Deree
School of Graduate and Professional Education

THE AMERICAN COLLEGE OF GREECE
Pierce Deree Alba
1875

7/19/2021

ITC 6010A1- NATURAL LANGUAGE PROCESSING

TERM PROJECT

Instructor: Polymenakos Lazaros

Diamantopoulos Thodoris (174430)
Kouvara Dimitra (246953)
Salmatanis Dimitris (253452)

CONTENTS

TABLE OF FIGURES

INTRODUCTION

We live in the era of high technology, and it is easily noticeable that new inventions pop up every day to bring comfort in our everyday life. With the evolution in computational and internet speed new technologies rise and companies from all kinds of different fields try to exploit these capabilities. Computers have become an essential tool for everyone, completing various tasks that a human would be incapable of. But while humans lack in computational power, storage capacity and more, they make up in their ability perform tasks and take decisions not only based on intelligence, but also based on their experiences. Humans have the innate motivation to create, tendencies that led them in our case to create a way to communicate and have what we call today language. We all know that people use language in a more abstract way, by not following exact rules, by fitting emotions inside it and generally communicating with a non-program like manner. By doing so computers would be unable to understand and gain any information out of this way of communicating because of their rule-based way of structure. But are they? With the advance of artificial intelligence and especially a subfield of it called Natural Language Processing, computers are now able to understand human language. By utilizing software designed specifically for these tasks, machines are now able to handle extremely large volumes of text and speech data fast without fatigue. They also have the ability to structure highly unstructured blocks of text data something extremely common in text data. All these helped the creation of translation tools, chatbots, speech-recognition tools and generally helped businesses achieve a more data-driven marketing.

All these new technologies were proven useful for sentiment analysis as nowadays people post texts online from product reviews, to posts on Twitter. Especially collecting, analyzing, and understanding those tweets plays a major role on determining the overall opinion of the public over a certain topic. But except the analysis of tweets about the public's opinion we get more and more interested about the tweets of individuals. More specifically, tweets of people that have a large audience and hold some levels of power are considered really important and are always analyzed. And who can have more impactful tweets than the president of the USA. Especially the former president Donald J. Trump had a really active Twitter account which people's lives in every conceivable way.

In our project we use Trump's tweets until June 2020 (kaggle, n.d.) and try to give structure to this highly unstructured data (after all we are talking about Donald Trump), detect word patterns in those data and cluster those words in groups that best characterize the different topics that the former president tweets about. This method used is called topic modeling and provides us with methods to organize, understand and summarize large collections of textual information. It helps us discover hidden topical patterns that are present across the collection, annotate documents according to these topics and organize, search, and summarize texts (David M. Blei, 2003).

To achieve this, we apply two different methods, a k-means clustering algorithm, a generative statistical model called LDA (Latent Dirichlet Allocation) and compare their results. Their results can be exploited later by analysts to predict market changes, foreign-policy predictions and generally get into the former president's way of thinking.

TEXT PREPROCESSING

SETTING THE FUNDAMENTALS

Our first step in our project was no other than importing the required libraries. Various libraries were imported from basic ones to nltk, gensim, sklearn and plot representation libraries. After that, some variables were initialized that would help later, on activating and deactivating various processes on demand by changing their value from 0 to 1 and vice versa. The next step was to define some regular expression patterns that would be used in tasks like removing hashtags, identifying different types of links and more. The stemmer and the lemmatizer were assigned into two variables for ease of use. Finally, several dictionaries were initialized that would be populated later with words, mentions, hashtags and more.

DEFINING FUNCTIONS

Writing code is one thing, but creating a piece of code that is efficient is a challenging task. To implement a top-down programming approach functions are needed. They offer the aspect of reusability and abstraction to the code, making it easier for future work implementation. So, our next step was to define several functions. There are functions responsible of preprocessing the text acquired by Trump's tweets, optimizing the k-means clustering and LDA model and visualizing the resulting topics.

