SUMMARIZED NEWS APPLICATION ALONG WITH NEWS DOCUMENT CLUSTERING FOR RECOMMENDATION USING DEEP LEARNING

A thesis (Phase-I) submitted in partial fulfillment of the requirements for the award of the degree of

M.Tech.

in

Data Analytics

By

ANUJ KUMAR JHA (205219007)

DEPARTMENT OF COMPUTER APPLICATIONS
NATIONAL INSTITUTE OF TECHNOLOGY
TIRUCHIRAPPALLI - 620015

DECEMBER 2020

BONAFIDE CERTIFICATE

This is to certify that the project work (phase I) titled **"Summarized News app along with News Recommendation and Document Clustering using Deep Learning"** is a bonafide record of the work done by

ANUJ KUMAR JHA (205219007)

in partial fulfillment of the requirements for the award of the degree of **Master of Technology** in **Data Analytics** of the **National Institute of Technology, Tiruchirappalli,** during the year 2020-2021.

Dr. (Mrs.) B. Janet	**Dr. P.J.A. Alphonse**
Project Guide	Head of the Department

Project Viva-voce held on ________________________________

Internal Examiner	**External Examiner**

ABSTRACT

“The newspaper is a greater treasure to the people than uncounted millions of gold.” **Henry Ward Beecher.** News has been one of the most significant part of human life since centuries. For some it is as significant as getting up and brushing teeth. News is the power that gives anyone the rights and authority to choose sides, to decide between good and bad, pass judgement on a government, and most importantly to be updated of all the things happening around the whole world.

Nowadays, the problem is that people are not invested in reading news as much as they should, and it can be bad for the society and the whole nation. The reasons for not being invested are as simple as news being too long, lot of useless news, too much bias, platform cluttered with lot of information which all could make the idea of reading news very overwhelming. If we investigate any news article, we can easily point out that there are only 4-5 lines which contains the main part of the news, rest it may include someone’s comment or reaction or information about some other related incident. One of the major objectives of the project is to summarize the news in few lines such that people could go through multiple summarized news instead of going through one single huge article. One other major problem is useless news being shown, one might be interested in Business, Sports and political news yet they are being shown Bollywood news. The other major objective of this project is to solve this problem by having classification of news, document clustering and recommendation.

Here initially I collected around 9000 news from various websites and in various categories. I used various state-of-the art models like BERT, XL-NET, GPT-2 and Newspaper3k for summarization of the news. After that I made use of summary extracted using BERT-Summarizer along with Doc2Vec embedding and PCA dimensionality reduction technique to convert words in proper 2-D vector format. And I also used K-Means document clustering algorithm to accumulate similar news article together and making the process of recommendation faster.

ACKNOWLEDGEMENT

I would like to extend my gratitude to **Dr. Mini Shaji Thomas, Director**, National Institute of Technology, Tiruchirappalli for having provided all the facilities to carry out this project work.

I would like to thank **Dr. P.J.A. Alphonse, Head of the Department**, Department of Computer Applications, National Institute of Technology, Tiruchirappalli for allowing me to avail the facilities of the department during project work.

I express my sincere thanks and gratitude to my guide **Dr. (Mrs.) B. Janet, Assistant Professor**, Department of Computer Applications, National Institute of Technology, Tiruchirappalli who directed me from time to time throughout the project with valuable guidance, constant encouragement and suggestions.

My hearty thanks to all the members of the Project Coordination Committee, Department of Computer Applications, National Institute of Technology, Tiruchirappalli, for their valuable suggestions during the project reviews.

I am also thankful to the faculty, staff members and research scholars of the Department of Computer Applications, National Institute of Technology, Tiruchirappalli, for their support and suggestions throughout the project.

Anuj Kumar Jha

(205219007)

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

BERT	Bidirectional Encoder Representation from Transformers
PCA	Principal Component Analysis
t-SNE	t-distributed Stochastic Neighbor Embedding
DBSCAN	Density-based spatial clustering of applications with noise
NLP	Natural Language Processing
MLM	Masked Language Modeling
PLM	Permutation Language Modeling
LSTM	Long Short-Term Memory
CNN	Convolution Neural Network
RNN	Recurrent Neural Network
PV-DM	Paragraph Vector-Distributed Memory
CBOW	Continuous Bag-Of-Words
LMs	Language Models
GPU	Graphics Processing Units
TPU	Tensor Processing Units

CHAPTER 1
INTRODUCTION

1.1 SCOPE

News is the power which gives us the authority to know and comprehend what is happening worldwide. News let us choose sides, let us decide what is good or bad, whether a government is worthwhile or not, who won the cricket match, when will the vaccines be released, and so much that we can't imagine a life without it. If it were not for news, then we wouldn't have been updated with everything going on in this pandemic.

The applications that are already present in the market right now lacks one of the two features, either it has summarization but no recommendation, or it has recommendation but no summarization. In this project I want to leverage the fact that both could be combined in making of a unique project. The reason why both summarization and recommendation are vital is that summarization not only saves time for users, but it provides the crisp information on a topic and readers are not overwhelmed with the huge content within an article. On the other hand, recommendation is crucial because even if we have summarization and show random news which a reader might not be interested in then there is no point of saving time because it will waste more time and also it will frustrate the readers.

Scope of this project lies in finding the best approach to combine summarization of news and recommending proper news articles. We will use various state-of-the art models like BERT which is encoder-based model, GPT-2 which is a decoder-based model, XL-Net an encoder-decoder based model and along with these various traditional models and algorithms like Doc2Vec for embedding, PCA and t-SNE for dimensionality reduction, and K-Means and DBSCAN for clustering.

1.2 TECHNIQUES

Deep learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural network. Deep learning architectures such as deep neural networks, recurrent neural networks and convolutional neural networks have been applied to fields including computer vision, speech recognition, natural language processing, audio recognition, social network filtering, machine translation, bioinformatics, etc. The progress made by attention models has led to growth in the field of machine comprehension and reading comprehension, to provide better reasoning of the context.

1.2.1 ATTENTION MECHANISM AND TRANSFORMER:

Attention models, or attention mechanisms, are input processing techniques for neural networks that allows the network to focus on specific aspects of a complex input, one at a time until the entire dataset is categorized. The goal is to break down complicated tasks into smaller areas of attention that are processed sequentially. Similar to how the human mind solves a new problem by dividing it into simpler tasks and solving them one by one. First, the computed attention weights are often used to extract the most relevant information from the context for answering the question by summarizing the context into a fixed-size vector. Second, in the text domain, they are often temporally dynamic, whereby the attention weights at the current time step are a function of the attended vector at the previous time step. Third, they are usually unidirectional, wherein the query attends to the context paragraph or the image.

The paper ‘Attention Is All You Need’ introduces a novel architecture called Transformer. As the title indicates, it uses the attention-mechanism. The Encoder is on the left and the Decoder is on the right. Both Encoder and Decoder are composed of modules that can be stacked on top of each other multiple times, which is described by *Nx* in the Fig1.1. We see that the modules consist mainly of Multi-Head Attention and Feed Forward layers. The inputs and outputs (target sentences) are first embedded into an n-dimensional space since we cannot use strings directly. One slight but important part of the model is the positional encoding of the different words. Since we have no recurrent networks that can remember how sequences are fed into a model, we need to somehow give every word/part in our sequence a relative position since a sequence depends on the order of its elements. These positions are added to the embedded representation (n-dimensional vector) of each word.

