    "ModelVersion": "0.1.0"
}
```

InferICD10CM also recognizes when an entity is negated in text. For instance, if a patient is not experiencing a symptom, both the symptom and negation are identified as traits and listed with a confidence score. Based on the input for the previous example, the symptom `Nausea` would be listed under `NEGATION` because the patient isn't experiencing nausea.

```
    {
            "Id": 3,
            "Text": "nausea",
            "Category": "MEDICAL_CONDITION",
            "Type": "DX_NAME",
            "Score": 0.9962648749351501,
            "BeginOffset": 210,
            "EndOffset": 216,
            "Attributes": [],
            "Traits": [
                {
                    "Name": "SYMPTOM",
                    "Score": 0.9296342730522156
                },
                {
                    "Name": "NEGATION",
                    "Score": 0.9620923399925232
                }
            ],
            "ICD10CMConcepts": [
                {
                    "Description": "Nausea with vomiting, unspecified",
                    "Code": "R11.2",
                    "Score": 0.8000147938728333
                },
                {
                    "Description": "Nausea",
                    "Code": "R11.0",
                    "Score": 0.7653312683105469
      }
```

RxNorm linking

Use the **InferRxNorm** operation to identify medications that are listed in a patient record as entities. The operation also links those entities to concept identifiers (RxCUI) from the RxNorm database from the National Library of Medicine. The source for each RxCUI is the UMLS Metathesaurus 2020AB. Each RxCUI is unique for different strengths and dose forms. Amazon Comprehend Medical lists the top potentially matching RxCUIs for each medication that it detects in descending order by confidence score. Use the RxCUI codes for downstream analysis not possible with unstructured text. Related information such as strength, frequency, dose, dose form, and route of administration are listed as attributes in JSON format.

InferRxNorm is well suited for the following scenarios:

- Screening for medications the patient has taken.
- Preventing potential negative reactions between newly prescribed drugs and drugs the patient is already taking.
- Screening for inclusion in clinical trials based on drug history using the RxCUI.
- Checking whether the dosage and frequency of a drug is appropriate.
- Screening for uses, indications, and side effects of drugs.
- Population health management.

Important notice

The **InferRxNorm** operation of Amazon Comprehend Medical is not a substitute for professional medical advice, diagnosis, or treatment. Identify the right confidence threshold for your use case, and use high confidence thresholds in situations that require high accuracy. All operations of Amazon Comprehend

Medical should only be used in patient care scenarios after review for accuracy and sound medical judgment by trained medical professionals.

RxNorm category

InferRxNorm detects entities in the `MEDICATION` category. Additional related information is also detected and linked as attributes or traits.

RxNorm types

The types of entity within the `Medication` category are:

- `BRAND_NAME`: The copyrighted brand name of the medication or therapeutic agent.
- `GENERIC_NAME`: Non-brand name, ingredient name, or formula mixture of the medication or therapeutic agent.

RxNorm attributes

- `DOSAGE`: The amount of medication ordered.
- `DURATION`: How long the medication should be administered.
- `FORM`: The form of the medication.
- `FREQUENCY`: How often to administer the medication.
- `RATE`: Primarily for medication infusions or IVs, the administration rate of the medication.
- `ROUTE_OR_MODE`: The administration method of a medication.
- `STRENGTH`: The medication strength.

RxNorm trait

- `NEGATION`: Any indication that the patient is not taking a medication.

Input and response examples

This section shows an example of using the **InferRxNorm** operation. The following text is an example of the input JSON:

```
{
    "Text": "fluoride topical ( fluoride 1.1 % topical gel ) 1 application
        Topically daily Brush onto teeth before bed time , spit , do not rinse,
        eat or drink for 20-30 minutes"
}
```

Based on the input above, you will receive the following result:

