Faculty of Media Engineering and Technology
German University in Cairo

Institute of Neural Information Processing
Faculty of Engineering, Computer Science and Psychology
Ulm University

Extractive Text Summarizer with Topic Identification

Bachelor Thesis

Author: Yomna Shaaban Mohamed Mahran
Supervisors: Prof.Dr.Friedhelm Schwenker
Submission Date: 1 October, 2021

Faculty of Media Engineering and Technology
German University in Cairo

Institute of Neural Information Processing
Faculty of Engineering, Computer Science and Psychology
Ulm University

Extractive Text Summarizer with Topic Identification

Bachelor Thesis

Author: Yomna Shaaban Mohamed Mahran
Supervisors: Prof.Dr.Friedhelm Schwenker
Submission Date: 1 October, 2021

This is to certify that:

(i) the thesis comprises only my original work toward the Bachelor Degree

(ii) due acknowlegement has been made in the text to all other material used

Yomna Shaaban Mohamed Mahran
1 October, 2021

Acknowledgments

First and foremost, I would like to praise and thank God, for all His granted blessings, countless opportunities and for giving me the strength to keep going. This thesis depends on the encouragement and support of many others. I would love to express my greatest appreciation to my supervisor Prof.Dr.Friedhelm Schwenker for his guidance and support throughout this bachelor project. It was a great privilege to study under his supervision. My deepest gratitude to my beloved parents and brothers for their belief in me and their unlimited love, patience and care. I hope I make them as proud as they make me. I also would like to thank my brother, Zeyad for having my back and for pushing me beyond my comfort zone. I am expanding my thanks to my bestfriend and sister Omnia for always showering me with love and kindness. My appreciations goes also to my precious friends Sandy, Monica, Sandra and Farah. I am grateful for the encouragement and motivation they gave me when I needed it. Last but least, I am thanking my dog Duke, for sharing with me the long nights.

Abstract

There is an exponential need for processing and analyzing the tremendous available amount of unstructured yet valuable text data. This thesis seeks to address that eternal need through implementing Extractive Text Summarization and Topic Modeling on long documents, hence, compressing the information required by many professionals to keep current with developments in their fields. So, the system takes as an input some text data, outputting a short summary for this data and a short list of tags including the most important words in it and identifying its general topic. For Text Summarization, the implementation passed through three main stages. Starting with data preprocessing, feature engineering and selection. Then finetuning our feature set by dimensionality reduction using UMAP and clustering using K-means. Finally, data is trained on a dataset of 10,000 records, summaries are predicted using three classifiers; Logistic Regression, Support Vector Machine (SVM) and Long Short Term Memory (LSTM) and results are evaluated using ROUGE-N and ROUGE-L measures. A strategy of choosing the first three sentences as a summary named Lead-3, is known to be exceedingly difficult to beat on standard metrics Recall-Oriented Understudy for Gisting Evaluation (ROUGE). In this spirit, we will use Lead-3 as the baseline for this approach. Moving to Topic Modeling, unsupervised learning is applied. Data is passed to BERTopic model after performing the needed data preprocessing steps. Best score was provided by LSTM model through a unidirectional network with 50 neurons.

Contents

Chapter 1

Introduction

1.1 Motivation

We are now living in a fast-evolving world where every second counts and saving time is worth million dollars. That's why any techniques that will help humans save time are fields of attention and topics of conversation. Text Summarization and Topic Modeling are two of those evolving fields to address the ever-growing amount of text data available online to both better help discover relevant information and to consume relevant information faster with such a big amount of data circulating in the digital space. Furthermore, applying those tasks reduces reading time, and increases the amount of information that can fit in an area.

Text summarization refers to the technique of shortening long pieces of text. The intention is to create a coherent and fluent summary having only the main points outlined. Since manual text summarization is time consuming and generally laborious task, automatization of tasks is gaining increasing popularity and therefore has drawn interest among the computer science researchers. There are important applications for text summarization in various Natural Language Processing (NLP) related tasks, such as medical cases, news summarization, headline generation, source code summarization, text classification, email summarization, question answering and chatbots. Text Summarization enables organizations and individuals to reuse already existing knowledge, which is stored and under-used in databases as unstructured data [7].

Researchers are in continuous investigation for summarization tools and methods that can automatically extract or abstract summaries from a range of information sources, yet automatic summarization is a very challenging task. Some of its challenges are the degree of redundancy, compression ratio, difficulty to generalize, different documents' length, sentence ordering and different genres included [4]. Another major challenge is the evaluation method; not due to the lack of methods, there are many available evaluators for evaluating automated summaries, however, it's not definite which sentences should be included in a summary, even by humans [7].

Artificial Intelligence (AI) uses numerous and powerful algorithms aiming to process language naturally, one of those is topic analysis. As the name suggests, topic modeling is a process of finding a group of words from a single document or multiple documents that best represents the associated information. However, extracting good quality topics that are meaningful and clear is the challenge here. This mainly depends on finding the optimal number of topics and having impactful preprocessing steps [32]. Definitely, topic modeling is much simpler and quicker than processing the entire document/documents and that's how it helps visualize things better and solves problems.

1.2 Objective

This thesis aims to study, observe and implement two of the current most popular NLP tasks. First one is, automated extractive text summarization on long texts. Secondly, topic modeling on data covering a variety of topics. The aim of this study is to make unused complex texts more readable and beneficial by using it to extract a short summary and predict a list of the most important words in this text. This project explores numerous concepts like different classifiers, sentence embeddings, dimension reduction and clustering techniques.

1.3 Thesis Outline

The thesis consists of five main chapters. Chapter 2 discusses important theoretical information about natural language processing, text summarization, topic modeling, machine learning, deep learning, sentence embedding, clustering, clustering validation methods, dimensionality reduction, classification models, similarity measures and evaluation metrics. In addition, it highlights some of the previous research that has been conducted in the field. After that, the methodology chapter elaborates and explores the dataset, and illustrates the workflow of the execution and the implementation. Next, the results chapter mentions the results obtained after applying multiple machine learning models, along with the challenges faced during the implementation. Finally, the last chapter discusses the conclusion and future work, sums up the work done throughout the thesis and gives an overview of the results and offers some recommendations for how they can be improved in the future.

Chapter 2

Background

2.1 Natural Language Processing

NLP refers to the branch of artificial intelligence that fills the gap between human communication and computer understanding. NLP tasks help computers understand, interpret and manipulate human language which draws many disciplines, including computer science and computational linguistics. NLP technologies give computers the ability to understand both spoken and written text in much the same way human beings can [26].

Despite years of research, more advanced AI and huge benefits, NLP still has its challenges, limitations and needs improvement consistently. One of these problems is the challenge of pragmatics, making it harder to be grasped by machines; a single phrase might have different intentions like informing, reminding, misleading, drawing attention, giving commands, etc. Another challenge in NLP is ambiguity; a single unique word can have a different meaning depending on the context it's in. Also, synonyms are an obstacle; because we can express the same idea by using many different words and terms. And the list of challenges still has a lot more including irony and sarcasm, colloquialisms and slang, errors in text and low-resource languages [26]. NLP has its barriers, however, it still offers wide-ranging and huge benefits to a lot of businesses. And with our vast evolving world introducing new technologies everyday, many of these limitations will fade away in the next few years. There are plenty of tasks fulfilled with the help of NLP like machine translation, topic modeling, information retrieval, question answering, text summarization and sentiment analysis. In this thesis, we will tackle text summarization and topic modeling.

2.2 Text Summarization

Automatic Text Summarization is one of the most challenging and interesting applications of NLP. It is a process of generating a meaningful and concise summary of long

pieces of text because who has the time to go through books or entire articles to decide whether they are useful or not.

There are alot of techniques when it comes to text summarization. As we can see in figure 2.1 text summarization can be classified based on input type, the purpose and output type. Input type can be categorized as a single document or multiple documents. Multi-document summarization can be considered as an extension of single document summarization. In multi-document summarization, search space is larger compared to single document summarization, which makes the output longer and the process more complex and challenging. Moving to the purpose, text summarization can aim for a generic, domain-specific or query-based summary. A generic summary gives the overall content of the inputted document/documents, while a domain aware summarizer is designed for a specific, restricted , given domain which indicates the genre of the inputted document/documents. On the other side, a query based summary is presented in a predefined and minimal number of words. Given some queries, the summerizar presents essential summarized information from the inputted document/documents that is most relevant and best covers the given queries. Last text summarization classification technique and the most popular one is the output type. The output type has two categories: extractive summarization and abstractive summarization [5].

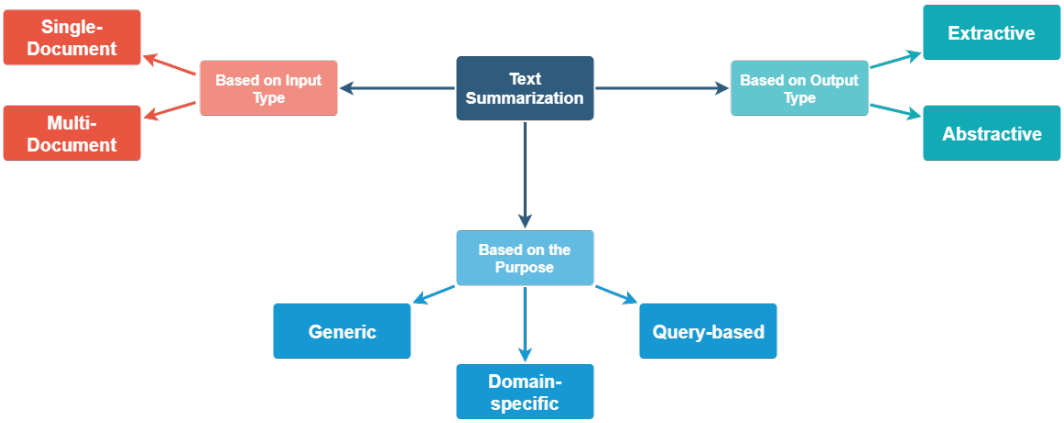

Figure 2.1: Text Summarization classification types. Source:[5]

2.2.1 Abstractive Summarization

Abstractive text summarization functions by understanding the original text and generating totally new phrases and sentences that rely on the most essential information from the original text, which is what humans will do to write a summary of a given text as shown in figure 2.2. Abstractive approaches result in more sophisticated summaries and can overcome grammar inconsistencies. On the other hand, abstraction needs various advanced algorithms to generate meaningful sentences, so it's more challenging to develop [2]. Research into abstractive implementation and true semantic understanding

of human language in general is a worthy pursuit and should not be ignored, however, much work is needed before we can confidently say that we have gained a true foothold in this endeavor. Consequently, in this thesis we will focus on extractive summarization methods.

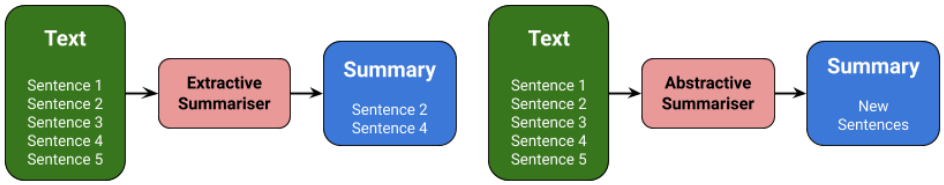

Figure 2.2: Abstractive vs Extractive Summarization. Source: [6]

2.2.2 Extractive Summarization

Extractive text summarization functions by extracting the important sentences and key-phrases from the source document/documents and combining those phrases to form a summary, no new text is formed as illustrated in Figure 2.2. Extractive-based summarization works as a highlighter and the generated summary is the highlighted parts. Extractive approaches are restricted to extracting, resulting in summaries that might be incoherent, however, extractive summarization is more popular as it will adapt better to larger sources and produce unbiased summaries [7].

Extractive Summarization can be tackled for a single document or multiple documents and can be accomplished by using supervised methods or unsupervised methods. Supervised algorithms depend on a set of training data and a model. The training data capture the importance of sentences by designing some features, so that the model can learn from our data to predict how important a sentence is and form a summary with the essential sentences. On the other hand, unsupervised methods assign scores using some algorithm to each word in a sentence and accordingly assign a score to each sentence in the text [50]. Extractive Summarizers usually have three missions to accomplish:

- Content selection, which covers the process of extracting the important sentences and phrases to form a summary. This phase is the most critical one. Then, the top-ranked sentences make it to the summary without using any training data.
- Information ordering which is simply ordering the outputted sentences from the first step.
- Sentence realization which is the process of forming a coherent summary by doing some sort of cleaning up on the extracted summary.

In the following chapters we will tackle Extractive Summarization with greater depth.

2.3 Topic Modeling

Topic Modeling is an unsupervised machine learning technique, capable of observing patterns and finding the most valuable words (called "topics") in large clusters of texts. The process usually passes by two main missions [32]:

- Clustering the documents, so that each cluster will contain similar topics documents together.
- Deriving topics using the clusters generated in step one, based on their content. This can be tackled by different term frequency algorithms (we will discuss it in the following sections).

Finally, a good topic model should result in – "band", "songs", "jazz", "tunes" for a topic – Music, and "chef", "sauce", "cooking", "restaurant" for a topic – "Food". Topic modeling can be used in organizing customer reviews, emails, etc. Consequently, businesses can save their employees' time to spend it on more important tasks by offloading tasks onto machines instead of overloading employees with too much data [32].

2.4 Machine Learning

Machine Learning (ML) is an application of AI based on the idea that systems can learn from data, discover patterns, evaluate data, and make better predictions with no or minimal human interaction [49]. The use of ML is growing significantly everyday because of the enormous quantity of available data and the growth of more powerful processing powers. ML has been beneficial to almost every real-world domain, including healthcare [45], financial services, autonomous vehicles (AV), transportation, gaming, climate modeling, speech, and image processing using different types of models such as regression analysis, support vector machines, Bayesian networks and many more. There are four types of ML algorithms: supervised, semi-supervised, unsupervised and reinforcement.

