ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
UNIVERSITY OF PIRAEUS

DEMOKRITOS

Document Clustering And Topic Mining

by

Ioannis Atlamazoglou

Submitted
in partial fulfilment of the requirements for the degree of Master of
Artificial Intelligence
at the
UNIVERSITY OF PIRAEUS

Athens, July 2021

Document Clustering And Topic Mining

Ioannis Atlamazoglou

MSc. Thesis, MSc. Programme in Artificial Intelligence

University of Piraeus, NCSR “Demokritos”, July 2021

Author: Ioannis Atlamazoglou

II-MSc “Artificial Intelligence”

Athens, July 2021

Approved by the examination committee

(Signature)	(Signature)	(Signature)
Γ. Πετάσης		
. . . .	. . . .	. . . .
Georgios Petasis Reasearcher	George Giannakopoulos Reasearcher	Maria Dagioglou Reasearcher

Document Clustering
And Topic Mining

by

Ioannis Atlamazoglou

Submitted to the II-MSc "Artificial Intelligence" on July 2021
in partial fulfillment of the
requirements for the MSc degree

Acknowledgments

To Dr. and Researcher Georgios Petasis for all the support and knowledge.

Περίληψη

Ο σκοπός αυτης της διατριβής είναι η μη εποπτευόμενη εξόρυξη θεματων από κείμενα στα ελληνικά και η ομαδοποίηση τους σύμφωνα με αυτά τα θεματα, έτσι ώστε τα κείμενα που αναφέρονται στο ίδιο θέμα ή είναι παρόμοια, να βρίσκονται στην ίδια ομάδα. Μετά από έρευνα σχετικών εργασιών, διερευνήθηκαν δημοφιλείς μέθοδοι εξαγωγής θεμάτων όπως το LDA αλλά και μέθοδοι αναπαράστασης κειμένου όπως το BERT και το FASTTEXT τα οποία συγκαταλέγονται στις τεχνολογίες αιχμής που χρησιμοποιούνται για εξαγωγή αναπαραστάσεων κειμένου σε μορφή διανυσμάτων. Για την αξιολόγηση της ομαδοποίησης των εγγράφων σύμφωνα με τις αναπαραστάσεις τους, εφαρμόζονται αρκετές μετρικές οι οποίες είναι ενδεδειγμένες για τέτοιου είδους εργασίες.

Abstract

The purpose of this thesis is topic of extraction from documents in Greek language and document clustering according to these topics, so that documents that that refer to the same topic or are similar, belong in the same cluster. After researching related work, popular methods of topic extraction models such as the LDA and text representation methods such as BERT and FASTTEXT, which are among the state if the art technologies used to export text representations in the form of vectors, were explored and applied. To evaluate the document clustering performance according to their vector embeddings, several metrics are applied which are suitable for such tasks.

Contents

List of Tables

List of Figures

List of Abbreviations

ML	Machine Learning
NLP	Natural Language Processing
LDA	Latent Dirichlet allocation
RNN	Recurrent neural network
LSTM	Long short-term memory
BERT	Bidirectional Encoder Representations from Transformers

Chapter 1

Introduction

1.1 Problem description

The problem this thesis addresses is document clustering and topic mining. This process aims to assign a document to one or more tags or to one or more categories because the document may consist of text from different topics. Each document consists of a number of words in a specific unique order and is simply a mixture of different topics in different percentages. That is, in a document that talks about violence in schools, the greatest percentage of the topic of the document may concern violence, the second greatest percentage may be about school and the third one may concern children. Assuming this ratio of topics, we could then assign to this document the labels ”violence”, ”school”, ”children”. It is also possible for this document to be classified in the ”violence” category.

The input of such a system could be, for example, documents from news sources or media, and thus it could export categories or clusters depending on the input. So, each cluster contains all those documents that talk about the same topic or event. When new document is published talking about the same topic could be automatically assigned to the corresponding cluster[46] containing documents that talk about this topic. This way a better automated organization of our data is achieved without requiring a lot of time or a lot of human resources to do so.

Of course, applications that can take advantage of this technology do not end

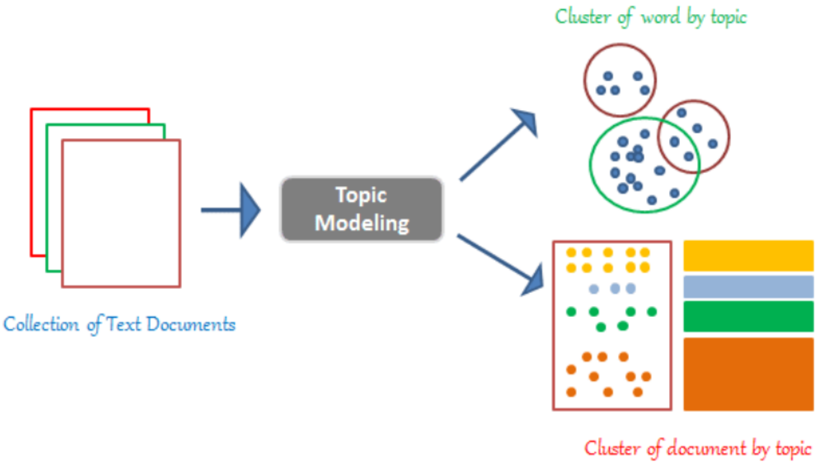


Figure 1.1: Topic modeling [18].

here. Natural Language Processing (NLP) [52] refers to any data source consisting of text. For example, managing and controlling large-scale service or product evaluations could be accomplished with this tool. That is, the evaluations could be categorized into positive and negative ones or by assigning the corresponding topic tag depending on the content of the evaluation, such as the label ”quality” or ”support”.

The application of this task is based on a pre-trained model in Greek documents since the set of data used in the experiments is in the Greek language. In order for the experimental methods used to be evaluated, special emphasis was placed on the data set being as close as possible to the work performed but also including some kind of validation tags that help evaluation in later stages.

Before deciding which methods were going to be used, research was done on which were the first topic-modeling methods when topic modeling [30] and topic extraction began to be widely used. Many research publications yielded interesting related work but, on the whole, the new methods presented were the most interesting. Historically, Latent Dirichlet Allocation (LDA)[25] has been a popular algorithm for the task of topic modelling.

According to LDA, each document is a distribution of topics and each topic is a distribution of words. So, each sentence inside a document consists of different topics. Hence, the main topic of a document is calculated by finding the most

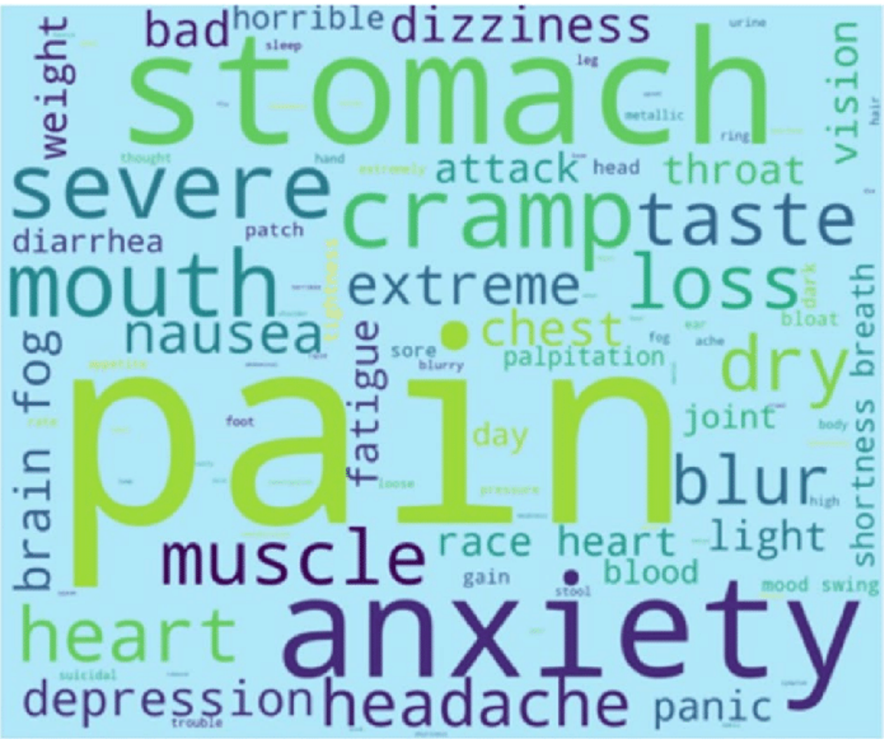
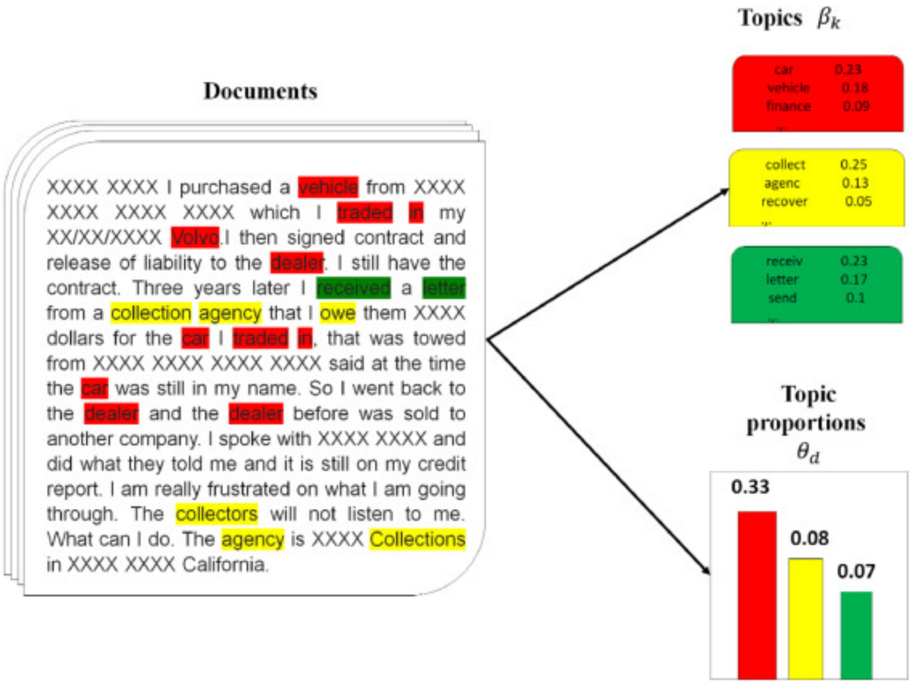


Figure 1.2: Digestion topic modeling visualization[2].

probable topic of the sentences.



XXXX XXXX I purchased a vehicle from XXXX XXXX XXXX XXXX which I traded in my XX/XX/XXXX Volvo.I then signed contract and release of liability to the dealer. I still have the contract. Three years later I received a letter from a collection agency that I owe them XXXX dollars for the car I traded in, that was towed from XXXX XXXX XXXX XXXX said at the time the car was still in my name. So I went back to the dealer and the dealer before was sold to another company. I spoke with XXXX XXXX and did what they told me and it is still on my credit report. I am really frustrated on what I am going through. The collectors will not listen to me. What can I do. The agency is XXXX Collections in XXXX XXXX California.

Figure 1.3: Topics[3].

The research then proceeded to more recent methods and was shown that transformer models performed better as these models deal with the idea of vector embeddings [51] where each word in a sentence is a unique vector, as its uniqueness is determined by the words on the left and right of that word (contextual embeddings). The same word therefore has different embedding in two different sentences and the more similar these sentences are then the more similar their vectors are.

This idea was started by the recurring neural network (RNN) [38] models where the memory feature first appeared, although over the years RNNs have been replaced mainly by models of attention following new papers leading up to the announcement of BERT, which revolutionized NLP and its applications as the accuracy on a wide range of tasks has increased significantly compared to older techniques.

In parallel with the experiments with BERT [28], which is based on the attention mechanism, it is decided that a comparison should be made with a non contextual embedding model called Fast-Text[36], which is nothing more than a dictionary that includes, for every single word it contains, only one vector and whose difference with BERT is that the vector of every word is not related to the context of the word before or after it but is the same for every sentence containing this word more specifically, this coding is based on the bag of words BOW [32] method.