```
[
    {
        [
    {
        "Id": 0,
        "Text": "fluoride",
        "Category": "MEDICATION",
        "Type": "GENERIC_NAME",
        "Score": 0.8712447285652161,
        "BeginOffset": 15,
        "EndOffset": 23,
        "Attributes": [
            {
                "Type": "ROUTE_OR_MODE",
                "Score": 0.8275620341300964,
                "RelationshipScore": 0.9997424483299255,
                "Id": 1,
                "BeginOffset": 24,
                "EndOffset": 31,
                "Text": "topical",
                "Traits": []
            },
            {
                "Type": "STRENGTH",
                "Score": 0.9978499412536621,
                "RelationshipScore": 0.9971910119056702,
                "Id": 3,
                "BeginOffset": 43,
                "EndOffset": 48,
                "Text": "1.1 %",
                "Traits": []
            },
            {
                "Type": "FORM",
                "Score": 0.9408299326896667,
                "RelationshipScore": 0.9997219443321228,
                "Id": 4,
                "BeginOffset": 49,
                "EndOffset": 60,
                "Text": "topical gel",
                "Traits": []
            },
            {
                "Type": "DOSAGE",
                "Score": 0.9950674772262573,
                "RelationshipScore": 0.9999516010284424,
                "Id": 5,
                "BeginOffset": 63,
                "EndOffset": 76,
                "Text": "1 application",
                "Traits": []
            },
            {
                "Type": "ROUTE_OR_MODE",
                "Score": 0.9967094659805298,
                "RelationshipScore": 0.9985153079032898,
                "Id": 6,
                "BeginOffset": 77,
                "EndOffset": 86,
                "Text": "Topically",
                "Traits": []
            },
            {
                "Type": "FREQUENCY",
                "Score": 0.9728374481201172,
                "RelationshipScore": 0.9999150037765503,
                "Id": 7,
                "BeginOffset": 87,
                "EndOffset": 92,
                "Text": "daily",
                "Traits": []
            }
        ],
        "Traits": [],
        "RxNormConcepts": [
            {
                "ConceptName": "Sodium Fluoride 0.011 MG/MG Oral Gel",
                "ConceptId": "1486566",
                "Score": 0.9764410257339478
            },
            {
                "ConceptName": "Sodium Fluoride 0.011 MG/MG Oral Gel [FluoriSHIELD]",
                "ConceptId": "1546240",
                "Score": 0.8418328762054443
            },
            {
                "ConceptName": "Hydrofluoric Acid 0.01 MG/MG / Phosphoric acid 0.0112 MG/
MG / Sodium Fluoride 0.018 MG/MG Oral Gel",
                "ConceptId": "1297381",
                "Score": 0.7141376733779907
            },
            {
                "ConceptName": "Stannous Fluoride 0.014 MG/MG Oral Gel",
                "ConceptId": "701932",
                "Score": 0.49637168645858765
            },
            {
                "ConceptName": "Sodium Fluoride 0.011 MG/MG Oral Gel [Neutracare]",
                "ConceptId": "1596938",
                "Score": 0.478127658367157
            }
        ]
    },
    {
        "Id": 2,
        "Text": "fluoride",
        "Category": "MEDICATION",
        "Type": "GENERIC_NAME",
        "Score": 0.9971680045127869,
        "BeginOffset": 34,
        "EndOffset": 42,
        "Attributes": [],
        "Traits": [],
        "RxNormConcepts": [
            {
                "ConceptName": "Fluorine",
                "ConceptId": "1310123",
                "Score": 0.9384168982505798
            },
            {
                "ConceptName": "Sodium Fluoride",
                "ConceptId": "9873",
                "Score": 0.9174549579620361
            },
            {
                "ConceptName": "magnesium fluoride",
                "ConceptId": "1435860",
                "Score": 0.8124921917915344
            },
            {
                "ConceptName": "Acidulated Phosphate Fluoride",
                "ConceptId": "236",
                "Score": 0.4174852967262268
            },
            {
                "ConceptName": "Sodium Fluoride 0.0025 MG/MG Toothpaste",
                "ConceptId": "1044547",
                "Score": 0.3444318175315857
            }
        ]
    }
```

```
]
```

InferRxNorm also recognizes negation, as shown in the following input example.

```
{
    "Text": "patient no longer taking warfarin"
}
```

Based on the input above, you receive the following result.