2.4.1 Supervised Learning

Supervised learning is a type of machine learning where the model is provided with labeled training data to classify or predict outcome accurately. As input, you feed the features and their corresponding labels into an algorithm in a process called training. During training, the algorithm gradually determines the relationship between features and their corresponding labels. This relationship is called the model [12]. Talking data mining, classification and regression are 2 types of supervised learning problems [28].

2.4.2 Unsupervised Learning

Unsupervised learning is a type of machine learning where the model is provided with unlabeled training data to identify meaningful patterns in the data and cluster unlabeled datasets. Some of the most common real-world applications of unsupervised learning are computer vision and recommendation systems. Dimensionality reduction and clustering are two of the most popular unsupervised learning approaches [35].

2.5 Deep Learning

Nowadays, deep learning drives our everyday AI services and applications. Deep learning is a broad sub-field of ML concerned with building models inspired by the structure and function of the brain's neural networks. Deep Learning models are composed of a neural network with more than two layers which learn from data and simulate human brain's behaviour. Those additional layers will fine tune and optimize the performance [14]. Furthermore, deep learning can automate feature extraction and process unstructured data, saving the data pre-processing effort that humans will do in case of machine learning as shown in figure 2.3.

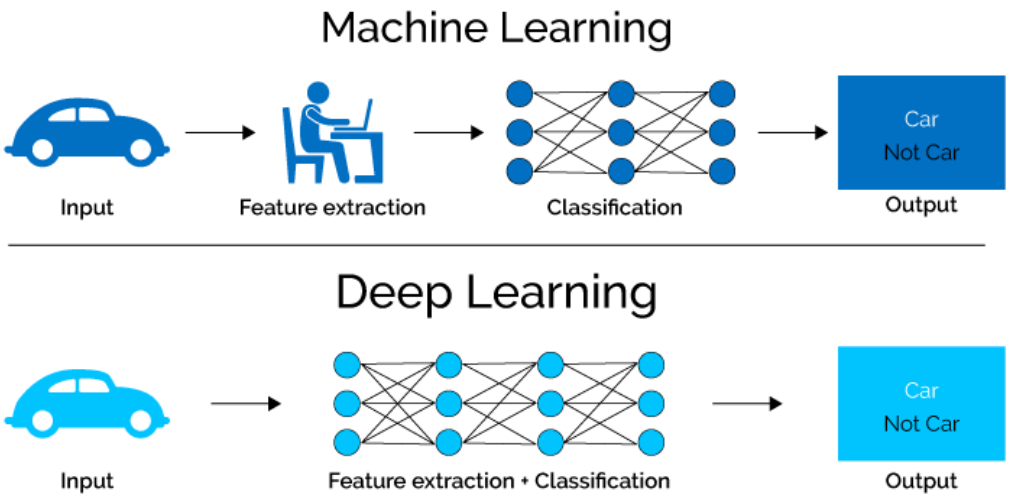

Figure 2.3: Machine Learning vs Deep Learning. Source: [3]

2.6 Text Embedding Techniques

Unlike human brains, machines can not understand sarcasm, negative sentiment, etc. That's why, converting the text to a machine understandable language is crucial and

as known, machines only understand numbers regardless of what datatype is your data. Embedding techniques represent text as numbers, so that the machine can process and understand different kinds of text [31].

2.6.1 Bert Word Embedding

Bidirectional Encoder Representations from Transformers (BERT) presented by Google [42] in 2018, had achieved state-of-the-art results in many NLP tasks. BERT's secret technical renovation key is applying the bidirectional training of Transformer to language modeling. A transformer is an encoder-decoder architecture model that adopts the mechanism of attention, forwarding to the decoder an idea of all the sequence at once instead of sequentially [55] as shown figure 2.4.

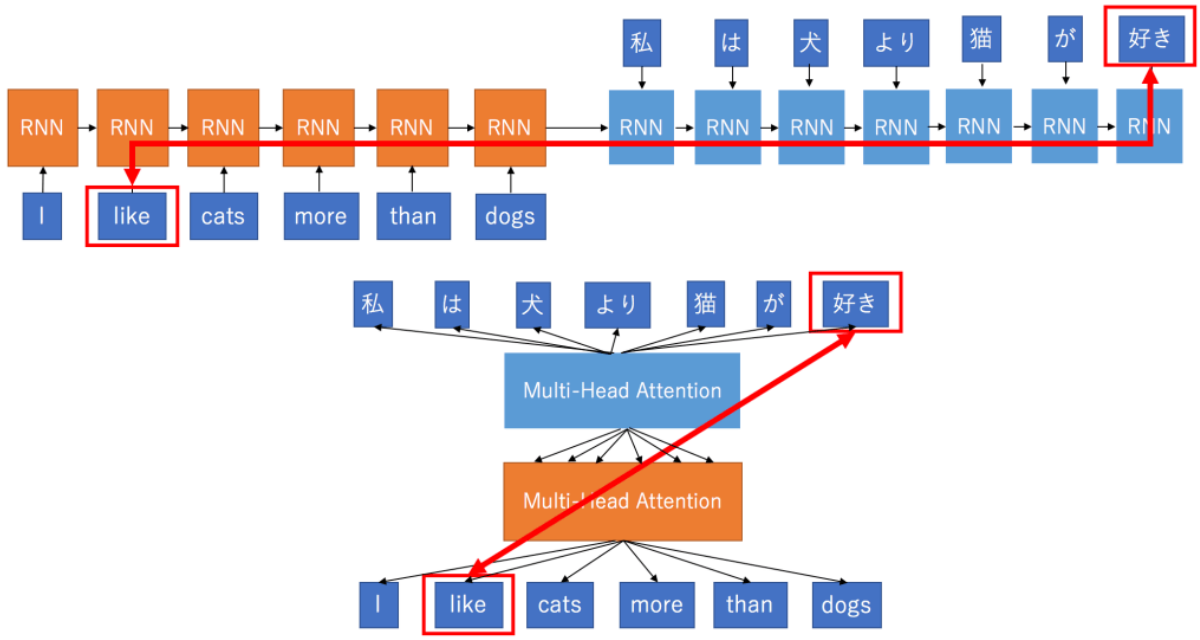

Figure 2.4: Transformer's encoder-decoder architecture. Source: [55]

After that, BERT model was introduced using encoders rather than decoders to build a bidirectional transformer-based language model. The main idea google's team worked on was to hide 15% of the words and infer them by using their position information [27]. The BERT architecture has two available variants:

- **The Base** with 12 transformer blocks (L), hidden layer of size 768 (H) and 12 attention heads (A).
- **The Large** with 24 transformer blocks (L), hidden layer of size 1024 (H) and 16 attention heads (A) [27].

Figure 2.5 shows a high-level view of BERT models, but of course each encoder is much more complex from the inside.

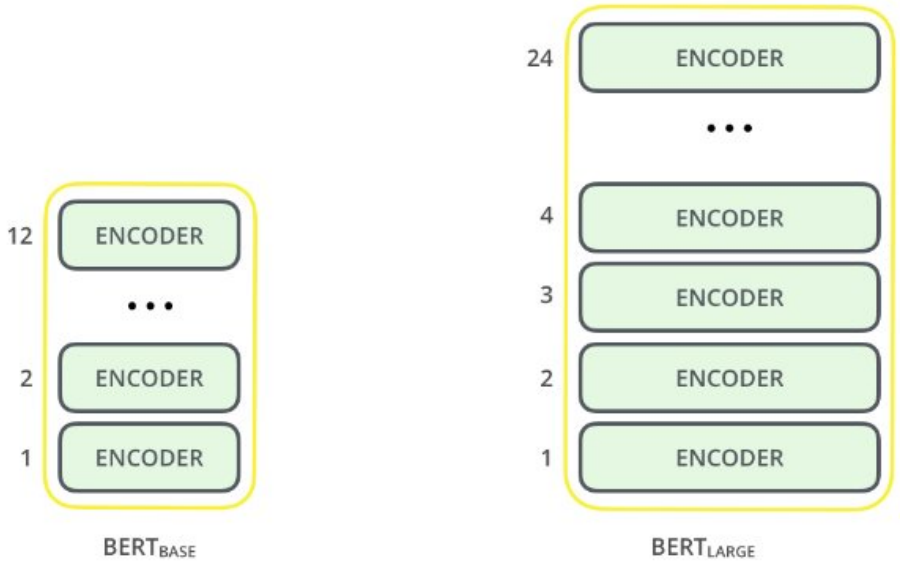

Figure 2.5: BERT Word Embedding two variants, the Base and the Large. Source: [55]

As for the input text for the model, it's the combination of three embeddings which are position embeddings, segment embeddings and token embeddings, as illustrated in figure 2.6. Position embeddings represent the position of each word in a sentence. BERT can take as an input two sentences, so segment embeddings are used to differentiate between sentence A and sentence B. For example in figure 2.6, tokens marked as EA belong to sentence A and the same for EB. Lastly from the WordPiece token vocabulary, token embeddings are learned for each specific token [15]. Now we are expecting the word embedding as an output. Actually, each output from each of the 12 layers (in case of BERT base model) can be used as a word embedding (depending on what's best for the task) or the summation of the last four layers can be used.

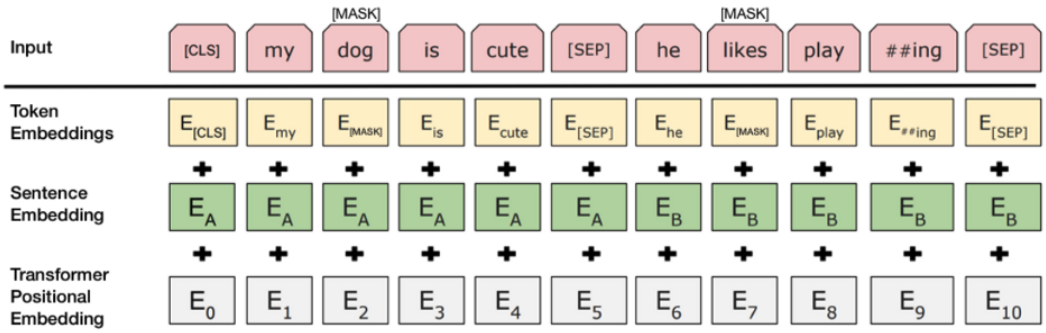

Figure 2.6: BERT Word Embedding general Architecture. Source: [55]

2.6.2 Bert Sentence Embedding

Initially, embedding techniques dealt with embedding words, generating an embedding for each word in a text. However, sometimes (like in the case of text summarization) we need to encode the meaning of the whole sentence to understand the context. That's why, sentence embedding techniques were proposed to represent sentence s' semantics as vectors.

Using BERT would be terribly expensive on both time and power levels, if we want to use BERT for similarity related tasks. That's why BERT's construction is not convenient to be used in tasks like clustering. So, we will use Bert Sentence Embedding (S-BERT) instead, which is a modified version of the pre-trained BERT network deriving semantically meaningful and fixed-sized sentence embeddings that can be computed using cosine similarity with a much lower cost [53]. S-BERT architecture from a classification objective and a regression objective is shown figure 2.7[53].

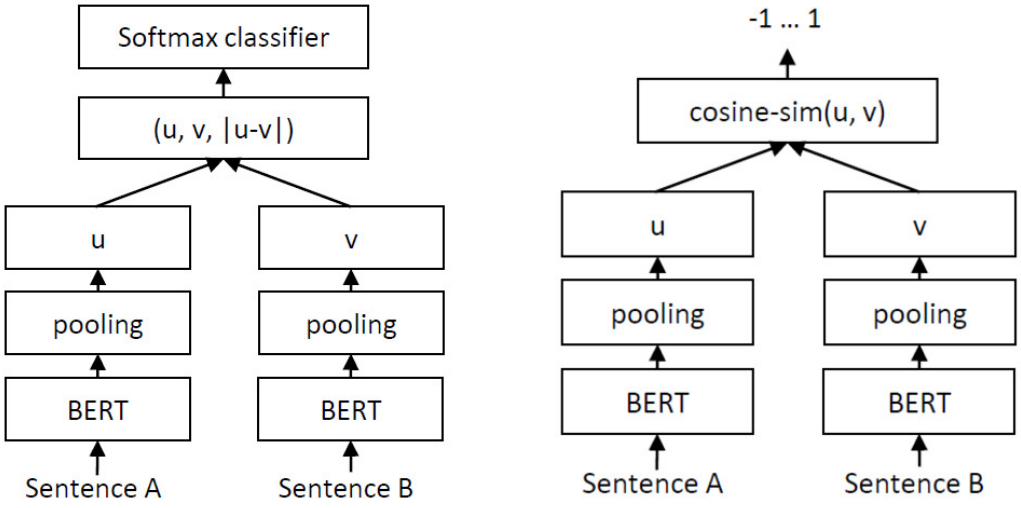

Figure 2.7: S-BERT Architecture. Source:[53]

2.7 Dimensionality Reduction

The reduction of features and input variables in a dataset is referred to as dimensionality reduction. The higher the number of input features, the more complex and challenging a predictive task is (curse of dimensionality) [8]. There are two main reasons why dimensionality reduction is utilized. The primary goal is to visualize data efficiently. Dimensionality reduction eases understanding the training set and subsequently work on it. The second goal is to help better fit a predictive model by simplifying a classification or regression dataset [8]. This is because dimensionality reduction helps in avoiding

overfitting as a model that is trained on a large number of features, becomes extremely dependent on the training data causing overfitting and inefficiency when trained on real data.