In order for the models to be evaluated, suitable evaluation metrics must be selected, for evaluating clustering results. Therefore, clustering algorithms are needed in order to identify clusters of documents that are similar, which means that the content they are talking about is about the same. For example, the embeddings of two identical sentences are exactly the same. These two embeddings in two dimensions could obviously be depicted as two points that are tangent to the two-dimensional space and therefore, since they are very close to each other, they belong to the same complex. A simple criterion for whether they belong to the same group could be Euclidean distance.

Thus, various clustering metrics are applied to evaluate whether the clustering algorithms are able to cluster the documents correctly. Finally, after applying the our selected metrics in performance, the majority of measurements shows that BERT yielded indeed better results compared to the FAST-TEXT.

The task of this thesis try to gather documents and classify them by topic. In particular, in addition to the words used to create vectors and, therefore, used in

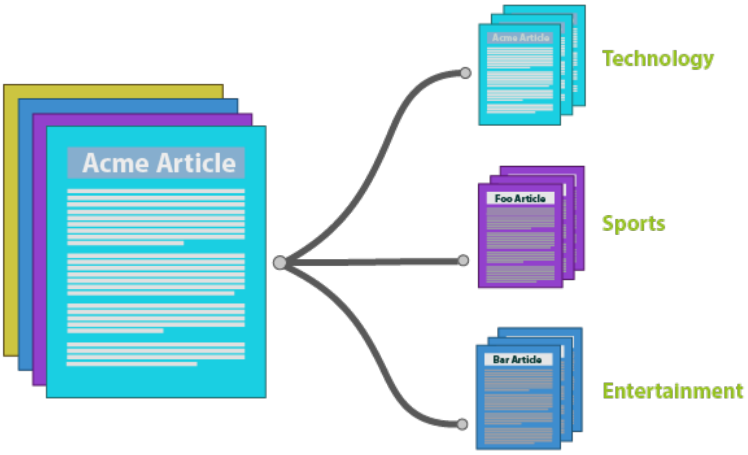


Figure 1.4: Document clustering[1].

clustering algorithms, the time entity could be included in data as a feature. Time is a very important characteristic in documents derived from news sources referring to events or news items, as the characteristic of time may be a strict condition of similarity in news items or events that can be connected many times.

1.2 Thesis structure

The thesis consists of five chapters: the first is an introduction, the second is about the related work which presents the background of topic modeling and document clustering. The third chapter deals with suggested approaches, includes the discussions about all these methods which are applied or tried and explains the implementation of this thesis.

The fourth chapter includes all the experiments, information about the data-set used, how algorithms were applied, the experimental process and results. At the end, the thesis finishes with the fifth chapter which includes the conclusions and future work, which includes a summary of the dissertation, conclusions and thoughts about future improvements.

Chapter 2

Related Work

2.1 Topic modelling

NLP is a research challenge in the field of computer science as it can enable computers to perceive some meaning of human natural language in various documents. A document could be anything from a book or an article or even an email. Topic modeling is very popular in the field of text mining and has many approaches which have been presented.

Many papers have been published in the field of NLP and many methods have been applied in other fields such as software engineering, political science, medicine, linguistics, etc. There are many topic modeling techniques and one of the most popular is Latent Dirichlet Allocation (LDA)[25].

Topic modeling is very important in natural language processing. Many researchers have just used or relied on the very popular LDA as a baseline to develop new methods. For example, some researchers have used this model to extract topics from political debates, while others have tried to extract opinions through a large number of evaluations on a particular topic.

Other researchers have also developed models that could understand code in the field of software engineering. These could detect differences in software code and eventually be able to categorize systems after training in millions of projects. Researchers have also focused on opinion mining systems, image categorization, recommendation systems and emotion categorization.

2.1.1 LDA as a baseline for other approaches

According to LDA, documents are treated as a mixture of topics. Each document is considered a probability distribution around a set of topics. Thus, a document can be represented by a possible distribution in different topics. That is, LDA considers documents as a mix of topics where each topic is a distribution over words. Thus, each sentence in a document has a distribution of topics according to the words that make it up.

Accordingly, each word of a sentence in the document represents a topic. So the word in the sentence with the largest proportion in the words that make it up represents the topic of this sentence. Consequently, the sentence with the highest proportion in the sentences of the entire document represents the topic that this document is ultimately related to.

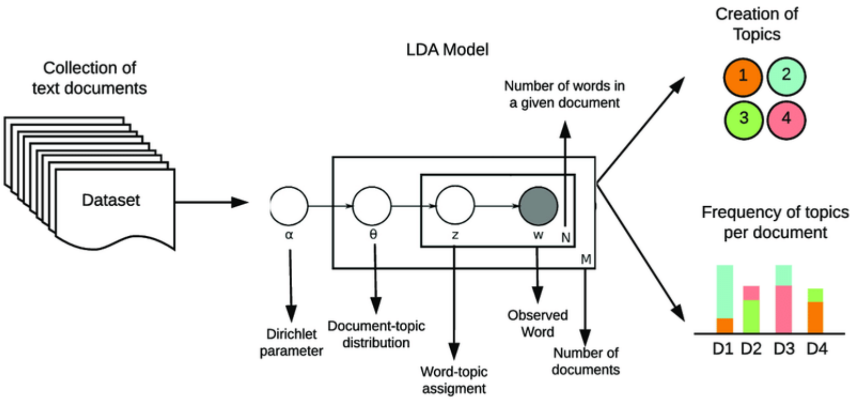


Figure 2.1: Schematic of LDA algorithm[16].

TopicSketch [55]is a very interesting approach to extracting topics from Twitter posts depending on the growth rate of tweets on a topic. In essence, this method tries to detect a large flow of tweets about a certain topic, that is, a big change in the growth rate of tweet creation for this particular news item.

This means that, at the time when there is a ”bursty” issue, many people simultaneously post a tweet about this topic so the tweet ratio in some time windows is much higher than the tweet ratio on a topic that is not generally flaming in the same time windows. It essentially tries to find the ”acceleration” of the flow of news about an event that is happening at that moment, thus signaling the existence of a fiery subject, such as the stormy leak of posts when an earthquake occurs as well as in several time windows after the event.

Another interesting approach is ET-LDA [21] which is a model that analyzes data from Twitter to extract facts but also to analyze tweet behaviors. TopicSpam [37] is also a model that tries to predict whether a review is true or false.

Another very important post is t-BERT [41] which combines LDA and BERT methods by merging their output as feature input into a deep-learning model. This paper examines whether topics along with word embeddings will ultimately improve Bert’s performance. Indeed, the performance of this model is clearly better than with the single use of Bert so a combination of these two methods shows better results.

Another very interesting approach is topic modeling in embedding spaces[29] which transfers topics to embedding space in order to see how topic-words are related to the words in the vocabulary which the model has learned.

In general, LDA has been used as an important baseline for creating models that perform various tasks such as being able to locate and categorize objects in an image, extracting political views such as a ratio, for example yes or no to a referendum according to tweeter’s data.

2.2 Clustering

There are different approaches and algorithms for clustering [57] tasks that can be divided into three subcategories: partition-based clustering such as k-means [58] , k-median [33], hierarchical clustering such as Agglomerative [22], Divisive and Density clustering such as HDBSCAN [39].

2.2.1 HDBSCAN

HDBSCAN[39] is an extension of DBSCAN and is a method that makes few assumptions about the data and does not depend on the noise, shape or size of the cluster areas. What it does is essentially try to locate very dense areas separated by sparse areas such as an island in the sea.

2.2.2 K-means

K-means is the simplest and most popular clustering algorithm. The number of clusters must be given as input to start the process. It initially assigns random points equal to a number obtained from the previously-mentioned parameter and now these points are the new centroids which are the centers of each cluster respectively.

In each iteration, it measures the distances between each point and the centers of the clusters and assigns the points to the corresponding cluster based on how close the points are around the corresponding center of the cluster. It also moves centroids appropriately trying to find equal distances between the center and the points belonging to this group. The algorithm stops when it completes the required number of iterations or when the centers of the clusters stop changing position in space.

2.2.3 Mean Shift

Mean Shift [26] is a hierarchical grouping algorithm. It does not make assumptions about data, nor does it need the number of clusters as input because it finds it on its own. The algorithm considers the points as a sample of some distribution and tries to locate the local maxima of the curve that symbolizes the density of the points using the Gradient descent method.

2.2.4 Agglomerative clustering

The agglomerative clustering [22] is one of the best-known types of hierarchical clustering algorithms and is based on point representation in a tree structure. More specifically, it recognizes each point as a separate cluster and then merges the nearest clusters, starting to form clusters of multiple points.

2.3 Embeddings

Texts are often not very easy to give as input to machine learning algorithms. This creates the need to convert words into numbers that are better understood by computers. A representation of words and therefore of texts are called embedding. Embeddings are just vectors representing words, sentences or even whole texts. There are many approaches that can turn an entire piece of text into vector embeddings.

A cluster refers to a collection of data points that are close together due to certain similarities. There are many clustering algorithms that work in different ways, but their common goal is to find clusters or texts that are similar. When the texts are similar they have similar embeddings and are close together within the embedding space. Therefore, clustering algorithms look for similarities or inequalities between data points.

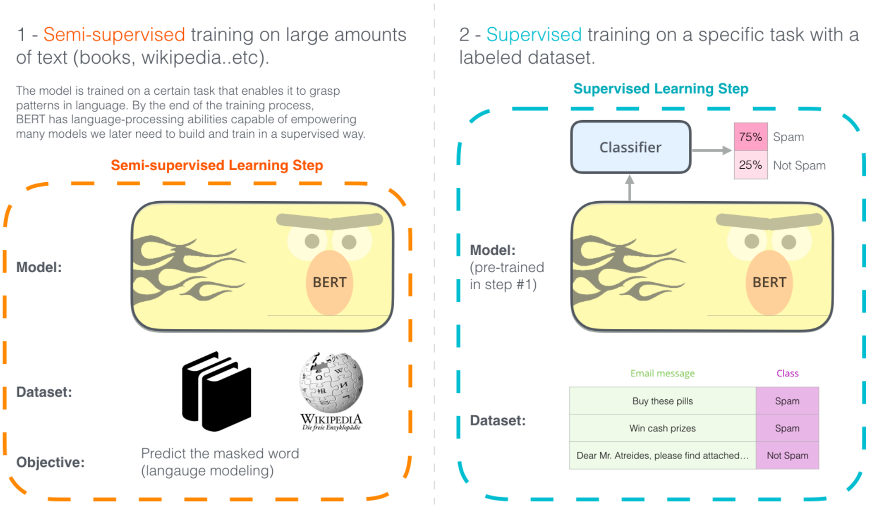
It is an unsupervised learning method as there is no label indicating which data points belong to which clusters. This means that the sole purpose of these methods is to be able to successfully detect the right labels. So the only thing that can be checked beforehand is the input that will be given to the respective algorithm and here the input is embeddings. So in order for the quality of the embeddings to be high enough to get the best results, enough focus must be given to the quality of the data-set to be experimented with.

Evaluating the performance of this task is quite arduous as the algorithm tries, taking all the embeddings as input, to render the clusters as output. Theoretically, each cluster should contain only similar documents. In practice, the clustering task is not perfect as some similar documents may not be so close eventually resulting in clustering errors.

2.3.1 BERT

In 2018, many deep-learning models gave a new twist to NLP, mainly in tasks such as question-answer or emotion classification, giving state-of-the-art results, with the state-of-the-art model being BERT (Bidirectional Encoder Representations from Transformers), which is based on Transformer architecture.

BERT is pre-trained through two unsupervised methods: the first one is mask modeling, i.e. the model tries to predict the missing word based on its left and right content, and the second is to predict the next sentence where the model tries to correctly predict if one sentence follows another. So here also comes the idea of transfer learning as BERT does not need to be trained from scratch in every new task. BERT uses multiple levels of attention as well as multiple attention heads, reaching hundreds of different attention mechanisms.



1 - Semi-supervised training on large amounts of text (books, wikipedia..etc).

The model is trained on a certain task that enables it to grasp patterns in language. By the end of the training process, BERT has language-processing abilities capable of empowering many models we later need to build and train in a supervised way.

2 - Supervised training on a specific task with a labeled dataset.

Email message	Class
Buy these pills	Spam
Win cash prizes	Spam
Dear Mr. Atreides, please find attached…	Not Spam

Figure 2.2: The idea of transfer learning[5].