PREPROCESSING AND CLEANING

To apply the topic modeling algorithms, we had to first prepare out dataset. We started by exploring our dataset so that we obtain useful information about the URLs, retweets, mentions, hashtags, links and pictures included in the former USA president's tweets. We found the ones that he uses most frequently, in order to get some intuition about the content in his tweets. By inspecting the impact of each information in his tweets we decided that we should remove everything except the hashtags and the actual text, as they did not provide us with any useful information. Then by utilizing regular expression patterns we followed the steps below to obtain meaningful results for our topic modeling keywords:

- We used the corresponding library to extend any existing contractions, like for example “we’re” to “we are”.
- All capital letters were set to lower case, to avoid any duplicate words.
- We removed the existing stop words which did not provide any useful insights for the goal of our task.
- We removed all punctuations except the hashtag (“#”) for the LDA model since it processes documents as “bag of words”. For the K-means algorithm, we did the same with the exception of also keeping the sentence ending punctuations.
- The hashtags were preserved as people tend to use them to provide their own topic inside their tweets.
- We also got rid of any retweets as they produce no core information about a topic.
- We removed any existing mentions for the same reason as we did with the retweets.
- As we observed that Donald Trump used plenty of links in his tweets (http and bit.ly) with no provision of information for out topics we decided to remove them.
- We applied the Porter stemming algorithm to transform the tokens collected from the tweets in a common stem and lemmatized them.

LDA MODEL

The first topic model that we applied was LDA. LDA is a generative probabilistic model for collections of discrete data such as text corpora (David M. Blei, 2003). It considers that all documents are made up of words that help in the creation of topics and by allocating each word to corresponding topics, it maps all the documents to a list of topics. This is its main advantage as by assigning one or more documents in each topic it enables a better understanding of the topics. LDA treats documents like a bag of words, disregarding the order that words appear (soft clustering) and any syntactical information, which is the main drawback of that method. To apply this method, we removed all punctuations except the hashtag.

INTERPRETABILITY OPTIMIZATION

Several basic techniques were implemented to enhance the interpretability of the model and we started the process to model our topics.

The first step was to identify phrases (multiple words) that commonly co-occur and act as single words, so that the topic model can recognize them. Therefore, we concatenated bigrams/trigrams (I.e., donald trump -> donald_trump) and counted as one word to improve topic modeling and create more meaningful features for the predictive LDA model. There are several methods to filter out the most meaningful collocations (nltk.collocations is a package that provides collocation finders). In this case, the Pointwise Mutual Information (PMI) score was applied. This score measures the likelihood that the words co-occur. The metric is sensitive to rare combination of words, so it is used with an occurrence frequency filter to ensure phrase relevance.

We detected the n-grams and filtered them with noun structures, exploiting the nltk package for tagging parts of speech. The reason behind this, is that since nouns are better indicators of a topic, more meaningful cluster topics are created by the LDA model. The selection of the PMI threshold, the score at which the n-grams stop making sense as pairs, was done after conscious reasoning. Lastly, we concatenated the n-grams into single words.

The second technique was filtering the remaining words for nouns. In sentences, nouns are mostly the "topic", as they are more interpretable. The rest of the words provide more context and explanation about the topic itself.

At this point, we tokenized the tweets, removed the stop words and the words with a length less than 2 characters.

The third technique was the optimization of the topics' number through the coherence measure and predictive perplexity. As the LDA model requires specifying the number of topics, finding the optimal coherence measure is considered essential. Literature suggests that maximizing the coherence measure, leads to better human interpretability.

The fourth technique was the adjustment of the LDA hyperparameters.