2.7.1 Uniform Manifold Approximation and Projection

Feature selection methods and feature engineering methods are used to apply dimensionality reduction. Feature selection is the method of analyzing and selecting relevant features for your sample and it can be done manually or programmatically. However, feature engineering is manually generating new features from existing features [8]. Now let us look at a programmatic approach for feature selection which is UMAP. UMAP is potentially one of the best approaches that creates meaningful features/variables and keeps a significant portion of the high-dimensional local structure in lower dimensionality [48]. Firstly, UMAP creates a weighted high dimensional graph representation (weights representing the likelihood that two points are connected). Then used it to optimize a low-dimensional graph to be as structurally similar as possible [48].

2.8 Clustering Models

Clustering, also known as the partitioning process, is an unsupervised machine learning approach. As the name suggests, clustering is the task of dividing data points into different clusters where the most similar points are grouped together [11]. There are various algorithms to apply clustering, two of the most effective approaches are K-means clustering and Hierarchical Density Based Clustering (HDBSAN) clustering.

2.8.1 K-means Clustering

K-means is one of the most popular unsupervised learning algorithms to solve the clustering problem. K-means partitions the data points into K different non-overlapping clusters through an iterative algorithm with the goal of keeping each clusters' data points as similar as possible and keeping each cluster as different from other clusters as possible [22]. The iterative algorithm goes through mainly three steps:

- Choosing the most optimal number of clusters K for the available dataset (we will dig deeper on how to achieve this step in the next part).
- Setting initial centroids by randomly assigning each datapoint to one cluster and taking the average of all cluster's data points to compute the centroid of each subgroup.

- Re-assign each data point to the closest cluster centroid and re-compute new cluster centroids. Finally, keep on repeating this step until no data points are changing their cluster group for two successive repeats, this means we have reached a global optima [19].

The k-means algorithm is shown clearly in figure 2.8.

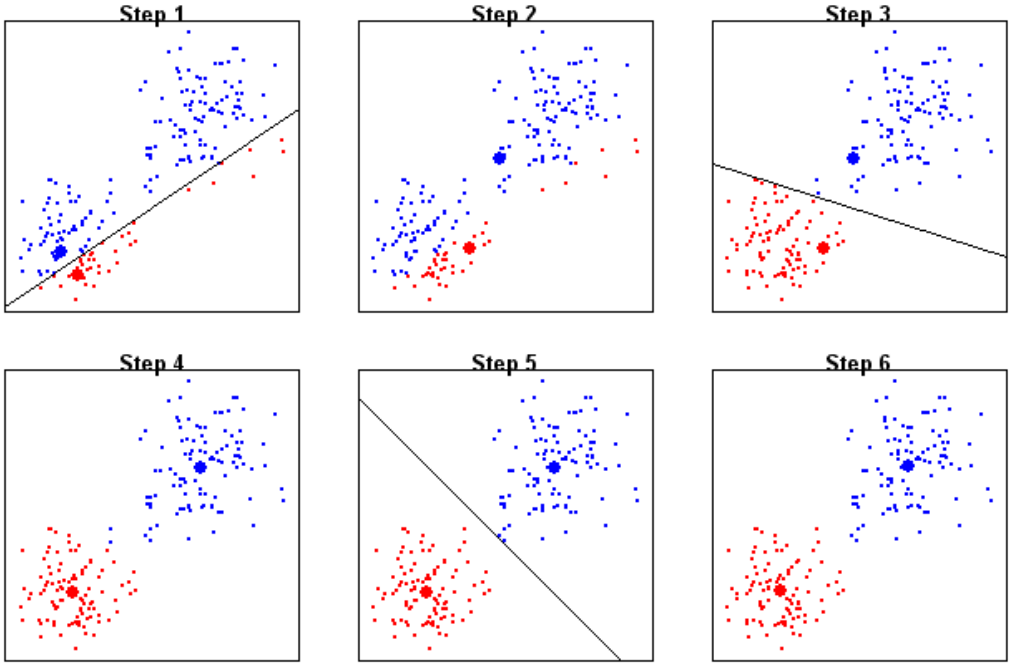

Figure 2.8: The k-means algorithm steps. Source:[20]

2.8.2 Hierarchical Density Based Clustering

Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSAN) is a clustering density-based algorithm [41]. Unlike many other clustering algorithms, HDBSAN can adapt and give good results even when the data is not clean and the clusters are weirdly shaped. The algorithm goes through two main steps. As the name suggests, the approach is density based, so, the first step would be to estimate the densities. One option to accomplish that would be to calculate the distance between a point and its K-th nearest neighbor, known as "core distance". As shown in figure 2.9, points in sparser regions would have a large core distance, while points in denser regions would have smaller ones [17]. Then, we can estimate the density by getting the calculated core distances' inverse. Second step is to simply select the clusters. This can be implemented by picking a global threshold for the calculated densities and group points accordingly. As illustrated in figure 2.10 which is densities' plotting, selecting the optimal threshold is a critical step. Based on that, we will determine the number of clusters (two clusters

on the left graph but three clusters on the right one) in figure 2.10. If the threshold is too low, we will overgroup points, if it's too high, we will under group the points [17].

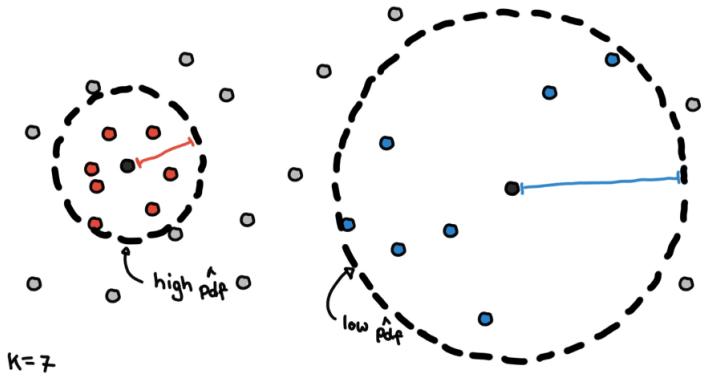

Figure 2.9: Calculating core distance in HDBSAN. Source:[17]

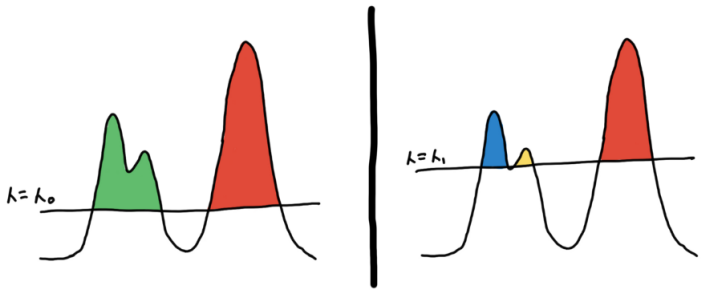

Figure 2.10: Picking a global threshold in HDBSAN. Source:[17]

2.9 Clustering Validation methods

Cluster Validation methods are also referred to as Evaluation Methods, used to evaluate how well the models are performing based on different K clusters since clusters are used in the downstream modeling [22]. The metrics used in this study are elbow method and silhouette analysis.

2.9.1 Elbow Method

Based on the Sum of Squared Distance (SSE) between data points and their assigned clusters' centroids, the elbow method gives us an estimate of what a suitable k number of clusters might be [22]. We pick k at the spot where SSE starts to flatten out and form an elbow as shown in figure 2.11. Because the curve is monotonically declining and may not exhibit any elbow or have an evident point where the curve starts flattening out, it might be difficult to determine an appropriate number of clusters to use.

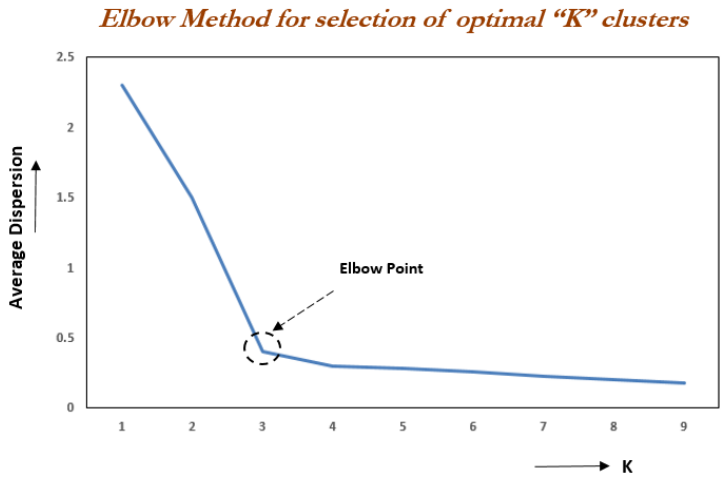

Figure 2.11: Elbow Method for selection of optimal K. Source:[16]

2.9.2 Silhouette Analysis

The degree of separation between clusters may be determined via silhouette analysis [22]. Each sample :

- Calculate
$$a^i$$
Which is the average distance from all data points in the same cluster.
- Calculate
$$b^i$$
Which is the average distance from all data points in the closest cluster.
- Compute the coefficient:
$$\frac{a^i - b^i}{max(a^i, b^i)}$$
Coefficients can take values ranging from -1,1. The bigger the coefficients and the closer to 1 the better the clusters.

2.10 Classification Models

In ML, classification is the task of approximating the mapping function from input variables to discrete output variables. Those output classes are referred to as targets or labels. The primary goal is to predict which class the new data instances will fall into [36]. If the number of outputted classes is two, then that's a binary classification. For example, classifying emails as spam or not spam. In this study, we will be working on a binary classification problem using the following classification models:

- Logistic Regression
- Support Vector Machine (SVM)
- Long term short Memory (LSTM)

2.10.1 Logistic Regression

Logistic Regression models are largely employed in Statistics and have demonstrated success in several real-world problems. Also, Logistic Regression is the go-to method for binary classification problems. It is a probability based predictive algorithm. 'Sigmoid function' or also known as the 'Logistic function' is the cost function used by logistic regression and what makes it more complex than linear regression. The Sigmoid function is limited to output values between 0 and 1 [21].

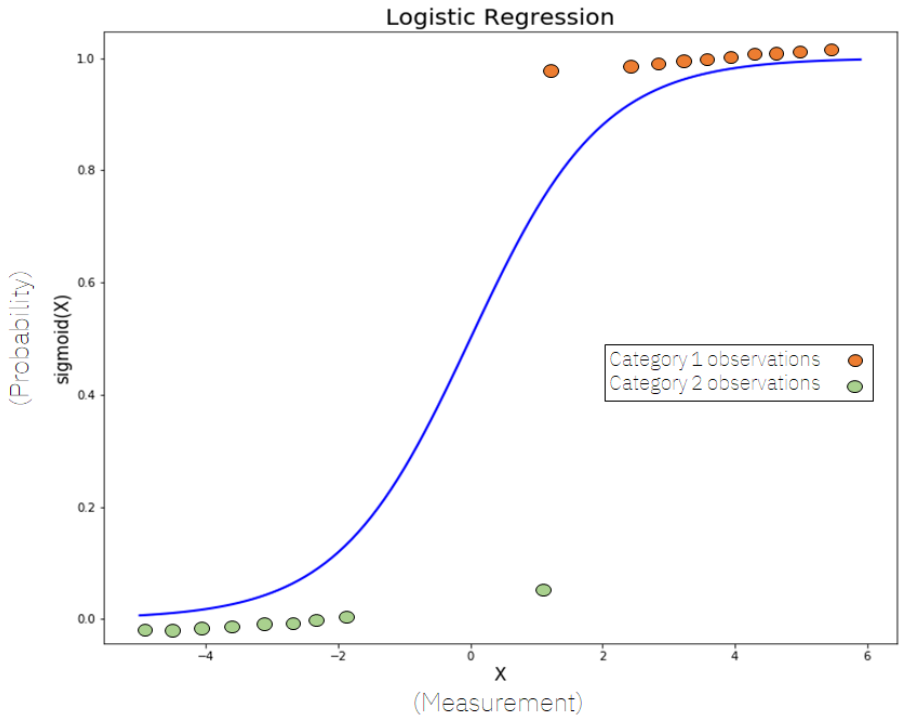

Figure 2.12: Sigmoid function used in Logistic Regression. Source:[23]

The formula for the Sigmoid function is:

$$h(x) = \frac{1}{1 + e^{-(a+bx)}}$$

Where h(x) is the probability of a 1, e is the base of the natural logarithms, while a and b are the parameters of the model.

So, it maps any real value into another value between 0 and 1; mapping predictions to probabilities. After that, according to a set threshold we start the classification phase. For example if the determined threshold is 0.6, all data points greater than 0.6 will belong to category one and all data points smaller than 0.6 will belong to category two as shown in figure 2.12 [21].

