



AWS Academy Natural Language Processing  
Module 06 Student Guide  
Version 0.1.0  
200-ACMNLP-01-EN-SG

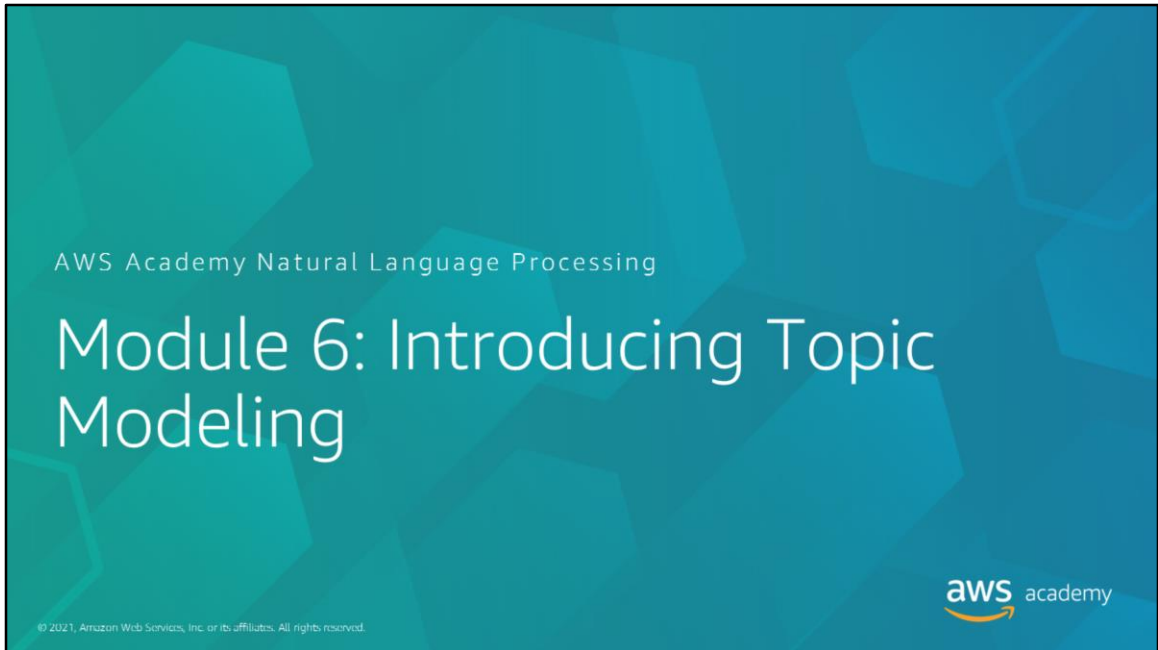
© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.


# Contents

Module 6: Introducing Topic Modeling	4
--------------------------------------	---



Welcome to Module 6: Introducing Topic Modeling.

## Module overview



Sections	Labs
<ul style="list-style-type: none"><li>• Section 1: Introduction to topic modeling</li><li>• Section 2: Identifying the approach</li><li>• Section 3: Implementing topic modeling</li></ul>	<ul style="list-style-type: none"><li>• Guided Lab: Implementing Topic Modeling with Amazon Comprehend</li><li>• Guided Lab: Implementing Topic Modeling Using Neural Topic Model (NTM)</li><li>• Challenge Lab: Implementing Topic Modeling</li></ul>

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.2

In this module, you will learn about the following topics:

- Introduction to topic modeling
- Identifying the approach
- Implementing topic modeling
- Making recommendations from text

Three labs are associated with this module. Two guided labs show you how to use Amazon Comprehend and the built-in Amazon SageMaker algorithms to obtain topics from a corpus of text. In the challenge lab, you will use the algorithms from the guided labs to make movie recommendations.

## Module objectives



At the end of this module, you should be able to:

- Understand the use cases for topic modeling
- Describe the steps for topic modeling
- Use Amazon Comprehend to implement a topic modeling solution
- Evaluate machine learning (ML) algorithms that are used in natural language processing (NLP) for topic modeling
- Create a solution to a topic modeling business problem

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

5

At the end of this module, you should be able to:

- Understand the use cases for topic modeling
- Describe the steps for topic modeling
- Use Amazon Comprehend to implement a topic modeling solution
- Evaluate machine learning (ML) algorithms that are used in natural language processing (NLP) for topic modeling
- Create a solution to a topic modeling business problem