```
[
    {
        "Id": 0,
        "Text": "warfarin",
        "Category": "MEDICATION",
        "Type": "GENERIC_NAME",
        "Score": 0.9997209906578064,
        "BeginOffset": 25,
        "EndOffset": 33,
        "Attributes": [],
        "Traits": [
            {
                "Name": "NEGATION",
                "Score": 0.7574219703674316
            }
        ],
        "RxNormConcepts": [
            {
                "ConceptName": "Warfarin",
                "ConceptId": "11289",
                "Score": 0.9439864158630371
            },
            {
                "ConceptName": "Warfarin Sodium 2 MG Oral Tablet",
                "ConceptId": "855302",
                "Score": 0.5045595169067383
            },
            {
                "ConceptName": "Warfarin Sodium 3 MG Oral Tablet",
                "ConceptId": "855318",
                "Score": 0.4175175130367279
            },
            {
                "ConceptName": "Warfarin Sodium 6 MG Oral Tablet",
                "ConceptId": "855338",
                "Score": 0.3222610652446747
            },
            {
                "ConceptName": "Warfarin Sodium 2.5 MG Oral Tablet [Coumadin]",
                "ConceptId": "855314",
                "Score": 0.1963760107755661
            }
        ]
    }
]
```

Ontology linking batch analysis

Use Amazon Comprehend Medical to detect entities in clinical text stored in an Amazon Simple Storage Service (Amazon S3) bucket and to link those entities to standardized ontologies. You can use ontology linking batch analysis to analyze either a collection of documents or a single document with up to 20,000 characters. By using either the console or the **InferRxNorm** and **InferICD10CM** ontology linking batch APIs, you can perform operations to start, stop, list, and describe ongoing batch analysis jobs.

For pricing information for batch analysis and other Amazon Comprehend Medical operations, see Amazon Comprehend Medical Pricing.

Important notice

Amazon Comprehend Medical ontology linking batch analysis operations are not intended as a substitute for professional medical advice, diagnosis, or treatment. Identify the right confidence threshold for your use case. In situations that require great accuracy, use high confidence thresholds. Use all Amazon Comprehend Medical operations in patient care scenarios only after the results have been reviewed for accuracy and sound medical judgment by trained medical professionals.

Performing batch analysis

You can run a batch analysis job using either the Amazon Comprehend Medical console or the Amazon Comprehend Medical Batch APIs.

Performing batch analysis using the APIs

Prerequisites

When you are using the Amazon Comprehend Medical API, create an AWS Identity Access and Management (IAM) policy and attach it to an IAM role. To learn more about IAM roles and trust policies, see IAM Policies and Permissions.

1. Upload your data into an S3 bucket.
2. To start a new analysis job, use either the **StartICD10CMInferenceJob** or the **StartRxNormInferenceJob** operation. Provide the name of the S3 bucket that contains the input files and the name of the S3 bucket where you want to send the output files.
3. Monitor the progress of the job by using **DescribeICD10CMInferenceJob** or **DescribeRxNormInferenceJob** operations. Additionally, **ListICD10CMInferenceJobs** and **ListRxNormInferenceJobs** enable you to see the status of all ontology linking batch analysis jobs.
4. If you need to stop a job in progress, use **StopICD10CMInferenceJob** or **StopRxNormInferenceJob** to stop analysis.
5. To view the results of your analysis job, see the output S3 bucket that you configured when you started the job.

Performing batch analysis using the console

1. Upload your data into an S3 bucket.
2. To start a new analysis job, select the type of analysis you will be performing. Then provide the name of the S3 bucket that contains the input files and the name of the S3 bucket where you want to send the output files.
3. Monitor the status of your job while it is ongoing. From the console, you are can view all batch analysis operations and their status, including when analysis was started and ended.
4. To see the results of your analysis job, see the output S3 bucket that you configured when you started the job.

IAM policies for batch operations

The IAM role that calls the Amazon Comprehend Medical batch APIs must have a policy that grants access to the S3 buckets that contain the input and output files. It must also be assigned a trust relationship that enables the Amazon Comprehend Medical service to assume the role. To learn more about IAM roles and trust policies, see IAM Roles.

The role must have the following policy.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::input-bucket/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::input-bucket",
                "arn:aws:s3:::output-bucket",
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                " arn:aws:s3:::output-bucket/*"
            ],
            "Effect": "Allow"
        }
    ]
}
```

The role must have the following trust relationship.

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

Batch analysis output files

Amazon Comprehend Medical creates one output file for each input file in the batch. The file has the extension `.out`. Amazon Comprehend Medical first creates a directory in the output S3 bucket using the *AwsAccountId*-*JobType*-*JobId* as the name, and then writes all of the output files for the batch to this directory. Amazon Comprehend Medical creates this new directory so that output from one job doesn't overwrite the output of another.

A batch operation produces the same output as a synchronous operation.

Each batch operation produces three manifest files that contain information about the job:

- `Manifest` – Summarizes the job. Provides information about the parameters used for the job, the total size of the job, and the number of files processed.
- `Success` – Provides information about the files that were successfully processed. Includes the input and output file name and the size of the input file.
- `Unprocessed` – Lists files that the batch job did not process. Typically, a file isn't processed because it was added to the input directory after the batch job started.

Amazon Comprehend Medical writes these files to the output directory that you specified for the batch job. The following sections show the structure of the manifest files.