2.10.2 Support Vector Machine

Support Vector Machine (SVM) is another approach to solve classification problems. However, unlike the logistic regression they are not probabilistic. This means that they assign data points to a certain class or category without providing a probability for each class or category [24]. Firstly, we plot the data points in an N-dimensional space where N is the number of data's features. The algorithm works by finding the best hyperplane, also known as the decision boundary which separates the data into distinct classes. The best decision boundary is the one that has the maximum margin, a margin is the distance between the hyperplane and the support vectors as shown in figure 2.13. A good decision

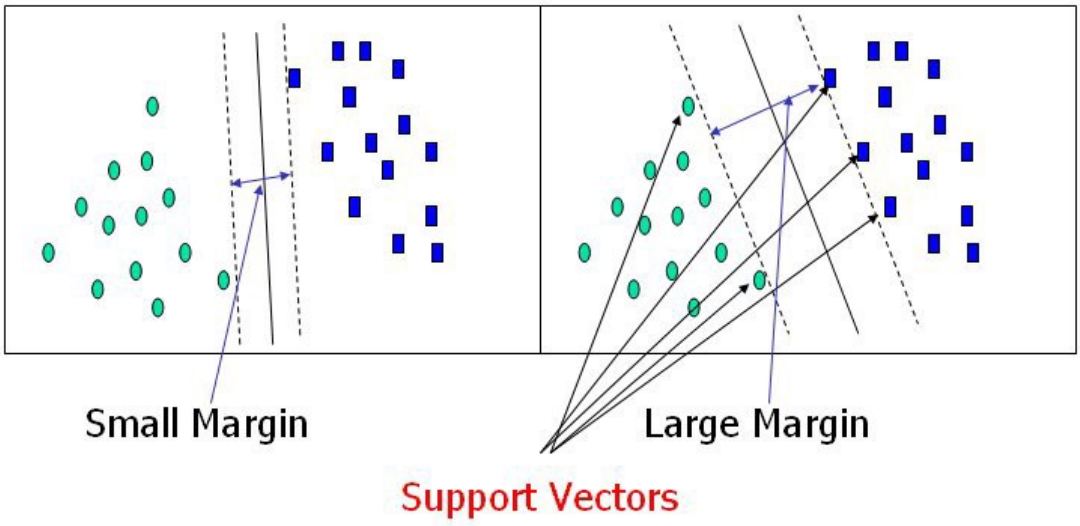

Figure 2.13: Support Vector Machine algorithm. Source:[24]

boundary will better generalize on new data instances. One of the key advantages of SVMs is that they stand out for their robustness to high dimensional data [29]. The formula of SVM is given by:

$$\left[\frac{1}{n}\sum_{i=1}^{n} max(0, 1 - y_i(w.x_i - b))\right] + \lambda||w||^2$$

2.10.3 Long Short Term Memory

One of ML models that successfully deal with sequential data are Recurrent Neural Networks (RNN). RNN are popular for solving text sequential problems where the output from the previous step is fed as input to the current step. If you think about it, a paragraph is a sequence of sentences and a sentence is a sequence of words. Figure 2.14 shows the RNN's architecture [25].

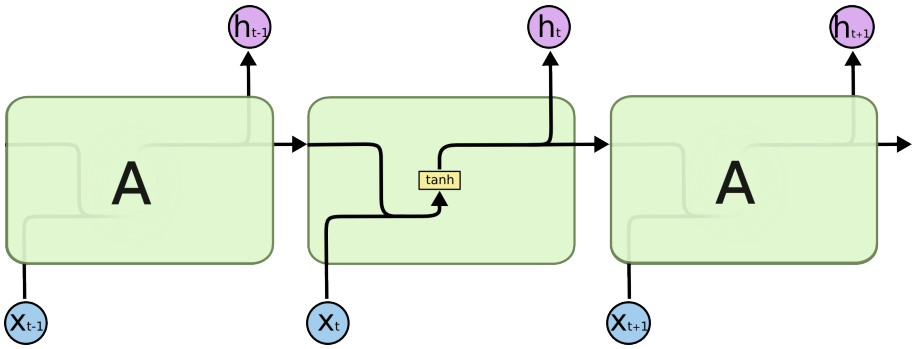

Figure 2.14: The repeating module in a standard RNN contains a single layer. Source:[34]

RNN works well for short sentences, however dealing with a long article, will cause a long term dependency problem. Long Short Term Memory (LSTM) is a special kind of RNN, capable of learning long-term dependencies. LSTM is used for classification of time-series data in the field of deep learning [25]. In this study, we have a 'many to one relationship' classification problem as we input a set of features and output a single label as illustrated in figure 2.15.

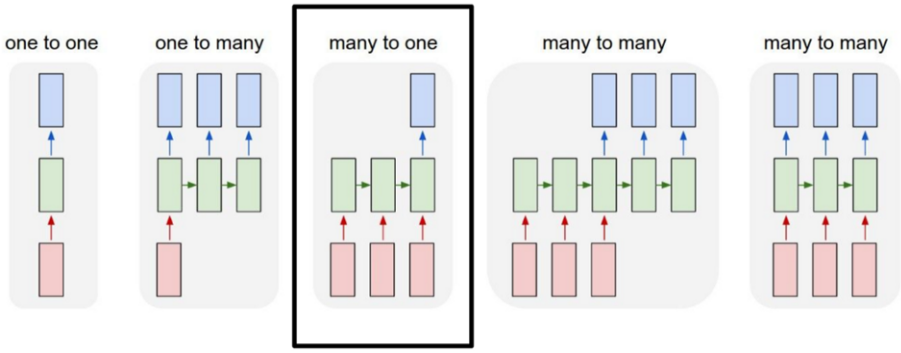

Figure 2.15: The input are sequences of words, output is one single class. Source:[25]

LSTMs have chain-like structure just like RNNs, but the repeating module has a more complex structure. Instead of having a single neural network layer, there are four, interacting in a very special way as shown in figure 2.16. In this diagram, the yellow boxes are learned neural network layers, the pink circles represent pointwise operations and each line carries an entire vector, from the output of one node to the inputs of others [34]. The main idea of LSTM is the cell state, which is the horizontal line shown in figure 2.17. The

cell state has a flow of information in it regulated by structures called gates. Those gates are composed out of a sigmoid neural net layer and a pointwise multiplication operation, outputting numbers between zero and one to describe how much of each component should be let through [34]. An LSTM has three of these gates, to protect and control the cell state.

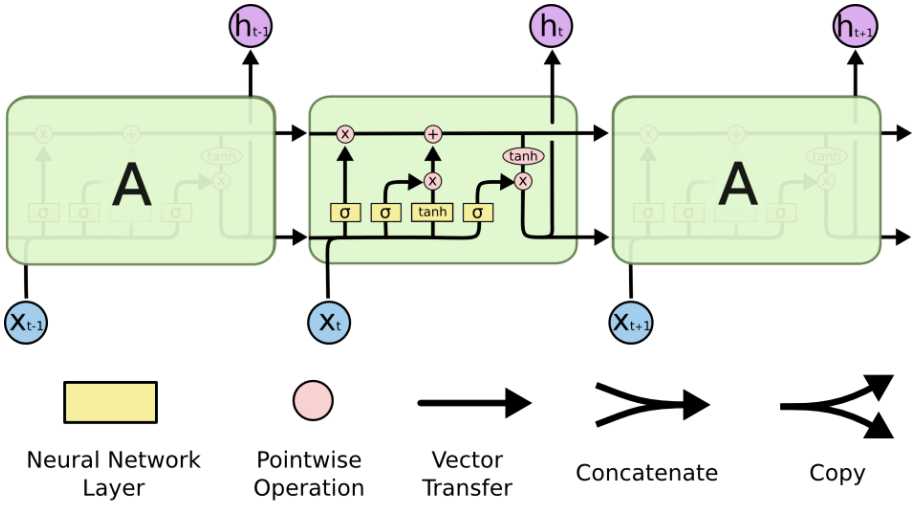

Figure 2.16: The repeating module in an LSTM contains four interacting layers. Source:[34]

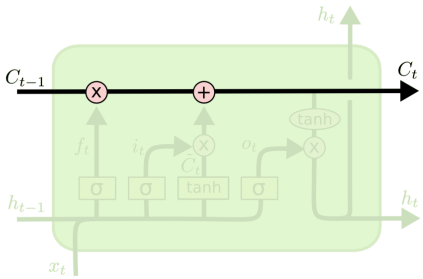

Figure 2.17: LSTM cell state. Source:[34]

2.11 Topic Representation

In the process of topic creation, a model named 'BERTopic' was used (we will discuss it in more details). This model uses two approaches to fine tune the topic modeling process, Class-based - Term Frequency - Inverse Document Frequency and Maximal Marginal Relevance.

2.11.1 Class-based - Term Frequency - Inverse Document Frequency

Starting with Term Frequency - Inverse Document Frequency (TF-IDF), is a widely used technique. Generally, we compute a weight to each word representing how important this word is in the document/sentence. TF-IDF is calculated using two measures, Term Frequency (TF) and Inverse Document Frequency (IDF) [30]. TF measures the frequency of a word in a document/sentence and calculated by the following formula:

$$\frac{x}{y}$$

Where x is the count of term t in document d & y is the number of words in document d.

The second measure is the IDF which is the inverse of Document Frequency (DF). DF is the count of occurrences of term t in the document set N [30]. Then we take its inverse to give a very low value for the most repeated words such as stopwords. IDF can be retrieved using the following equation:

$$\log \frac{N}{df + 1}$$

Finally by multiplying TF and IDF, we get our TF-IDF score given by this equation [30]:

$$TF * IDF$$

In the BERTopic model, the used metrics is Class-based Term Frequency - Inverse Document Frequency (c-TF-IDF). The only difference is that instead of applying TF-IDF to a document, we apply TF-IDF to a group of documents to get the most important terms in a cluster of documents, not a single one [10].

2.12 Maximal Marginal Relevance

Maximal Marginal Relevance (MMR) helps reduce redundancy and hence help in a better performance. It not only considers the similarity between keywords and the document, but also considers the similarity between already selected keywords to maximize the diversity of the chosen keywords. It also improves the coherence of words by finding the most coherent words without having too much overlap between the words themselves. This results in the removal of words that do not contribute to a topic like the author's signature [9].

2.13 Similarity Metrics

Similarity Learning is the area of learning a similarity function that measures how similar or related two objects are. By this measurement, we can tell how similar or different our data samples are. Similarity Metrics usually output a numerical value between zero and one where high similarity maps to one and low similarity maps to zero [1]. The metric that will be used in this study is the Cosine Similarity.

2.13.1 Cosine Similarity

Cosine Similarity is a mathematical measurement, however, its advantages extend beyond that. This measurement became too valuable in several ML applications like information retrieval and text matching. Mathematically, cosine similarity measures the cosine angle between two vectors in a multi-dimensional space [13]. In our study, those two vectors map to two sentences, each of them is an array of words. The cosine similarity metric is concerned about the orientation not the magnitude. So in figure 2.18, the two vectors will have a similarity of one, if they are aligned in the same orientation. However, perpendicularly aligned two vectors will have a similarity of zero [37].

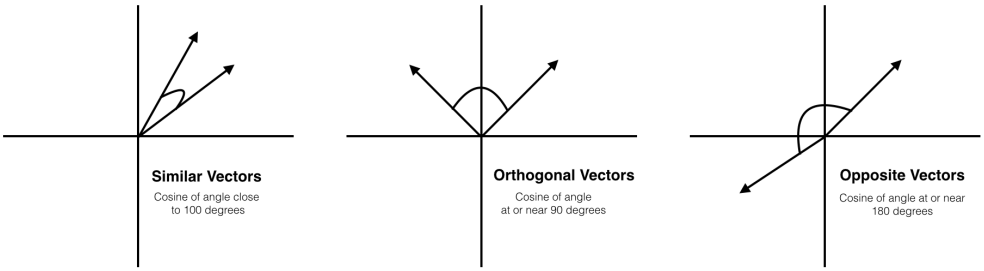

Figure 2.18: Mapping Similarity Measurements. Source:[37]

The Cosine Similarity is calculated using the following formula :

$$\frac{A.B}{||A||.||B||}$$

2.14 Evaluation Metrics

To evaluate the prediction accuracy of each classifier, ROUGE score is used. ROUGE is the most popular set of metrics for evaluating automatic summarization. It measures the accuracy by comparing the predicted summaries and the golden summaries (reference summaries, can be human produced) [18]. More specifically, we will use ROUGE-N and ROUGE-L.

2.14.1 ROUGE-N

ROUGE-N measures the number of matching 'n-grams' between automatically generated summary and reference summary. The N can vary from 1 to n. However, the greater our n, the higher is the computational cost. That's why the mostly used n-gram metrics are uni and bi-gram. For example, if our summary sentence is "my dog is so sleepy", its list of unigrams would be ['my', 'dog', 'is', 'so', 'sleepy'] and bigrams would be ['my dog' , 'dog is', 'is so' , 'so sleepy'] [33]. Once we have decided which N to use, we would think about whether we'd like to calculate the ROUGE recall, precision, or F1 score. To get good valid metrics, we will compute precision, recall and F1. Starting with the recall, it references how much does the predicted summary cover from the golden summary. Recall can be computed using the following formula:

$$\frac{x}{y}$$

Where x is the number of n-grams found in model and reference and y is the number n-grams in reference.

However, a good recall does not necessarily mean that we are having a good predicted summary as the machine generated summary can be extremely long, that it captures all words in the golden summary and a lot more useless words. That's where precision comes to the rescue. Precision measures how relevant/necessary is every word in the predicted summary [33]. Precision can be calculated using the following formula:

$$\frac{x}{y}$$

Where x is the number of n-grams found in model and reference and y is the number n-grams in model.

Consequently, it is always best to merge both recall and precision metrics to compute the ROUGE F1 score [33]. It can be calculated like so:

$$2 * \frac{precision * recall}{precision + recall}$$

That gives us an accurate measure of our model performance that relies on both capturing as many words as possible (recall) and making sure the captured words are relevant (precision).

2.14.2 ROUGE-L

Moving to ROUGE-L, this metric measures the Longest Common Subsequence (LCS) shared between our model output and reference. Its approach supports that the longer shared sequence would indicate more similarity between the two sequences. Recall, precision and F1 scores can be calculated just as illustrated before but with replacing the

'number of overlapping words' with the calculated LCS [33]. In this case, Recall will be calculated as :

$$\frac{\text{LCS}(gram_n)}{ReferenceCount(gram_n)}$$

And precision will be calculated as follows:

$$\frac{\text{LCS}(gram_n)}{ModelCount(gram_n)}$$

And the F1 score equation stays the same.

