Document Search using Doc2Vec

Sudhanshu Chib | Nov 12th

Introduction

The project is an attempt to explore the utility of Doc2Vec algorithm for the task of Document Searching. The project aims to address a problem faced by my current organization where there is a need to have a system that can help trace information stored in files across multiple folders.

PROBLEM DEFINITION:

Business and Financial research teams in my organization collect data from multiple sources like mails, analyst reports, newspaper articles, sector reports, syndicate sources like Bloomberg etc.

Information drawn from these sources is used to build reports that are then provided to clients as per their need. To keep stock of the information available an inventory of these information sources (specially the reports) is maintained in folder structures and on share point.

With time (as people move out or switch teams) the transient knowledge about these project reports and information sources is lost. In this case, any new research project has to start from scratch and in most cases is not able to leverage similar work done in the past.

This project aims to explore the possibility of using Doc2Vec to build a system where an analyst is assisted in his pursuit of information gathering by reducing time spend by them in looking for the right information

Note:

As it would not be possible to get the exact data that is stored in share drives of the firm hence a proxy data set is used to evaluate the approach.

Data Set Used: The proxy data set used is the book summaries data set that has 16,559 book summaries across different genres. Dataset contains plot summaries for 16,559 books extracted from Wikipedia, along with aligned metadata from Freebase, including book author, title, and genre.

The aim of the analysis will be to find most similar books based on their summaries. This will serve as a proxy for the real world problem of finding the most similar report to the report that the analyst needs to research on.

Data Source=http://www.cs.cmu.edu/~dbamman/booksummaries.html

Column names: WikipediaArticleID, FreebaseID, BookTitle, Author, PublicationDate, BookGenres and Summary

METHODOLOGY:

Data cleaning:

Data for this project was read in as a csv and stored in pandas data frame for processing. As we are working with text data in this case a combination of genism and spacy was used to massage the data and make it useable for consumption.

English vocab for Spacy was used to tokenize text. Build in spacy functions were used to tag tokens that were pure punctuations or white spaces. These tokens are removed from the analysis as they have little value to the output.

Next, all reviews were looped over and broken down into sentences. Sentences were tagged again by the build in functionality of Spacy. Individual sentences were then written as an output to a text file on the computer.

Few of the key identifier's in case of book summaries are character names (Harry Potter, Tom Sawyer), geography names (city or country names where the plot is set), events (Operation desert storm) etc. To tag these elements phrase modelling was used. Phrases utility of Spacy was run thrice to form a tri gram model in a sequential manner. This was done with the assumption that frequently occurring names, events etc. will be tagged as one token.

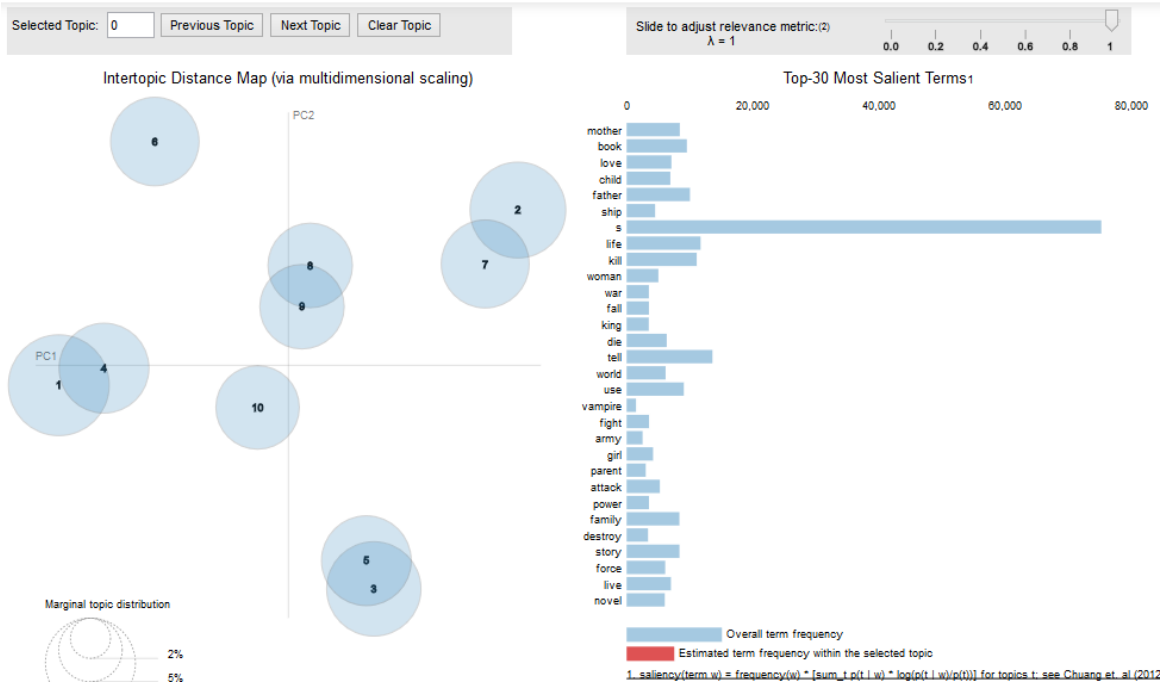
Post application of phrase model, stop-words were removed from the corpus.