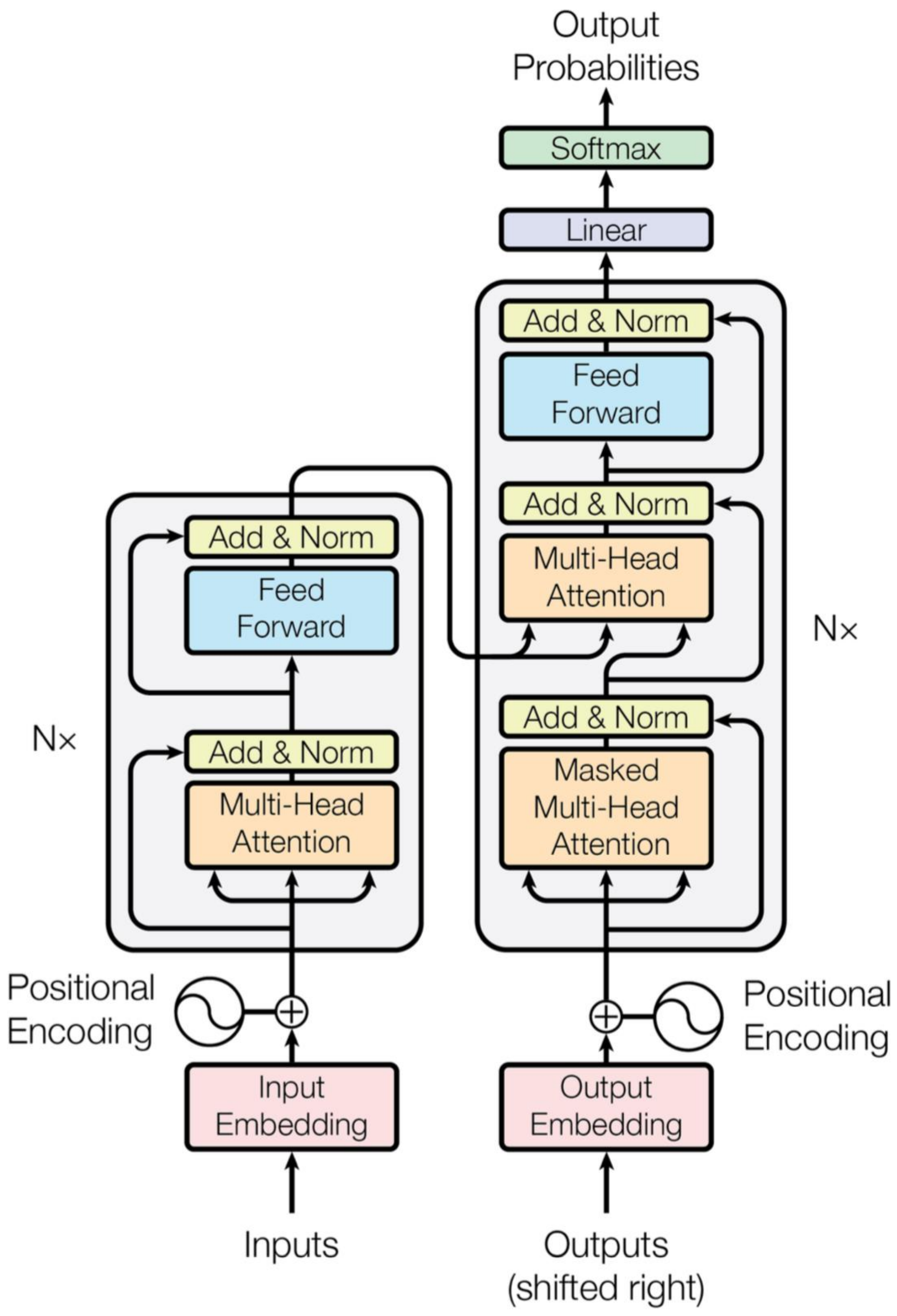

Fig 1.1: Transformer Architecture

- **BERT**

BERT (Bidirectional Encoder Representations from Transformers), released at the end of 2018, is a system of pre-trained representations of the language used to create models that can be downloaded and used free of charge by NLP practitioners. The BERT is built using transformer encoder blocks. BERT's key technical innovation is applying the bidirectional training of Transformer, a popular attention model, to language modelling. This contrasts with previous efforts which looked at a text sequence either from left to right or combined left-to-right and right-to-left training. The paper's results show that a language model which is bidirectionally trained can have a deeper sense of language context and flow than single-direction language models. In the paper, the researchers detail a novel technique named Masked Language Modeling (MLM) which allows bidirectional training in models in which it was previously impossible. BERT makes use of Transformer, an attention mechanism that learns contextual relations between words (or sub-words) in a text. To help the model distinguish between the two sentences in training, the input is processed in the following way before entering the model:

1. A [CLS] token is inserted at the beginning of the first sentence and a [SEP] token is inserted at the end of each sentence.
2. A sentence embedding indicating Sentence A or Sentence B is added to each token. Sentence embeddings are similar in concept to token embeddings with a vocabulary of 2.
3. A positional embedding is added to each token to indicate its position in the sequence. The concept and implementation of positional embedding are presented in the Transformer paper.

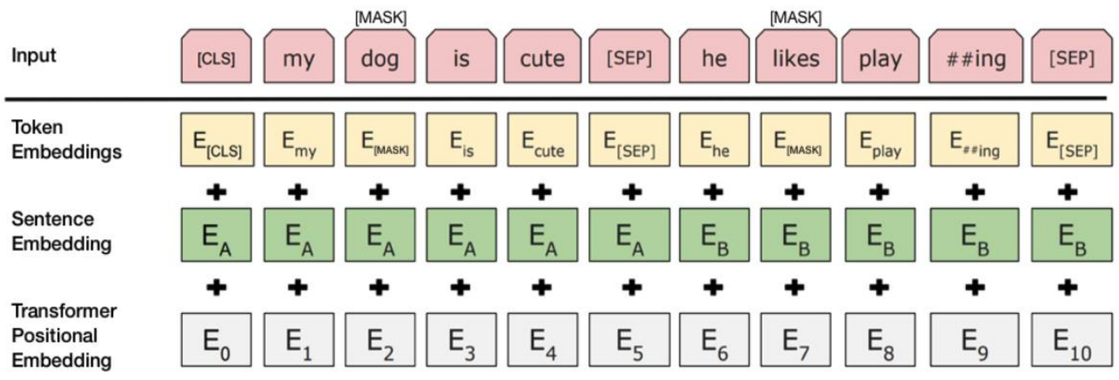
Input
[MASK]
[MASK]
[CLS] my dog is cute [SEP] he likes play ##ing [SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} $E_{[MASK]}$ E_{is} E_{cute} $E_{[SEP]}$ E_{he} $E_{[MASK]}$ E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Sentence Embedding
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Transformer Positional Embedding
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Fig 1.2: BERT Embedding mechanism

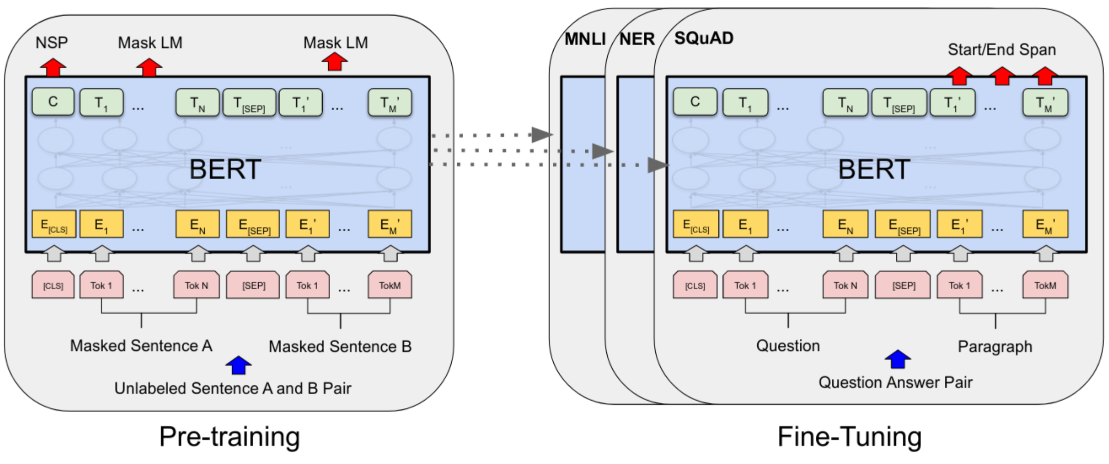

Fig 1.3: BERT Pre-training and Fine Tuning

- **GPT-2**

The OpenAI GPT-2 exhibited impressive ability of writing coherent and passionate essays that exceed what we anticipated current language models can produce. The GPT-2 wasn't a particularly novel architecture – its architecture is very similar to the decoder-only transformer. The GPT2 was, however, a very large, transformer-based language model trained on a massive dataset. The GPT-2 is built using transformer decoder blocks. GPT2, like traditional language models, outputs one token at a time. The way these models work is that after each token is produced, that token is added to the sequence of inputs. And that new sequence becomes the input to the model in its next step. This is an idea called "auto-regression". One key difference in the self-attention layer here, is that it masks future tokens – not by changing the word to [mask] like BERT, but by interfering in the self-attention calculation blocking information from tokens that are to the right of the position being calculated. It's important that the distinction between self-attention (what BERT uses) and masked self-attention (what GPT-2 uses) is clear. A normal self-attention block allows a position to peak at tokens to its right. Masked self-attention prevents that from happening:

