Topic Classification of Amazon Articles using NLP techniques

In this article, we discuss a state of the art NLP pipeline that enables the grouping of randomly selected articles from [www.amazon.com](http://www.amazon.com) into relevant topics. We use webhose.io for data ingestion, IBM Watson developer cloud for named entity recognition, MongoDB for storage and a Flask app to display the results.

The following diagram depicts the various components of the NLP pipeline and their inter-connections.

A close up of a sign

Description generated with very high confidence

As seen in the above figure, the dataset which is a collection of articles from amazon.com is ingested into MongoDB using a web service called WebHose.io. The stored articles are then queried from mongoDB using PyMongo and are pre-processed before being fed into two independent topic modelling algorithms -namely LatentDirichletAllocation and KMeans. The topics outputted by these algorithms are then used to build a topic taxonomy, which has the outputted topics mapped to a list of relevant daughter items defined by the user. A word2vec model is then used to find similarity scores that define the strength of relationship between the items in the topic taxonomy and the titles of the articles in the dataset. A flask application is then built on AWS to read titles entered by the user and output a list of top three most relevant topics associated with the inputted title.

As an aside, named entity recognition is performed on the dataset using Watson NLU. We establish a connection between python and Watson NLU using the Watson developer cloud API. The tool classifies named entities in the articles into pre-define categories such as type, text and relevance. The figure below shows a snippet of the entities extracted from the articles.

A screenshot of a cell phone

Description generated with very high confidence

With the help of the pymongo package, data is stored in MongoDB and MongoClient is used by Pymongo for connecting to a MongoDB instance, replica set or sharded cluster. The data is then pre-processed using the nltk package in python. The steps involved in pre-processing is outlined in the diagram below

A close up of a logo

Description generated with very high confidence

The raw text is first tokenized into sentences and the sentences are in turn tokenized into individual words. The stop words from the tokenized word list are removed and then lemmatization is performed to retain only the dictionary form of the word, called the lemma. The lemmatized data is then vectorized, in other words it is transformed into a matrix containing the counts of the words in the articles. This matrix is then fed as input to the topic modelling algorithms.

We use two independent topic modelling techniques-namely k-means and LDA to cluster the articles into a finite number of topics. A snapshot of the k-means and LDA results are shown below

A black sign with white text

Description generated with high confidence A screenshot of a cell phone

Description generated with very high confidence

Snippet of LDA output Snippet of K-means output

The outputs obtained from LDA and K-means are then used to build the topic taxonomy. The figure below shows a snapshot of the topic taxonomy

A screen shot of a smart phone

Description generated with high confidence

A word2vec model then compares the keywords in the topic taxonomy with the title of the article that is queried to come up with a list containing the top three similarity scores between the topics in the topic taxonomy and the title ordered in descending order. A screenshot of the classification results is shown below

A white sign with black text

Description generated with high confidence

A flask application is then built on AWS to recommend topics for any user-defined title. A snapshot of the flask app is shown below

A screenshot of a cell phone

Description generated with high confidence

In sum, we have discussed a state of the art NLP pipeline used to recommend topics for titles pertaining to articles in amazon.com. The code to the project is available on github for your reference.