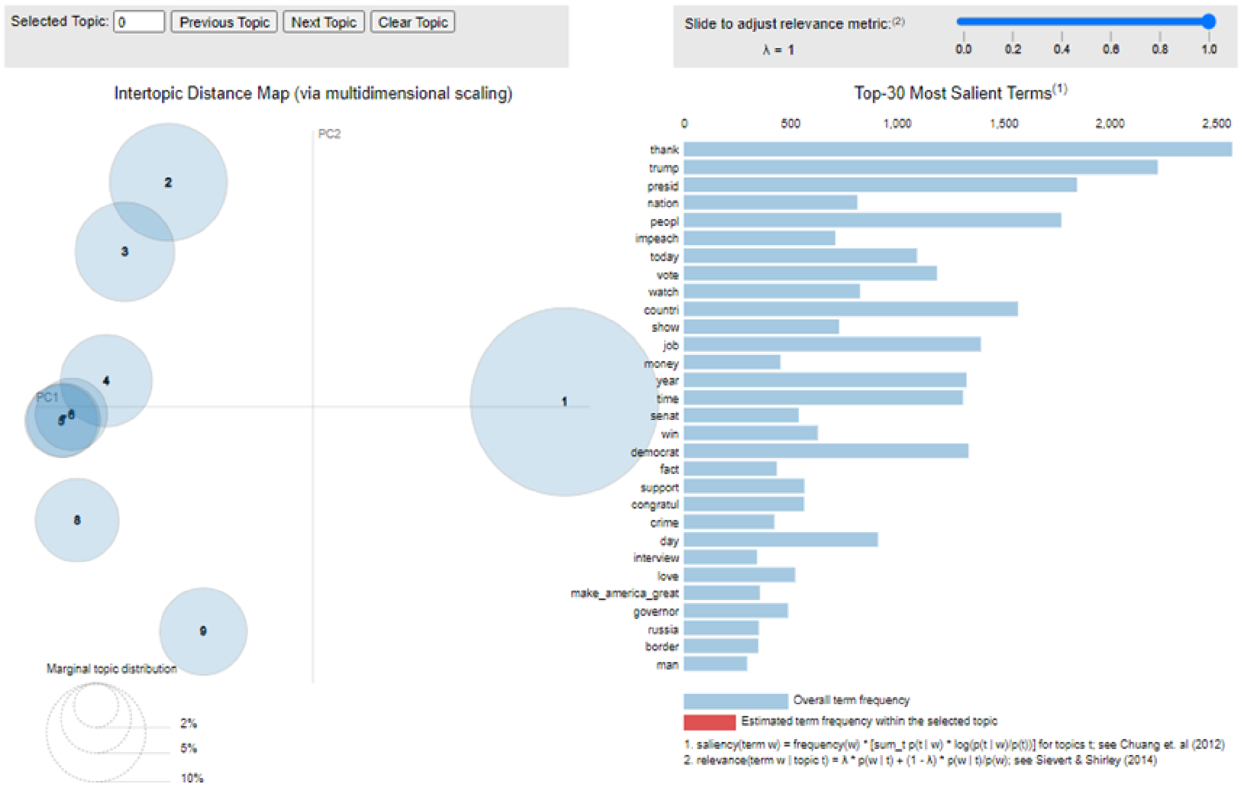
- Chunksize that controls how many documents are processed at a time in the training algorithm.
- Passes, that controls how often the model is trained on the entire corpus.
- Iterations, that essentially it controls how often a particular loop over each document is repeated.

Our first attempt provided us with topics with less sense that we would expect, so we increased the passes and iterations while increasing the chunksize, always taking into consideration our machine’s memory capacities.

By doing so, the execution time was higher, therefore we adjusted our method and decided to follow a trial-and-error approach in order to define the best possible number of topics. The parameters that were chosen, provided us with a better coherence score. Then we tested our model by both applying the TextBlob tool and not, to correct any misspelled words. Even though we should expect better results, TextBlob was actually correcting words that proved to be valuable for our model, so the coherence score was worse. For example, “donald trump” would be transformed into “donald plump” and other politicians’ names would be messed up, thus losing valuable information that would define our topics.