2.15 Literature Review

2.15.1 Text Summarization

Extractive summarizers aim at picking the most relevant sentences in a document/documents to form an informative summary while solving the two major problems in extractions which are long sentences and relevance. Extractive Text Summarization has been presented and implemented in various techniques for single document and multiple documents, in this section, we will discuss different extractive summarization techniques implemented in previous work. They can be classified into four main categories, which are statistical, semantic, fuzzy-logic and machine-learning approaches.

Statistical Approaches

These are the approaches that use statistical-based algorithms like sentence position, term frequency, title similarity, terms polarity and sentence length to score the sentences and then extract top k scored sentences to generate our summary, without trying to understand the meaning behind the sentences.

TF-IDF, assign a score to each word based on two algorithms TF and IDF. TF measures the number of times a word appears in a given document/documents with respect to all words and IDF measures how rare and unique a word is. The output of both algorithms is used to assign each word a score, that will then be used to calculate each sentence score. The assumption is that the more essential a piece of information is, the higher its words' score. Sentence length and TF-IDF were used as the scoring algorithms in A.Agrawal and Gupta (2014) [39].

An unsupervised learning technique was used which is k-means clustering technique. The sentences' scores were used to represent the sentences as unique coordinates in the single dimensional Cartesian plane. These coordinates were used as input data for unsupervised k-means clustering algorithm. Simulating the algorithm on the input data generates k cluster centers. Then each sentence was classified into different clusters based on

the scores computed for each sentence. Then, from the groups of sentences, the most representative sentence was selected for composing the summary [39]. The proposed approach does not need a lot of golden summaries for training as in supervised training. Also, when compared to humans abstractive summary, this approach produces a very similar result unlike other extractive-based techniques.

Machine Learning Approaches

There are a wide variety of machine learning approaches when it comes to extractive text summarization classified into supervised, unsupervised and semi supervised learning techniques. Supervised learning techniques are such as Multilayer neural network, Regression, Support vector machine, Decision tree, and Naïve bayesian classifier. Machine learning approaches model the summarization as a classification problem; sentences are classified as summary sentences or non-summary sentences based on the features that they possess. Artificial neural networks are used for creating the summary of the arbitrary length articles. The network is trained according to the style of the human reader and to which sentences the human reader deems to be important in a paragraph.

In (Nallapati 2017), SummaRuNNer focuses on sentential extractive summarization of single documents using neural networks [51]. The model used consists of a two-layer bi-directional GRU RNN, which is a recurrent network with two gates, 'U' called the update gate and 'R', the reset gate. The first layer (bottom one) of the RNN runs at the word level, and the second layer (top one) of bi-directional RNN runs at the sentence level. Evaluations were computed using different variants of the Rouge metric with respect to the gold summaries on the CNN/Daily Mail corpus and DUC 2002 corpus. SummaRuNNer outperforms or matches the state-of-art models for the summarization and also, it allows interpretable visualization of its decision.

The approach presented in (Yin 2015), is a powerful model in sentence representation based on a neural network language model for an input of multiple documents [56]. Firstly, Convonutional neural networks apply project sentences into distributed representation. Secondly, cosine similarity measurement is applied for representing and modeling the sentence's redundancy. Finally, as for optimization, a sentence selection method called diversified selection is used to pick up the high quality sentences by minimizing the prestige and diversity cost of them.

Semantic Approaches

Those approaches form summaries by understanding the semantics behind given sentences, where every sentence is represented in the form of a vector in a representation space, placing the words which have similar semantic meaning near to each other in the space. It then compares different sentences and ranks them in terms of their importance using similarity measures like cosine similarities, then use the top k sentences to give the

summary. Using Glove model and Latent Semantic Analysis (LSA) for text summarization is an example of semantic approach.

In Lsa(2011), LSA technique is used [52]. LSA is a fully unsupervised learning technique composed of three main steps including: input matrix creation, Singular Value Decomposition (SVD) and sentence selection. In input matrix creation, the input document is represented by a matrix in which columns are mapped to sentences, rows are mapped to words and cells represent the importance of words in sentences. The relations between the words in different sentences is depicted by the SVD, which also has the capability to reduce the noise to improve the accuracy. Then using the results of SVD, different algorithms are used to select important sentences. Different LSA methods are executed using different input matrix creation methods. Evaluation of the LSA-based summarization systems in this thesis was carried out using Turkish and English datasets and evaluated by ROUGE scores. Unfortunately, LSA has some limitations. One of those is that, the performance decreases sharply with larger and more inhomogeneous data, this decrease in performance is caused by SVD, which is a very complex algorithm. Also, LSA does not use the information about word order, syntactic relations, and morphologies.

Fuzzy Logic Approaches

Sometimes, it is not exact to classify the sentences using a binary classification as zeros and ones, that's why fuzzy logic is used. The fuzzy logic approach mainly contains four components: defuzzifier, fuzzifier, fuzzy knowledge base and inference engine. Fuzzy values can range from zero to one, allowing a sentence to be partially included in the summary after ranking the sentences and choosing top k ones to generate our automated summary.

In (Anfis 2017), Adaptive Neuro-Fuzzy Inference System (ANFIS) is used to summarize single documents [46]. A vector of nine features for each sentence including: title similarity, proper noun, sentence position, numerical data and sentence similarity will be input to nine neurons in ANFIS model. Then using a membership function, each input will be converted to a fuzzy value. This model tackles the problem of needing the human experts for building fuzzy rules by using subtractive clustering methods to automatically generate rules. ANFIS, a hybrid approach which takes the advantages from both neural network and fuzzy logic has improved the accuracy of the summarization model in determining the sentences which should be included in the summary. The model is trained on the DUC-2002 corpus and achieved average precision of 0.7128 and average recall of 0.698.

2.15.2 Topic Modeling

Topic modeling is an unsupervised machine learning technique, used for finding coherent topics to a set of documents. Two of the most popular techniques to apply topic modeling are Latent Dirichlet Allocation (LDA) proposed in 2003 [40] and Non-Negative Matrix Factorization (NMF) [47] . LDA is a generative probabilistic model which describes each document as a mixture of topics and each topic as a distribution of words. NMF reduces the dimensionality of non negative matrices to discover the underlying relationships between texts. Despite these approaches popularity, they have several weaknesses. In order to achieve optimal results they often require the number of topics to be known, custom stop-word lists, stemming, and lemmatization. Additionally these methods rely on bag-of-words representation of documents which ignore the ordering and semantics of words. That's why, techniques based on BERT for topic modeling are now getting more considered [43].

In 2021, BERTopic was applied for an Arabic dataset using different Pre-trained Language Models and compared with LDA and NMF [38]. The experiments were applied on 108789 documents which were categorized into five classes: sport, politics, culture, economy and diverse, and evaluated using Normalized Pointwise Mutual Information (NMPI). The overall results generated by BERTopic showed better results compared to NMF and LDA.

Moreover, BERTopic was used to describe and evaluate topics in legal documents via domain-specific embeddings pre-trained from LEGAL-BERT [54]. The pre-trained embeddings provide a more fine tuned representation of the text. Then the semantic representation of the documents is extended with inserting the United States Code cited in the document. Adding the references to laws in embedding representation of the text improves the quality of topic modeling. The input of the system is a text of some legal document and the output is k-top words of the topic that represents the document's theme.

Chapter 3

Methodology

3.1 System Overview

As mentioned before, this thesis studies two NLP tasks, Extractive Text Summarization and Topic Modeling, using the same dataset and almost the same techniques but with different preprocessing steps.

Starting with Text Summarization, the workflow starts by the 'Data Exploration' step. It would be interesting to discover and visualize the data we will use, along with the relationships between different features. Secondly, we move to the 'Data Preprocessing' step, in order to prepare the data by fetching it from files, cleaning it and highlighting important details. Then, we go through the 'Feature Engineering and Selection' phase where we calculate, create and select the features that will help obtain accurate and efficient results. After that, the features go through the 'Dimensionality Reduction' step using UMAP. And here we have reached the clustering phase where we apply 'Clustering Validation methods' to find the optimal number of clusters. Then using the number of clusters obtained, we apply 'K-means Clustering'. Last not least, training our model using multiple 'Classifiers' as Logistic Regression, SVM and LSTM, followed by their 'Evaluation'.

Moving to Topic Modeling, the workflow starts by some 'Data Preprocessing' steps, however those steps are different from those applied for text summarization (we will discuss why in the following sections). Then, a predefined model named 'BERTopic' is used to identify special tags (set of words) that best describes the given document. BERTopic go through an 'Embedding Documents' step where documents are turned into a set of vectors. Followed by, 'Dimensionality Reduction' using UMAP and 'Clustering' using HDBSAN steps. Finally, c-TF-IDF and MMR to improve the coherence of the outputted words. Figure 3.1 shows the exact work approach for both Text Summarization and Topic Modeling.

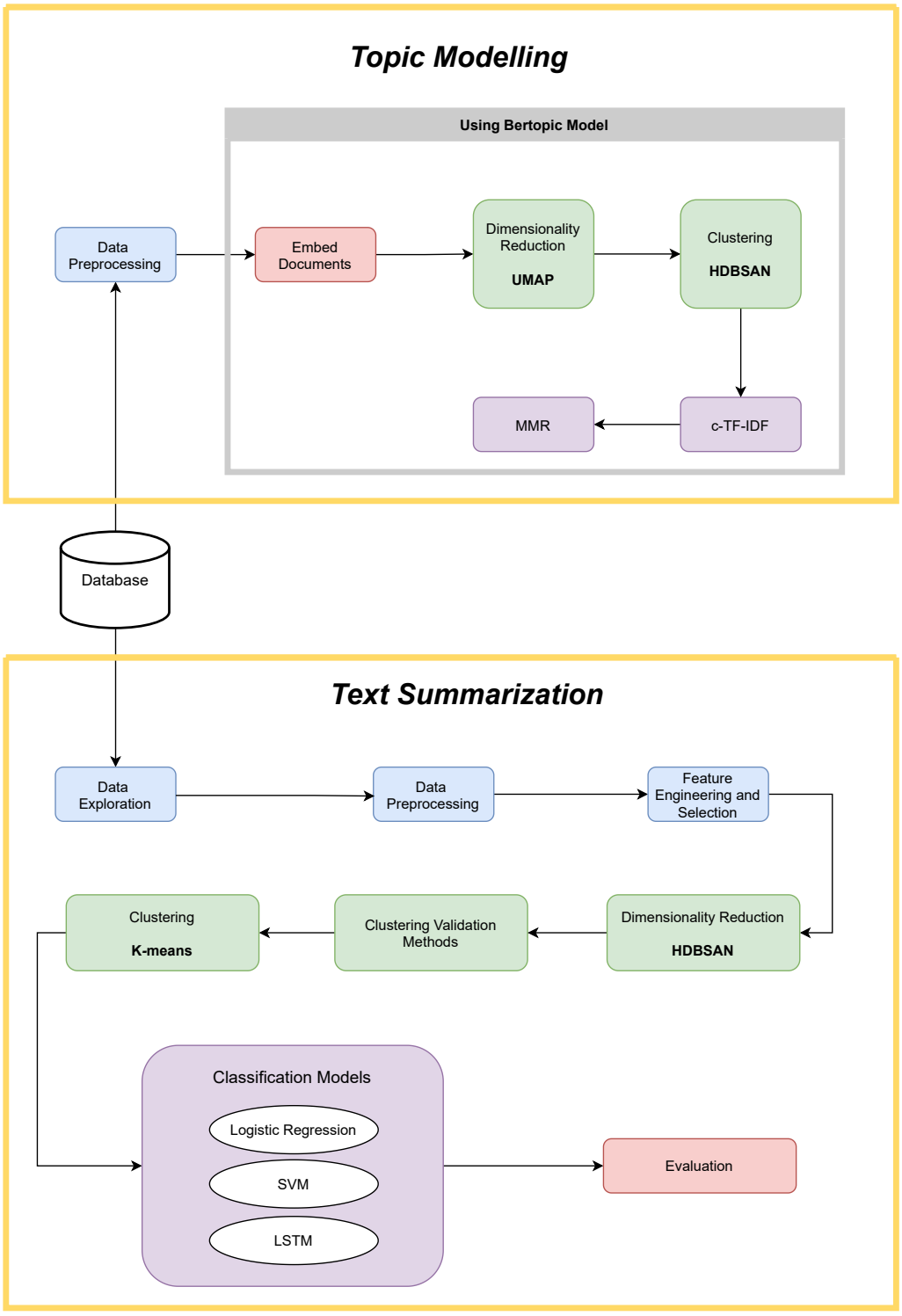

Figure 3.1: Workflow Overview

3.2 Dataset

Training data for supervised extractive summarization models has always been a great challenge. That's because we require sets of texts and their gold summaries (human produced summaries). While the canonical news corpus for summarization is the CNN / Daily Mail database, this study makes use of Cornell University's Newsroom database which is larger, more diverse and contains interesting metadata labels. The full dataset contains 1.3 million articles and summaries sources from 39 publications and features a wide array of domain subjects and summarization strategies. In addition, Cornell's dataset contains metrics to assign three strategy types to each summary: extractive, mixed or abstractive which is a great advantage in our case (implementing Extractive Summarization). Domain subject of each article is not labelled, so, we used unsupervised learning techniques to implement topic modelling. A partition of 10,000 records from the dataset is used. The database was obtained on request, after agreeing to certain copyrite conditions, from Cornell and is delivered in JSONL format [44].

3.3 Data Exploration

Let's discover our dataset statistically:

- In the following figure 3.2, articles and summaries are measured in sentence number. As shown, the average and median article lengths are 33 and 26 respectively.
- Average and median sentences for the summaries are 3 and 2 respectively.
- Another note here is that 75% of summaries have only 4 or less summary sentences.