BERT actually learns multiple attention heads that work in parallel with each other. Multi-head attention allows the model to capture a wider range of word relationships than could be done with a single attention mechanism. BERT also accumulates several levels of attention, each of which acts as an output to the next level entry. Through this repetitive synthesis of word embedding, BERT is able to form very rich representations as it reaches the deeper levels of the model. Because attention heads do not share the parameters, each head learns a unique attention pattern.

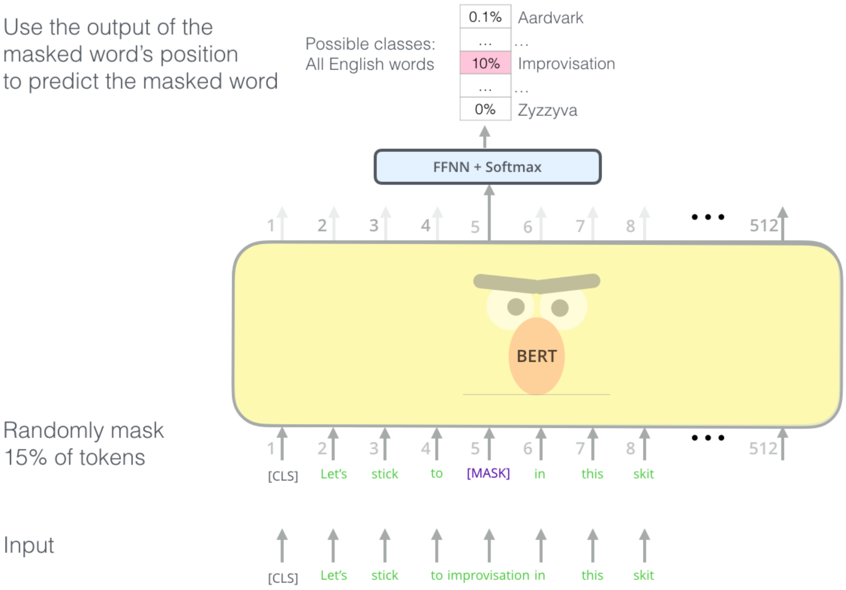

Use the output of the masked word's position to predict the masked word
Possible classes: All English words
0.1% Aardvark
...
10% Improvisation
...
0% Zyzzyva
FFNN + Softmax
1 2 3 4 5 6 7 8 ... 512
BERT
Randomly mask 15% of tokens
[CLS] Let's stick to [MASK] in this skit
Input
[CLS] Let's stick to improvisation in this skit

Figure 2.3: BERT during language modeling while one word is masked and trying to predict it[4].

2.3.2 FASTTEXT

Another different approach model have also been proposed which is called FASTTEXT. This can achieve a very good performance in word representations, especially in the case of rare words by using information at the character level. The algorithm is a non- contextual approach because each word is converted into a vector embeddings without any dependence on the other words surrounding it in a sentence. So, unlike BERT, the same word in two completely different sentences based on FASTTEXT will be represented with exactly the same vector which, of course, always depends on the training done before.

2.4 Technical background

2.4.1 RNN & LSTM

While LDA is primarily about statistical calculations, innovation began when machine learning and especially departmental learning gave a different approach to NLP. The first deep neural networks [47] attempted to give another representation to the data in general. Long-term memory (LSTM) [50] was the first recurrent neural network (RNN) model to introduce the concept of memory.

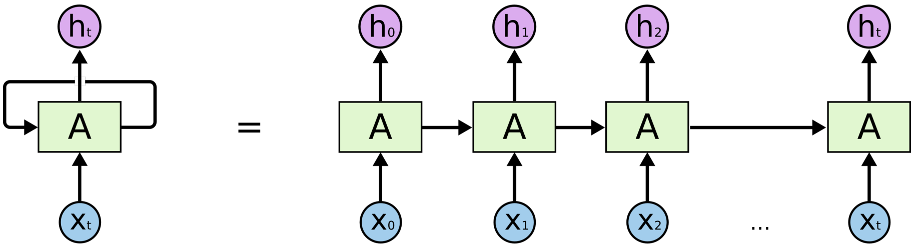


Figure 2.4: RNN architecture[20].

RNN could read a phrase word by word serially without resetting the parameters from the beginning and this gave them the great advantage that they could see all the words of the sentence without changing the parameters of the model. Thanks to the attention [34] mechanism that essentially provided a representation of the relationship between the words of a sentence, RNNs were allowed to emphasize specific input words in the process of predicting the next output word, which increased the quality of these models.

A task example based on RNNs could be a chat bot or automatic translation system. The main problem of LSTM is that the model works in serial mode which makes training slow and also that, due to the vanishing gradient [35] issue, it is inefficient in large sentences or big text.

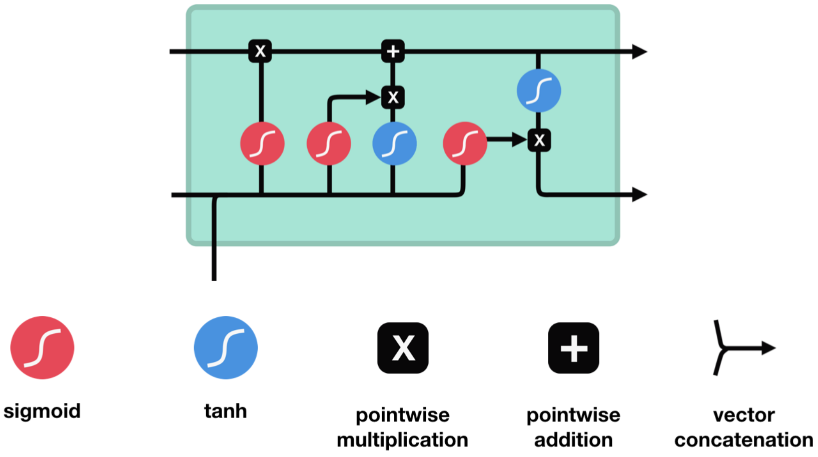


Figure 2.5: LSTM architecture[6].

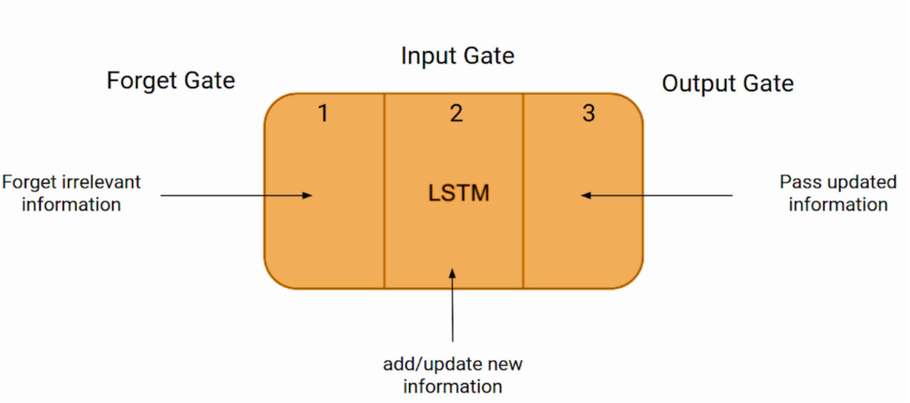


Figure 2.6: LSTM architecture[17].

2.4.2 Attention mechanism

However, the most important component of LSTM that helped in its further development of NLP architectures is as mentioned above in the attention mechanism. In LSTM, the final state of the RNN or encoder must contain information about the entire input sequence. An important disadvantage of this architecture is that the encoding must represent the entire input sequence as a single vector[48], which can cause information loss as all information must be compressed in this vector.

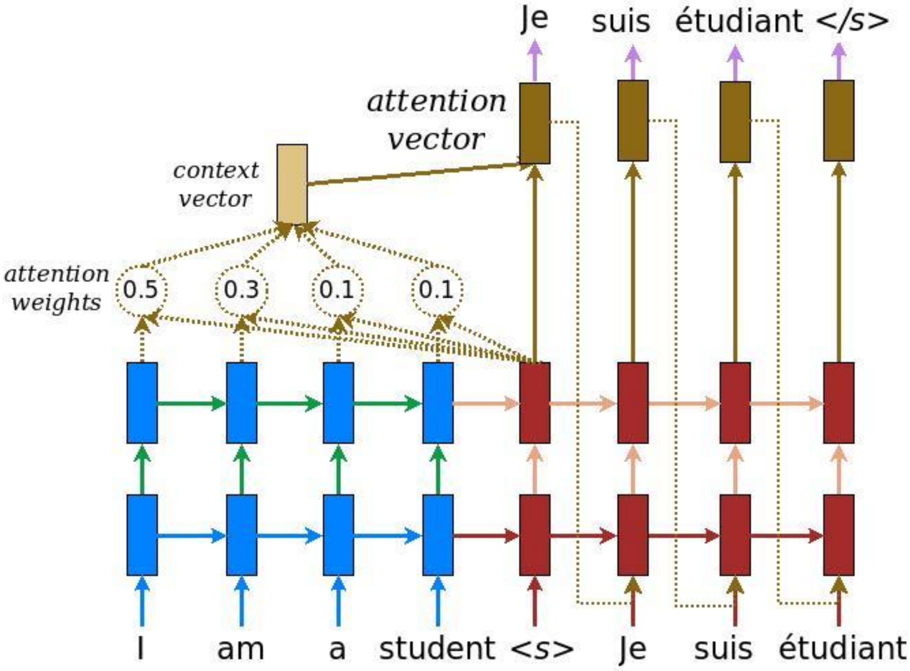


Figure 2.7: LSTM attention mechanism.

As explained in the paper of attention in 2015 [44], a possible issue with the encoder-decoder approach [27] is that a neural network must be able to compress all the necessary information of a sentence in a fixed length vector. This can make it difficult for the neural network to deal with large sentences, especially those that are larger than the sentences in the training.

2.4.3 Transformer

In 2017, a new network architecture is proposed called Transformer[53] and instead of using recurrence, it relies entirely on attention mechanism to find input- output relationships. The transformer is a stack of encoders and decoders.

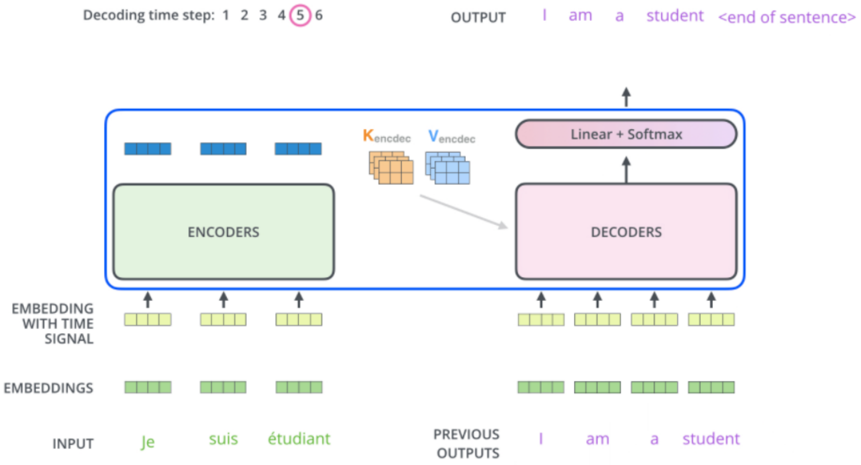


Figure 2.8: The transformer architecture[7].

2.5 Evaluation metrics

2.5.1 Homogeneity

Homogeneity metric[10] computes how much clear the cluster are. High homogeneity means that each cluster don't include many documents from other clusters. It's like measuring precision on pair permutations.

2.5.2 Completeness

Completeness metric[11] computes how many documents clustered are correctly clustered and there is no dependence in the existence of documents which belongs to other clusters. It's like measuring recall on pair permutations.

2.5.3 V-measure

V-measure metric[12] is the harmonic mean between Homogeneity and Completeness. It's like F1 on homogeneity and completeness.

