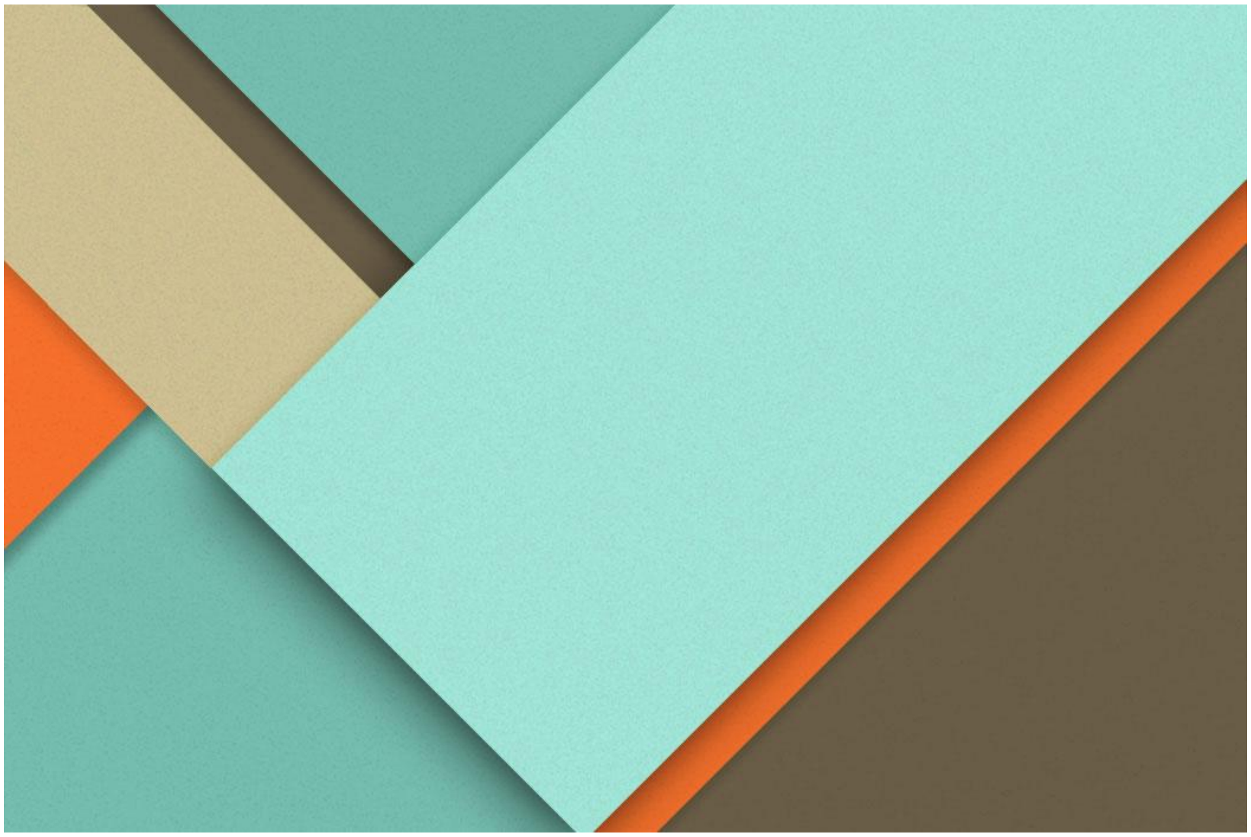


Technology Review (Amazon Comprehend)

11.08.2021

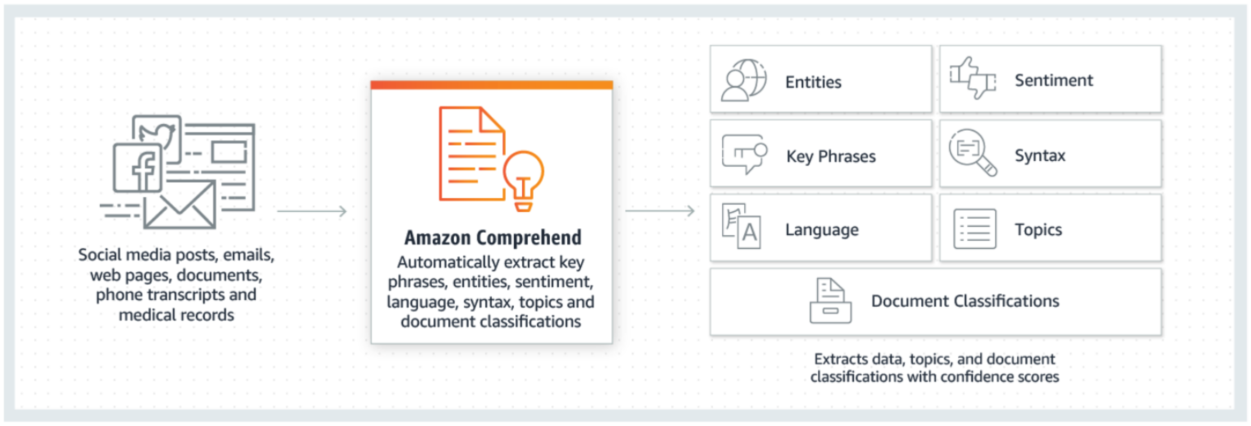
Ratan Bajpai
MCS-DS Student
University of Illinois at Urbana-Champaign
Champaign, IL