		Number of Sentences	Article	Summary
		Percentile		
		0.00	1.00	1.0
	mean	0.25	14.00	1.0
Article	33.195	0.50	26.00	2.0
Summary	3.268	0.75	42.00	4.0
		0.95	76.00	8.0
		0.99	153.01	14.0
		1.00	1147.00	55.0

Figure 3.2: Articles and Summaries length comparison.

- It will be interesting too to examine the relationship between lengths of articles and their summaries, as illustrated in figure 3.3.
- Apart from a small percentage, we see that there is a stable relationship between article and sentence length. Article lengths could then be used by the model to predict how many sentences to output as a summary.

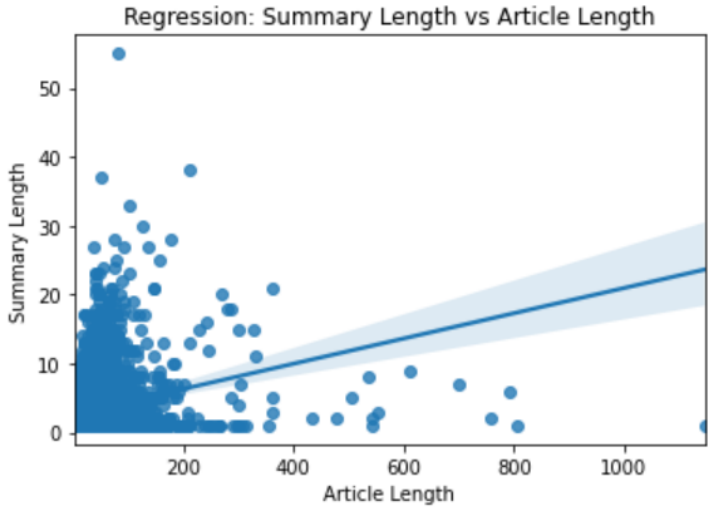

Figure 3.3: Articles and Summaries lengths relationship.

- Lastly, we look at the percentage breakdown of publications across the corpus and track the leading sentence distribution across the top 10 news vendors, shown in figure 3.4. The top 10 account for 80% of the corpus and the NY Times accounts for nearly 25%.

	% Breakdown	Cumulative
nytimes	0.2383	0.2383
bostonglobe	0.1356	0.3739
nydailynews	0.0837	0.4576
cnbc	0.0828	0.5404
p://fortune	0.0512	0.5916
9news	0.0510	0.6426
sfgate	0.0452	0.6878
tmz	0.0426	0.7304
p://nypost	0.0386	0.7690
people	0.0364	0.8054

Figure 3.4: Publications across the Corpus

3.4 Text Summarization

3.4.1 Data Preprocessing

Extractive strategies are a pure classification problem where the goal is to label sentences within the full text as either belonging to a summary or not. So firstly, we have to filter our dataset to only Extractive labelled summaries to have a dataset of all extractive summaries of length 10000. After that, we splitted every article and summary into a list of sentences using spacy. Then, we cleaned the implemented lists by ignoring any sentence containing less than 3 words. Removal of stopwords, lemmatization and stemming are intentionally ignored because these words and nuances provide semantic and contextual information. For example, the use of "and" versus "or" can give a sentence entirely different meanings.

3.4.2 Feature Engineering and selection

In this section, we will discuss how and why every added feature is calculated, selected and used. The following is our list of features:

- **Text_clean:** A list of sentences representing article sentences.
- **Text_embedding:** We applied S-BERT Sentence Embeddings on every sentence in the text_clean, this outputs a vector of size 768 representing every sentence, so if our article contains 33 sentences, applying S-BERT embeddings to this article will output an array (length 33) of arrays (each of length 768), every one of the 33 arrays represents the semantics of a sentence in the article.
- **Summary_clean:** A list of sentences representing summary sentences.
- **Summary_embedding:** We applied BERT Embeddings to the summary_clean, mapping every sentence to its embedding in a 768 vector space (just as text_embedding).
- **Document_embedding:** Text summarization requires each sentence to have some knowledge about the article as a whole or its relative embedding to the other sentences. So, this is simply calculated as the mean of the sentence embeddings for the article (the mean of text_embeddings calculated before), by using the .mean() function. Again, a vector of size 768.
- **Sent_Number:** The sentence order in an article. This feature had a very positive impact on the performance. That's because the earlier a sentence is in the article, the more likely it will appear in its summary.
- **Doc_Length:** Number of sentences per article.
- **Sum_Length:** Number of sentences per summary.

- **Total_words_count:** Number of words in a sentence. This feature also showed great potential because the longer a sentence is, the more informative it might be, and hence the more likely this sentence will appear in the summary.

- **Clusters_idx:** We used the k-means clustering algorithm to cluster the sentences (will be discussed in more details). Then the cluster produced for each sentence was added to the feature set, this feature had a very positive impact on the system's performance.

- **Labels_idx_list:** We are applying supervised learning techniques, so we will have to get our labels ready to pass it to the training models. As mentioned before, our goal is to classify each sentence to whether it will be included in the summary (1) or it will not (0). So, our labels would be an array of sentences' numbers that are included in the summary. For example, a Labels_idx_list = [11, 12, 20], shows that this article's summary consists of the 11th, 12th and 20th sentences in the article. To accomplish this, we need to know which sentences/part of sentences from the articles are included in the summary, however not all summary sentences are fully extractive; the text "I came to work late yesterday because my dog was sick all night" might be included in the summary as "I came to work late yesterday". That's why cosine similarity is used to identify the closest match to each summary sentence. SK-Learn's inbuilt cosine similarity function is used to calculate the similarity between every summary sentence and every article sentence using text_embedding and summary_embedding calculated in previous steps. And hence output the most similar article sentences to our summary. Figure 3.5 shows an example, we have an article of length 15 sentences and its summary has length equal to 4 sentences. Cosine Similarity will output the following cos_sim_mat, then we will look for the highest similarity between each of the 4 summary sentences and all article sentences, which in this case will be sentences 0,1,1,2.

```
cos_sim_mat
```

```
array([[0.99999976, 0.50047994, 0.5196508 , 0.59983295],
       [0.6196736 , 0.7722818 , 0.7966124 , 0.5275056 ],
       [0.599833  , 0.29468644, 0.52074134, 0.99999976],
       [0.56438607, 0.33038598, 0.55166763, 0.6393672 ],
       [0.4886982 , 0.45817882, 0.51193637, 0.624065  ],
       [0.46047905, 0.35522765, 0.4688924 , 0.48108268],
       [0.28927967, 0.3464931 , 0.5418632 , 0.40989965],
       [0.5305566 , 0.32012153, 0.54381216, 0.5782098 ],
       [0.46696132, 0.32249698, 0.70351344, 0.66038245],
       [0.2646217 , 0.31237996, 0.51453644, 0.3949415 ],
       [0.61532736, 0.36732113, 0.54515684, 0.77773935],
       [0.34573337, 0.33674398, 0.3050996 , 0.35545546],
       [0.48274225, 0.2860468 , 0.5660532 , 0.7047001 ],
       [0.63601404, 0.3220192 , 0.6802253 , 0.57318306],
       [0.27251178, 0.37319538, 0.42750496, 0.35306516]], dtype=float32)
```

```
idx_arr = np.argmax(cos_sim_mat, axis=0)
idx_arr
```

```
array([0, 1, 1, 2], dtype=int64)
```

Figure 3.5: Calculating cosine similarity

To sum it up, figure 3.6 shows our feature set which is composed of text_embedding (vector of size 768), document_embedding (vector of size 768), sentence number, document length, summary length, total words count, and sentence 's cluster. (each record represent a sentence in a document, that's why sentences in the same document will have same doc embeddings, doc length and summary length).

	Sent_BERT_D_0	Sent_BERT_D_1	Sent_BERT_D_2	Sent_BERT_D_3	...	Sent_BERT_D_764	Sent_BERT_D_765	Sent_BERT_D_766	Sent_BERT_D_767
0	0.294639	-0.595545	-0.077182	-1.181117	...	0.212790	-0.446712	0.661156	-0.783301
1	-0.148342	-0.487103	0.633056	-1.041132	...	-0.005954	0.235457	0.398671	-0.450129
2	0.184422	-0.550722	0.341682	-0.959729	...	-0.061294	0.385642	0.354645	-0.727727
3	-0.434744	-0.549924	-0.124533	-0.674852	...	-0.090414	0.194115	1.001167	-0.097112
4	0.587411	-0.441652	0.307622	-0.231736	...	-0.564287	0.017638	-0.143863	0.076410

	Doc_BERT_D_0	Doc_BERT_D_1	Doc_BERT_D_2	Doc_BERT_D_3	...	Doc_BERT_D_764	Doc_BERT_D_765	Doc_BERT_D_766	Doc_BERT_D_767
0	-0.057089	-0.370998	0.226461	-0.737085	...	-0.074801	-0.052847	0.221650	-0.414717
1	-0.057089	-0.370998	0.226461	-0.737085	...	-0.074801	-0.052847	0.221650	-0.414717
2	-0.057089	-0.370998	0.226461	-0.737085	...	-0.074801	-0.052847	0.221650	-0.414717
3	-0.057089	-0.370998	0.226461	-0.737085	...	-0.074801	-0.052847	0.221650	-0.414717
4	-0.057089	-0.370998	0.226461	-0.737085	...	-0.074801	-0.052847	0.221650	-0.414717

	Sent_Number	Doc_Length	total_words_count	sum_length	clusters_idx
0	0.0	33.0	34	1	0
1	1.0	33.0	29	1	0
2	2.0	33.0	39	1	0
3	3.0	33.0	21	1	0

Figure 3.6: Fearure set

As for the features selection, manual selection was performed by selecting the features manually based on trial and error with the aim to maximize the predictive accuracy of classifiers. Data exploration phase assisted in better understanding the data and the importance of the features used to help in the selection of the features. Unnecessary features for this study, available in the dataset were ignored like title, release_date, url, density, etc.

3.4.3 Dimensionality Reduction using UMAP

The text_embedding and document_embedding features are passed to the dimensionality reduction algorithm UMAP. We reduced every set of embeddings' dimensions from 768 to only 5 while keeping the size of the local neighborhood at 15 as UMAP keeps a significant portion of the high-dimensional local structure in lower dimensionality. The optimal number of dimensions should not be too high, resulting in poorer clustering results (the following step) but also, not too low to not lose information.

3.4.4 Clustering using K-means

K-means Clustering algorithm was applied to calculate the centroid and add this assignment cluster label for each sentence to the feature set. The aim is to cluster the sentences

based on sentence reduced embeddings (not document embeddings), sentence number in the article and words count in the sentence. To find the optimal number of clusters, two techniques were applied, elbow method and silhouette analysis.

3.4.5 Classification Models

For summary predictions, data is split using 80% for training and 20% for testing. Then, three classifiers with variations of hyperparameters in each one, were tried and compared. First classifier is a classical Logistic Regression model with two variations, setting class weights to the default and to be balanced. Second classifier is a Support Vector Machine using gaussian kernel. Third and last classifier is the Long Short Term Memory neural network. A variety of unidirectional and bidirectional networks were tested with different numbers of neurons.

3.4.6 Evaluation

ROUGE-N is the standard evaluation metrics for text summarization. It measures the n-gram overlap between the predicted and gold summaries. F1 metric is calculated to balance between recall and precision as one disadvantage of the recall metric is that very long predicted summaries could contain all the words in the gold summary (earning a perfect 100% recall) but also have many unnecessary words. That's why, rouge precision was executed too, to capture how concise the predicted summary and then both were used to calculate ROUGE F1. ROUGE-L was also considered as an evaluating measure. We will use the rouge-score package to implement metric calculations.

3.5 Topic Modeling

3.5.1 Data Preprocessing

There are additional data preprocessing steps integrated before applying Topic Modeling. The goal here is to output a coherent topic for every set of documents.

- Punctuations are removed as they do not matter in extracting the most important words and will create a problem in differentiating with other words. However, we do need those punctuations in summarization.
- All characters were converted to lowercase, because we do not want to differentiate between the words 'DOG' and 'dog', they are both the same word, with the same importance.

- Stopwords like he, she, but, was, have, etc were removed. We do not need them in the case of topic modelling. From one side, we do not want the output of topic modelling to have a stop word as an important word (without this step this could happen because stop words are frequently repeated words), from the other side, stop words will not affect the importance of other words. However, stopwords are important in the case of Text Summarization.

- Lemmatization was applied to take the word to its root and bring words to their dictionary form. By that, our model will not treat the words 'excited' and 'exciting' as two different words, varying in importance.

3.5.2 BERTopic Model

Following the preprocessing step, data is split using 80% for training, 20% for testing and passed to a model named BERTopic [43]. BERTopic is a topic modeling technique that allows easy interpretable topics while keeping important words. BERTopic pipeline has several stages. Firstly, documents are embedded using BERT embedding technique. Second step is reducing those calculated document embeddings using UMAP dimensionality reduction technique and then cluster reduced embeddings using HDBSAN. Finally, creating topic representation by extracting, reducing topics with c-TF-IDF and improving coherence of words with MMR.

Chapter 4

Results

4.1 Data Dimensionality Reduction

Using UMAP, we reduced document BERT embeddings and sentence BERT embeddings from 768 dimensions to 5 dimensions. If number of dimensions smaller than 5 is used, information would have been lost (based on trial and error), so five is an optimal number. Figure 3.6 shows the initial 768 dimensions and figure 4.1 shows the updated feature set after reducing the dimensions.