Batch manifest file

The following is the JSON structure of the batch manifest file.

```
{
  "Summary" : {
    "Status" : "COMPLETED | FAILED | PARTIAL_SUCCESS | STOPPED",
    "JobType" : "ICD10CMInferenceJob | RxNormInferenceJob",
    "InputDataConfiguration" : {
      "Bucket" : "input bucket",
      "Path" : "path to files/account ID-job type-job ID"
    },
    "OutputDataConfiguration" : {
      "Bucket" : "output bucket",
      "Path" : "path to files"
    },
    "InputFileCount" : number of files in input bucket,
    "TotalMeteredCharacters" : total characters processed from all files,
    "UnprocessedFilesCount" : number of files not processed,
    "SuccessFilesCount" : total number of files processed,
    "TotalDurationSeconds" : time required for processing,
    "SuccessfulFilesListLocation" : "path to file",
    "UnprocessedFilesListLocation" : "path to file"
  }
}
```

Success manifest file

The following is the JSON structure of the file that contains information about successfully processed files.

```
{
        "Files": [{
               "Input": "input path/input file name",
               "Output": "output path/output file name",
               "InputSize": size in bytes of input file
        }, {
               "Input": "input path/input file name",
               "Output": "output path/output file name",
               "InputSize": size in bytes of input file
        }]
}
```

Unprocessed manifest file

The following is the JSON structure of the manifest file that contains information about unprocessed files.

```
{
        "Files": [
                "input path/input file name",
                "input path/input file name"
        ]
}
```

Security in Amazon Comprehend Medical

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to Amazon Comprehend Medical, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using Comprehend Medical. The following topics show you how to configure Comprehend Medical to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your Comprehend Medical resources.

Topics

Data protection in Amazon Comprehend Medical

The AWS shared responsibility model applies to data protection in Amazon Comprehend Medical. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. This content includes the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM). That way each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We recommend TLS 1.2 or later.

- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form fields such as a **Name** field. This includes when you work with Comprehend Medical or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Identity and access management in Amazon Comprehend Medical

Access to Comprehend Medical requires credentials that AWS can use to authenticate your requests. Those credentials must have permissions to access Comprehend Medical actions. AWS Identity and Access Management (IAM) can help secure your resources by controlling who can access them. The following sections provide details on how you can use IAM with Comprehend Medical.

Authentication

Authentication

You can access AWS as any of the following types of identities:

- **AWS account root user** – When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the best practice of using the root user only to create your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.
- **IAM user** – An IAM user is an identity within your AWS account that has specific custom permissions (for example, permissions to create in Amazon Comprehend Medical). You can use an IAM user name and password to sign in to secure AWS webpages like the AWS Management Console, AWS Discussion Forums, or the AWS Support Center.

  In addition to a user name and password, you can also generate access keys for each user. You can use these keys when you access AWS services programmatically, either through one of the several SDKs or by using the AWS Command Line Interface (CLI). The SDK and CLI tools use the access keys to cryptographically sign your request. If you don't use AWS tools, you must sign the request yourself. Amazon Comprehend Medical supports *Signature Version 4*, a protocol for authenticating inbound API requests. For more information about authenticating requests, see Signature Version 4 Signing Process in the *AWS General Reference*.

- **IAM role** – An IAM role is an IAM identity that you can create in your account that has specific permissions. An IAM role is similar to an IAM user in that it is an AWS identity with permissions policies that determine what the identity can and cannot do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. Also, a role does not have standard long-term credentials such as a password or access keys associated with it. Instead, when you assume a role, it provides you with temporary security credentials for your role session. IAM roles with temporary credentials are useful in the following situations:

  - **Federated user access** – Instead of creating an IAM user, you can use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as *federated users*. AWS assigns a role to a federated user when access is requested through an identity provider. For more information about federated users, see Federated users and roles in the *IAM User Guide*.

  - **AWS service access** – A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Creating a role to delegate permissions to an AWS service in the *IAM User Guide*.

  - **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances in the *IAM User Guide*.

Access Control

You must have valid credentials to authenticate your requests. The credentials must have permissions to call an Amazon Comprehend Medical action.

The following sections describe how to manage permissions for Amazon Comprehend Medical. We recommend that you read the overview first.

- Overview of managing access permissions to Amazon Comprehend Medical resources (p. 50)
- Using Identity-Based policies (IAM policies) for Amazon Comprehend Medical (p. 52)

Overview of managing access permissions to Amazon Comprehend Medical resources

Permissions policies govern the access to an action. An account administrator attaches permissions policies to IAM identities to manage access to actions. IAM identities include users, groups, and roles.

> **Note**
> An *account administrator* (or administrator user) is a user with administrator privileges. For more information, see IAM Best Practices in the *IAM User Guide*.

When you grant permissions, you decide both who and what actions get the permissions.

Topics

Managing access to actions

A *permissions policy* describes who has access to what. The following section explains the options for permissions policies.

> **Note**
> This section explains IAM in the context of Amazon Comprehend Medical. It doesn't provide detailed information about the IAM service. For more about IAM, see What Is IAM? in the *IAM User Guide*. For information about IAM policy syntax and descriptions, see AWS IAM Policy Reference in the *IAM User Guide*.

Policies attached to an IAM identity are *identity-based* policies. Policies attached to a resource are *resource-based* policies. Amazon Comprehend Medical supports only identity-based policies.

Identity-based policies (IAM policies)

You can attach policies to IAM identities. Here are two examples.