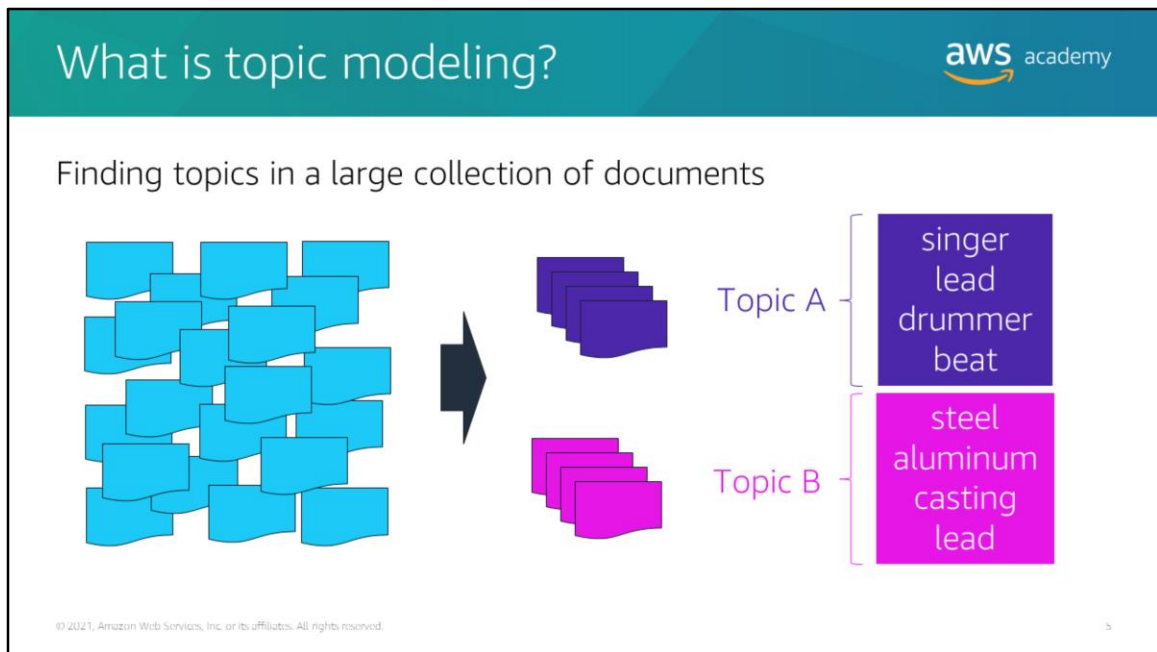
## Module 6: Introducing Topic Modeling

### Section 1: Introduction to topic modeling

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.



In this section, you look at what topic modeling is and some of the use cases for topic modeling.



*Topic modeling* is the NLP process of extracting topics from a large collection of documents. Topic modeling is an unsupervised machine learning model that scans a set of documents to detect word and phrase patterns. These words and patterns are grouped into clusters that best characterize the documents.

In the diagram, two possible topics have been identified, Topic A and Topic B. Topic A contains the words singer, lead, drummer, and beat. Topic B contains steel, aluminum, casting, and lead. Topic A could be about music or bands. Topic B could be about metallurgy or manufacturing. A key note is that the topics are represented as a collection of related words or phrases. Topics themselves are not usually identified, and they require human intervention to interpret the suggested topics. Notice that the topics identified by the model might not be the same as a human would identify.



## Topic modeling use cases



- Document classification
- Automatic content tagging
- Document summarization
- Information retrieval by using topics
- Content recommendation based on topics

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

b

Topic modeling has many practical use cases, including the following:

- Document classification based on the topics that are detected
- Automatic content tagging, which uses tags that are mapped to a set of topics
- Document summarization, which uses the topics that are found in the document
- Information retrieval by using topics
- Content recommendation based on topic similarities

Topic modeling can also be used as a feature engineering step for downstream text-related ML tasks.

## Section 1: Summary



- What is topic modeling?
- Topic modeling use cases:
  - Document classification
  - Automatic content tagging
  - Document summarization
  - Information retrieval by using topics
  - Content recommendation based on topics

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

1

In this section, you looked at what topic modeling is and some of the use cases for topic modeling.