TOPICS VISUALIZATION

We used pyLDAvis to visualize the topics and the clusters created as shown in the figures (Figure 1).



1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

Figure 1: Topics visualized with pyLDAvis

The pyLDAvis package in Python gives two important pieces of information. The circles represent each topic, while the distance between the circles visualizes topic relatedness. These are mapped through dimensionality reduction (PCA/T-SNE) on distances between each topic's probability distributions into 2D space. This shows whether the model developed distinct topics. The idea is that the model parameters and the number of topics should be tuned in such way, so that the circle overlap is minimized.

Wordcloud plots were created as an extra option of visualizing the topics (Figures 2&3).

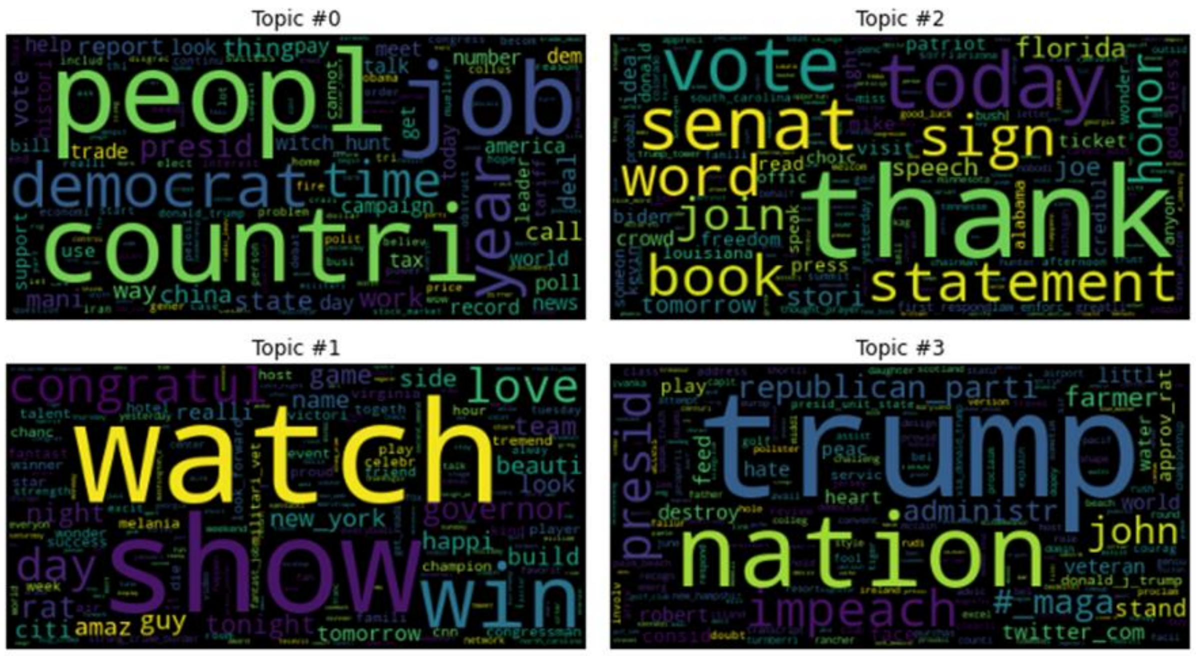


Figure 2: Topics visualized with WordCloud

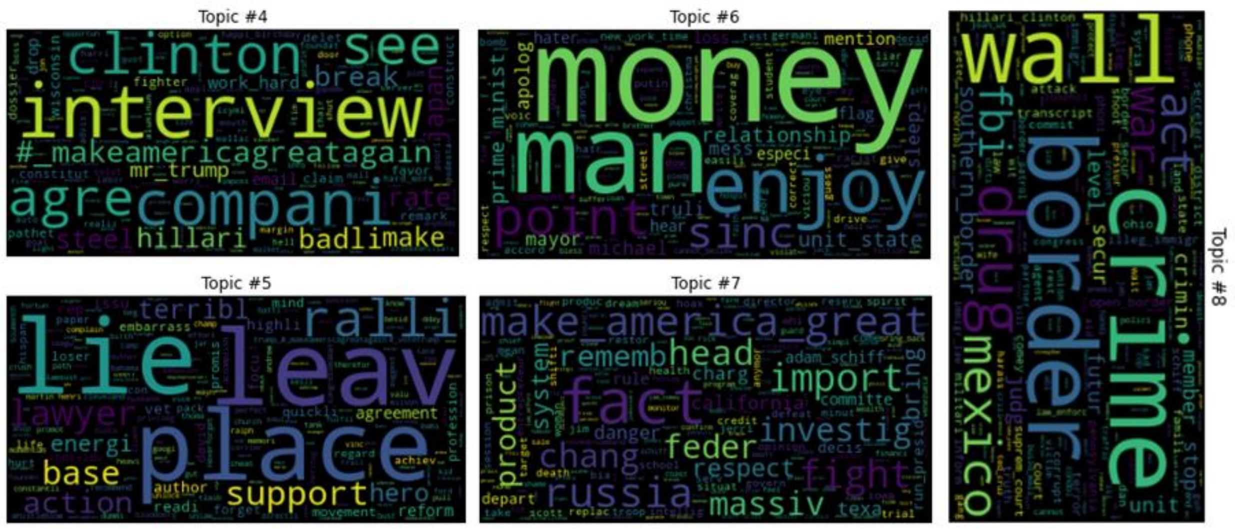


Figure 3: Topics visualized with WordCloud (LDA model)

K-MEANS CLUSTERING

The more dated model of bag-of words, even if it sometimes, depending on the documents, performs smoothly, it does a poor job in making sense of text data as it treats words as frequency counts without any sequence or meaning. This can be avoided with a newer technique called word embeddings that interprets in its model how words are semantically correlated to each other in a document.