2.5.4 Rand index

Rand Index (ARI)[13] creates all possible pairs in order to calculate the percentage of correct decisions made by the algorithm. It can be computed using the following formula where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives.

$$ARI = (TP + TN)/(TP + FP + FN + TN) \tag{2.1}$$

2.5.5 Adjusted Rand index

Adjusted Rand Index[14] calculates the percentage of how much better is our clustering compared with a random permutation model, defining random clustering, which requires that we have a distribution of clusters, and each cluster has same number of instances. In practice, there is no chance this is true, because it can affect cases like the number of clusters is known from beginning of the experiment, so ARI may not fit so good our task. So, ARI compares its random clustering performance with the real clustering performance both based on the true labels. If the random clustering is better than the clustering result of the experiment the number is negative, if the is not better is positive and if are equal then the number is zero.

$$adjusted_rand_score(a, b) == adjusted_rand_score(b, a) \tag{2.2}$$

2.5.6 Adjusted Mutual Information

Adjusted Mutual Information[15] may then be defined to be the below formula. It is mutual information corrected for chance.

$$AMI(U,V) = [MI(U,V) - E(MI(U,V))]/[avg(H(U),H(V)) - E(MI(U,V))] \quad (2.3)$$

2.5.7 Normalized Mutual Information

Normalized Mutual Information (NMI)[8] calculates the mutual information between two clusterings. It is based on the probability a point to belongs to both clusters U and V.

$$MI(U,V) = \sum_{i=1}^{R} \sum_{j=1}^{C} P_{UV}(i,j) \log \frac{P_{UV}(i,j)}{P_U(i) P_V(j)}$$

Figure 2.9: Normalized Mutual Information formula[8].

2.5.8 Fowlkes-Mallows index

Fowlkes-Mallows Index (FMI)[9] is defined as the below formula where TP is the number of true positives, FP is the number of false positives, and FN is the number of false negatives. TPR is the true positive rate, also called sensitivity or recall, and PPV is the positive predictive rate, also known as precision.

$$FM = \sqrt{PPV \cdot TPR} = \sqrt{\frac{TP}{TP+FP} \cdot \frac{TP}{TP+FN}}$$

Figure 2.10: Fowlkes-Mallows index formula[9].

Chapter 3

Proposed Approaches

3.1 Document clustering with LDA

Document clustering is a very difficult task and different approaches have been tried to get the best results. So in this thesis, many applied approaches have been tried such as LDA, BERT and FastText. As a first set of experiments, we used a corpus with known (labelled) topics (i.e. 20-newsgroup corpus), in order to evaluate an unsupervised approach, more specifically LDA, on the task of topic extraction.

The purpose was to find topics and classify some documents into these topics using LDA. The corpus was split in train and test sets, where the training set of documenst were analysed by LDA to extract the topics, and the held out test documents were classified into the extracted topics. Initial results were not satisfactory. Further attempts to tune hyperparameters did not help, as the performance of LDA has not improved significantly.

3.2 Embeddings-based clustering

An alternative approach for extracting topics from documents is been proposed. This approach tries to exploit embeddings representing documents, and applies several clustering algorithms (i.e. KMeans, HDBSCAN, Agglomerative clustering and Mean Shift) in order to cluster documents into topics. (We assume that each cluster is a topic). This approach involves the following tasks: Document preprocessing, including text cleaning, tokenisation, document represntation through embeddings, clustering.

The application of this thesis includes a complete pipeline in which the embedding process for each document is the key part. Flair[23] library is used to handle the basic functions of the BERT model, such as the embedding process. Flair is based on the "hugging face" library of "transformers"[54] which also provides thousands of pre-trained models for performing tasks on texts such as classification, topic-extracting and question-answering.

3.3 Text pre-processing

Another pre-processing function is document preparation. Text is the main input for any type of NLP job such as sorting, question-answer, emotion analysis. The text contains different symbols and words that do not convey meaning to the model during training and need to be removed before being given to the model effectively.

This method is called text preprocessing. Text preprocessing involves ASCII[56] characters removing, lowercase convertion, punctuation removal and any symbol that is not alphabetical or numeric. Another method is to remove stop words which are words that do not add much meaning to a sentence, i.e. yes, it will.

3.4 Text tokenization and lemmatization

After the above basic refinement, tokenization [31] is sometimes used, i.e. turning a sentence into a list of its words; stemming is another technique, which is the process of reducing words to their roots. Also, lemmatization is another method of text purification which, unlike stemming, reduces inflected words correctly ensuring that the root word belongs to the language. In the implementation of this thesis, only punctuation, ASCII and conversion of words from uppercase to lowercase have been used.

3.5 Using embeddings to represent documents

After text preprocessing, documents are ready to be converted to embeddings. The basic idea is to get the vector embeddings of each document as input to clustering algorithms for the purpose of document clustering. First each document breaks into sentences and each sentence feeds the pre-trained BERT model to return embeddings vector in size 512. Thus, each document is converted into n vectors embeddings where is the number of sentences.

Sentences that are too small, such as two words or less, or more than 512 words (exceeding the number of input tokens a typical transformer such as BERT can accept) are deleted from the dataset. Typically, the first sentences contain the main topic of the whole document but we take the average of the vectors of all the sentences. For this reason, the implementation initially uses only the title of the text at first, then uses the title and the first sentence and gradually increases the information window seen by the model to study the relationship of the first sentences to the content of the central idea of the entire document.

Another parameter is the type of employed embeddings. Embeddings can be separated into two major categories: 1) contextual embeddings exploit the context arround each word in the calculation of its embedding. Thus, the same word can have different embeedings based on the context the word has been used in. A typical approach for obtaining contextual embeddings is BERT. 2) Non contextual embddings, where the context of words is not considered during embeddings generation. A word has the same embeedings vector, no matter the context it appears in. Popular approaches for this kind of embeddings are Word2Vec[40], GloVE[42], and FastText[24]. For the work presented in this thesis, we used only FastText, because of its ability to hadle better unknown (out-of-vocabulary) words, as it uses sub-word units that include individual characters.

3.6 Embeddings clustering

Once all the documents have been converted into embedding vectors, the final step of the proposed approach is to cluster the embeddings representing the documents. Several clustering algorithms have been evaluated on the task of topic extraction starting with Kmeans waiting for the number of input clusters, which number, thanks to the data-set created especially for this task, is known from the beginning.

Then, HDBSCAN is applied by reducing the dimensionality of the integration vectors before using UMAP[45] function. Mean-shift and agglomerative clustering follow without the need for the number of clusters as an input. Some visualizations is developed using PCA [49] and TSNE [43] to show how document embedding vectors are displayed, in which some documents are very close together giving the impression that they belong to the same topic (cluster).

Chapter 4

Methodoly, Experiments And Results

4.1 Thesis approach

The problem that this thesis addresses is topic modeling and, therefore, the creation of a model that is capable of clustering similar documents. What is understandable is that topic modeling tries to extract the topics that a document consists of. As a result, the topic of a document can be visible with labels, without having to read it in order for a human to draw conclusions.

The task of topic-modeling model has concerned many researches in the field of natural language processing. Taking language as an important parameter and the fact that we need embeddings in the language of the documents, the availability of pre-trained embeddings for the target language is important. In the case of this thesis, the language is Greek and, therefore, all experimentation are done with a Greek data-set. Transformers through the Hugging Face platform offers many pre-trained models in many languages (including Greek). FASTTEXT also has a Greek model. For each task an appropriate, labeled with topics, dataset is required, which can be used for evaluating and comparing the clustering algoriths. Document clustering is an unsupervised task and a good data-set is required.

4.2 Topic modeling methods

Many methods have been published and one of them is LDA. This method assumes that a document is written based on certain topics proportions. That is, it assumes that every word leads to a topic and each sentence is the sum of its proportions of the topic of each word. Thus, each document is the sum of the proportions of the topics of the sentences where the main topic of a document is the most frequently occurring topic.

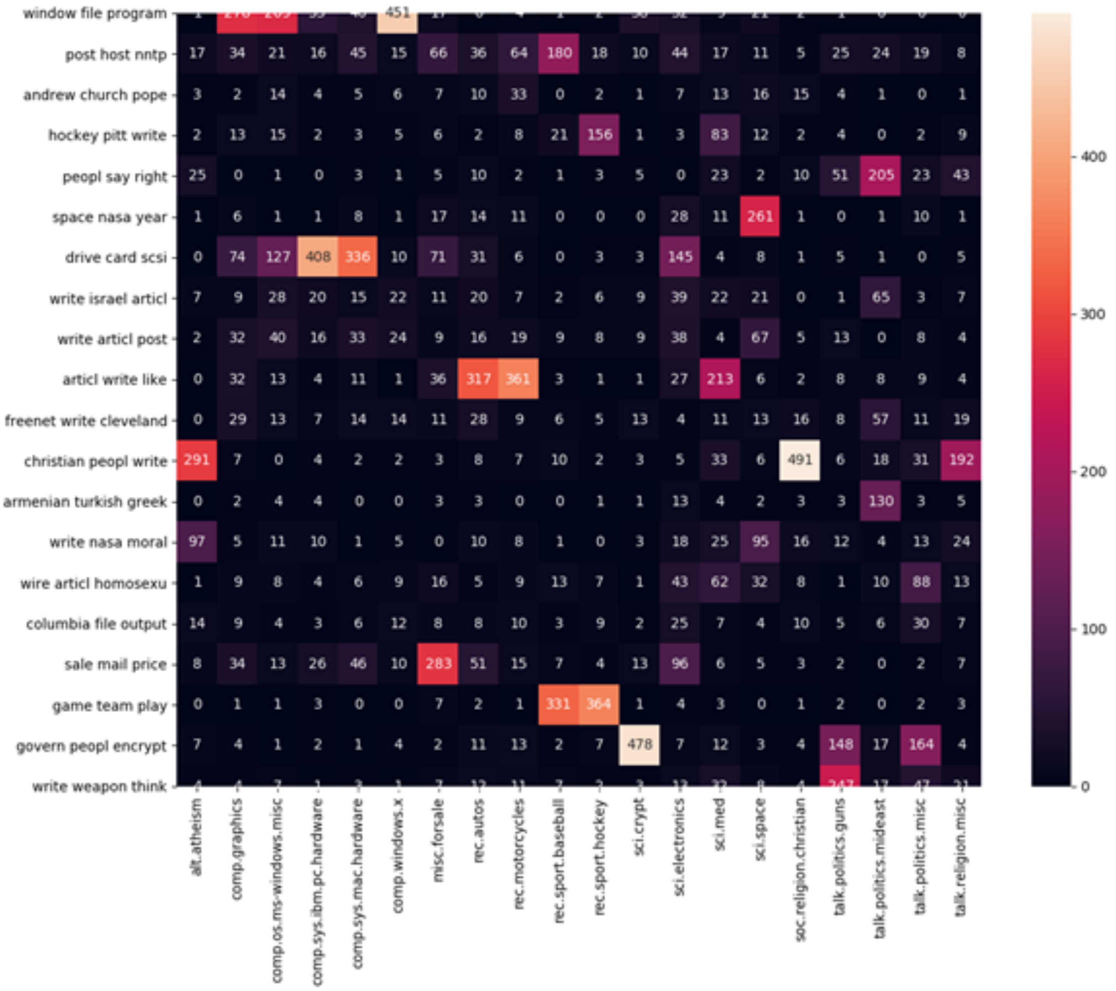
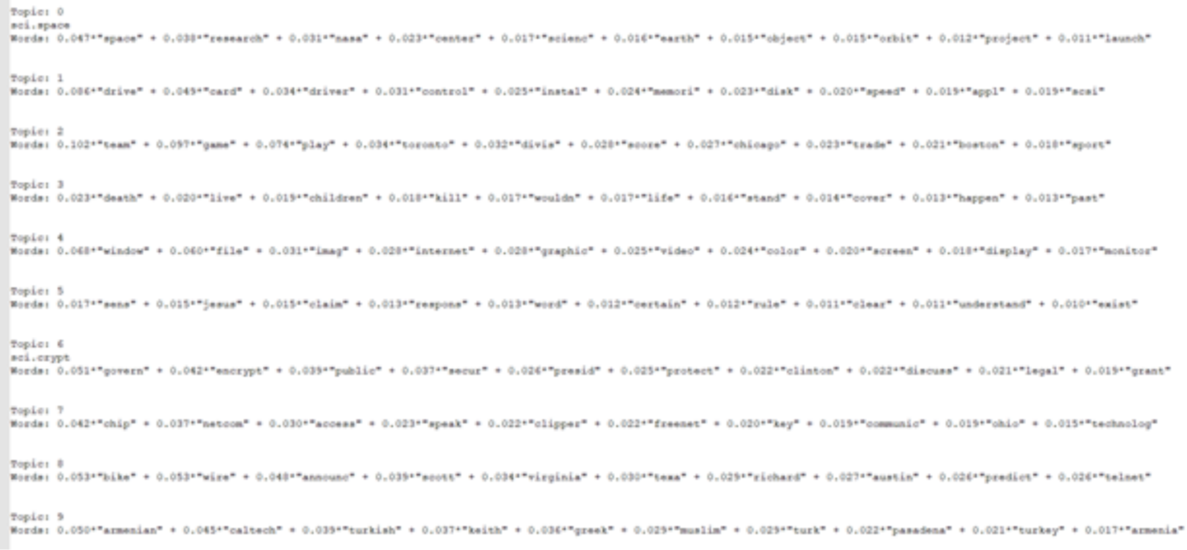


Figure 4.1: Prediction results of LDA with unseen documents.