Overview

In this document we're giving an overview of Amazon Comprehend which is a high-level NLP service by AWS. It is a relatively newer service and was introduced around November 2017. It uses machine learning to gather insights and relationships from text documents. It provides a rich set of APIs so that the users can accomplish various NLP tasks such as entity / event detection, personal information identification, sentiment analysis, topic modeling etc. Here we will look specifically at sentiment analysis using Amazon Comprehend.

How It Works

Amazon Comprehend continuously trains models using a large collection of text data. It then uses this pre-trained model to analyze a set of input documents to gather and generate various insights about them. This general overview is depicted in the figure below (source: Amazon AWS)

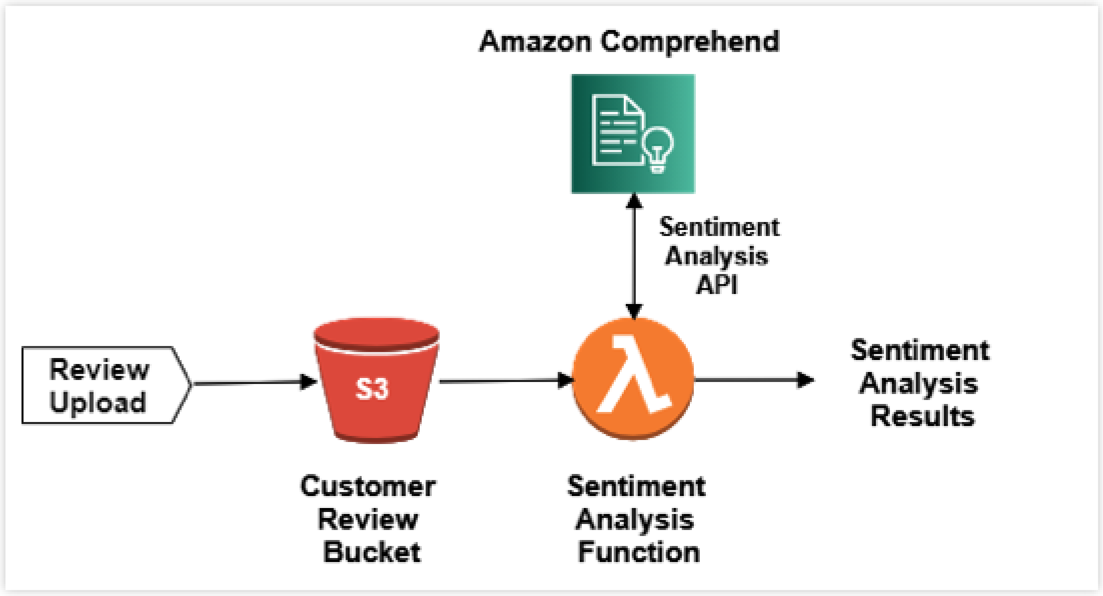

Social media posts, emails, web pages, documents, phone transcripts and medical records
Amazon Comprehend
Automatically extract key phrases, entities, sentiment, language, syntax, topics and document classifications
Entities
Sentiment
Key Phrases
Syntax
Language
Topics
Document Classifications
Extracts data, topics, and document classifications with confidence scores

In the figure above, Amazon Comprehend ingests large quantities of a variety of text data. It then continuously trains models and automatically extracts key phrases, entities, topics etc. To use this service we need to invoke a specific API for a specific task and give it the input text document or set of documents. The API processes the input data based on pre-trained machine learning models and returns the output which could be dominant language, entities (names, places, items), positive or negative sentiment and topic phrases. Amazon Comprehend uses specific machine learning models for specific tasks. For example, for topic modeling it uses Latent Drichlet Allocation or LDA.