A measure that is frequently utilized in topic modeling is TF-IDF (Term Frequency Inverse Document Frequency) which weighs the relevance of a word to a document in a collection of documents. By assigning a score based the word's frequency in a document and also a score of the same word in the entire set of the documents and multiplying them, TFIDF is calculated, providing us the most significant words in a document.

OPTIMAL NUMBER OF CLUSTERS

Firstly, we had to determine the optimal number of clusters that the k-means algorithm would use to cluster the tweets. To do so, we had to turn the tweets into matrices using word embeddings and the TF-IDF vectorizer. We used two different methods to discover the number of clusters that should be used on the k-means clustering algorithm. The first one was to apply Gaussian Mixture Models (GMM) on the embeddings to approximate the number of clusters that the algorithm dictates us to use. Due to the enormous number of matrices (shape of embeddings) we needed to apply some dimensionality reduction technique to gain simplicity and speed, trading off some accuracy. We applied Principal Component Analysis (PCA) and Uniform Approximation and Projection (UMAP) to reduce the matrix to 10 dimensions, thus making our model more time efficient. UMAP seemed to work better and provided with a better silhouette score than PCA. The criterion determining the best number of clusters was the Bayesian Information Criterion (BIC) which penalizes models with big number of clusters to avoid overfitting the model (Lavorini, 2018).The best number of clusters were found to be 6 (Figure 4).