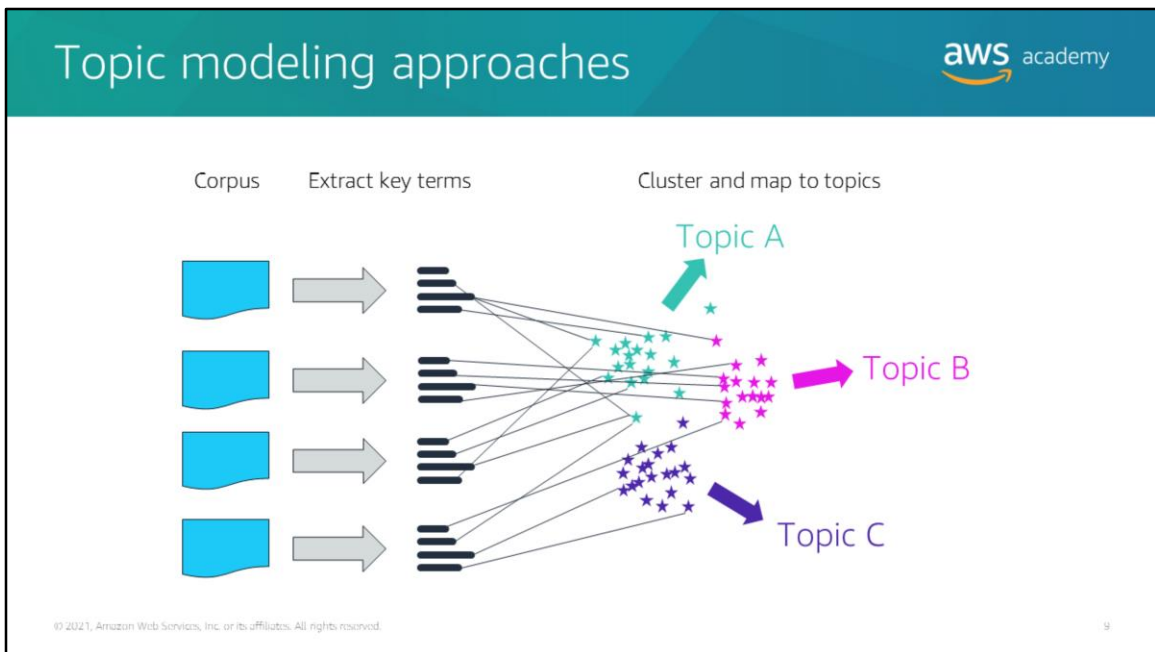
## Module 6: Introducing Topic Modeling

### Section 2: Identifying the approach

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.



In this section, you look at approaches to topic modeling. You also review steps to prepare data for topic modeling.



Documents can contain a mixture of topics. Topic modeling tries to extract words that are relevant in a document. Documents that share common terms share a topic. Documents might contain more than one topic.

Two documents with similar topics will not have the exact same content. However, you would expect these documents more frequently to share a subset of words, compared to documents from a different topic mix.

In Module 3, you learned about term frequency-inverse document frequency (TF-IDF), which is a way to extract key terms that are important to the document. You can use this method on a set of documents to extract those terms and then group documents with similar terms together. You can also visualize the words as vectors. When terms share the same vector space, the indication is that they are related. Topic modeling merely tries to identify which terms belong together.

## Processing steps



- Clean the data
- Remove stopwords
- Lemmatize the data
- Vectorize or tokenize the documents

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

10

As with any NLP task, it is helpful to understand how the model works so that you can prepare the data. Because the models look for common words across documents, you might want to take a number of key steps:

- **Clean the data:** For example, remove HTML tags, markup, numerical data, and whitespace to have a better set of text to work with.
- **Remove stopwords:** These words tend not to add value in these types of NLP tasks.
- **Lemmatize the data:** This is important because you want to find groups of words that are common across documents. Lemmatization can help reduce the number of word variations that are needed.
- **Vectorize or tokenize the documents:** This depends on the requirements of the algorithm that you are using.

## Section 2: Summary



- Topic modeling approaches
- Processing steps
  - Clean the data
  - Remove stopwords
  - Lemmatize the data
  - Vectorize and tokenize

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

11

In this section, you learned a high-level approach to topic modeling. You also reviewed some of the key processing steps that are required to prepare your data. These steps include cleaning your data, removing stopwords, lemmatization, and vectorizing or tokenizing your text.