	Doc_Reduced_Bert_0	Doc_Reduced_Bert_1	Doc_Reduced_Bert_2	Doc_Reduced_Bert_3	Doc_Reduced_Bert_4
0	0.546492	-6.864014	5.945311	2.252664	10.815696
1	0.546487	-6.863972	5.945272	2.252696	10.815673
2	0.546499	-6.864021	5.945284	2.252662	10.815671
3	0.546472	-6.864006	5.945293	2.252688	10.815742
4	0.546495	-6.863969	5.945285	2.252681	10.815670
	Sent_Reduced_Bert_0	Sent_Reduced_Bert_1	Sent_Reduced_Bert_2	Sent_Reduced_Bert_3	Sent_Reduced_Bert_4
0	10.019340	6.174984	3.051572	5.871604	1.613014
1	10.156852	6.835186	2.867902	5.821199	3.274038
2	10.497073	6.439278	3.560450	7.187341	2.713980
3	9.945075	6.511126	3.482961	5.384308	0.913819
4	9.794641	6.186894	3.085572	4.806489	1.465862

Figure 4.1: Feature set after applying UMAP

4.2 Data K-means Clustering output

To determine the optimal number of clusters Elbow Method and Silhouette Method were implemented to cluster sentence BERT embeddings. The two techniques were applied

after using UMAP. Elbow method was not clear if the optimal k equals 3 or 4 as shown in figure 4.2. Silhouette suggests k equals 3, however as illustrated in figure 4.3 the average silhouette score for k equals 4 was pretty good too. Consequently, we will apply our models using the two k values 3 and 4, then compare results.

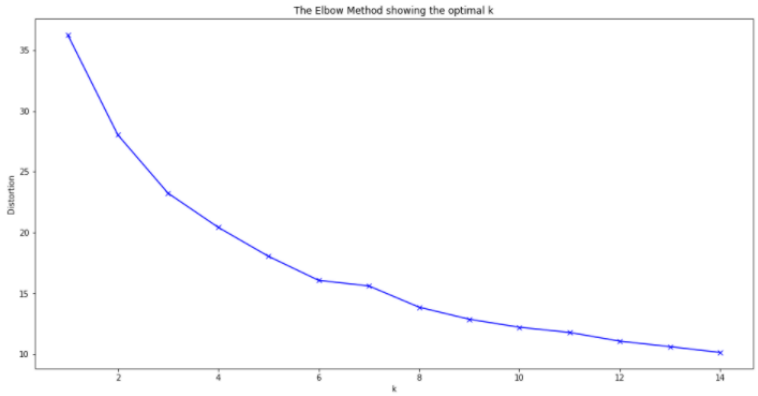

Figure 4.2: Using Elbow Method to predict optimal k value in k-means clustering

Figure 4.3: Using Silhouette Method to predict optimal k value in k-means clustering

4.3 Classification using Logistic Regression

Logistic Regression will be used as a classifier with some hyperparameter tuning; default class weights and balanced class weights. Logistic Regression was imported from SKLearn.

4.3.1 Training & Evaluating Logistic Regression Model

Using K-means clustering where k = 3:

ROUGE-1	F1	Recall	Precision
Default	0.560	0.540	0.755
Balanced	0.544	0.520	0.744

ROUGE-L	F1	Recall	Precision
Default	0.541	0.521	0.727
Balanced	0.522	0.500	0.711

Using K-means clustering where k = 4:

ROUGE-1	F1	Recall	Precision
Default	0.561	0.543	0.756
Balanced	0.544	0.521	0.743

ROUGE-L	F1	Recall	Precision
Default	0.542	0.524	0.728
Balanced	0.523	0.501	0.711

4.3.2 Results Discussion

In Logistic Regression, the best performed variant was the one with default class weights and k = 4 in k-means clustering by achieving ROUGE-1 F1 score of 56.1%. We can observe that:

- Precision is higher than recall with almost 20% in all cases.
- Default class weights outperformed balanced class weights.
- Clustering with k = 4 had higher accuracies than k = 3 in all cases.

Now, let's pick three random predicted summaries by the best performed model and compare them with the golden summaries.

Gold Summaries	Predicted Summaries
All day, every day, Cheryl Bernstein thanks her 16-month-old son. "I gave life to Reid, but he gave me life - a reason to get clean and go on,"she said yesterday after graduating from the Manhattan Family Treatment Court program. Bernstein, 41, and her husband, Doug Flaumenbaum, 33, both recovering crack and heroin addicts, were among three dozen men and women who regained custody of their children.	All day, every day, Cheryl Bernstein thanks her 16-month-old son. "I gave life to Reid, but he gave me life - a reason to get clean and go on," she said yesterday after graduating from the Manhattan Family Treatment Court program.Bernstein, 41, and her husband, Doug Flaumenbaum, 33, both recovering crack and heroin addicts, were among three dozen men and women who regained custody of their children.
Lionel Messi is named in Argentina's squad, less than two months after retiring from international football.	Lionel Messi has been named in Argentina's World Cup qualifiers squad, less than two months after retiring from international football.Messi, 29, quit after Argentina's Copa America final defeat by Chile in June. But in a statement reported by media in Argentina, the Barcelona striker said he made the U-turn for the "love of the country" and "to help from within".
The father of a baby girl whose Australian mother took her out of Canada is suing the Royal Canadian Mounted Police to court, claiming they were negligent in helping her leave.	The father of a baby girl whose Australian mother took her out of Canada is taking the Royal Canadian Mounted Police to court, claiming they were negligent in helping her leave.Craig Johnstone, 28, an electrician from North Vancouver, met Natasha Bride, from Melbourne, in 2010 while she was travelling in Canada, CBC News reports.But the relationship disintegrated and on April 15 last year Johnstone discovered Bride, 33, had packed up her belongings and taken their daughter to Vancouver's international airport, according to his lawsuit.

4.4 Classification using Long Short Term Memory

After that, LSTM model was trained and evaluated using epochs equal to one and batch size equals to one. Four network varieties were tested; unidirectional network with 25 neurons, bidirectional network with 25 neurons, unidirectional network with 50 neurons

and bidirectional network with 50 neurons. Followed by a dense layer in the four cases with sigmoid activation function. Layers were imported from keras.

4.4.1 Training & Evaluating LSTM Model

Using K-means clustering where k = 3:

ROUGE-1	F1	Recall	Precision
Uni 25	0.580	0.560	0.766
Uni 50	0.601	0.577	0.765
Bi 25	0.584	0.578	0.751
Bi 50	0.590	0.579	0.765

ROUGE-L	F1	Recall	Precision
Uni 25	0.561	0.541	0.739
Uni 50	0.572	0.559	0.739
Bi 25	0.566	0.560	0.725
Bi 50	0.572	0.561	0.739

Using K-means clustering where k = 4:

ROUGE-1	F1	Recall	Precision
Uni 25	0.582	0.574	0.754
Uni 50	0.590	0.582	0.760
Bi 25	0.586	0.574	0.760
Bi 50	0.588	0.582	0.761

ROUGE-L	F1	Recall	Precision
Uni 25	0.564	0.555	0.727
Uni 50	0.572	0.563	0.734
Bi 25	0.568	0.555	0.734
Bi 50	0.570	0.564	0.734

4.4.2 Results Discussion

In LSTM, the best performed network architecture was unidirectional network with 50 neurons and k = 3 in k-means clustering by achieving ROUGE-1 F1 score of 60.1%. We can observe that:

- Unidirectional networks with 50 neurons performed best when k was equal 3 and when k was equal 4.
- Difference between precision and recall in LSTM is slightly less than in Logistic Regression.
- LSTM performed better than Logistic Regression by 4%.

Now, let's pick random predicted summaries by the best performed model and compare them with the golden summaries.

Gold Summaries	Predicted Summaries
The U.S. House approved legislation Thursday to stem Puerto Rico's escalating debt crisis, capping an unusually bipartisan course on a fraught and technically complex compromise measure.	The House approved legislation Thursday to stem Puerto Rico's escalating debt crisis, capping an unusually bipartisan course on a fraught and technically complex compromise measure. The bill, which must still pass the Senate, establishes a process to handle what is shaping up as the largest municipal-debt workout in U.S. history. Puerto Rico's government has begun defaulting on $70 billion in debts.
The incoming owners of 1510 WMEX-AM said Tuesday they will pass on Rush Limbaugh and go instead with a new programming lineup meant to bring more humor and local coverage to conservative talk radio in Boston.	The incoming owners of 1510 WMEX-AM said they will pass on Rush Limbaugh and go instead with a new programming lineup meant to bring more humor and local coverage to conservative talk radio in Boston. It's a surprise move by a station that radio industry analysts pegged as the presumptive landing spot for Limbaugh's nationally syndicated show after his distributor, Premiere Networks.
Carson Palmer connected with Larry Fitzgerald for three of the TDs, giving him seven in two games after returning last week against New Orleans from a torn anterior cruciate ligament that cut short his season after just six games a year ago.	CHICAGO – As long as Carson Palmer is in the lineup, the Arizona Cardinals believe they can beat anybody. And they will be tough to knock off if he keeps playing like this.Palmer connected with Larry Fitzgerald for three of his TDs, giving the quarterback seven in two games after returning last week against New Orleans from a torn anterior cruciate ligament that cut short his season after just six games a year ago.

4.5 Classification using Support Vector Machine

Last but not least, SVM was trained with a gaussian kernel as a classifier. That's why, it consumed a lot of computational power and time. SVM was imported from SKlearn.

4.5.1 Training & Evaluating SVM Model

Using K-means clustering where k = 3:

ROUGE-1	F1	Recall	Precision
rbf	0.568	0.558	0.749

ROUGE-L	F1	Recall	Precision
rbf	0.552	0.541	0.725

Using K-means clustering where k = 4:

ROUGE-1	F1	Recall	Precision
rbf	0.568	0.558	0.749

ROUGE-L	F1	Recall	Precision
rbf	0.552	0.541	0.725

4.5.2 Results Discussion

During using SVM as a classifier, we faced computional power and time challenges. Best ROUGE-1 F1 score was at 56.8%. We can observe that:

- There's no difference between clustering into 3 or 4 clusters, they both outputted exactly same accuracies.
- SVM performed better than Logistic Regression by 0.8%
- However, LSTM performed better than SVM by 3.3%.

Now, let's pick three random predicted summaries by the best performed model and compare them with the golden summaries.

Gold Summaries	Predicted Summaries
The four-year-long fiesta of unbridled public spending, imports and foreign borrowing that followed confirmation of Mexico's new-found oil wealth has now given way to a morning-after feeling of regrets at the prodigality. Suddenly, ebullient toasts have been deflated by the sobering appearance of the bill for the fiesta: 30 percent inflation, a dangerously overvalued currency, stagnant nonoil exports and a public sector foreign debt that has doubled to $48.7 billion since 1977. Yet in reality, things were neither as good as they seemed a year ago, when Mexicans were reveling in annual growth rates of 8 percent, a tripling of oil production and a tenfold increase in proven hydrocarbon reserves, nor as bad as they seem today. Confidence, though, is shaped by perceptions rather than realities, and, in a matter of months, the mood among government economists and foreign bankers has gone from optimistic to pessimistic.	The four-year-long fiesta of unbridled public spending, imports and foreign borrowing that followed confirmation of Mexico's new-found oil wealth has now given way to a morning-after feeling of regrets at the prodigality. Suddenly, ebullient toasts have been deflated by the sobering appearance of the bill for the fiesta: 30 percent inflation, a dangerously overvalued currency, stagnant nonoil exports and a public sector foreign debt that has doubled to $48.7 billion since 1977. Yet in reality, things were neither as good as they seemed a year ago, when Mexicans were reveling in annual growth rates of 8 percent, a tripling of oil production and a tenfold increase in proven hydrocarbon reserves, nor as bad as they seem today.
As videogames become a spectator sport, Amazon.com just bought the world's largest arena. The e-commerce giant said Monday it agreed to acquire Twitch, a popular Internet video channel for broadcasting, and watching, people play videogames, for about $970 million in cash.	As videogames become a spectator sport, Amazon.com Inc. just bought the world's largest arena. The e-commerce giant said Monday it agreed to acquire Twitch Interactive Inc., a popular Internet video channel for broadcasting, and watching, people play videogames, for about $970 million in cash.
The French city of Nice says it will not host the European Road Cycling Championships, planned for 14-18 September, for security reasons.	The French city of Nice says it will not host the European Road Cycling Championships, planned for 14-18 September, for security reasons. Mayor Philippe Pradal said the event required a large police presence, but the southern city had not "received any guarantees" about their deployment. It is the latest event in France to be called off after July's lorry attack in Nice in which 85 people died.

4.6 Comparison to Lead-3 Baseline

We used the Lead-3 baseline, in which the first three sentences of the text are returned as a summary. Though simple, this baseline is competitive with state-of-the-art systems following prior work in (Nallapati 2017) [51]. Lead-3 was implemented across the same test sample and compared to the best performed Logistic Regression, SVM and LSTM models.

ROUGE-1	F1	Recall	Precision
Lead-3	56.1%	58.4%	70.6%
LR	56.1%	54.3%	75.6%
SVM	56.8%	55.8%	74.9%
LSTM	60.1%	57.7%	76.5%

ROUGE-L	F1	Recall	Precision
Lead-3	54.5%	56.6%	68.4%
LR	54.2%	52.4%	72.8%
SVM	55.2%	54.1%	72.5%
LSTM	57.2%	55.9%	73.9%

LSTM and SVM were both able to beat the high standard Lead-3 baseline being implemented on the same dataset. Also, Lead-3 recall is higher than LSTM and SVM but its precision is far lower. This suggests that Lead-3 produces longer summaries and, LSTM and SVM do a good job in picking out relatively shorter or more concise sentences.