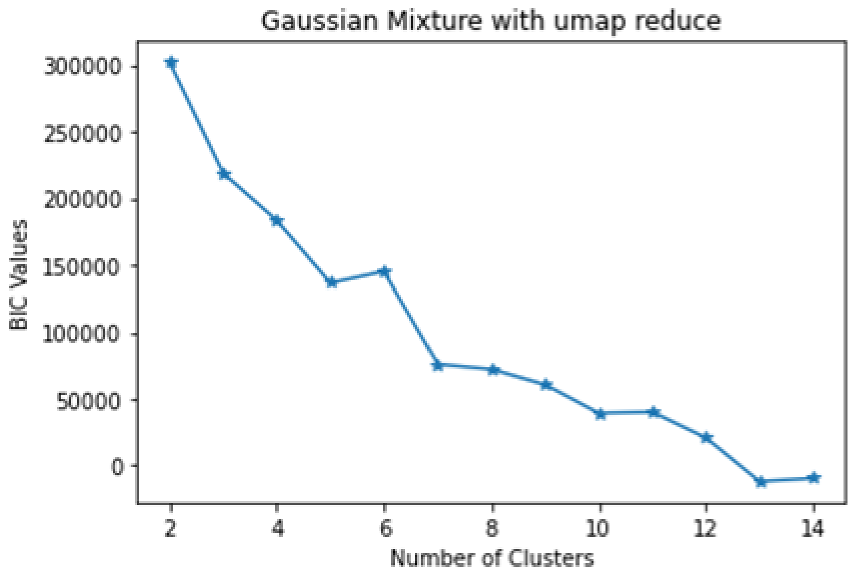


Figure 4: BIC scores vs. Number of Clusters after UMAP reduction

The second method was carried out using the elbow method (SSE) on the k-means algorithm for a range of cluster sizes. The point, where the "elbow" is detected on the plot, dictates the optimal number of clusters. Since the regular k-means implementation failed due to memory loss, MiniBatchKMeans was exploited as the alternative. The MiniBatchKMeans is a variant of the k-means algorithm which uses mini batches; subsets of the input data, randomly sampled in each training iteration. This algorithm reduces the convergence time of k-means, whereas at the same time attempts to optimize the same objective function. In contrast to similar algorithms, mini-batch k-means produces results that are slightly worse than the standard algorithm.

As shown on the figure below (Figure 5), the optimal number of clusters based on the inertia score is 7.

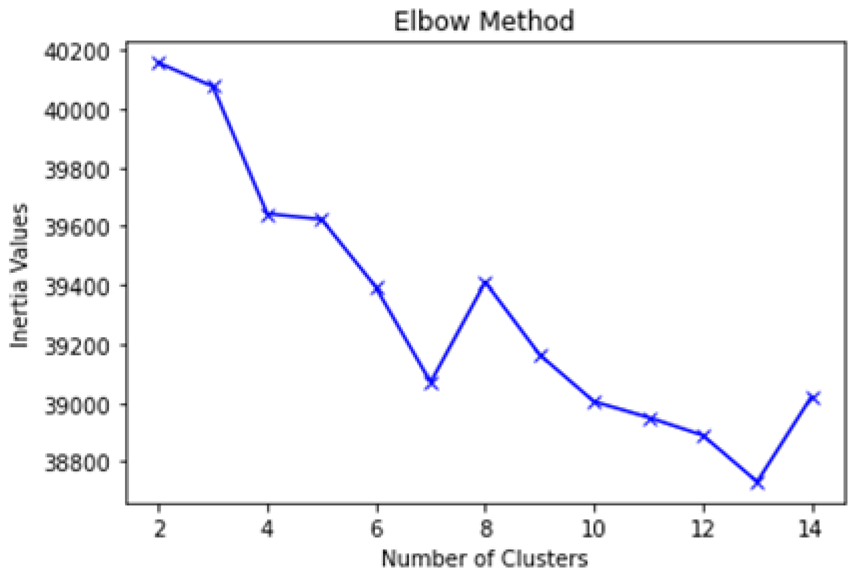


Figure 5: Inertia scores vs. Number of Clusters after UMAP reduction

PLOTTING THE CLUSTERS

At this point, we plotted the clusters generated by the k-means operation. The proposed algorithms are highly stochastic and very much dependent on choice of hyperparameters and are non-linear reductions. The main difference between T-distributed Stochastic Neighbor Embedding (T-SNE) and Uniform Manifold Approximation and Projection (UMAP) is the interpretation of the distance between clusters. One plot uses T-SNE which preserves local structure in the data. The second one uses UMAP, which preserves both local and most of the global structure in the data (Figures 6&7).