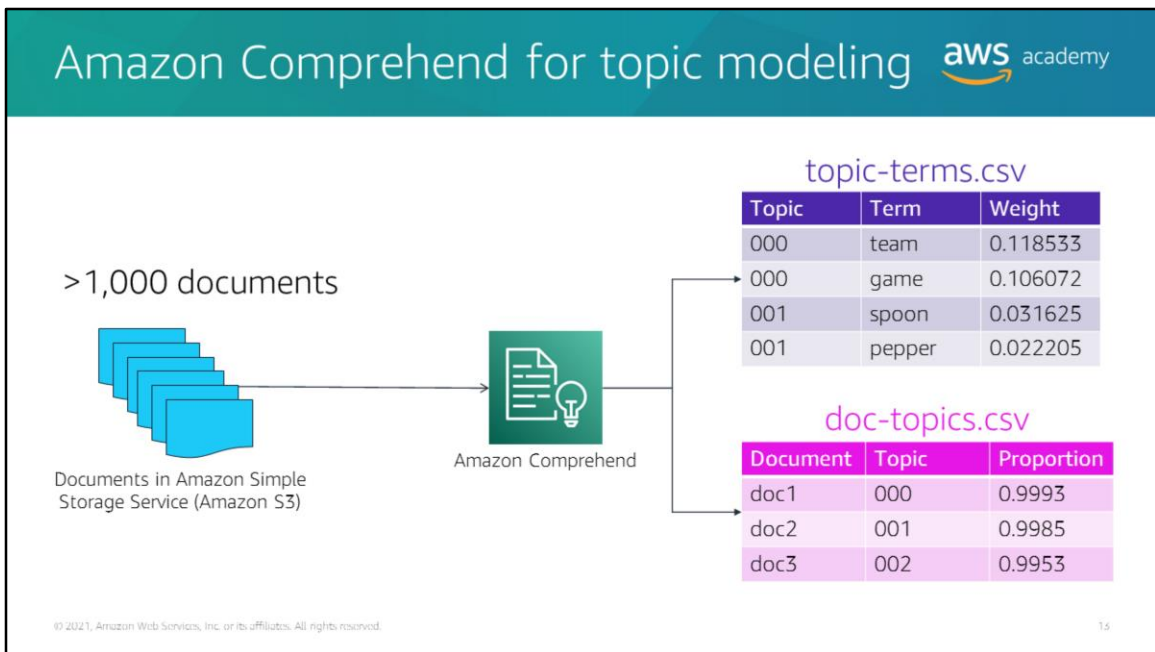
## Module 6: Introducing Topic Modeling

### Section 3: Implementing topic modeling

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.



In this section, you learn about three different options to implement topic modeling as well as trade-offs among the options.



You can use Amazon Comprehend to examine a collection of documents to determine common topics. For the best results, you must use at least 1,000 documents in each topic modeling job. With Amazon Comprehend, you do not need to train a model or preprocess your data. You can create UTF-8 text files in Amazon Simple Storage Service (Amazon S3) and create an Amazon Comprehend topic modeling job.

When a job is complete, you have two output files. The first file, `topic-terms.csv`, contains a list of all of the topics and the top terms for the topic according to their weights. The weight represents a probability distribution over the words in a given topic. Amazon Comprehend returns only the top 10 words for each topic. Note that the weights won't add up to 10 unless you have fewer than 10 terms. The 10 terms should be enough for you to determine whether a topic makes sense. Amazon Comprehend can detect up to 100 topics per job, but you can specify how many you want to find based on your domain knowledge.

The second file is the `doc-topics.csv` file. This file lists the documents that are



associated with a topic and the proportion of the document that is concerned with the topic.

Notice that the names of the topics are not given. Naming the topics is still a task for a human to complete.

## Module 6 Guided Lab 1: Implementing Topic Modeling with Amazon Comprehend

14



© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

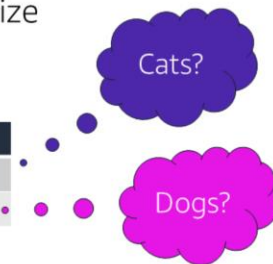
You will now complete Module 6 Guided Lab 1: Implementing Topic Modeling with Amazon Comprehend. In this lab, you use Amazon Comprehend to extract topics from a dataset.

## Latent Dirichlet Allocation (LDA)



- Unsupervised learning algorithm
- Describes a set of documents as a mixture of distinct topics
- Topics are **NOT** specified up front
- Might not align with how a human would categorize
- Word groups = topics

Topic	eat	sleep	play	meow	bark
Topic 1	0.1	0.3	0.2	0.4	0.0
Topic 2	0.2	0.1	0.4	0.0	0.3



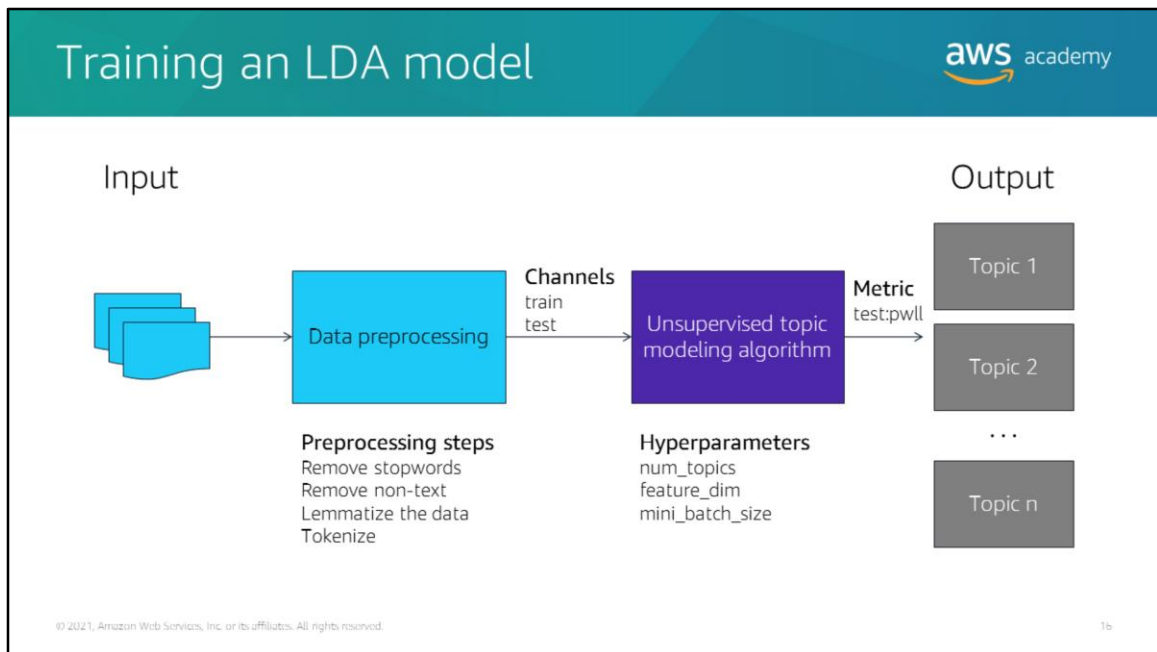
© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

15

Amazon SageMaker contains two built-in algorithms for extracting topics from text. The first algorithm is *Latent Dirichlet Allocation (LDA)*, and the second is the *Neural Topic Model (NTM)*.