As these summaries are documented and have proper editing, there was no real danger of having misspelled words polluting the corpus.

Data Exploration:

Latent Dirichlet allocation (LDA) was used to explore the corpus. LDA supported by genism was used to cluster text on 6 to 20 clusters. For each iteration top 30words were studied to check if a logical theme could be placed on the cluster.

After evaluating different values 10 clusters were fixed as beyond this number the clusters were breaking out into smaller sets that were hard to understand. Below is the cluster breakout diagram for 10 selected clusters:



Key themes that emerged from this analysis is that most of the books in the corpus broadly talk about violence, tragedy and relationships.

To check if the word association is working for this corpus a word2vec model was trained. Parameters used were size=100, window=5, min_count=20, sg=1 and workers=4.

Choice of parameters:

Size: Signifying dimensionality of the feature vectors was chosen as 100. Earlier attempts were made with lesser dimensions like 30 and 50 but the logical interpretation of results was better with vector dimension 100. Also articles available online suggest choice of value to range between 100 and 300.

Window: Represents maximum distance between the current and predicted word within a sentence or a floating window of defined size used to predict word. 5 was used as the value for this parameter as average sentence length was 5.

Min_count: Value was set to 20 to ignore words that appear less than 20 times.

Sg: Skip-Gram was chosen to be the algorithm of choice as most of the article refer this to be more accurate than CBOW

Workers: Value was set to 4 as the machine used to train the model had 4 cores

Model was trained over 12 epochs and the final model was stored as a file on disk.

Post training number of words remaining in the corpus were 335,152. To check the accuracy of the word2vec model checks were performed with words like “man”, “war”, ”love” etc.

Result for the same are shared below:

get_related_terms(u'love')

```
fall                  0.674
affection             0.622
marry                 0.621
each_other            0.601
married               0.583
friendship            0.57
romance               0.566
truly_love            0.561
feeling               0.556
kiss                  0.547
```

get_related_terms(u'war')

```
world_war_i           0.66
war_against           0.628
civil_war             0.626
world_war_ii          0.616
plot_outline_description 0.615
war_between           0.609
battle                0.608
fighting              0.601
fight                 0.591
wars                  0.576
```

get_related_terms(u'murder')

```
crime                 0.738
killing               0.679
victim                0.655
arrest                0.655
killer                0.653
suspect               0.64
murderer              0.636
suicide               0.631
commit                0.63
case                  0.623
```

get_related_terms(u'man')

```
woman                0.752
an_old               0.552
young_man            0.541
stranger             0.537
englishman           0.523
boy                  0.52
person               0.498
thug                 0.493
soldier              0.492
priest               0.483
```

Next, some basic word algebra was attempted to establish logical association. For example:

word_algebra(add=[u'tank',u'troop'])

```
army
```

word_algebra(add=[u'man',u'knife'])

```
stab
```

word_algebra(add=[u'husband', u'wife'], subtract=[u'happy'])

```
her_husband
```

As most of the results seem to have a logical connection to the searched term it was established with fair confidence that word association are being identified by the algorithm and hence doc2vec could be a good fit for this corpus.