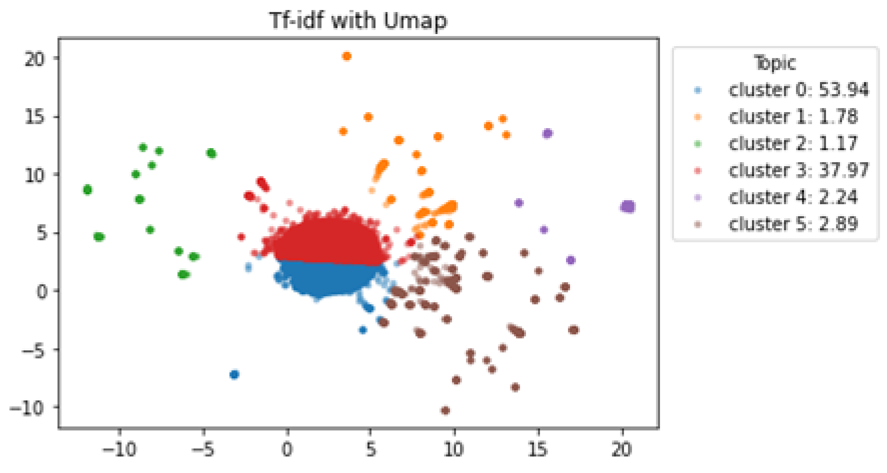


Figure 6: Visualizing the clusters after UMAP reduction

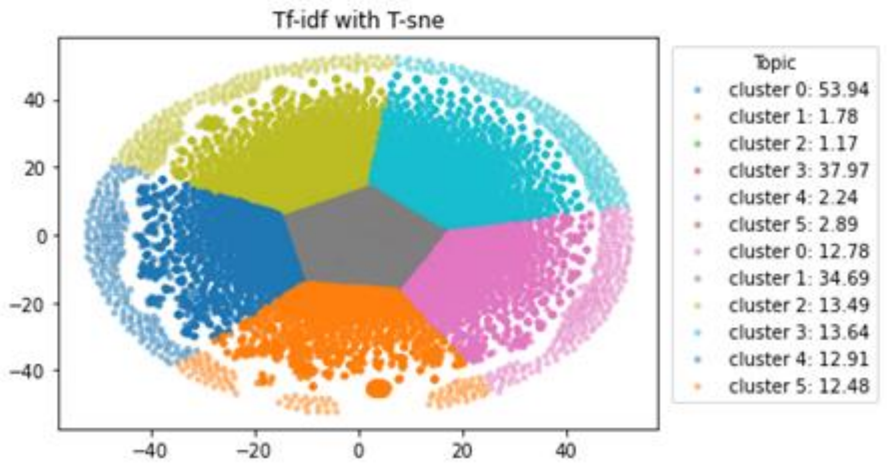


Figure 7: Visualizing the clusters after T-SNE reduction

TOPICS VISUALIZATION

Both k-means and MiniBatchKMeans algorithms predicted the tweets' topics in a rather short execution time, and so it was decided that the topics will derive from k-means clustering. We cycled through the clusters and printed out the top keywords based on their TFIDF score to check for any trends in the tweets. Using numpy, finding the top words was done by simply sorting the average values for each row, and taking the top N.

Finally, we used word cloud to visualize the results (Figure 8). From a detailed observation of the words that describe each topic, it was decided that the 6 clusters make more sense topic-wise.