LDA is an unsupervised learning algorithm that attempts to describe a set of documents as a mixture of distinct topics. Because this algorithm is unsupervised, the topics are **not** specified up front. The topics might not line up with how a human would categorize documents. The topics themselves are learned as a probability distribution over the words that occur in each document. Each document, in turn, is described as a mixture of topics.

For the example on the slide, simple documents contain only the words **eat**, **sleep**, **play**, **meow**, and **bark**. LDA might produce the topics shown in the table. Topic 1 is probably about cats (who are likely to meow and sleep) and Topic 2 is likely about dogs (who prefer to play and bark).



To train an LDA model, you need a set of documents. You should preprocess these documents appropriately, depending on your documents. The typical steps that you would take are to clean the text, remove stopwords, remove non-text, and perform lemmatization. Next, you would tokenize the text by using bag of words (BOW) or some other tokenizer. Notice that lemmatization significantly increases algorithm performance and accuracy.

To train, you must supply the data on the train channel. You can optionally supply test data, which the final model scores, to provide the metric test:pwll. Per-word log-likelihood (pwll) on the test dataset is the likelihood that the learned LDA model accurately describes the test dataset. The goal is to maximize this value.

The data can be in a recordIO-wrapped-protobuf or CSV file.

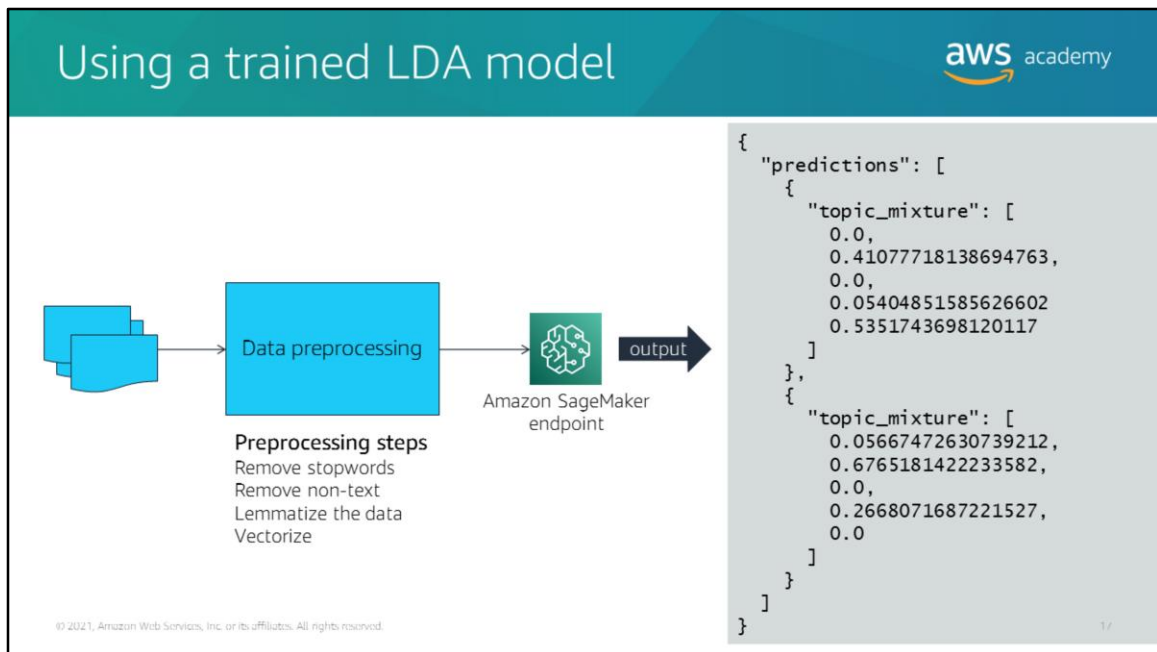
You must supply a number of hyperparameters:

- num\_topics – The number of topics for LDA to find within the data
- feature\_dim – The size of the vocabulary of the input document corpus
- mini\_batch\_size – The total number of documents in the input document corpus

LDA supports only single central processing unit (CPU) training.

The output of the training job will be a model that can identify topics that are discovered from the input corpus.

For more information about LDA, see [Latent Dirichlet Allocation \(LDA\) Algorithm](#).