4.7 Topic Modeling

After applying various preprocessing steps, data is passed to BERTopic model. Topics' number was reduced by merging topics with minimum similarity of 0.915. After that, we are left with 61 topics. Figure 4.4 visualizes topics clusters and their modeling. Each cluster is modelled by the most weighted words in this cluster's documents and hence give insights about what topic a cluster's documents is about. In figure 4.4, we can tell that topic 6 (cluster 6) is about cinema and media as the most important words in this cluster's documents are film, movie, show,..etc

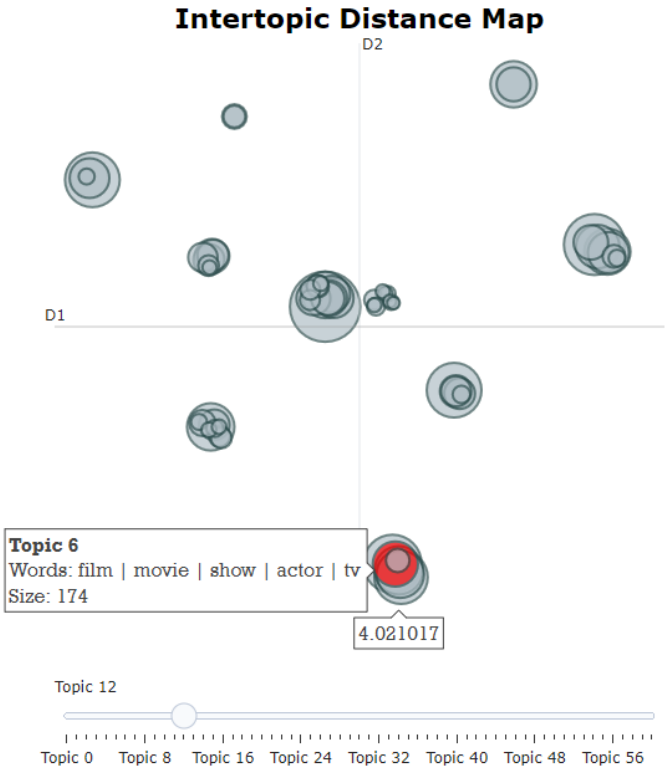

Figure 4.4: Visualizing topics clusters and their modeling

In figure 4.5, we can visualize the c-TF-IDF scores for each topic representation.

Figure 4.5: c-TF-IDF scores for each topic representation

4.7.1 Results Discussion

The following table shows some random documents and their predicted list of tags.

Articles	List of Tags
SEOUL—For the launch of its latest premium smartphone, Samsung Electronics Co. is sticking to its winning playbook. The smartphone maker introduced on Tuesday a refresh of its large-size Galaxy Note series that hews to the same formula that made its flagship Galaxy S7 a hit with consumers earlier this year: Add water-resistance and expandable memory to a sleek metallic curved-screen smartphone...	[Iphone, Mobile, App, Android, Ipad, Tablet, Smartphone, Store]
Since Novak Djokovic started to dominate tennis in 2011, his gluten-free diet, fanatical fitness regimen and love of meditation have been the talk of the sport. But let's not overlook the obvious: Djokovic is No. 1 because he's exceptional at everything—even the serve, a shot that used to drive him mad. In the last few years, as Djokovic has banished baguettes and pizza from his dinner plate, he has transformed a ho-hum serve with suspect technique into one of the most accurate and deadly shots in the game...	[Olympic, Sport, Goal, Olympics, Medal, Player, Score]
Leo Apotheker is barnstorming the world as he embarks on fixing the culture and focus of the huge technology company. One to-do stands out above all the rest.Here's what new Hewlett-Packard CEO Leo Apotheker's to-do list might look like: The first two tasks should be easy enough to cross off the list. Finding real estate in the Silicon Valley can't be hard for a tech mogul with a $4.6 million relocation allowance, and HP HPQ certainly has the cash to pour into R&D. But building a successful software business will be tough for a company whose core business is selling PCs...	[Business, Company, Innovation, Marketing, CEO, Industry, Enterprise]

Chapter 5

Conclusion

In this work, we propose an Extractive Text Summarizer. The classifiers used are Logistic Regression, SVM and LSTM. Highest performed model was LSTM unidirectional network with 50 neurons at ROUGE-1 F1 score of 60.1%. Lead-3 baseline was used to compare results, and tested on the same test data. Given the high bar set by Lead-3, LSTM and SVM show better performance than or are comparable to this state-of-the-art system and they were able to beat it on both ROUGE-1 and ROUGE-L F1 which is impressive. LSTM exceeds the Lead-3 performance by 4% . The dataset was limited to 10,000 records due to computational time and power challenges, however, if we managed to use more training data, those scores could go even higher. We also propose a Topic Modeling system where a list of tags describing a topic is outputted for every set of documents. The approach presents very good results, revealing topics consistent with the document's theme. The aim of the whole study is to make unstructured data more beneficial and usable by taking a large amount of text data, outputting a summary and a list of tags for every text document.

5.1 Future Work

As a recommendation for future work in the areas of Text Summarization and Topic Modeling, there are some points that should be considered for a better performance.

- **Mixing Corpuses:** One challenge of the chosen news corpus is that its overwhelming sequential structure limits the generalizability of the model. This can be solved by mixing corpuses from other domains.
- **More Training:** The database is huge, however the computational time and power limited us to only 10,000 of the dataset.
- **Clustering using HDBSAN:** Instead of using K-means clustering in Text Summarization, HDBSAN should be considered as it can better adapt to high dimensional data.
- **Topic Modeling clusters:** Add Topic Modeling clusters to Text Summarization feature set.
- **Add Feature Selection algorithm:** For Example, Recursive Feature Elimination Algorithm.
- **Adding Topic Modeling output to Text Summarization features set:** Adding vectors representing the most important words in a document (generated from topic modeling) to text summarization training, will definitely enhance the performance because sentences including words close to the list of tags in the vector space would have higher opportunity to be included in the summary.

Appendix

Appendix A

Lists

NLP	Natural Language Processing
ML	Machine Learning
BERT	Bidirectional Encoder Representations from Transformers
S-BERT	Bert Sentence Embedding
UMAP	Uniform Manifold Approximation and Projection
HDBSAN	Hierarchical Density-Based Spatial Clustering of Applications with Noise
SSE	Sum of Squared Distance
SVM	Support Vector Machine
LSTM	Long Short Term Memory
RNN	Recurrent Neural Networks
TF-IDF	Term Frequency - Inverse Document Frequency
c-TF-IDF	Class-based Term Frequency - Inverse Document Frequency
TF	Term Frequency
IDF	Inverse Document Frequency
DF	Document Frequency
MMR	Maximal Marginal Relevance
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
LCS	Longest Common Subsequence
AI	Artificial Intelligence
LSA	Latent Semantic Analysis
SVD	Singular Value Decomposition
ANFIS	Adaptive Neuro-Fuzzy Inference System

LDA	Latent Dirichlet Allocation
NMF	Non-Negative Matrix Factorization
NMPI	Normalized Pointwise Mutual Information

List of Figures

Bibliography

[1] 17 types of similarity and dissimilarity measures used in data science. https://towardsdatascience.com/17-types-of-similarity-and-dissimilarity-measures-used-in-data-science-3eb914d2681.

[2] Ai-driven abstractive text summarization. https://medium.com/swlh/ai-driven-abstractive-text-summarization-56a094b722d4.

[3] A.i. technical: Machine vs deep learning. https://lawtomated.com/a-i-technical-machine-vs-deep-learning/.

[4] Automated text summarization: A short overview of the task and its challenges. https://www.dataminingapps.com/2016/02/automated-text-summarization-a-short-overview-of-the-task-and-its-challenges/.

[5] Automated text summarization techniques. https://www.kdnuggets.com/2019/01/approaches-text-summarization-overview.html.

[6] Automatic text summarisation. https://towardsdatascience.com/automatic-text-summarisation-ccc98d2b323f.

[7] Automatic text summarization with machine learning. https://medium.com/luisfredgs/automatic-text-summarization-with-machine-learning-an-overview-68ded5717a25.

[8] A beginner's guide to dimensionality reduction in machine learning. https://towardsdatascience.com/dimensionality-reduction-for-machine-learning-80a46c2ebb7e.

[9] Bert topic, the algorithm. https://maartengr.github.io/BERTopic/tutorial/algorithm/algorithm.html.

[10] c-tf-idf. https://maartengr.github.io/BERTopic/api/ctfidf.html.

[11] Clustering in machine learning. https://www.geeksforgeeks.org/clustering-in-machine-learning/.

[12] Common ml problems. https://developers.google.com/machine-learning/problem-framing/cases.

[13] Cosine similarity – understanding the math and how it works. https://www.machinelearningplus.com/nlp/cosine-similarity/.

[14] Deep learning. https://www.ibm.com/cloud/learn/deep-learning.

[15] Demystifying bert: A comprehensive guide to the groundbreaking nlp framework. https://www.analyticsvidhya.com/blog/2019/09/demystifying-bert-groundbreaking-nlp-framework/.

[16] The elbow method. https://www.oreilly.com/library/view/statistics-for-machine/9781788295758/c71ea970-0f3c-4973-8d3a-b09a7a6553c1.xhtml.

[17] A gentle introduction to hdbscan and density-based clustering. https://towardsdatascience.com/a-gentle-introduction-to-hdbscan-and-density-based-clustering-5fd79329c1e8.

[18] An intro to rouge, and how to use it to evaluate summaries. https://www.freecodecamp.org/news/what-is-rouge-and-how-it-works-for-evaluation-of-summaries-e059fb8ac840/.

[19] An introduction to clustering and different methods of clustering. https://www.analyticsvidhya.com/blog/2016/11/an-introduction-to-clustering-and-different-methods-of-clustering/#h2_5.

[20] Introduction to k-means clustering. https://medium.com/@dilekamadushan/introduction-to-k-means-clustering-7c0ebc997e00.

[21] Introduction to logistic regression. https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148.

[22] K-means clustering: Algorithm, applications, evaluation methods, and drawbacks. https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a.

[23] Logistic regression explained. https://towardsdatascience.com/logistic-regression-explained-9ee73cede081.

[24] Logistic regression vs support vector machines (svm). https://medium.com/axum-labs/logistic-regression-vs-support-vector-machines-svm-c335610a3d16.

[25] Multi class text classification with lstm using tensorflow 2.0. https://towardsdatascience.com/multi-class-text-classification-with-lstm-using-tensorflow-2-0-d88627c10a35.

[26] Natural language processing (nlp). https://www.ibm.com/cloud/learn/natural-language-processing.

[27] Nlp: Contextualized word embeddings from bert. https://towardsdatascience.com/nlp-extract-contextualized-word-embeddings-from-bert-keras-tf-67ef29f60a7b.

[28] Supervised learning. https://www.ibm.com/cloud/learn/supervised-learning.

[29] Support vector machines (svm) algorithm explained. https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/.

[30] Tf-idf from scratch in python on real world dataset. https://towardsdatascience.com/tf-idf-for-document-ranking-from-scratch-in-python-on-real-world-dataset-796d339a4089.

[31] Top 4 sentence embedding techniques using python! https://www.analyticsvidhya.com/blog/2020/08/top-4-sentence-embedding-techniques-using-python/#h2_8.

[32] Topic modeling: An introduction. https://monkeylearn.com/blog/introduction-to-topic-modeling/.

[33] The ultimate performance metric in nlp. https://towardsdatascience.com/the-ultimate-performance-metric-in-nlp-111df6c64460.

[34] Understanding lstm networks. https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[35] Unsupervised learning. https://www.ibm.com/cloud/learn/unsupervised-learning.

[36] What is classification in machine learning. https://www.edureka.co/blog/classification-in-machine-learning/.

[37] What is cosine similarity? https://deepai.org/machine-learning-glossary-and-terms/cosine-similarity.

[38] Abeer Abuzayed and Hend Al-Khalifa. Bert for arabic topic modeling: An experimental study on bertopic technique. *Procedia Computer Science*, 189:191–194, 2021.

[39] A. Agrawal and Utsav Gupta. Extraction based approach for text summarization using k-means clustering. 2014.

[40] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[41] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 160–172. Springer, 2013.

[42] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[43] Maarten Grootendorst. Bertopic: Leveraging bert and c-tf-idf to create easily interpretable topics., 2020.

[44] Max Grusky, Mor Naaman, and Yoav Artzi. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies, 2020.

[45] Hsiao-Lin Hwa, Wen-Hong Kuo, Li-Yun Chang, Ming-Yang Wang, Tao-Hsin Tung, King-Jen Chang, and Fon-Jou Hsieh. Prediction of breast cancer and lymph node metastatic status with tumour markers using logistic regression models. *Journal of evaluation in clinical practice*, 14(2):275–280, 2008.

[46] Yogan Jaya Kumar, Fong Kang, Ong Sing Goh, and Atif Khan. *Text Summarization Based on Classification Using ANFIS*, pages 405–417. 03 2017.

[47] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[48] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2020.

[49] Marvin Minsky. Steps toward artificial intelligence. *Proceedings of the IRE*, 49(1):8–30, 1961.

[50] N. Moratanch and S. Chitrakala. A survey on extractive text summarization. In *2017 International Conference on Computer, Communication and Signal Processing (ICCCSP)*, pages 1–6, 2017.

[51] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*, 2017.

[52] Makbule Gülçin Özsoy, F. Alpaslan, and I. Çiçekli. Text summarization using latent semantic analysis. *Journal of Information Science*, 37:405 – 417, 2011.

[53] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[54] Raquel Silveira, Carlos GO Fernandes, João A Monteiro Neto, Vasco Furtado, and José Ernesto Pimentel Filho. Topic modelling of legal documents via legal-bert1. *Proceedings http://ceur-ws. org ISSN*, 1613:0073, 2021.

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

[56] Wenpeng Yin and Yulong Pei. Optimizing sentence modeling and selection for document summarization. In *IJCAI*, 2015.