- **Attach a permissions policy to a user or a group in your account**. To allow a user or a group of users to call an Amazon Comprehend Medical action, attach a permissions policy to a user. Attach a policy to a group that contains the user.
- **Attach a permissions policy to a role to grant cross-account permissions**. To grant cross-account permissions, attach an identity-based policy to an IAM role. For example, the administrator in Account A can create a role to grant cross-account permissions to another account. In this example, call it Account B, which could also be an AWS service.
  1. Account A administrator creates an IAM role and attaches a policy to the role that grants permissions to resources in Account A.
  2. Account A administrator attaches a trust policy to the role. The policy identifies Account B as the principal who can assume the role.
  3. Account B administrator can then delegate permissions to assume the role to any users in Account B. This allows users in Account B to create or access resources in Account A. If you want to grant an AWS service the permissions to assume the role, the principal in the trust policy can also be an AWS service principal.

  For more information about using IAM to delegate permissions, see Access Management in the *IAM User Guide*.

For more information about using identity-based policies with Amazon Comprehend Medical, see Using Identity-Based policies (IAM policies) for Amazon Comprehend Medical (p. 52). For more information about users, groups, roles, and permissions, see Identities (Users, Groups, and Roles) in the *IAM User Guide*.

Resource-based policies

Other services, such as AWS Lambda, support resource-based permissions policies. For example, you can attach a policy to an S3 bucket to manage access permissions to that bucket. Amazon Comprehend Medical doesn't support resource-based policies.

Specifying policy elements: actions, effects, and principals

Amazon Comprehend Medical defines a set of API operations. To grant permissions for these API operations, Amazon Comprehend Medical defines a set of actions that you can specify in a policy.

The four items here are the most basic policy elements.

- **Resource** – In a policy, use an Amazon Resource Name (ARN) to identify the resource to which the policy applies. For Amazon Comprehend Medical, the resource is always `"*"`.
- **Action** – Use action keywords to identify operations that you want to allow or deny. For example, depending on the specified effect, `comprehendmedical:DetectEntities` either allows or denies the user permission to perform the Amazon Comprehend Medical `DetectEntities` operation.
- **Effect** – Specify the effect of the action that occurs when the user requests the specific action—either allow or deny. If you don't explicitly grant access to (allow) a resource, access is implicitly denied. You can also explicitly deny access to a resource. You might do this to make sure that a user cannot access the resource, even if a different policy grants access.
- **Principal** – In identity-based policies, the user that the policy is attached to is the implicit principal.

To learn more about IAM policy syntax and descriptions, see AWS IAM Policy Reference in the *IAM User Guide*.

For a table showing all of the Amazon Comprehend Medical API actions, see Amazon Comprehend Medical API Permissions: actions, resources, and conditions reference (p. 57).

Specifying conditions in a policy

When you grant permissions, you use the IAM policy language to specify the conditions under which a policy should take effect. For example, you might want a policy to be applied only after a specific date. For more information about specifying conditions in a policy language, see Condition in the *IAM User Guide*.

AWS provides a set of predefined condition keys for all AWS services that support IAM for access control. For example, you can use the `aws:userid` condition key to require a specific AWS ID when requesting an action. For more information and a complete list of AWS keys, see Available Keys for Conditions in the *IAM User Guide*.

Amazon Comprehend Medical does not provide any additional condition keys.

Using Identity-Based policies (IAM policies) for Amazon Comprehend Medical

This topic shows example identity-based policies. The examples show how an account administrator can attach permissions policies to IAM identities. This enables users, groups, and roles to perform Amazon Comprehend Medical actions.

Important
To understand permissions, we recommend Overview of managing access permissions to Amazon Comprehend Medical resources (p. 50).

This example policy is required to use the Amazon Comprehend Medical document analysis actions.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Sid": "AllowDetectActions",
      "Effect": "Allow",
      "Action": [
                "comprehendmedical:DetectEntitiesV2",
                "comprehendmedical:DetectEntities",
                "comprehendmedical:DetectPHI",

                "comprehendmedical:StartEntitiesDetectionV2Job",
                "comprehendmedical:ListEntitiesDetectionV2Jobs",
                "comprehendmedical:DescribeEntitiesDetectionV2Job",
                "comprehendmedical:StopEntitiesDetectionV2Job",

                "comprehendmedical:StartPHIDtectionJob",
                "comprehendmedical:ListPHIDetectionJobs",
                "comprehendmedical:DescribePHIDetectionJob",
                "comprehendmedical:StopPHIDetectionJob",

                "comprehendmedical:StartRxNormInferenceJob",
                "comprehendmedical:ListRxNormInferenceJobs",
                "comprehendmedical:DescribeRxNormInferenceJob",
                "comprehendmedical:StopRxNormInferenceJob",

                "comprehendmedical:StartICD10CMInferenceJob",
                "comprehendmedical:ListICD10CMInferenceJobs",
                "comprehendmedical:DescribeICD10CMInferenceJob",
                "comprehendmedical:StopICD10CMInferenceJob",

                "comprehendmedical:InferRxNorm",
                "comprehendmedical:InferICD10CM",

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

The policy has one statement that grants permission to use the `DetectEntities` and `DetectPHI` actions.