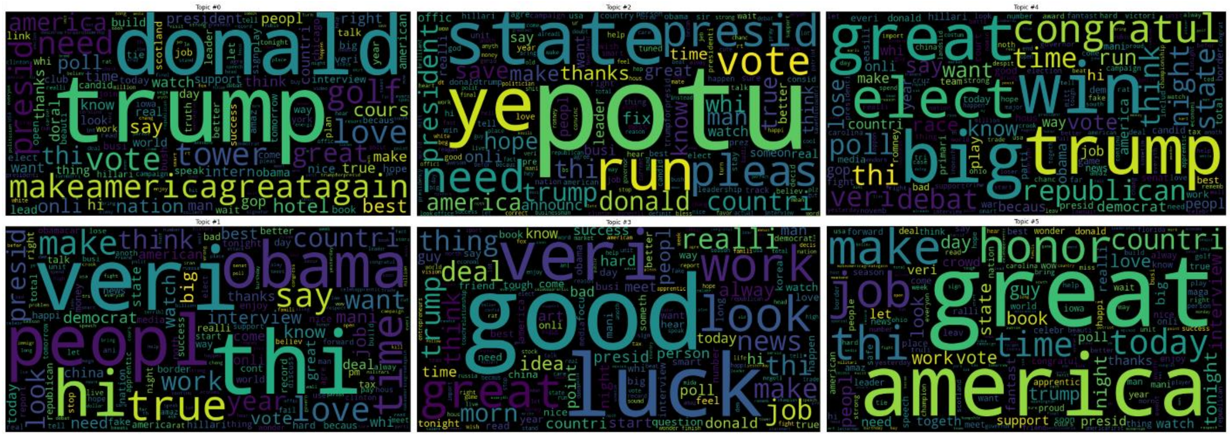


Figure 8: Topics visualized with WordCloud (k-means clustering)

FUTURE WORK

In our project we managed to collect, preprocess, clean and cluster our data with two different topic modeling techniques, but surely in a field like natural language processing there will always be complications and more work that can be done. Firstly, we could fine tune the hyperparameters of our LDA model to produce the best results possible. We could also implement various topic modeling techniques like Pachinko Allocation Model (PAM) and Parallel Latent Dirichlet Allocation (PLDA). PAM is a model that works with the same principals as LDA, identifying topics based on the words present in the corpus, but then improvises by modeling correlation between the generated topics (Wei Li). PLDA is the probabilistic version of LDA which has also the ability to handle more complexity in data (Yi Wang). As the computational power of our machines was limited, we had to make compromises with our approach. An alternative to executing the .py file locally would be the Google Colab. As a hosted Jupyter notebook service, it requires no setup to use, while at the same time it provides free access to computing resources including GPUs. A major issue that we faced was Donald Trump's peculiar way of tweeting that led to weird word production and topics with similar content. This could of course be avoided by choosing to work on a different dataset with better wording and more distinct topics.

CONCLUSION

By applying Natural Language Processing techniques and especially topic modeling we were able to give structure, categorize and finally draw information out of the tweets of the former USA president. The task of transforming natural language which may contain errors in text, ambiguities, irony or sarcasm, colloquialisms, slang and more, to data that can be understood by a computer is really challenging. But the results of these tasks prove to be extremely useful as they allow computers and humans to communicate effectively and proceed to the future of data-driven intelligence. After all, these changes in artificial intelligence may MAKE THE WORLD GREAT AGAIN.

REFERENCES

(n.d.). Retrieved from kaggle: https://www.kaggle.com/austinreese/trump-tweets

David M. Blei, A. Y. (2003). Latent Dirichlet Allocation.

Lavorini, V. (2018, 11 21). Retrieved from towardsdatascience: https://towardsdatascience.com/gaussian-mixture-model-clusterization-how-to-select-the-number-of-components-clusters-553bef45f6e4

Wei Li, A. M. (n.d.). Pachinko Allocation:DAG-Structured Mixture Models of Topic Correlations.

Yi Wang, H. B.-Y. (n.d.). PLDA: Parallel Latent Dirichlet Allocation for Large-scale Applications.