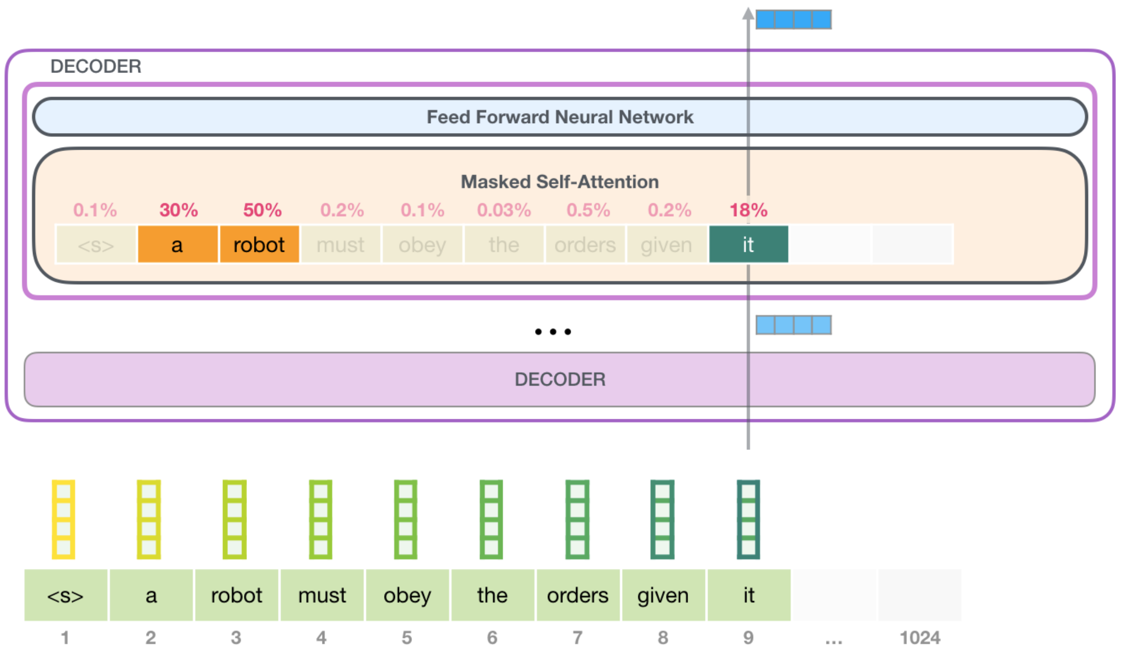

Figure 1.4: A GPT-2 example with Masked Self-Attention

- **XLNet**

  XLNet integrates ideas from Transformer-XL, the state-of-the-art autoregressive model into pretraining. Transformer is a model used for language translation purposes by google. It basically revolves around “attention”. It is an encoder-decoder model where you map one sequence to another.

  The decoder weighs each of the hidden states of the encoder to know which hidden state it should look up at any point. The weights are determined by a simple feed forward neural network. These are called attention weights, or values in the terminology of the paper. XLNet is “generalized” because it captures bi-directional context by means of a mechanism called “permutation language modeling” or PLM. PLM is the idea of capturing bidirectional context by training an autoregressive model on all possible permutation of words in a sentence. Instead of fixed left-right or right-left modeling, XLNet maximizes expected log likelihood over all possible permutations of the sequence.

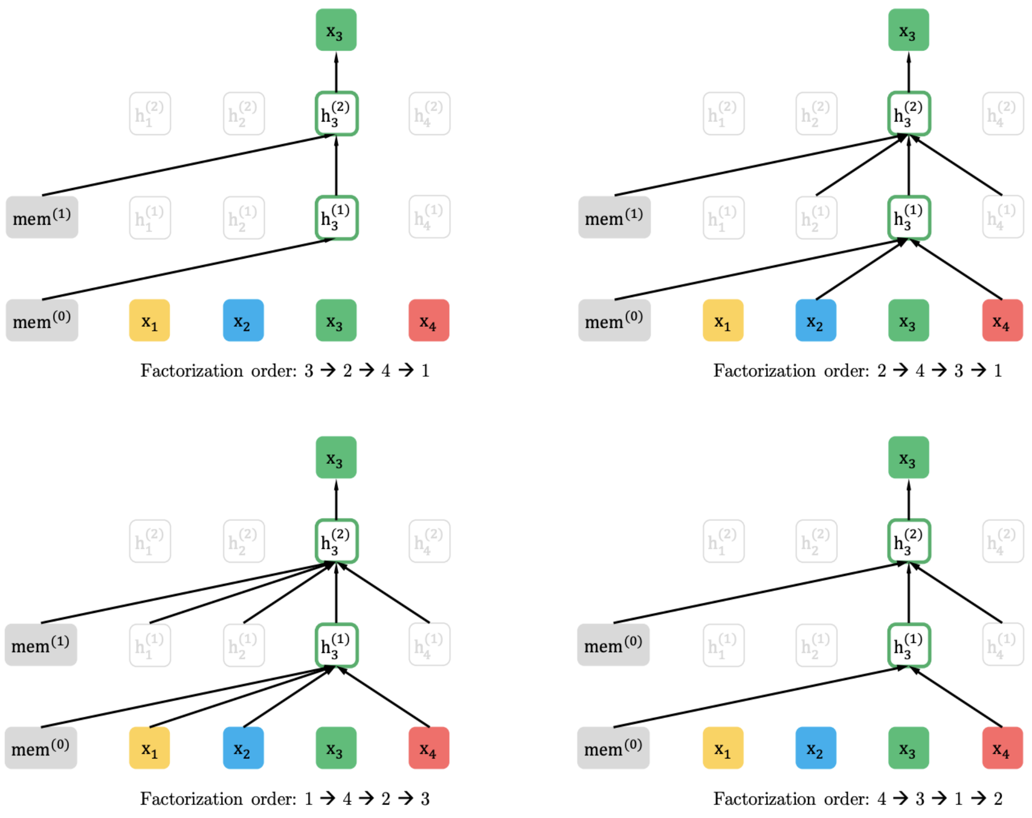
x_3
$h_1^{(2)}$ $h_2^{(2)}$ $h_3^{(2)}$ $h_4^{(2)}$
mem$^{(1)}$ $h_1^{(1)}$ $h_2^{(1)}$ $h_3^{(1)}$ $h_4^{(1)}$
mem$^{(0)}$ x_1 x_2 x_3 x_4
Factorization order: 3 → 2 → 4 → 1
Factorization order: 2 → 4 → 3 → 1
Factorization order: 1 → 4 → 2 → 3
Factorization order: 4 → 3 → 1 → 2

Figure 1.5: Illustration of the permutation language modeling objective for predicting x_3 given the same input sequence x but with different factorization order in XLNet.

1.2.2 EMBEDDINGS:

Embedding is the process of mapping a natural language element into its corresponding vector in the vector space such that it captures the syntactic and semantic similarity relation with other elements in the space. Applications in image and audio domain encode the data into highly dense vectors. However, treating natural language words as discrete symbols will result in a sparse vector which makes it difficult for statistical model training. Embeddings are being used at word, sentence and document level. However, the latter ones are mostly derived from the word embedding vectors.

- **BERT Embedding**

BERT (Bidirectional Encoder Representations from Transformers), is an encoder-based model released at the end of 2018. We will use BERT to extract characteristics from text information, namely word and sentence embedding vectors. What can we do with these vector embedding words and phrases? These embeddings are useful for the expansion of the keyword/query, semantic search, and retrieval of data.