The policy doesn't specify the `Principal` element because you don't specify the principal who gets the permission in an identity-based policy. When you attach a policy to a user, the user is the implicit principal. When you attach a policy to an IAM role, the principal identified in the role's trust policy gets the permission.

To see all the Amazon Comprehend Medical API actions and the resources that they apply to, see Amazon Comprehend Medical API Permissions: actions, resources, and conditions reference (p. 57).

Permissions required to use the Amazon Comprehend Medical console

The permissions reference table lists the Amazon Comprehend Medical API operations and shows the required permissions for each operation. For more information, about Amazon Comprehend Medical API permissions, see Amazon Comprehend Medical API Permissions: actions, resources, and conditions reference (p. 57).

To use the Amazon Comprehend Medical console, grant permissions for the actions shown in the following policy.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "iam:CreateRole",
            "iam:CreatePolicy",
            "iam:AttachRolePolicy"
         ],
         "Resource": "*"
      },
      {
         "Effect": "Allow",
         "Action": "iam:PassRole",
         "Resource": "*",
         "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "comprehendmedical.amazonaws.com"
                }
         }
      }
   ]
}
```

The Amazon Comprehend Medical console needs these permissions for the following reasons:

- `iam` permissions to list the available IAM roles for your account.
- `s3` permissions to access the Amazon S3 buckets and objects that contain the data.

When you create an asynchronous batch job using the console, you can also create an IAM role for your job. To create an IAM role using the console, users must be granted the additional permissions shown here to create IAM roles and policies, and to attach policies to roles.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "iam:CreateRole",
        "iam:CreatePolicy",
        "iam:AttachRolePolicy"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

The Amazon Comprehend Medical console needs these permissions to create roles and policies and to attach roles and policies. The `iam:PassRole` action enables the console to pass the role to Amazon Comprehend Medical.

AWS managed (predefined) policies for Amazon Comprehend Medical

AWS addresses many common use cases by providing standalone IAM policies that are created and administered by AWS. These AWS managed policies grant necessary permissions for common use cases so that you can avoid having to investigate what permissions are needed. For more information, see AWS Managed Policies in the *IAM User Guide*.

The following AWS managed policy, which you can attach to users in your account, is specific to Amazon Comprehend Medical.

- **ComprehendMedicalFullAccess** – Grants full access to Amazon Comprehend Medical resources. Includes permission to list and get IAM roles.

You must apply the following additional policy to any user using Amazon Comprehend Medical:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*"
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "comprehendmedical.amazonaws.com"
                }
            }
        }
    ]
}
```

You can review the managed permissions policies by signing in to the IAM console and searching for specific policies there.

These policies work when you are using AWS SDKs or the AWS CLI.

You can also create your own IAM policies to allow permissions for Amazon Comprehend Medical actions and resources. You can attach these custom policies to the IAM users or groups that require them.

Role-based Permissions required for batch operations

To use the Amazon Comprehend Medical asynchronous operations, grant Amazon Comprehend Medical access to the Amazon S3 bucket that contains your document collection. Do this by creating a data access role in your account to trust the Amazon Comprehend Medical service principal. For more information about creating a role, see Creating a Role to Delegate Permissions to an AWS Service in the *AWS Identity and Access Management User Guide*.

The following is the role's trust policy.

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

After you have created the role, create an access policy for it. The policy should grant the Amazon S3 `GetObject` and `ListBucket` permissions to the Amazon S3 bucket that contains your input data. It also grants permissions for the Amazon S3 `PutObject` to your Amazon S3 output data bucket.