Initially, this method was evaluated using a known 20-news-groups data-set containing various documents, taken either from emails or other sources, and a label depending on the topic to which it refers. LDA analyzed this data-set and some documents was applied for evaluation of its ability to finally be able to find the topic to which it belongs. Due to the weak performance of LDA on the topic modelling task for the 20newsgoups dataset, our bibliographic reaserh has been extended to identify other approaches suitbale for our task.



```
Topic: 0
sci.space
Words: 0.047*"space" + 0.038*"research" + 0.031*"nasa" + 0.023*"center" + 0.017*"scienc" + 0.016*"earth" + 0.015*"object" + 0.015*"orbit" + 0.012*"project" + 0.011*"launch"

Topic: 1
Words: 0.086*"drive" + 0.049*"card" + 0.034*"driver" + 0.031*"control" + 0.025*"instal" + 0.024*"memori" + 0.023*"disk" + 0.020*"speed" + 0.019*"appl" + 0.019*"scsi"

Topic: 2
Words: 0.102*"team" + 0.097*"game" + 0.074*"play" + 0.034*"toronto" + 0.032*"divis" + 0.028*"score" + 0.027*"chicago" + 0.023*"trade" + 0.021*"boston" + 0.018*"sport"

Topic: 3
Words: 0.023*"death" + 0.020*"live" + 0.019*"children" + 0.018*"kill" + 0.017*"wouldn" + 0.017*"life" + 0.016*"stand" + 0.014*"cover" + 0.013*"happen" + 0.013*"past"

Topic: 4
Words: 0.068*"window" + 0.060*"file" + 0.031*"imag" + 0.028*"internet" + 0.028*"graphic" + 0.025*"video" + 0.024*"color" + 0.020*"screen" + 0.018*"display" + 0.017*"monitor"

Topic: 5
Words: 0.017*"sens" + 0.015*"jesus" + 0.015*"claim" + 0.013*"respons" + 0.013*"word" + 0.012*"certain" + 0.012*"rule" + 0.011*"clear" + 0.011*"understand" + 0.010*"exist"

Topic: 6
sci.crypt
Words: 0.051*"govern" + 0.042*"encrypt" + 0.039*"public" + 0.037*"secur" + 0.026*"presid" + 0.025*"protect" + 0.022*"clinton" + 0.022*"discuss" + 0.021*"legal" + 0.019*"grant"

Topic: 7
Words: 0.042*"chip" + 0.037*"netcom" + 0.030*"access" + 0.023*"speak" + 0.022*"clipper" + 0.022*"freenet" + 0.020*"key" + 0.019*"communic" + 0.019*"ohio" + 0.015*"technolog"

Topic: 8
Words: 0.053*"bike" + 0.053*"wire" + 0.048*"announc" + 0.039*"scott" + 0.034*"virginia" + 0.030*"texa" + 0.029*"richard" + 0.027*"austin" + 0.026*"predict" + 0.026*"telnet"

Topic: 9
Words: 0.050*"armenian" + 0.045*"caltech" + 0.039*"turkish" + 0.037*"keith" + 0.036*"greek" + 0.029*"muslim" + 0.029*"turk" + 0.022*"pasadena" + 0.021*"turkey" + 0.017*"armenia"
```

Figure 4.2: Output of LDA after training with the NewsGroups data-set.

In order to apply clustering, we need to convert documents into vectors, whose distance can be measured by the various clustering algorithms (K-means, HDBSDCAN, Meanshift and Agglomerative clustering). For representing documents as vectors, pre-trained embeddings have been used, both non-contextual (focusing on FASTETX) and contextual (focusing on BERT). Thus, by converting the texts into numbers, and specifically into vectors, they could later be given as input to clustering algorithms which, by taking these vectors, could create clusters. It is expected that similar documents will be close to each other, and belong to the same cluster, and finally classify similar documents together.

4.3 The data-set used

Since we did not have access to an existing dataset, labelled with topics, in the Greek language, we followed an approach to create such a corpus. For the purposes of this thesis, we collected 200 topics from Google news, along with 4 news items for each topic. The data-set includes similar documents and, a label that reveals which documents are in the same cluster, that is, they talk about exactly the same thing.

Specific script is created which takes some of the article links under each topic that appears on the main page of the Google news site automatically every eight hours. More specifically, by taking the link of the first five similar articles but from a different source under each cluster, i.e. under each Google news topic, the document could be extracted from the corresponding source found by Google.

Essentially, documents from each link of the same news item are quite similar, if not exactly the same, so along with the extraction of the documents, labels are included that indicate the unique number of the cluster, or otherwise the news item that belongs to it, so it is known from the beginning which documents or otherwise which news items are the same and belong to the same cluster.

The reason why documents are exported every eight hours is because there needs to be uniqueness in the content of the documents and, consequently, the news as Google news often brought to the fore news that had reappeared in the past. This would result in extracting documents from multiple clusters that would address the same issue, which would distort the purpose of the process but also create problems of unreliability of the model metrics. So the choice of the eight-hour interval was considered ideal after testing so that most, if not all, of the topics were unique within the data-set.

The process of extracting documents is completed when about two hundred clusters are collected, i.e. different topics that were identified by Google news through various sources - news articles; a manual formal check begins to ensure that these two hundred topics are unique and do not contain garbage. For example, a cluster from the dataset contains about four documents, all of which refer to just one topic, which may be a new earthquake activity. It is an absolute prerequisite that all these four articles that belong to this cluster only talk about this news item. For the purpose of evaluating the performance of the model, the cluster ID was created for each article in order to know which cluster each document belongs to.

Pygooglenews library was used to export HTML content because it is partially exported every eight hours to avoid duplicate articles and to make sure that only different news items are exported. The format of the file that contains the document links is as follows: cluster ID, article title, and the URL that contains the document.

Scrappy is then used as a library to retrieve the document via the URLs given as input. This is immediately followed by the saving of the documents in a file, in which each record is a document or, in other words, an article, and on the left of the document, i.e. in the next column, the unique cluster number is included, which indicates which cluster the document belongs to.

4.4 Dataset clean up

The extracted documents may contain punctuation marks and some strange characters which could reduce the quality of the embeddings. So with the help of the text preprocessing function that has been created according to the needs of this process, all the weird characters, punctuation marks and numbers are removed from the documents. So when it is ensured that the dateset does not contain unacceptable characters, then the process of converting the documents to vectors begins. What clustering algorithms need are the vector representations of the documents. The model used for the export of embeddings is BERT, which is pre-trained, i.e. its weights have already been adjusted based on the training made by the Greek community in Greek data. So the model does not need to be trained from scratch as the model is already able to convert the documents to vectors.

4.5 Documents handle for embeddings

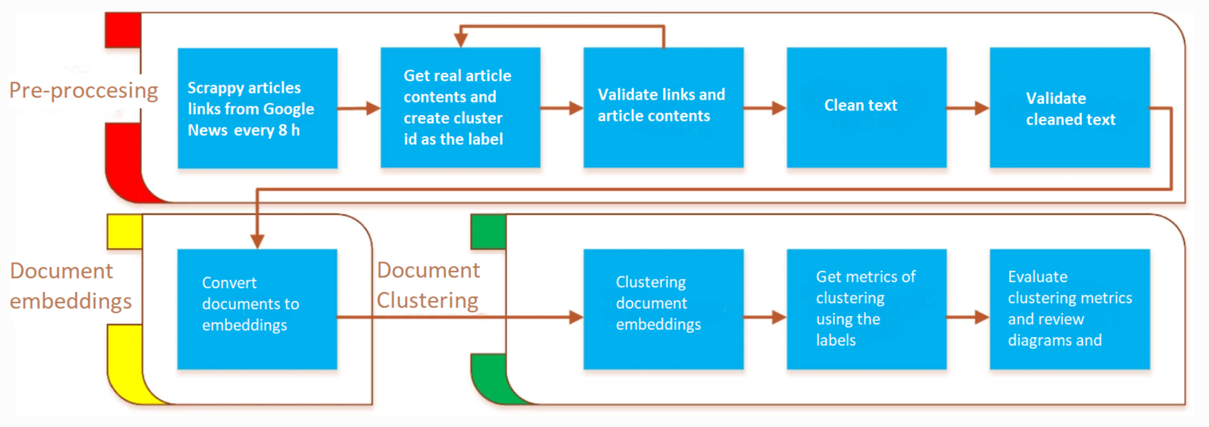

Pre-proccesing
Scrappy articles links from Google News every 8 h
Get real article contents and create cluster id as the label
Validate links and article contents
Clean text
Validate cleaned text
Document embeddings
Convert documents to embeddings
Document Clustering
Clustering document embeddings
Get metrics of clustering using the labels
Evaluate clustering metrics and review diagrams and

Figure 4.3: Thesis' document clustering approach architecture.

The creation of embeddings is performed in a loop n times, where n is the number of sentences of each document that the model processes in each iteration in order to create the vector representations. For example, in the first iteration, the model is fed with the title of each document without the text. In the next iteration, the first sentence of the document is added and the number of sentences the model sees gradually increases over the iterations.

So, little by little, the whole document is fed to the model. This is one way for the relationship of the number of sentences in a document with the performance of the model to be checked. The clustering method is tested with the help of four algorithms of different philosophy in order to measure the performance of each one, also of course in relation to the increase of the sentences that the model is fed with. The results concerning each number of sentences for each clustering algorithm are stored in a file for later use in the visualization of each metric. The algorithms used to create embeddings are BERT and FASTTEXT.

There is no need to optimize the clustering algorithms as, from what was proved after experiments, the algorithms give better results with default parameters. The purpose of embeddings clustering is to find documents that belong to the same cluster. Labels created together with the dataset are used to measure clustering algorithms performance.

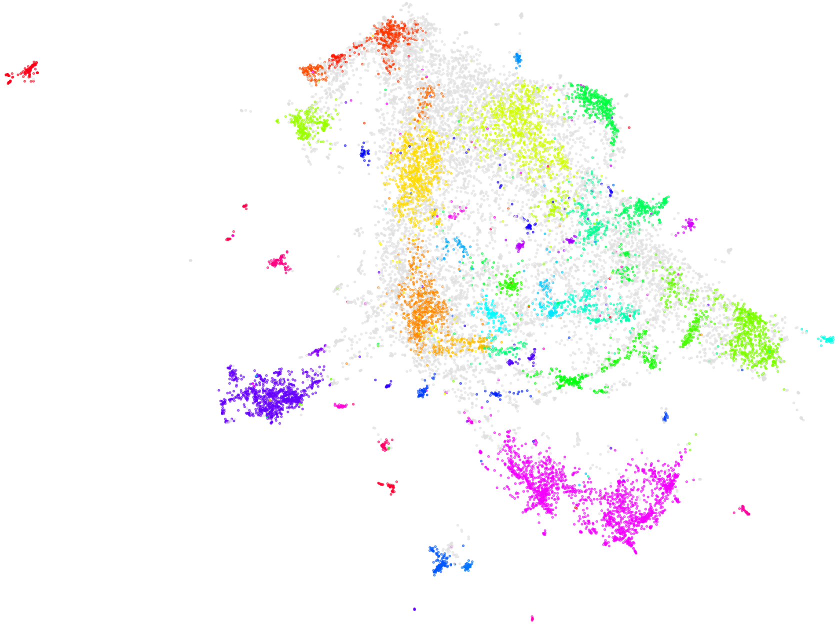


Figure 4.4: Documents embedding space visualization[19].