Additionally, and perhaps more significantly, these vectors are used in downstream models as high-quality function inputs. NLP models such as LSTMs or CNNs require numerical vector inputs, usually converting features such as vocabulary and vocabulary sections into numerical representations. Words have been interpreted in the past either as specific indexed values (one-hot encoding) or as neural word embedding where vocabulary words are matched against fixed-length embedding features resulting from models such as Word2Vec or FastText. For example, given two sentences: "The man was charged with stealing a bank," and "The man went fishing by the river bank." Word2Vec would generate the same word embedding in both sentences for the word "bank," although the word embedding for "bank" in each sentence would be different under BERT. In addition to capturing obvious differences such as polysemy, the context-informed word embedding captures other types of information resulting in more accurate representations of features, resulting in better performance of the model.

To start with, each input embedding is a combination of 3 embeddings:

Position Embeddings: BERT learns and uses positional embedding to express the word position in a sentence. These are introduced to address the Transformer limitation that, unlike an RNN, cannot capture information about "line" or "order".

Segment Embedding: BERT may also take sentence pairs as task inputs (Question-Answering). That's why it learns a special embedding to help the model differentiate between the first and second sentences.

Token Embeddings: These are the embeddings from the WordPiece token vocabulary acquired for the particular token.

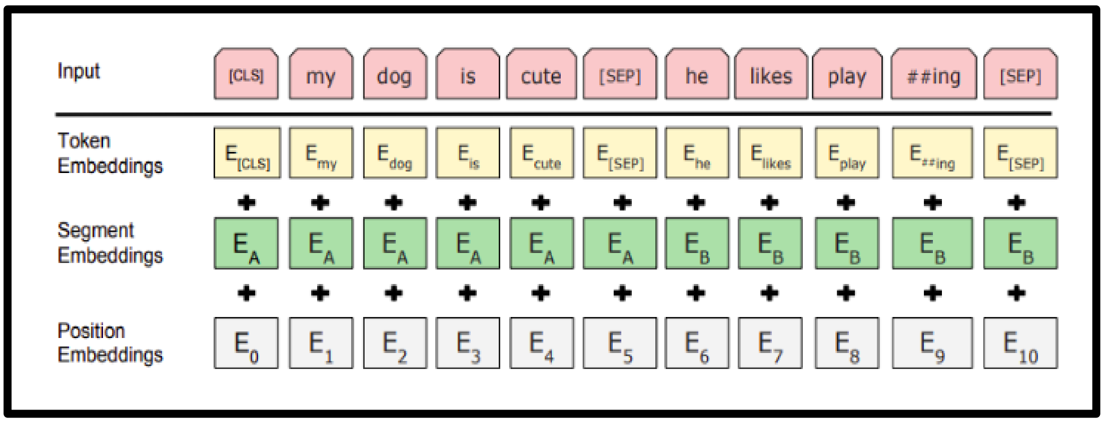
Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Figure 1.6: BERT Embedding mechanism

- **Doc2Vec Embedding**

  Paragraph Vector (more popularly known as Doc2Vec) is supposed to be an extension to Word2Vec such that Word2Vec learns to project words into a latent d-dimensional space whereas Doc2Vec aims at learning how to project a document into a latent d-dimensional space. The basic idea behind PV-DM is inspired from Word2Vec. In the CBOW model of Word2Vec, the model learns to predict a center word based on the context.

  Let's look at the model diagram for some more clarity. In the given model, we see Paragraph Matrix, Average/Concatenate and Classifier sections. Paragraph matrix is the matrix where each column represents the vector of a paragraph. By Average/Concatenate, it means whether the word vectors and paragraph vector are averaged or concatenated. Lastly, the Classifier part takes the hidden layer vector (the one that was concatenated/averaged) as input and predicts the centre word.

  Matrix D has the embeddings for "seen" paragraphs (i.e. arbitrary length documents), the same way Word2Vec models learns embeddings for words. For unseen paragraphs, the model is again ran through gradient descent (5 or so iterations) to infer a document vector.

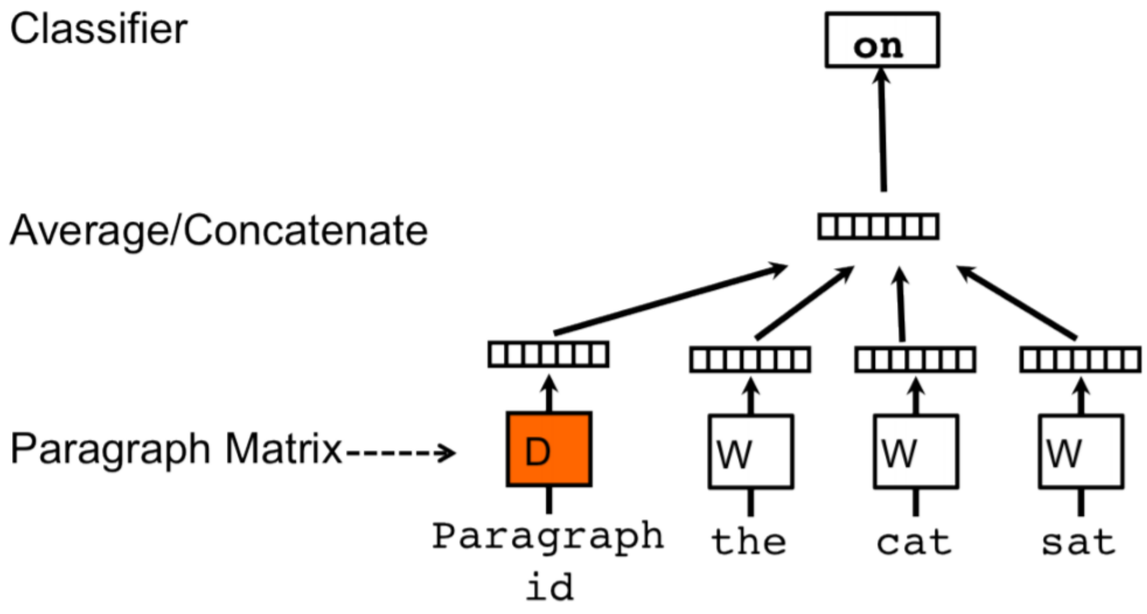

Figure 1.7: Doc2Vec Embedding example

1.2.3 CLUSTERING:

Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters). It is a main task of exploratory data mining, and a common technique for statistical data analysis, used in many fields, including pattern recognition, machine learning, image analysis, information retrieval, bioinformatics, data compression and computer graphics.

- **K-Means Clustering**

  K-Means is an unsupervised learning technique in machine learning. It is a centroid based clustering method. A centroid is the imaginary or real location representing the center of the cluster. Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares. In other words, the K-means algorithm identifies *k* number of centroids, and then allocates every data point to the nearest cluster, while keeping the centroids as small as possible. The 'K' in K-means is a hyperparameter which refers to the number of clusters or regions and the 'means' in the K-means refers to averaging of the data; that is, finding the centroid.

- **DBSCAN Clustering**

  Density-based spatial clustering of applications with noise (DBSCAN) is a data clustering algorithm. It is a density-based clustering non-parametric algorithm: given a set of points in some space, it groups together points that are closely packed together (points with many nearby neighbors), marking as outliers points that lie alone in low-density regions (whose nearest neighbors are too far away). DBSCAN is one of the most common clustering algorithms and also most cited in scientific literature.

1.2.4 DIMENSIONALITY REDUCTION:

In machine learning, there are often too many factors based on which the final result is achieved. These factors are basically variables called features. The higher the number of features, the harder it gets to visualize the training set and then work on it. Sometimes, most of these features are correlated, and hence redundant. This is where dimensionality reduction algorithms come into play. Dimensionality reduction is the process of reducing the number of random variables under consideration, by obtaining a set of principal variables. It can be divided into feature selection and feature extraction.

- **Principal Component Analysis (PCA)**

This method was introduced by Karl Pearson. It works on a condition that while the data in a higher dimensional space is mapped to data in a lower dimension space, the variance of the data in the lower dimensional space should be maximum.

- **t-distributed Stochastic Neighbor Embedding (t-SNE)**

t-SNE is a nonlinear dimensionality reduction technique that is well suited for embedding high dimension data into lower dimensional data (2D or 3D) for data visualization. Stochastic is not definite but random probability, Neighbour is concerned only about retaining the variance of neighbour points, and Embedding is plotting data into lower dimensions

In short, t-SNE is a machine learning algorithm that generates slightly different results each time on the same data set, focusing on retaining the structure of neighbour points.

