Capstone 1: Document Search using Doc2Vec

Introduction (1/2):

- The project aims to address a problem that my current organisation is facing.
- The business and financial research teams collect data from multiple sources like mails, syndicate sources like Bloomberg, analyst reports, newspaper articles, sector reports etc.
- An inventory of these information sources (specially the reports) is maintained over time 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
- Hence for a new research project it becomes really difficult to trace information about similar work done in the past that can be referred for background research or analysis

- The aim of this project is to build a system that help an analyst by reducing time spend by them in looking for the right information

Introduction (2/2):

- As it would not be possible to get the exact data that is stored in share drives of the firm hence I am building a proxy for the problem by taking a dataset of book summaries
- Book summary data set has 16,559 book summaries across different genres
- The aim of the analysis will be to find most similar books based on their summaries and the technique applied for the same is Doc2Vec

- Dataset overview:
  - Data Source="http://www.cs.cmu.edu/~dbamman/booksummaries.html"
  - Data set details: This dataset contains plot summaries for 16,559 books extracted from Wikipedia, along with aligned metadata from Freebase, including book author, title, and genre.
  - Column names: WikipediaArticleID, FreebaseID, BookTitle, Author, PublicationDate, BookGenres and Summary

Method (1/3):

- The primary technique used to address the problem is Doc2Vec
- Understanding of data and validating the quality of data for use in Doc2Vec is done using a combination of LDA and Word2Vec model
- As we are dealing with textual data, primary cleaning procedures like lemmatization and removal of special characters is performed
- Two step phrase modelling is used to identify and tag key words that appear together for eg. post two step phrase modelling “New York City” will not be seen as three separate tokens but as a single token New_York_City
- Stopwords are removed post application of phrase modelling
- LDA is used to observe the key themes in the data
- For LDA, 6 to 20 clusters were evaluated. But I have settled for 10 as beyond this number the clusters were breaking out into smaller sets that were hard to understand

Method (2/3):

- The key themes observed through LDA is that the corpus has books that broadly talk about violence, tragedy and relationships
- Post LDA, Word2Vec model is used to understand word associations. A positive result on Word2Vec output serves as evidence that Doc2Vec can be a good candidate for this data set
- Example of the word association that Word2Vec model was able to figure out is shown below. Example of word algebra are also provided in the iPython notebook

  - 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
```

Method (3/3):

- With the base data ready, Doc2Vec model is build 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 learning rate of 0.002 (fixed)
- Some of the words associations picked by the model are:
  - model.most_similar("father")

```
[('mother',0.9025280475616455),
('son', 0.8302043080329895),
('brother', 0.8022996187210083),
('daughter', 0.778820276260376),
('family', 0.7785707712173462),
('he', 0.7773062586784363),
('in', 0.7636514902114868),
('be', 0.7592247128486633),
('have', 0.7495785355567932),
('and', 0.7495403289794922)]
```

- With the Doc2Vec model build, now recommendation for related books can be queried by providing book name (or index in this case) to the model

Model Output:

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 Ringworld Throne
The Witch Hunters
Mr. Fairlie's Final 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 Araby in Medieval England
China Sky
Goldfinger
From Russia with Love

Choice of Doc2Vec

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)

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

a. As 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

a. As we are working with unlabelled 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

I have tried to measure quality of model output by analysing 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

b. 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 analyse

Ways to improve and Next Steps:

- One possible way to improve the accuracy of the model is to add meta data to collected reports and articles. For example, analyst name or team name, project id or client id etc.
- Typically a team or an analyst cover only a narrow domain so any meta data 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
- The other possibility is to use Skip-Gram algorithm to in place of CBOW(continuous bag of words) that is currently used to build the model

Appendix:

- Introduction to Doc2Vec
- D**oc2Vec** is a model that creates a numeric representation of a document, regardless of it’s 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