4.6 Experiments

The experiments are performed on an NVIDIA GTi GPU with 11 Gb ram with very good performance as it managed to perform the full experimental task in just six hours while on a simple personal computer with dual core CPU, it takes a week, so the difference is great. Below are the diagrams showing the clustering accuracy for each cluster algorithm for the number of model process propositions.

All the measurements agreed on the changes in accuracy as, in the beginning, the more the number of sentences fed by the model increased, the more the accuracy increased. The second observation is that the performance is constant when the number of sentences fed by the model is more than five, so it can be understood that this document-clustering task only needs the first five sentences and that, without the rest of the document, the performance remains constant and does not increase.

There is an iteration based on the number of sentences encoded in embeddings from the begining of the document. As we want to measure the effect of the portion of a document that should be considered for extracting topics.

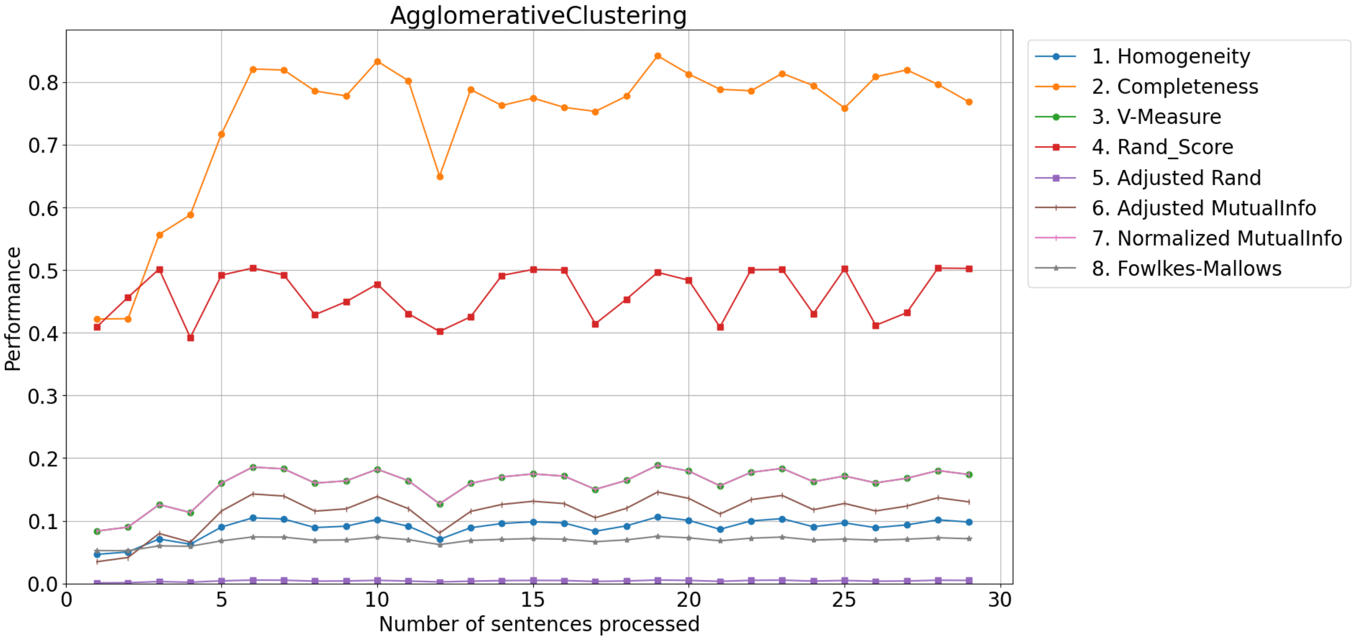


Figure 4.5: BERT embeddings with agglomerative clustering accuracy.

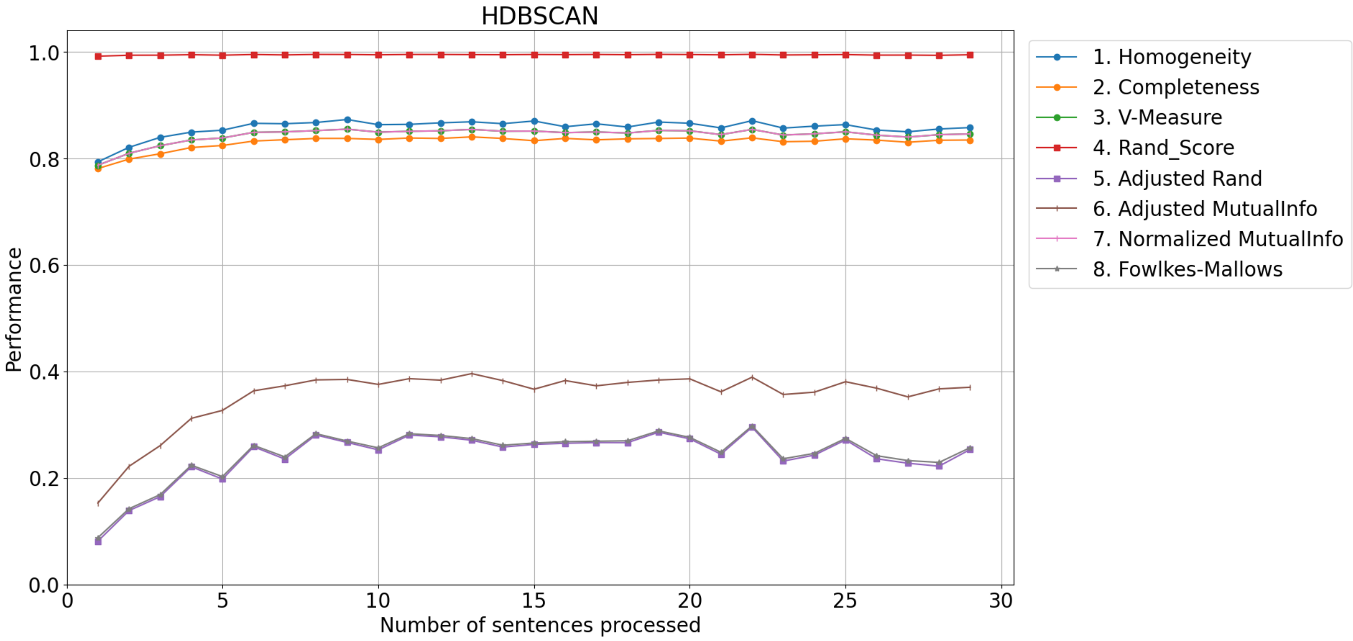


Figure 4.6: BERT embeddings with HDBSCAN clustering accuracy.

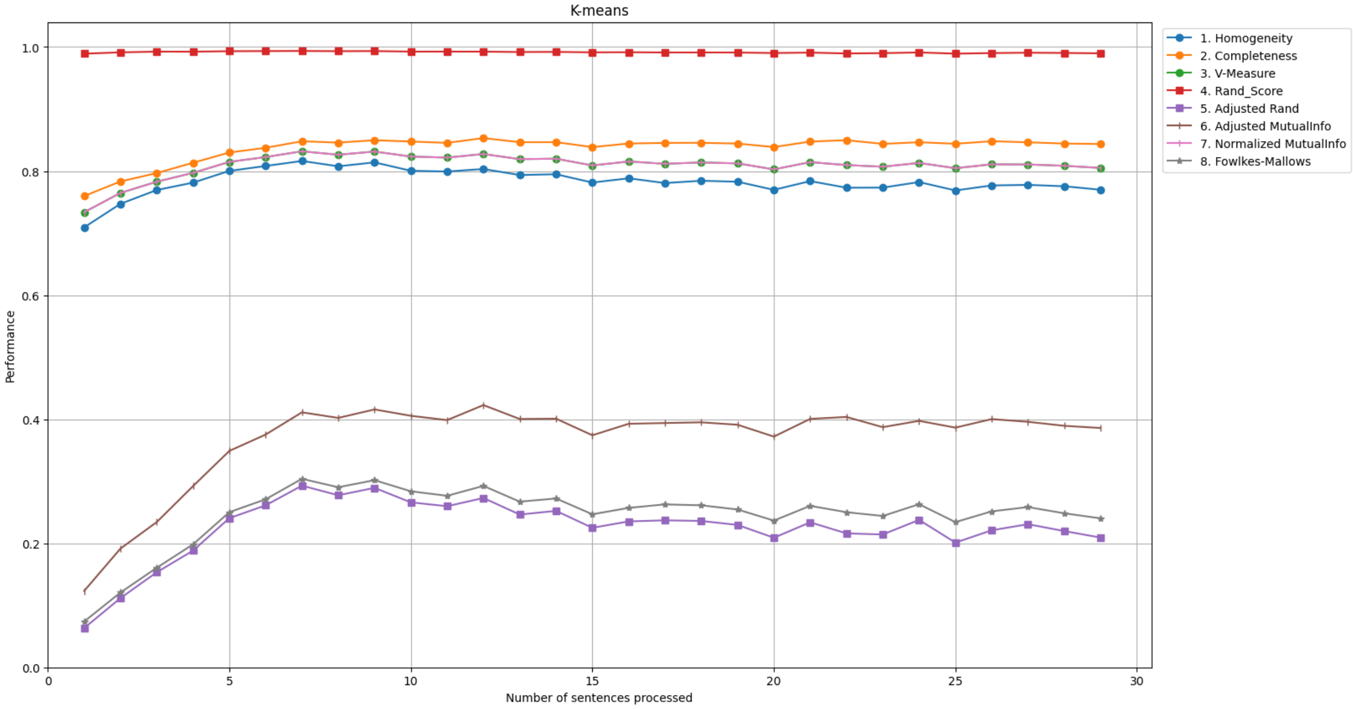


Figure 4.7: BERT embeddings with K-means clustering accuracy.

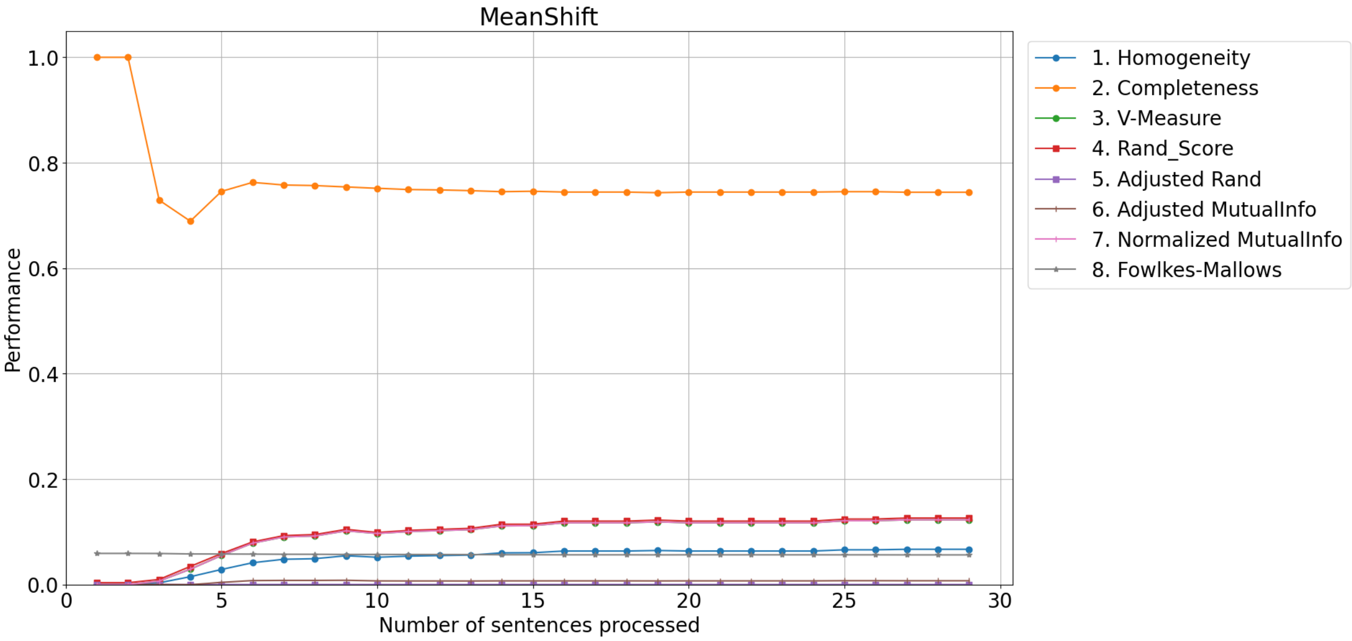


Figure 4.8: BERT embeddings with MeanShift clustering accuracy.