To infer from the model, you must deploy the model. As soon as it is deployed, you should run your documents to infer through the same preprocessing pipeline before calling the model. The output of the model is a Python dictionary that contains predictions, and each **topic\_mixture** refers to an input document. Each element of the **topic\_mixture** corresponds to a topic that was learned when the original model was trained.

In the example that is shown, the first document is most likely about Topic 2 and Topic 5. The second document is likely about Topic 2 and possibly Topic 4, depending on your threshold.

## Neural Topic Model (NTM) algorithm



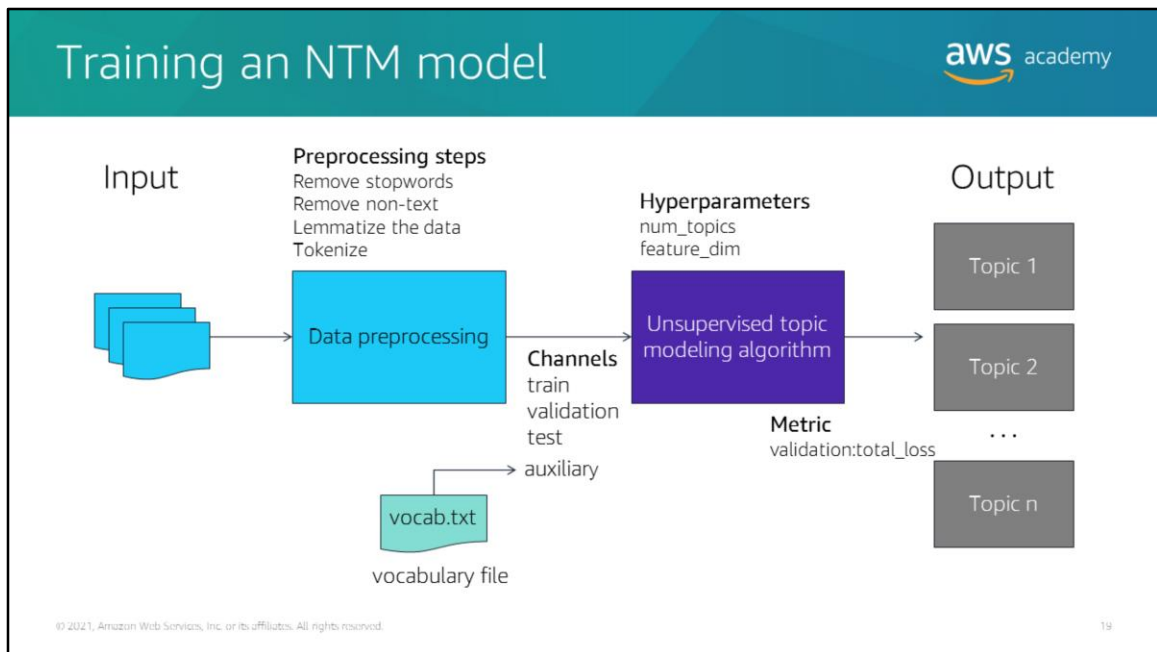
- Unsupervised learning algorithm
- Organizes a corpus of documents into *topics* that contain word groupings based on their statistical distribution
- Topics are **NOT** specified up front
- Might not align with how a human would categorize
- Can produce different results than LDA

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

18

The second Amazon SageMaker algorithm is the Neural Topic Model (NTM) algorithm. It is another unsupervised learning algorithm that organizes documents into topics that contain word groupings based on statistical distribution. Again, because it is unsupervised, the topics are not specified up front. Only the number of topics to find is specified up front. The topics that are discovered might not align with how a human with domain knowledge would categorize them.

NTM and LDA are distinct algorithms, and as such you can expect them to produce different results.



The Neural Topic Model (NTM) algorithm requires similar processing to LDA. You must remove stopwords, remove non-text, lemmatize the data, and tokenize the text.

NTM supports four channels: train, validation, test, and auxiliary. Train is the only required channel, and you should already be familiar with the validation and test channels. However, auxiliary is likely new to you.

*Auxiliary* is used to supply a text file that contains vocabulary. When you supply the vocabulary file, users are able to see the top words in each topic instead of only the integer IDs. This result is similar to the output from Amazon Comprehend. In addition, NTM can use the vocabulary file to compute the Word Embedding Topic Coherence (WTEC) score. This new metric is displayed in the log that captures similarity among the topic words in each document. The vocabulary file is a plaintext file (text/plain ContentType). A word on each line corresponds to the integer IDs that are provided in the data. This file must be named `vocab.txt` and be UTF-8 encoded.