Sentiment Analysis

Amazon Comprehend has a “DetectSentiment” API to categorize sentiment for a single document. It also has “BatchDetectSentiment” to output the sentiment result for a set of documents. The sentiment values that are returned are: “positive”, “negative”, “neutral” or “mixed” along with their confidence scores. Amazon Comprehend supports a number of primary languages (English, Spanish, German, Italian, French etc.). All the documents for an API call must be in the same language.

If we want to do sentiment analysis on a set of product reviews from users, the below figure (source: Amazon AWS) illustrates the high-level architecture of such a system.



Each review can be uploaded as a separate document, or as a line item in a single document to an S3 bucket. Each review string must contain less than 5000 bytes of UTF-8 encoded characters. A custom lambda function needs to be implemented that will construct the request to invoke the “DetectSentiment” or “BatchDetectSentiment” APIs. The results returned by Amazon Comprehend can be analyzed for the set of product reviews and some statistics can be generated to show key insights, i.e. percentage of users that like the product etc. The below section shows sample requests and responses for the “DetectSentiment” and “BatchDetectSentiment” APIs.

Sentiment Analysis Request / Response

DetectSentiment API

This API takes the following request / response structure (source: Amazon Comprehend website):

- Request

```
{
   "LanguageCode": "string",
   "Text": "string"
}
```

Here the “LanguageCode” denotes what language the text string is in, i.e. “en”, “de”, “es” etc. A list of these codes along with description is available on Amazon Comprehend website. The “Text” is the actual text string for which the sentiment analysis has to be done.

- Response

```
{
   "Sentiment": "string",
   "SentimentScore": {
      "Mixed": number,
      "Negative": number,
      "Neutral": number,
      "Positive": number
   }
}
```

The response contains the “Sentiment” result, which could be “POSITIVE”, “NEGATIVE”, “NEUTRAL” or “MIXED”. Also it contains a score for each of those sentient values. Below is an example of such a response.

- Example Response

```
{
    "SentimentScore": {
        "Mixed": 0.0033542951568961143,
        "Positive": 0.9869875907897949,
        "Neutral": 0.008563132025301456,
        "Negative": 0.0010949420975521207
    },
    "Sentiment": "POSITIVE",
 }
}
```

In this example response, the score of the positive sentiment is clearly the highest, hence the text is characterized as positive sentiment. These kinds of sentiment scores are useful as the API user can use their own thresholding mechanisms to decide the sentiment value in case the scores are not clearly in favor of one of the sentiment values.

BatchDetectSentiment API

This API takes the following request / response structure (source: Amazon Comprehend website):

- Request

```
{
   "LanguageCode": "string",
   "TextList": [ "string" ]
}
```

Here each document that needs to be analyzed is passed in as an array element. The language for all the documents have to be the same.

- Response

```
{
   "ErrorList": [
      {
         "ErrorCode": "string",
         "ErrorMessage": "string",
         "Index": number
      }
   ],
   "ResultList": [
      {
         "Index": number,
         "Sentiment": "string",
         "SentimentScore": {
            "Mixed": number,
            "Negative": number,
            "Neutral": number,
            "Positive": number
         }
      }
   ]
}
```

Here the response is an array where each item is a response for a particular document or text string. The index denotes the order of the request / response. Typically the input order is preserved for the responses. Also there is an error list which indicates if there is a problem with any of the input document or text string. Again the errors have an index which correlates to the input.

Conclusions

Amazon Comprehend provides a rich set of simple APIs that can generate insights from a set of text documents. These APIs can perform various NLP tasks such as entity / event detection, personal information identification, sentiment analysis, topic modeling etc. These APIs work at a high-level. The models are pre-trained, so the user does not have to worry about generating training data and training the models. In this document, we looked at the mechanics of how these APIs work for the sentiment analysis task. The API is fairly easy to use and can work at scale. The “SentimentScore” for various sentiment values is useful and can be used to override the sentiment conclusion in case of a “MIXED” sentiment value.

Although the APIs are high-level and easy to use, the user does not have any control over the models that are used for accomplishing various tasks. For example for topic modeling, if we want to use some other algorithm instead of LDA, that will not be possible. Also, there is no way to tweak any of the parameters in the models that are used. So the ease of use comes at the cost of lack of control. If the user wants more control over these models and their parameters, it might be better to use the Natural Language Toolkit (NLTK) or some other tools.