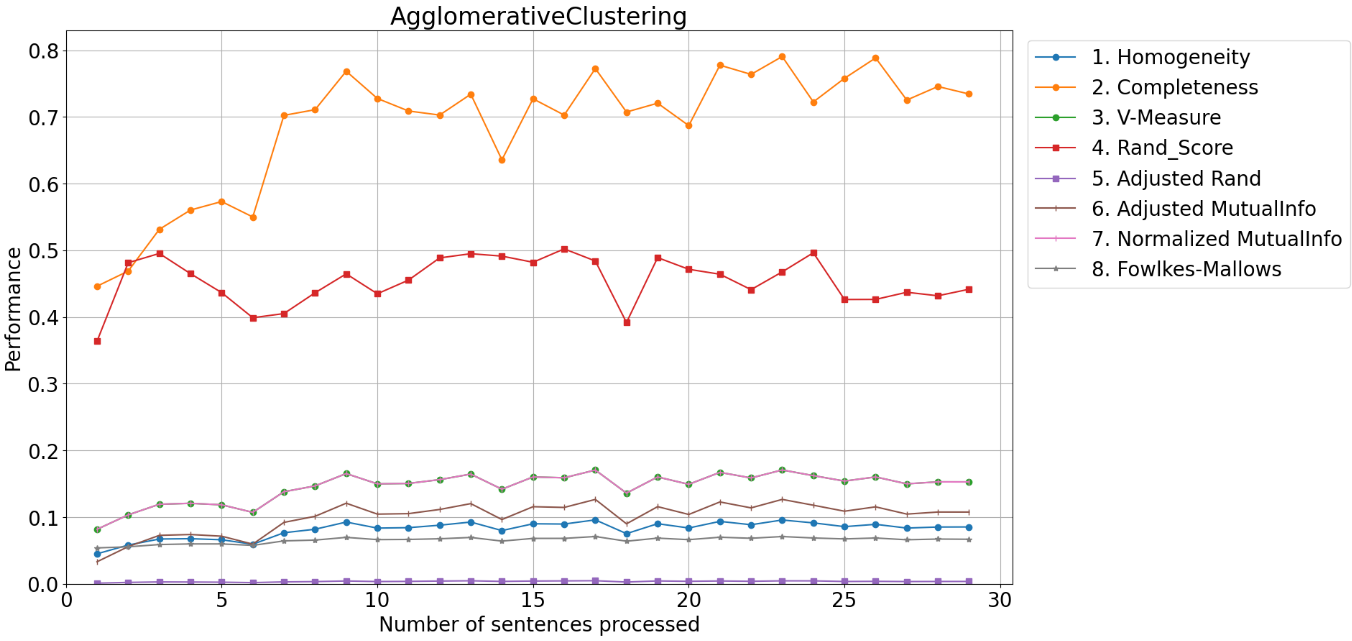


Figure 4.9: FASTTEXT embeddings with agglomerative clustering accuracy.

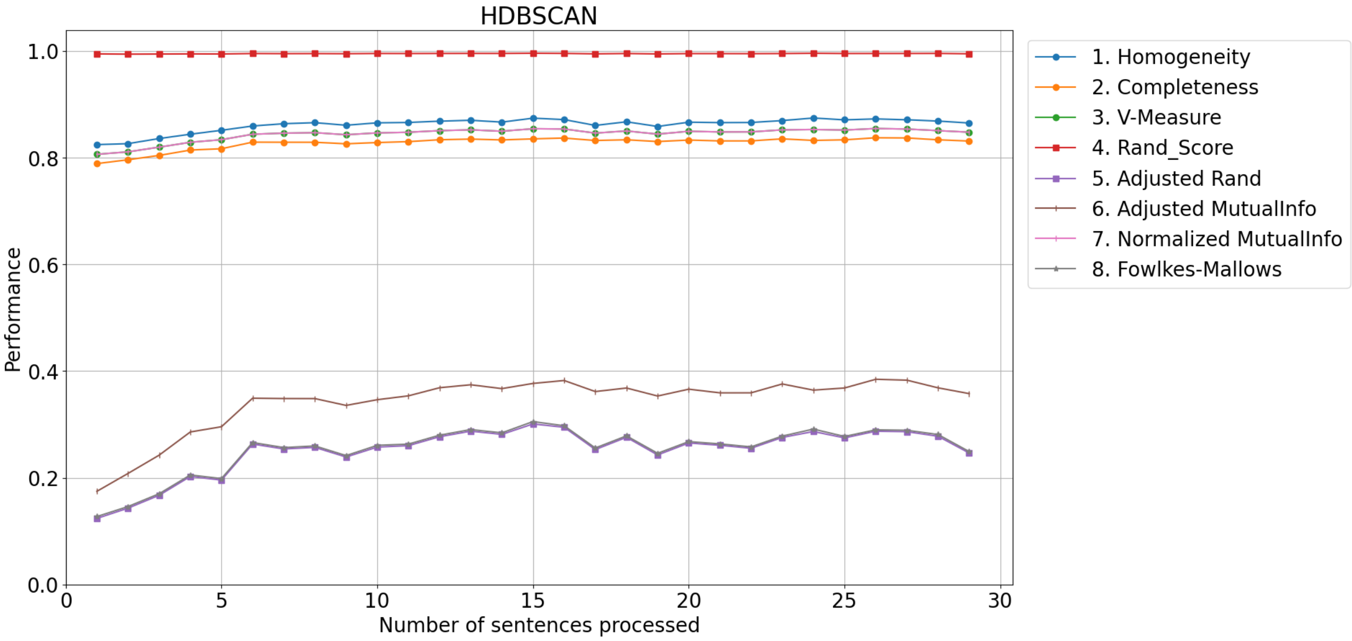


Figure 4.10: FASTTEXT embeddings with HDBSCAN clustering accuracy.

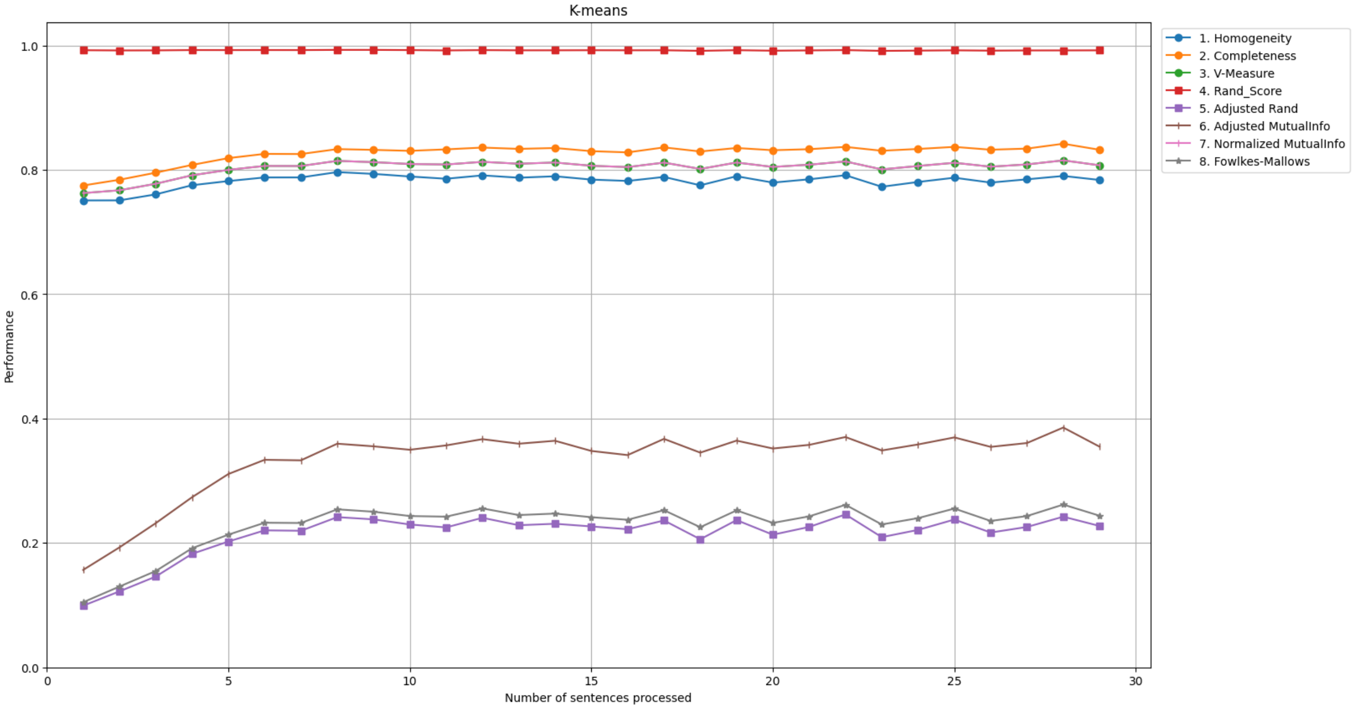


Figure 4.11: FASTTEXT embeddings with K-means clustering accuracy.

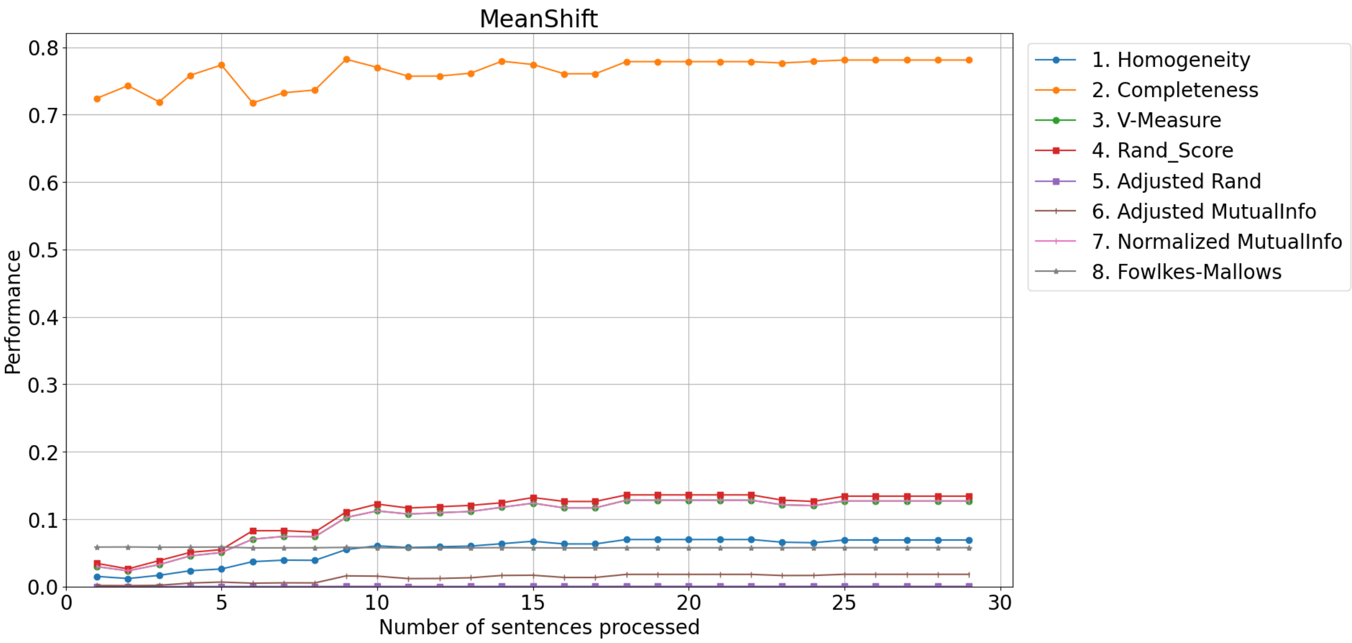


Figure 4.12: FASTTEXT embeddings with MeanShift clustering accuracy.

4.7 Results

The experiments performed focused on the comparison of the BERT and FASTTEXT models as well as the clustering algorithms. Each diagram shows the performance of the clustering in relation to the number of sentences that each time the models make embeddings for all the data-set and with respect to several evaluation metrics such as homogeneity, completeness, V-measure, Rand index, Adjusted Rand index, Adjusted Mutual Information, Normalized Mutual Information and Fowlkes-Mallows index .