1.3 PROBLEM STATEMENT

This project deals with two major parts:

A) Collecting unique news data from various news websites. Transforming the data by cleaning and preprocessing. Finding the summary of the news. The main objective is summarization of the news articles of more than 20 lines into 3-4 lines using various state-of-the art models and techniques.

B) Document clustering and news recommendation is the other major part of this project. Here, we will use various embedding techniques to convert the summaries into vector form and then apply clustering on those vectors. Instead of using the whole news article for clustering, we will use summary from BERT, GPT-2 and XLNet to cluster similar news items and use that for recommendation.

1.4 REPORT OUTLINE

This chapter summarized the scope of the work, an introduction to some of the natural language processing techniques (embeddings) that are used in this work. The rest of the report is organized as follows:

- Chapter 2 presents the already existing solutions and their drawbacks.
- Chapter 3 presents the proposed solution.
- Chapter 4 presents the implementation details along with the results obtained.
- Chapter 5 concludes the work done and presents the possible future work.

CHAPTER 2
LITERATURE SURVEY

This section discusses some of the earlier works related to Transformers, Attention Mechanism and Transformer based models as well as some traditional models. The main objective here is to leverage the state-of-the art models and combine them to find the best possible models for our project.

Transformers

In paper Attention is all you need [1], transformers were first introduced. Transformer is an encoder-decoder based model which completely relies on the Attention Mechanism, dispensing with recurrence and convolutions entirely.

Encoder: The encoder is composed of a stack of N = 6 identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position wise fully connected feed-forward network. It employs a residual connection around each of the two sub-layers, followed by layer normalization.

Decoder: The decoder is also composed of a stack of N = 6 identical layers. In addition to the two sub-layers in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack. Similar to the encoder, it employs residual connections around each of the sub-layers, followed by layer normalization. We also modify the self-attention sub-layer in the decoder stack to prevent positions from attending to subsequent positions. As we are not directly taking anything, so I encourage readers to go through the paper [1] for more architectural details.

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

BERT was proposed in the paper [2]. BERT structure includes two steps: **pre-training** and **fine-tuning**. The model is trained on unlabeled data over various pre-training tasks during pre-training. The BERT model is initialized with the pre-trained parameters for fine-tuning, and all parameters are fine-tuned using marked downstream tasks information. Each downstream function has different fine-tuned models, although with the same pre-trained parameters they are initialized.

BERT's integrated architecture across different tasks is a distinctive feature. The pre-

trained architecture and the final supervised architecture are minimally different.
The software architecture of BERT is a bidirectional multi-layer transformer encoder based on the original implementation defined in this paper [2] and published in the library of tensor2tensor.

BERT is the current state-of-the-art method to represent words and sentences in a way that best captures the underlying semantics and relations. BERT was trained in a language modeling setting, with the help of a deep bidirectional network.

Text Summarization with Pretrained Encoders

This paper [3] showcases how BERT can be usefully applied in text summarization and propose a general framework for both extractive and abstractive models. It introduces a novel document-level encoder based on BERT which can express the semantics of a document and obtain representations for its sentences. There extractive model is built on top of this encoder by stacking several inter-sentence Transformer layers. For abstractive summarization, they propose a new fine-tuning schedule which adopts different optimizers for the encoder and the decoder as a means of alleviating the mismatch between the two.
They design a new training schedule which separates the optimizers of the encoder and the decoder in order to accommodate the fact that the former is pretrained while the latter must be trained from scratch. Finally, motivated by previous work showing that the combination of extractive and abstractive objectives can help generate better summaries (Gehrmann et al., 2018 [17]), they present a two-stage approach where the encoder is fine-tuned twice, first with an extractive objective and subsequently on the abstractive summarization task.
Each token wi is assigned three kinds of embeddings: token embeddings indicate the meaning of each token, segmentation embeddings are used to discriminate between two sentences (e.g., during a sentence-pair classification task) and position embeddings indicate the position of each token within the text sequence. These three embeddings are summed to a single input vector xi and fed to a bidirectional Transformer with multiple layers:

$$\tilde{h}^l = \mathrm{LN}(h^{l-1} + \mathrm{MHAtt}(h^{l-1})) \qquad (1)$$

$$h^l = \mathrm{LN}(\tilde{h}^l + \mathrm{FFN}(\tilde{h}^l)) \qquad (2)$$

where h0 = x are the input vectors; LN is the layer normalization operation (Ba et al., 2016 [18]); MHAtt is the multi-head attention operation (Vaswani et al., 2017 [1]); superscript l indicates the depth of the stacked layer. On the top layer, BERT will generate an output vector ti for each token with rich contextual information.

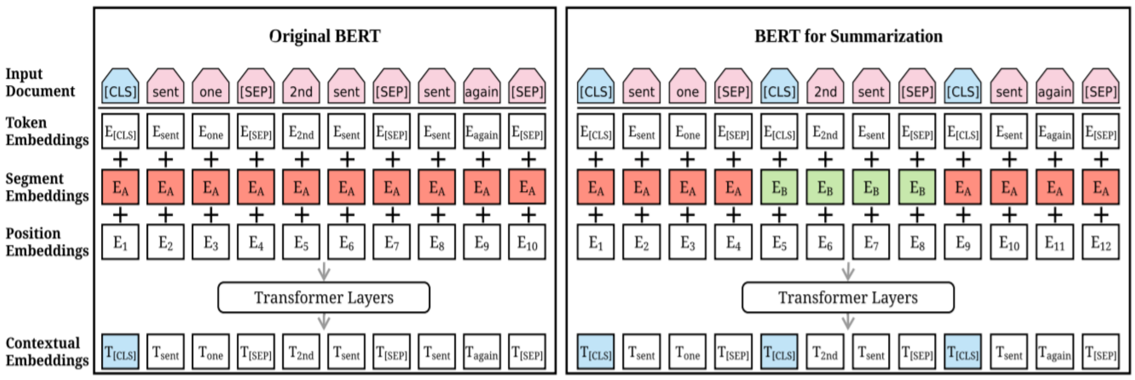

Fig 2.1: Embedding comparison of Original BERT and BERT for Summarization

Language Models are Unsupervised Multitask Learners

This paper [6] introduces GPT-2 with around 1.5B parameters. Natural language processing tasks, such as question answering, machine translation, reading comprehension, and summarization, are typically approached with supervised learning on task specific datasets. They demonstrate that language models begin to learn these tasks without any explicit supervision when trained on a new dataset of millions of webpages called WebText. They use a Transformer (Vaswani et al., 2017 [1]) based architecture for their LMs. The model largely follows the details of the OpenAI GPT model with a few modifications. Layer normalization (Ba et al., 2016 [18]) was moved to the input of each sub-block, similar to a pre-activation residual network and an additional layer normalization was added after the final self-attention block. A modified initialization which accounts for the accumulation on the residual path with model depth is used. They scale the weights of residual layers at initialization by a factor of 1/ √ N where N is the number of residual layers. The vocabulary is expanded to 50,257. They also increase the context size from 512 to 1024 tokens and a larger batch size of 512 is used.

Transformer-XL

Transformers have a potential of learning longer-term dependency but are limited by a fixed-length context in the setting of language modeling. They propose a novel neural architecture Transformer-XL that enables learning dependency beyond a fixed length without disrupting temporal coherence. It consists of a segment-level recurrence mechanism and a novel positional encoding scheme. Their method not only enables capturing longer-term dependency, but also resolves the context fragmentation problem. As a result, Transformer-XL learns dependency that is 80% longer than RNNs and 450% longer than vanilla Transformers, achieves better performance on both short and long sequences, and is up to 1,800+ times faster than vanilla Transformers during evaluation. The limitations of fixed-length contexts in various models, they propose a new architecture called Transformer-XL (meaning extra-long) [4]. They introduce the notion of recurrence into their deep self-attention network. Instead of computing the hidden states from scratch for each new segment, they reuse the hidden states obtained in previous segments. The reused hidden states serve as memory for the current segment, which builds up a recurrent connection between the segments. As a result, modeling very long-term dependency becomes possible because information can be propagated through the recurrent connections. Meanwhile, passing information from the previous segment can also resolve the problem of context fragmentation.