Application of Doc2Vec model:

With the base data ready, Doc2Vec model was built with vector length specified as 100 for each document and context window specified to 50 words. Model was trained for 20 epoch or cycles at a fixed learning rate of 0.002. Learning rate value was decided by checking value over a range between 0.001 and 0.005. These range parameters were again drawn from articles available online describing Doc2Vec applications.

Doc2Vec model took approximately two hours for training and the resultant model was stored on the disk as a file. Book names were used as labels for training hence they were queried to check the resulting outputs.

Book Selected:
Dracula

Books Recommended:
House of Hell
Lord Kelvin's Machine
The Chemistry of Death
The Painted Man
The Soft Whisper of the Dead
Assassin
The RingworldThrone
The Witch Hunters
Mr. Fairlie'sFinal Journey
Ravenloft

Book Selected:
The Wars
Books Recommended:
Helmet for My Pillow
If I Die in a Combat Zone: Box Me Up and Ship Me Home
The Doctor In War
Insurrection
Zak's Lunch
Rage
Soldier's Heart
The Hallo-Wiener
Petey
The Settlers

Book Selected:
Harry Potter and the Philosopher's Stone
Books Recommended:
Fantastic Beasts and Where to Find Them
Harry Potter and the Prisoner of Azkaban
Harry Potter and the Half-Blood Prince
Harry Potter and the Goblet of Fire
GoodKnyght!
Talking to Dragons
Mistborn: The Alloy of Law
Dragons of Summer Flame
A Taste of Blackberries
Hop o' My Thumb

Book Selected:
Dr.No
Books Recommended:
The Man with the Golden Gun
Razor's Edge
Thunderball
A Concise Treatise on the Art of Angling
The Loveday Trials
Lucky Starr and the Oceans of Venus
The Matter of Arabyin Medieval England
China Sky
Goldfinger
From Russia with Love

Choice of model algorithm:

The main challenge with identifying contextually similar documents is that there realistically can be no labelled data that could be used to guide the algorithm in the learning phase. Hence supervised learning algorithms are not an option

Also, using text similarity measures like cosine, jaccard etc. would require comparison of tokens in isolation (without context) and for each queried entity (n-1) comparisons will be required per similarity measure (here n is the total number of entities). As the number of unique words in the corpus would increase and number of entitles increase the process will become slower).

Most similar books identified by Doc2Vec model are identified using Cosine Similarity computation. This should not be confused with traditional text similarity measures as Doc2Vec model similarity compares document vectors (computed by neural network) as against using direct word counts, Tf-Idf or presence or absence of the words.

With these considerations in mind it was decided to use Doc2Vec model. The main advantages of the model are:

a. It represents each document as a vector of specified size the model has a fast response time

b. Adding new document to the model do not require recalibration of the complete model hence it can scale better

Drawbacks of the model:

As we are working with unlabeled data the biggest challenge is defining a way to judge model performance. Model outputs can be critiqued as it is a subjective judgement of the person viewing the results. Attempt is to measure quality of model output by analyzing

results for books that are part of a series for ex. Harry Porter or James Bond and by viewing books that have similar genre.

It is a subjective way and does come with its fair share of criticism.

The order of the results is not the most accurate always, hence the most similar book (as per human knowledge) may not always be in the top three recommendations. Hence in practical applications we may need to provide the user a broader set of results to analyze.

Next Steps:

• One possible way to improve the accuracy of the model is to add metadata to collected reports and articles. For example, analyst name or team name, project id or client id etc.

• Typically a team or an analyst covers only a narrow domain so any metadata that provides this information can be used to improve predictions

• Another way to improve the model could be to add keyword search where the user provides not just article name or id but a few qualifying keywords that help narrow down the search

Appendix:

Brief introduction of Doc2Vec:

Doc2Vecis a model that creates a numeric representation of a document, regardless of its length. Each document is tagged by its identifier and represented as a vector of defined length. With the document represented as a number, it becomes possible to compare similarity between different documents. Under the hood Doc2Vev runs a neural network that tries to predict the probability of a word based on the words that surround that word (context). The number of words in context are specified by the window parameter in the model. Doc2Vec uses the words and the title as tag for input to the neural network to understand the vector representation of document.