The following example access policy contains those permissions.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::input bucket/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::input bucket"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::output bucket/*"
            ],
            "Effect": "Allow"
        }
    ]
}
```

Customer managed policy examples

In this section, you can find example user policies that grant permissions for various Amazon Comprehend Medical actions. These policies work when you are using AWS SDKs or the AWS CLI. When you are using the console, you must grant permissions to all the Amazon Comprehend Medical APIs. This is discussed in Permissions required to use the Amazon Comprehend Medical console (p. 53).

Note
All examples use the us-east-2 Region and contain fictitious account IDs.

Examples

Example 1: Allow all Amazon Comprehend Medical actions

After you sign up for AWS, you create an administrator to manage your account, including creating users and managing their permissions.

You can choose to create a user who has permissions for all Amazon Comprehend actions. Think of this user as a service-specific administrator for working with Amazon Comprehend. You can attach the following permissions policy to this user.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Sid": "AllowAllComprehendMedicalActions",
      "Effect": "Allow",
      "Action": [
         "comprehendmedical:*"],
      "Resource": "*"
      }
   ]
}
```

Example 2: Allow only DetectEntities actions

The following permissions policy grants user permissions to detect entities in Amazon Comprehend Medical, but not to detect PHI operations.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Sid": "AllowDetectEntityActions",
      "Effect": "Allow",
      "Action": [
             "comprehendedical:DetectEntities"
          ],
         "Resource": "*"
         ]
     }
    ]
}
```

Amazon Comprehend Medical API Permissions: actions, resources, and conditions reference

Use the following table as a reference when setting up Access Control (p. 50) and writing a permissions policy that you can attach to an IAM identity (an identity-based policy). The list includes each Amazon Comprehend Medical API operation, the corresponding action for which you can grant permissions to perform the action, and the AWS resource for which you can grant the permissions. You specify the actions in the policy's `Action` field, and you specify the resource value in the policy's `Resource` field.

To express conditions, you can use AWS condition keys in your Amazon Comprehend Medical policies. For a complete list of keys, see Available Keys in the *IAM User Guide*.

Note
To specify an action, use the `comprehendmedical:` prefix followed by the API operation name, for example, `comprehendmedical:DetectEntities`.

Logging Amazon Comprehend Medical API calls by using AWS CloudTrail

Amazon Comprehend Medical is integrated with AWS CloudTrail. CloudTrail is a service that provides a record of actions taken by a user, role, or an AWS service from within Amazon Comprehend Medical. CloudTrail captures all API calls for Amazon Comprehend Medical as events. The calls captured include calls from the Amazon Comprehend Medical console and code calls to the Amazon Comprehend Medical API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for Amazon Comprehend Medical. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine several things such as:

- The request that was made to Amazon Comprehend Medical
- The IP address from which the request was made
- Who made the request
- When the request was made
- Other details

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

Amazon Comprehend Medical information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in Amazon Comprehend Medical, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History.

For an ongoing record of events in your AWS account, including events for Amazon Comprehend Medical, create a trail. A *trail* enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- Overview for Creating a Trail
- CloudTrail Supported Services and Integrations
- Configuring Amazon SNS Notifications for CloudTrail
- Receiving CloudTrail Log Files from Multiple Regions and Receiving CloudTrail Log Files from Multiple Accounts

All Amazon Comprehend Medical actions are logged by CloudTrail and are documented in the Amazon Comprehend Medical API Reference. For example, calls to the `DetectEntitiesV2`, `DetectPHI` and `ListEntitiesDetectionV2Jobs` actions generate entries in the CloudTrail log files.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or AWS Identity and Access Management (IAM) user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the CloudTrail userIdentity Element.

Understanding Amazon Comprehend Medical log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source. The event includes information about the requested action, such as the date and time or request parameters. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the `DetectEntitiesV2` action.

```
        {
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/Mateo_Jackson",
        "accountId": "123456789012",
        "accessKeyId": "ASIAXHKUFODNN8EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::123456789012:user/Mateo_Jackson",
                "accountId": "123456789012",
                "userName": "Mateo_Jackson"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2019-09-27T20:07:27Z"
            }
        }
    },
    "eventTime": "2019-09-27T20:10:26Z",
    "eventSource": "comprehendmedical.amazonaws.com",
    "eventName": "DetectEntitiesV2",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "702.21.198.166",
    "userAgent": "aws-internal/3 aws-sdk-java/1.11.590
Linux/4.9.184-0.1.ac.235.83.329.metal1.x86_64 OpenJDK_64-Bit_Server_VM/25.212-b03
java/1.8.0_212 vendor/Oracle_Corporation",
    "requestParameters": null,
    "responseElements": null,
    "requestID": "8d85f2ec-EXAMPLE",
    "eventID": "ae9be9b1-EXAMPLE",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```

Compliance validation for Amazon Comprehend Medical

Third-party auditors assess the security and compliance of Amazon Comprehend Medical as part of multiple AWS compliance programs. These include PCI, FedRAMP, HIPAA, and others. You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using Comprehend Medical is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.
- Architecting for HIPAA Security and Compliance Whitepaper – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.
- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
- AWS Config – This AWS service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- AWS Security Hub – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.

For a list of AWS services in scope of specific compliance programs, see AWS Services in Scope by Compliance Program. For general information, see AWS Compliance Programs.