You must supply a number of hyperparameters:

- **num\_topics** – The number of topics for NTM to find within the data, with a maximum of 1,000,000
- **feature\_dim** – The size of the vocabulary of the input document corpus

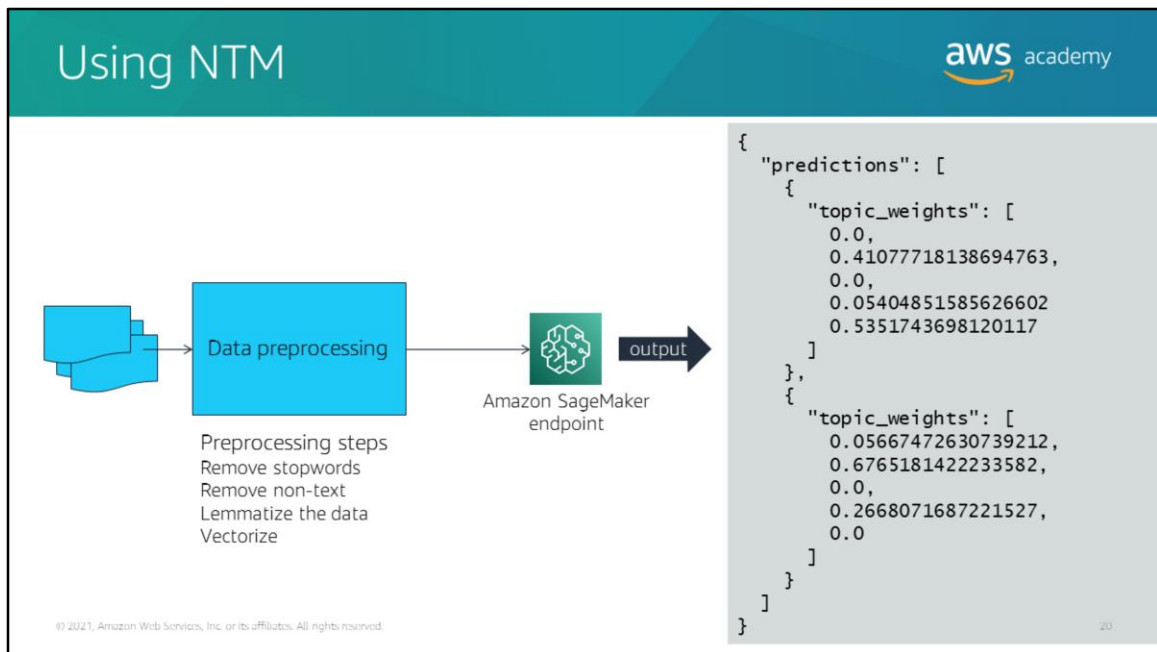
The metric that is used for tuning is **validation:total\_loss**. It is the sum of the



reconstruction loss and Kullback-Leibler divergence. The Kullback-Leibler divergence is a measure of how one probability distribution is from a second probability distribution. Reconstruction loss is a measure of the difference between inputs and outputs. In general, you want to tune the model to reduce the value for this metric.

NTM supports both graphics processing unit (GPU) and CPU instance types, and GPUs are recommended.


For more information about NTM, see [Neural Topic Model \(NTM\) Algorithm](#).



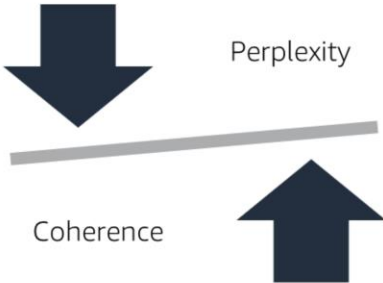
With a trained model, you must deploy it to perform inferences. We recommend that you preprocess your documents with the same pipeline.

The results are almost identical to the LDA results except that instead of the **topic\_mixture**, you get the **topic\_weights**. They are essentially the same, and they indicate which documents belong to which topics.

## Perplexity and coherence



- **Perplexity:** Metric for the likelihood that a topic is included in a document
- **Coherence:** Metric for the similarity of words in a topic
- Metrics are useful to evaluate topic models
  - Lower perplexity scores = model correctly identifies the subject
  - Higher coherence = topics in the document are similar



© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.


21



As with any machine learning problem, choosing the right algorithm is a function of your business goal. To choose between LDA and NTM, you can consider two metrics: perplexity and coherence.

*Perplexity* is a measure of the likelihood that a word will appear in a topic. A lower perplexity score indicates that you are more likely to be able to generalize the model. However, if you rely on perplexity alone to select an algorithm, it might not align with how a human would categorize the topics.

*Coherence* is another metric that you can use to help overcome the problem of relying on perplexity alone. With coherence, you calculate the average or median similarity of pairs of words in a defined set of words in a topic. When you minimize perplexity and maximize coherence, you are more likely to optimize the model that you use for topic modeling.