For example, in the first iteration, the models receive as input only the title of each document within the corpus. In the second iteration the models receive the title and the first sentence of the document and so on. So in a total of thirty iterations the models run through the entire corpus and in each iteration the performance of the algorithm is measured in a set of four different clustering algorithms and a set of eight different metrics.

K-Means and HDBSCAN algorithms show very good results in general in the

Highest Performance clustering metrics based on BERT				
	K-Means	HDBSCAN	Mean Shift	Agglom.
Homogeneity	0.82	0.87	0.07	0.11
Completeness	0.85	0.84	**1.00**	**0.84**
V-measure	0.83	0.85	0.12	0.19
Rand-index	**0.99**	**0.99**	0.13	0.50
Adjusted-Rand-index	0.29	0.29	0.00	0.00
Adjusted-Mutual-Info	0.42	0.40	0.01	0.15
Normalized-Mutual-Info	0.83	0.85	0.12	0.20
Fowlkes-Mallows	0.30	0.30	0.06	0.08

Table 4.1: Highest Performance clustering metrics based on BERT

majority of metrics in both document representation models. According to the metrics, Mean-shift and Agglomerative clustering give lower performance results for both representation models. In general both models present similar performance over the clustering algorithms.

Highest Performance clustering metrics based on FastText				
	K-Means	HDBSCAN	Mean Shift	Agglom.
Homogeneity	0.80	0.87	0.07	0.10
Completeness	0.84	0.84	**0.78**	**0.79**
V-measure	0.82	0.85	0.13	0.17
Rand-index	**0.99**	**0.99**	0.14	0.50
Adjusted-Rand-index	0.25	0.30	0.00	0.00
Adjusted-Mutual-Info	0.39	0.38	0.02	0.13
Normalized-Mutual-Info	0.82	0.85	0.13	0.17
Fowlkes-Mallows	0.26	0.29	0.06	0.07

Table 4.2: Highest Performance clustering metrics based on FastText

According to the metrics, it is also noticed that first five sentences approximately can result in highest performance and there is no need to feed models with the entire document. So, it seems that each document contains the main topic at the first sentences including the title. It can be told that clustering algorithms get best results with first five approximately sentences.

Chapter 5

Conclusions and Future Work

5.1 Resume

The issue this thesis addresses is topic extraction from document in the Greek language and document clustering according to their content so that documents that refer to the same topic are in the same cluster. After researching related tasks, state of the art methods of topic modelling, extraction and document representation are explored and applied. BERT and FASTTEXT are among the state-of-the-art technologies used to export embeddings, making BERT give better results. Experiments show that BERT creates higher quality embeddings and therefore represents documents better than FASTTEXT. The optimal number of sentences in a document so that the clustering performance is the highest is five, taken from the beginning of the document. To evaluate the clustering performance, several clustering metrics are applied.

5.2 Conclusions

In this thesis, the use of document representation models are tried: BERT model and FAST-TEXT. Both by incorporating document embeddings into the clustering process extracting document relatedness information captured in word embeddings. The experimental comparison on custom document corpus which is created from scratch for this thesis shows that BERT improved clustering performance results but based on the metrics non context embedding methods like FastText can challenge context-based embeddings method like BERT. One conclusion that comes out after experiment is that news article documents usually mention the topic within the first five sentences as the rest of the text does not add any value to the news identification. In general document clustering task works in K-Means and HDBSCAN based on the metrics. K-means and HDBSCAN are more combatable on this task in comparison with Mean-shift and Agglomerative clustering and also, good quality corpus leads to better clustering results.

5.3 Future work

In the approach of this thesis, time did not exist as a variable in the representation of documents. From papers that were researched, it is tried to integrate time as an entity and tried to locate in time windows various news articles that could be linked to each other so that, in addition to the representation of the documents, criterion for the similarity between documents would be the similarity in the time of publication of an article. This is something that may be able to increase clustering performance and get better results if the time entity is integrated into the document embeddings process. It could be found out alternative document embeddings calculations with weights over the sentences of the documents. It would also be an asset, adding other NLP methods and combine for example word frequency with embeddings. One approach that also can increase performance would be choosing better quality document streams like posts from Twitter or enhancing text pre-processing in order to get more clear text or even normalizing embedding vectors through dimensionality reduction would probable help K-means to give better performance.

References

[1] Applying machine learning to classify an unsupervised text document. https://towardsdatascience.com/applying-machine-learning-to-classify-an-unsupervised-text-document-e7bb6265f52

[2] Digestion topic modeling visualization digestion topic modeling. https://towardsdatascience.com/understanding-nlp-and-topic-modeling-part-1-257c13e8217d.

[3] Getting to the point with topic modeling — part 1 - what is lda? https://community.alteryx.com/t5/Data-Science/Getting-to-the-Point-with-Topic-Modeling-Part-1-What-is-LDA/ba-p/611874.

[4] Gpt-2: Understanding language generation through visualization. https://towardsdatascience.com/openai-gpt-2-understanding-language-generation-through-visualization-8252f683b2

[5] The illustrated bert, elmo, and co. (how nlp cracked transfer learning). http://jalammar.github.io/illustrated-bert/.

[6] Long short-term memory. https://en.wikipedia.org/wiki/Long_short-term_memory.

[7] Reproducible and efficient benchmarks for hyperparameter optimization of neural machine translation systems. https://www.researchgate.net/figure/The-Transformer-model-architecture_fig1_323904682.

[8] Sklearn clustering metrics.

[9] Sklearn clustering metrics.

[10] Sklearn clustering metrics.

[11] Sklearn clustering metrics.

[12] Sklearn clustering metrics.

[13] Sklearn clustering metrics.

[14] Sklearn clustering metrics.

[15] Sklearn clustering metrics.

[16] Text mining of open-ended questions in self-assessment of university teachers: An lda topic modeling approach. https://www.researchgate.net/figure/Schematic-of-LDA-algorithm_fig1_339368709.

[17] Text prediction with tensorflow and long short-term memory—in six steps. https://www.altoros.com/blog/text-prediction-with-tensorflow-and-long-short-term-memory-in-six-steps/.

[18] Topic modeling. https://medium.com/technovators/topic-modeling-art-of-storytelling-in-nlp-4dc83e96a987.

[19] Unsupervised word embedding learning by incorporating local and global contexts. https://www.researchgate.net/figure/Word-and-document-embedding-visualization_fig4_339857996.

[20] basic architecture of rnn and lstm. https://pydeeplearning.weebly.com/blog/basic-architecture-of-rnn-and-lstm.

[21] ET-LDA: joint topic modeling for aligning, analyzing and sensemaking of public events and their twitter feeds. *CoRR*, abs/1210.2164, 2012. Withdrawn.

[22] Marcel R. Ackermann, Johannes Blömer, Daniel Kuntze, and Christian Sohler. Analysis of agglomerative clustering. *CoRR*, abs/1012.3697, 2010.

[23] Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[24] Ben Athiwaratkun, Andrew Wilson, and Anima Anandkumar. Probabilistic FastText for multi-sense word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1–11, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[25] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3(null):993–1022, March 2003.

[26] Miguel Á. Carreira-Perpiñán. A review of mean-shift algorithms for clustering. *CoRR*, abs/1503.00687, 2015.

[27] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

[28] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[29] Adji B. Dieng, Francisco J. R. Ruiz, and David M. Blei. Topic modeling in embedding spaces. *CoRR*, abs/1907.04907, 2019.

[30] Adji B. Dieng, Francisco J. R. Ruiz, and David M. Blei. Topic modeling in embedding spaces. *Transactions of the Association for Computational Linguistics*, 8:439–453, 2020.

[31] Miguel Domingo, Mercedes García-Martínez, Alexandre Helle, Francisco Casacuberta, and Manuel Herranz. How much does tokenization affect neural machine translation? *CoRR*, abs/1812.08621, 2018.

[32] Yao Fu, Yansong Feng, and John P. Cunningham. Paraphrase generation with latent bag of words. *CoRR*, abs/2001.01941, 2020.

[33] Xiangyu Guo, Janardhan Kulkarni, Shi Li, and Jiayi Xian. Consistent k-median: Simpler, better and robust. *CoRR*, abs/2008.06101, 2020.

[34] Thomas Hollis, Antoine Viscardi, and Seung Eun Yi. A comparison of lstms and attention mechanisms for forecasting financial time series. *CoRR*, abs/1812.07699, 2018.

[35] Yuhuang Hu, Adrian E. G. Huber, Jithendar Anumula, and Shih-Chii Liu. Overcoming the vanishing gradient problem in plain recurrent networks. *CoRR*, abs/1801.06105, 2018.

[36] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomás Mikolov. Bag of tricks for efficient text classification. *CoRR*, abs/1607.01759, 2016.

[37] Jiwei Li, Claire Cardie, and Sujian Li. TopicSpam: a topic-model based approach for spam detection. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 217–221, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[38] Zachary Chase Lipton. A critical review of recurrent neural networks for sequence learning. *CoRR*, abs/1506.00019, 2015.

[39] Claudia Malzer and Marcus Baum. Hdbscan(ϵ

): An alternative cluster extraction method for HDBSCAN. *CoRR*, abs/1911.02282, 2019.

[40] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013.

[41] Nicole Peinelt, Dong Nguyen, and Maria Liakata. tBERT: Topic models and BERT joining forces for semantic similarity detection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7047–7055, Online, July 2020. Association for Computational Linguistics.

[42] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[43] Nicola Pezzotti, Boudewijn P. F. Lelieveldt, Laurens van der Maaten, Thomas Höllt, Elmar Eisemann, and Anna Vilanova. Approximated and user steerable tsne for progressive visual analytics. *CoRR*, abs/1512.01655, 2015.

[44] Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. *CoRR*, abs/1509.00685, 2015.

[45] Tim Sainburg, Leland McInnes, and Timothy Q. Gentner. Parametric UMAP: learning embeddings with deep neural networks for representation and semi-supervised learning. *CoRR*, abs/2009.12981, 2020.

[46] Philipp Scharpf, Moritz Schubotz, Abdou Youssef, Felix Hamborg, Norman Meuschke, and Bela Gipp. Classification and clustering of arxiv documents, sections, and abstracts, comparing encodings of natural and mathematical language. *CoRR*, abs/2005.11021, 2020.

[47] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *CoRR*, abs/1404.7828, 2014.

[48] Robert Schwarzenberg, Lisa Raithel, and David Harbecke. Neural vector conceptualization for word vector space interpretation. *CoRR*, abs/1904.01500, 2019.

[49] Jonathon Shlens. A tutorial on principal component analysis. *CoRR*, abs/1404.1100, 2014.

[50] Ralf C. Staudemeyer and Eric Rothstein Morris. Understanding LSTM - a tutorial into long short-term memory recurrent neural networks. *CoRR*, abs/1909.09586, 2019.

[51] Sho Takase and Sosuke Kobayashi. All word embeddings from one embedding. *CoRR*, abs/2004.12073, 2020.

[52] Amirsina Torfi, Rouzbeh A. Shirvani, Yaser Keneshloo, Nader Tavaf, and Edward A. Fox. Natural language processing advancements by deep learning: A survey. *CoRR*, abs/2003.01200, 2020.

[53] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[54] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771, 2019.

[55] Wei Xie, Feida Zhu, Jing Jiang, Ee-Peng Lim, and Ke Wang. Topicsketch: Real-time bursty topic detection from twitter. *IEEE Transactions on Knowledge and Data Engineering*, 28(8):2216–2229, 2016.

[56] Jiangyan Yi, Jianhua Tao, Ye Bai, Zhengkun Tian, and Cunhang Fan. Adversarial transfer learning for punctuation restoration. *CoRR*, abs/2004.00248, 2020.

[57] Lorijn Zaadnoordijk, Tarek R. Besold, and Rhodri Cusack. The next big thing(s) in unsupervised machine learning: Five lessons from infant learning. *CoRR*, abs/2009.08497, 2020.

[58] Md. Zubair, MD. Asif Iqbal, Avijeet Shil, Enamul Haque, Mohammed Moshiul Hoque, and Iqbal H. Sarker. An efficient k-means clustering algorithm for analysing COVID-19. *CoRR*, abs/2101.03140, 2021.