XLNet

XLNet [5], a generalized autoregressive pretraining method that enables learning bidirectional contexts by maximizing the expected likelihood over all permutations of the factorization order and overcomes the limitations of BERT thanks to its autoregressive formulation. Furthermore, XLNet integrates ideas from Transformer-XL [4], the state-of-the-art autoregressive model, into pretraining. Typically, these methods first pretrain neural networks on large-scale unlabeled text corpora, and then finetune the models or representations on downstream tasks. Under this shared high-level idea, different unsupervised pretraining objectives have been explored in literature. Among them, autoregressive (AR) language modeling and autoencoding (AE) have been the two most successful pretraining objectives. XLNet, a generalized autoregressive method that leverages the best of both AR language modeling and AE while avoiding their limitations.

Firstly, instead of using a fixed forward or backward factorization order as in conventional AR models, XLNet maximizes the expected log likelihood of a sequence w.r.t. all possible permutations of the factorization order. Secondly, as a generalized AR language model, XLNet does not rely on data corruption. Hence, XLNet does not suffer from the pretrain-finetune discrepancy that BERT is subject to.

2.1 SUMMARY

This chapter enlisted the recent works and techniques related to language modeling task and other various Natural Language specific task. A few of these models were used to build my proposed model.

There are various approaches and models related language modeling task, so our main idea is to leverage the state-of-the art models and combine it along with some traditional models to build something novel.

CHAPTER 3
PROPOSED WORK

As already mentioned, the whole project is divided into two major parts, first is summarization and second is document clustering and recommendation. In the first part, as we know that most of the state-of-the art models that are present and are in use has been trained for days with several GPUs and TPUs in distributed environment by giant companies. Like BERT was created by people at Google, training of BERTBASE (110M parameters) was performed on 4 Cloud TPUs in Pod configuration (16 TPU chips total) and training of BERTLARGE (340M parameters) was performed on 16 Cloud TPUs (64 TPU chips total). Each pretraining took 4 days to complete. In contrast GPT-2 has 1.5B parameters. Now, creating something of our own in this field might take a huge toll on everything and it might not be quite possible too, but instead what we can do is to make use of these extraordinary models for our task and find a combination of these along with traditional models to best suit our model.

3.1 METHODOLOGY:

3.1.1 News Summarization

BERT is an encoder-based model, GPT-2 is a decoder-based model, and XLNet is encoder-decoder based model. The major feature of these models is that all these models extract unique kind of details from provided languages. Unlike earlier RNN models like LSTM or GRU, these models are completely based on transformers and attention mechanism. Each of these models single handedly have the capacity to defeat any RNN models in almost every NLP task. What we will do here is instead of building our model from scratch we will use all of three models and find the summary one by one. Since we don't have any set of reference for summary or gold standard for this unsupervised task, as we have collected our own data from various websites to make it more authentic so we will have to just keep the summaries and see how it works in later task like document clustering and recommendation task.

Extractive summarization is a challenging task that has only recently become practical.

Like many things NLP, one reason for this progress is the superior embeddings offered by transformer models like BERT. The library bert-extractive-summarizer uses BERT sentence embeddings to build an extractive summarizer taking two supervised approaches. The first considers only embeddings and their derivatives. This corresponds to our intuition that a good summarizer can parse meaning and should select sentences based purely on the internal structure of the article. The baseline for this approach is the unsupervised TextRank model. The other approach incorporates sequential information and takes advantage of the well-known Lead3 phenomena particular to news corpuses. This is the observation that the first three sentences typically do a good job in summarizing the article. In fact, this strategy is explicitly deployed by many publishers. Lead3 is used as the baseline for this second approach.

GPT-2 is essentially a decoder-only transformer. The model is built by stacking up the transformer decoder blocks. Based on the number of layers, there are four variants of GPT-2- 117M, 345M, 762M, and 1542M. Unlike the self-attention that transformers use, GPT-2 uses **masked self-attention.** A normal self-attention block allows a position to peak at tokens to its right. Masked self-attention prevents that from happening, which means that they only use the left context to predict the next word. And for the tokenization of inputs, GPT-2 uses **byte pair encoding(BPE).** BPE is a simple form of data compression in which the most common pair of consecutive bytes of data is replaced with a byte that does not occur within that data. BPE is a middle ground between character and word-level encodings which helps it in managing the vocabulary of large corpora. This behavior also enables the encoding of any rare words in the vocabulary with appropriate subword tokens without introducing any "unknown" tokens. In the paper *Fine-Tuning Language Models from Human Preferences* that I talked about earlier, it is shown how the GPT-2 774M model was fine-tuned to summarize texts according to human preferences. The model was trained by combining supervised fine-tuning with human fine-tuning on 60k labels. As a result, the summaries from the supervised fine-tuned version of GPT-2 are more novel as measured by n-grams or sentences, they are also more novel in terms of content. That is, they're not just copying from the input text.

XLNet and TransformerXL are the two recurrent language models currently available in the Transformers NLP library. "Recurrent" in this context means that they were designed to model extremely long sequences by breaking those sequences into chunks and processing them one at a time. The chunks are then tied together via a "memory" that is recursively passed between from forward pass to the next. XLNET is an abstractive summarizer. XLNet is particularly interesting for language generation because it is pre-trained in a regressive manner similar to the GPT family of models. This holds the promise for more coherent text output than what you would typically find with MLM models like Transformer XL. And that is what makes it more special for abstractive summarization. In abstractive summarization we just don't copy paste whatever is present, instead we create new summary by understanding the whole context of the text.

In this part we will extract summary using these three models and later on use those summaries for recommendation purpose, because the more context a summary covers the better recommendation will occur.

3.1.2 Document Clustering

In this part, we will use the summary extracted from BERT, GPT-2, and XLNet in order to cluster the various news articles, using two clustering technique, one where we need to imply the number of clusters i.e. K-Means clustering and the other where we don't need to mention the cluster size DBSCAN. We will also use two embedding technique where one of them being state-of-the art BERT Embedding and other being traditional model Doc2Vec Embedding and along with this we will experiment the whole setup with and without dimensionality reduction, we will use two dimension reduction techniques Principal Component Analysis (PCA) and t-SNE.

3.2 EXPERIMENT SETUP

We will use various combination of summaries with embedding techniques, dimensionality reduction and clustering. K-Means clustering supports clustering even when the data's dimension is too high, so we are going to experiment with and without dimensionality reduction in K-Means but in case of DBSCAN we must use dimensionality reduction. Following is the list of experimental setups on the proposed architecture:

Model Type	**Stacked Models**
BERT Summary + Embedding + Clustering	BERT Summary + Doc2Vec + K-Means
	BERT Summary + BERT EMBEDDING + K-Means
BERT Summary + Embedding + Dimensionality Reduction + Clustering	BERT Summary + Doc2Vec + PCA + K-Means
	BERT Summary + BERT Embedding + PCA + K-Means
	BERT Summary + Doc2Vec + t-SNE + K-Means
	BERT Summary + BERT Embedding + t-SNE + K-Means
	BERT Summary + Doc2Vec + PCA + DBSCAN
	BERT Summary + BERT Embedding + PCA + DBSCAN
GPT-2 Summary + Embedding + Clustering	GPT-2 Summary + Doc2Vec + K-Means
	GPT-2 Summary + BERT EMBEDDING + K-Means
GPT-2 Summary + Embedding + Dimensionality Reduction + Clustering	GPT-2 Summary + Doc2Vec + PCA + K-Means
	GPT-2 Summary + BERT Embedding + PCA + K-Means
	GPT-2 Summary + Doc2Vec + t-SNE + K-Means
	GPT-2 Summary + BERT Embedding + t-SNE + K-Means

	GPT-2 Summary + Doc2Vec + PCA + DBSCAN
	GPT-2 Summary + BERT Embedding + PCA + DBSCAN
XLNet Summary + Embedding + Clustering	XLNet Summary + Doc2Vec + K-Means
	XLNet Summary + BERT EMBEDDING + K-Means
XLNet Summary + Embedding + Dimensionality Reduction + Clustering	XLNet Summary + Doc2Vec + PCA + K-Means
	XLNet Summary + BERT Embedding + PCA + K-Means
	XLNet Summary + Doc2Vec + t-SNE + K-Means
	XLNet Summary + BERT Embedding + t-SNE + K-Means
	XLNet Summary + Doc2Vec + PCA + DBSCAN
	XLNet Summary + BERT Embedding + PCA + DBSCAN

Table 3.1: List of stacked models experimented

3.3 DATASET

Dataset for this project was gathered and collected by me using various tools, and the reason behind doing so was that I wanted to make this app more authentic and real-time for Indian audience. There are lots of dataset for news present, but I choose not to go with them because they were collected from either overseas news agencies or it didn’t had the format that I required it to be in.