Resilience in Amazon Comprehend Medical

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability Zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

Infrastructure security in Amazon Comprehend Medical

As a managed service, Amazon Comprehend Medical is protected by the AWS global network security procedures that are described in the Amazon Web Services: Overview of Security Processes whitepaper.

To access Comprehend Medical through the network, you use AWS published API calls. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS), such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems, such as Java 7 and later, support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an AWS Identity and Access Management (IAM) principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

Guidelines and quotas

Keep in mind the following information when using Amazon Comprehend Medical.

Important notice

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. For certain use cases, results should be reviewed and verified by appropriately trained human reviewers. Use Amazon Comprehend Medical in patient care scenarios only after trained medical professionals have reviewed results for accuracy and sound medical judgment.

Supported regions

For a list of AWS Regions where Amazon Comprehend Medical is available, see AWS Regions and Endpoints in the *Amazon Web Services General Reference*.

Throttling

For information about throttling and quotas for Amazon Comprehend Medical, and to request a quota increase, see AWS Service Quotas.

Overall quotas

Character encoding for Amazon Comprehend Medical is in UTF-8. Amazon Comprehend Medical operations have the following quotas for transactions per second(TPS) or characters per second (CPS):

Resource	Default
Transactions per second (TPS) for the `DetectEntities-v2`, `DetectEntities`, `DetectPHI`, `InferRxNorm`, and `InferICD10CM` operations	100
Characters per second (CPS) for the `DetectEntities-v2`, `DetectEntities`, `DetectPHI`, `InferRxNorm`, and `InferICD10CM` operations	40,000
Transactions per second (TPS) for the `StartEntitiesDetectionV2Job`, `StartPHIDetectionJob`, `StopEntitiesDetectionV2Job`, `StopPHIDetectionJob`, `StartICD10CMInferenceJob`, `StartRxNormInferenceJob`, `StopICD10CMInferenceJob`, and `StopRxNormInferenceJob` operations	5

Resource	Default
Transactions per second (TPS) for the `ListEntitiesDetectionV2Jobs`, `ListPHIDetectionJobs`, `DescribeEntitiesDetectionV2Job`, `DescribePHIDetectionJob`, `ListICD10CMInferenceJobs`, `ListRxNormInferenceJobs`, `DescribeICD10CMInferenceJob`, and `DescribeRxNormInferenceJob` operations	10

The size quotas for files as is shown in the table below:

Description	Quota
Maximum document size (UTF-8 characters) for `DetectEntities`, `DetectEntities-v2` and `DetectPHI` operations.	20,000 bytes
Maximum document size (UTF-8 characters) for `InferICD10-CM` and `InferRxNorm` operation	10,000 bytes
Maximum size of text analysis batch jobs (all files)	10 Gb
Maximum size of ontology linking batch analysis jobs (all files)	5 Gb
Minimum size of batch jobs (all files)	1 byte
Maximum individual file size for batch jobs	40 Kb
Maximum number of files for batch jobs	5 million

If your text is larger than the character quotas, use segment.py to create smaller segments that can be analyzed.

Only documents in English (EN) are supported.

Document History for Amazon Comprehend Medical

The following table describes the documentation for this release of Amazon Comprehend Medical.

update-history-change	update-history-description	update-history-date
New Feature for Amazon Comprehend Medical	Amazon Comprehend Medical now allows you to establish a private connection with your Virtual Private Cloud (VPC) by creating an interface VPC endpoint. For more information, see VPC endpoints(PrivateLink).	June 13, 2021
New feature	Amazon Comprehend Medical now provides batch operations for ontology linking. This enables the service to detect entities in medical text stored in an S3 bucket and link those entities to standardized ontologies. For more information, see Ontology Linking Batch Analysis.	May 4, 2020
New feature	Amazon Comprehend Medical now enables users to extract and relate the date or time expression to any of the entities currently detected by Amazon Comprehend Medical. For more information, see Detect Entities Version 2.	March 2, 2020
New features	Amazon Comprehend Medical now detects both medications and medical conditions and links the entities to established national ontologies RxNorm and ICD-10-CMs. For more information, see Ontology Linking APIs.	December 16, 2019
New feature	Amazon Comprehend Medical now provides batch operations so that you can process medical text stored in an S3 bucket. It also provides a new model that you can use for examining your medical text. For more information, see Detect Entities Version 2.	September 24, 2019

Major new feature	Amazon Comprehend Medical is a new service that detects useful information in unstructured clinical text: physician's notes, discharge summaries, test results, case notes, and so on. Amazon Comprehend Medical uses Natural Language Processing (NLP) models to leverage the latest advances in machine learning to sort through this enormous quantity of data and retrieve valuable information that is otherwise difficult to retrieve and use without significant manual effort. For more information, see Comprehend Medical	November 27, 2018