## Choosing between Amazon Comprehend, LDA, and NTM



 <b>Amazon Comprehend</b>	 <b>Amazon SageMaker built-in algorithms</b>				
<ul style="list-style-type: none"><li>• Fully managed service</li><li>• Easier if you are new to machine learning</li><li>• Tight deadlines</li></ul>	<ul style="list-style-type: none"><li>• Require training</li><li>• Customized vocabulary</li><li>• Regulatory hurdles</li></ul>				
	<table border="1"><thead><tr><th>LDA</th><th>NTM</th></tr></thead><tbody><tr><td><ul style="list-style-type: none"><li>• Single CPU</li><li>• Better at perplexity</li></ul></td><td><ul style="list-style-type: none"><li>• Multiple GPUs</li><li>• Better at coherence</li></ul></td></tr></tbody></table>	LDA	NTM	<ul style="list-style-type: none"><li>• Single CPU</li><li>• Better at perplexity</li></ul>	<ul style="list-style-type: none"><li>• Multiple GPUs</li><li>• Better at coherence</li></ul>
LDA	NTM				
<ul style="list-style-type: none"><li>• Single CPU</li><li>• Better at perplexity</li></ul>	<ul style="list-style-type: none"><li>• Multiple GPUs</li><li>• Better at coherence</li></ul>				

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

22

With three choices for topic modeling, you might be wondering when to use each.

Amazon Comprehend provides a fully managed service that you can use quickly without any data preparation. This option is good for projects when you need quick results or you are working within tight deadlines. You can use Amazon Comprehend even if you are new to machine learning

Built-in algorithms, such as NTM and LDA, require more work up front to train the models. You must preprocess the data and tune the models, which can take some time to complete. However, the advantage is models that can use domain-specific vocabulary. You might also need to use these algorithms for compliance reasons where you need detailed audits of data access and processing.

LDA and NTM can give you different results based on the same dataset. LDA is faster than other implementations, and NTM can use multiple GPUs. Try both algorithms to see which one gives you the best results.

## Module 6 Guided Lab 2: Implementing Topic Modeling Using Neural Topic Model (NTM)

25



© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

You will now complete Module 6 Guided Lab 2: Using Neural Topic Model (NTM) for Topic Modeling. In this lab, you will use both the LDA and NTM algorithms to examine a dataset.

## Module Challenge Lab 3: Implementing Topic Modeling

24



© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

You will now complete Module 6 Challenge Lab: Implementing Topic Modeling.

## Section 3: Summary



- Using Amazon Comprehend
- Using Amazon SageMaker algorithms
  - LDA
  - NTM
- Choosing between Amazon Comprehend, LDA, and NTM

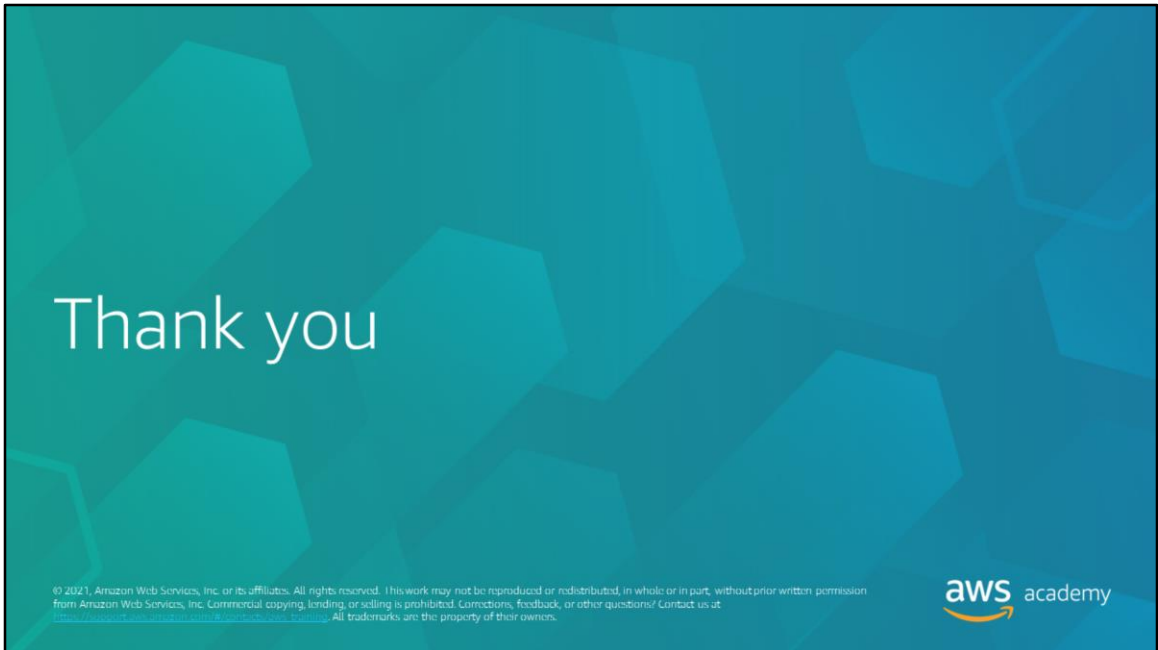
© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

25

In this section, you looked at three different ways to implement topic modeling.

Amazon Comprehend provides a fully managed service that requires little processing. LDA and NTM are both Amazon SageMaker algorithms and require training before you can perform inferences.

You also looked at some trade-offs among the three options.



Thank you for completing this module.