In total, I collected around 9000 unique news articles from 6 different websites namely ANI, Hindustan Times, Indian Express, NDTV, Times Now, and Times of India. The news gathered from these websites were collected in various categories like Business, Politics, Entertainment, Sports, World, and India. Initially, I collected URLs from these websites and used BeautifulSoup, Newspaper3k, and NLTK library to extract articles and other features from the news URLs.

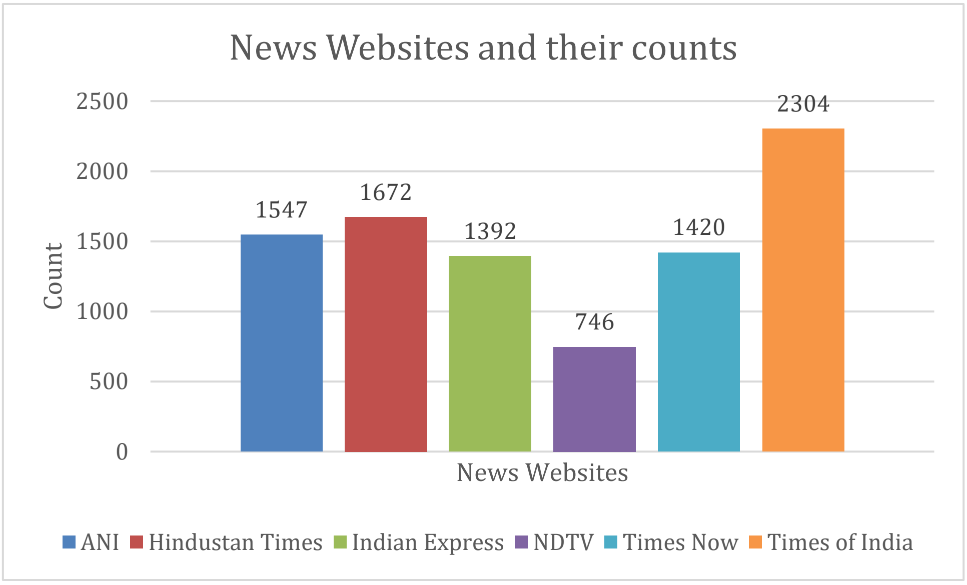

Figure 3.1: Count of news for each website in the dataset

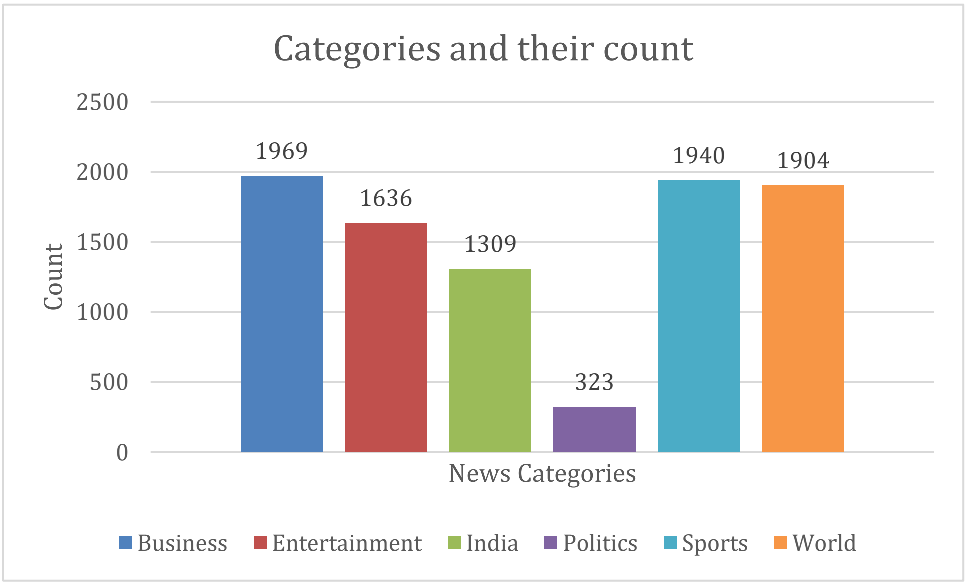

Figure 3.2: Count of news for each category in the dataset

NOTE: The reason that politics news is so low is because generally websites don’t categories them into politics separately, instead they include them in India category.

3.4 SUMMARY

This chapter enlisted the details of the proposed framework of applying different recent state-of-the art summarization techniques with recent and classical embeddings and various other methodologies in the area of NLP which are applied to the news articles. It clearly explains all the models used, architecture and the dataset used.

CHAPTER 4
IMPLEMENTATION AND RESULTS

4.1 IMPLEMENTATION DETAILS

The system used for the experiments has the following configuration:

● Processor	Intel i7 9th Generation 2.6 GHz
● RAM	8 GB
● System Type	64-bit
● Operating System	Windows 10
● GPU	NVIDIA GeForce GTX 1650 4 GB

4.2 SOFTWARE ENVIRONMENT

The following software tools and libraries were used:

● IDE	Google Collaboratory
● Programming Language	Python
● Libraries	Numpy, PyTorch, Matplotlib, Pandas, Sklearn, Sentence Transformers, Bert Extractive Summarizer, Spacy, NLTK, Newspaper3k

4.3 MODEL TRAINING

Most of the models that we are using here comes pre-trained and we just have to fine-tune it for our purpose. Models for summarization came pre-trained as it is unsupervised task and we didn't have summary of our own, so we had to use pre-trained models. But there are some which needed training. The Doc2Vec embedding had to be trained, it was trained for 100 epochs vector size of 100 and 0.001 alpha.

We also used number of dimensions as 2 for dimensionality reduction in both PCA and t-SNE. And for K-Means clustering we had 200 maximum iteration, for K values ranging for 10 to 30, we used elbow method to decide the best k value (which is 20 clusters) and then compared the performance. And for DBSCAN we used various eps and minimum sample values according to the requirements. There is not any metrics to see performance of DBSCAN unlike K-Means which can be done using sum of squared distance, so we had to visualize and see how good or bad the results are.

4.4 RESULTS

Following figures indicate the results obtained for different stacked model:

BERT Summary + BERT Embedding + K-Means:

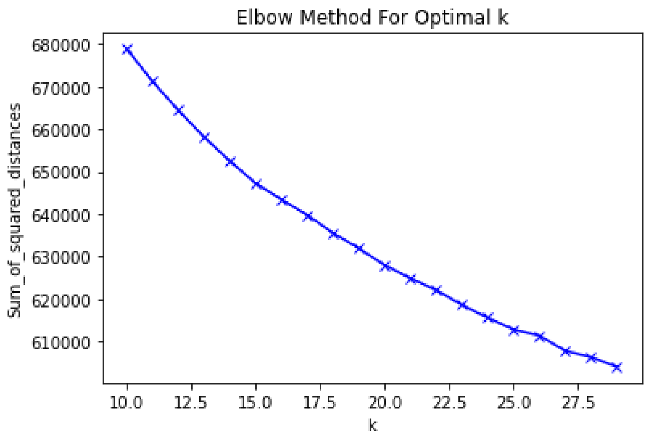

Fig 4.1: Sum of squared distance for various k values

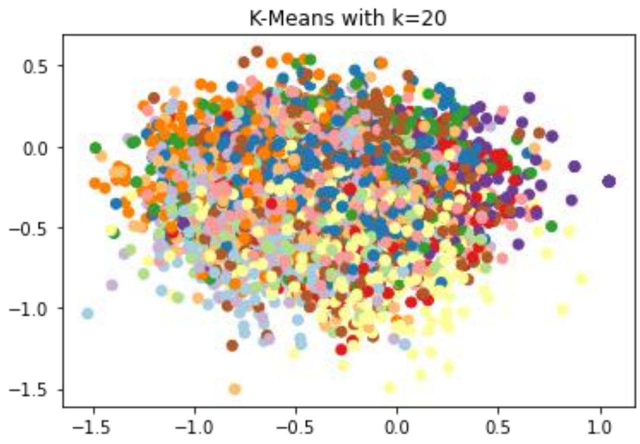

Fig 4.2: Clustering result at k=20 clusters

BERT Summary + Doc2Vec Embedding + K-Means:

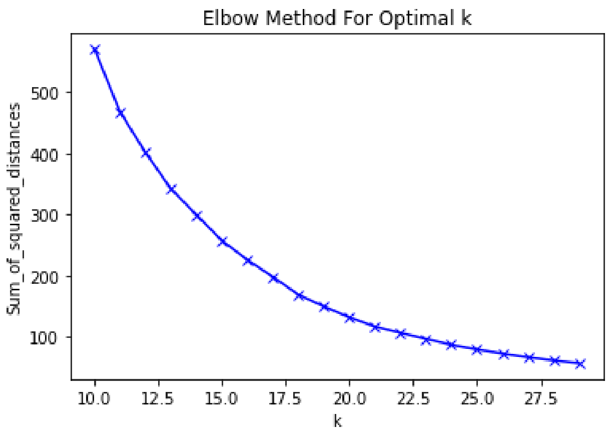

Fig 4.3: Sum of squared distance for various k values

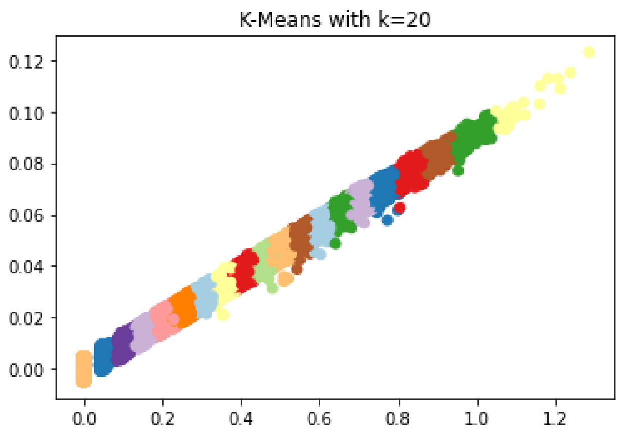

Fig 4.4: Clustering result at k=20 clusters

BERT Summary + BERT Embedding + PCA + K-Means:

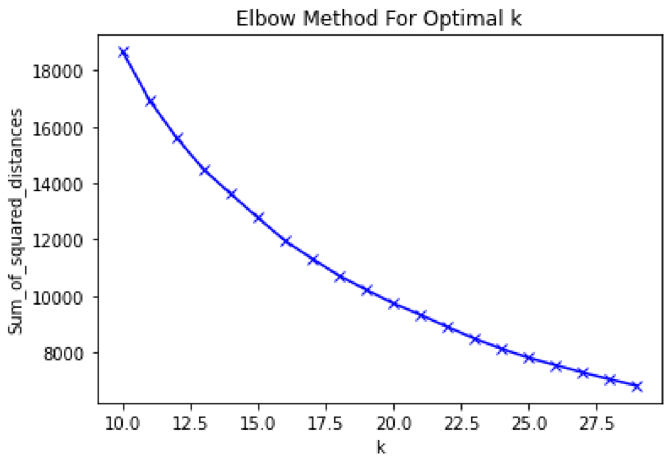

Fig 4.5: Sum of squared distance for various k values

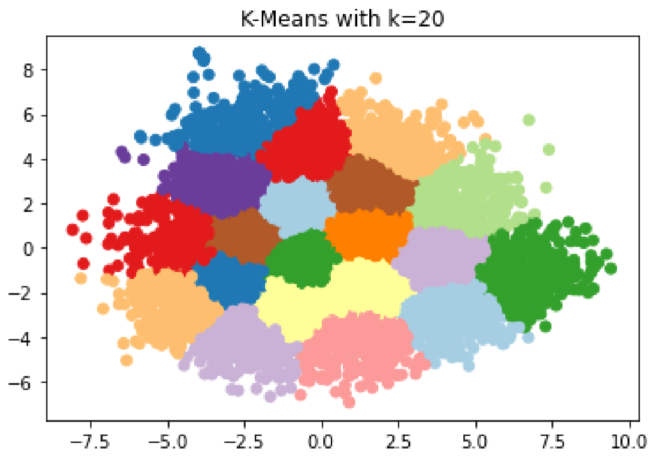

Fig 4.6: Clustering result at k=20 clusters

BERT Summary + Doc2Vec Embedding + PCA + K-Means:

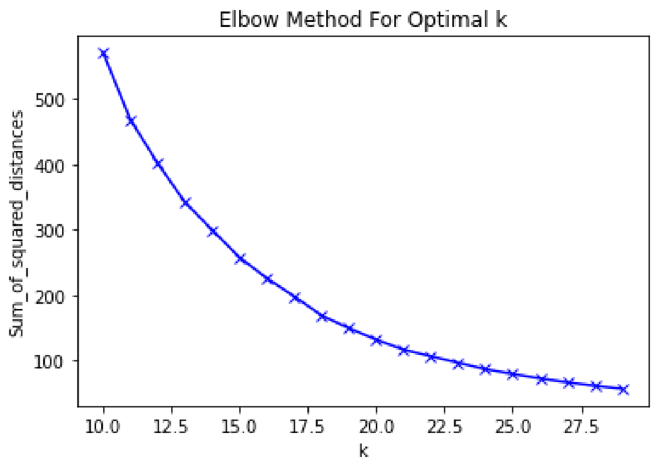

Fig 4.7: Sum of squared distance for various k values

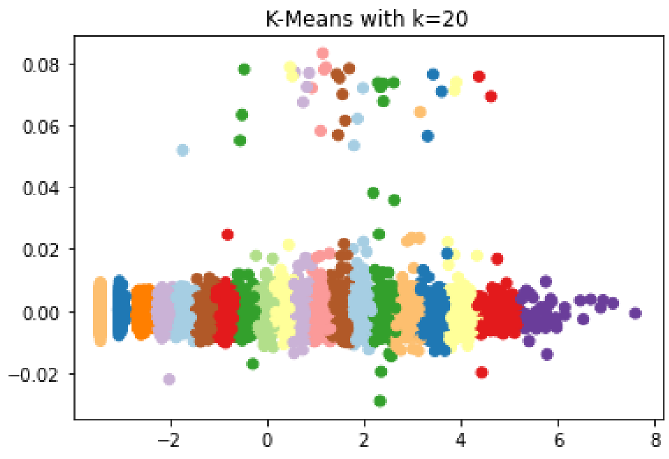

Fig 4.8: Clustering result at k=20 clusters

BERT Summary + BERT Embedding + t-SNE + K-Means:

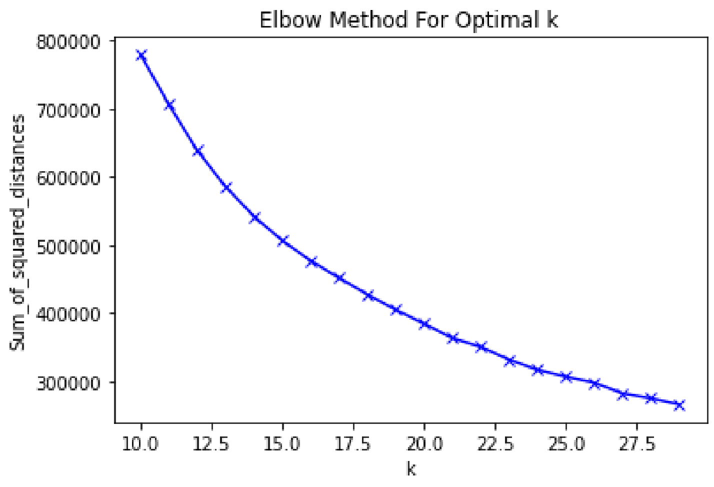

Fig 4.9: Sum of squared distance for various k values

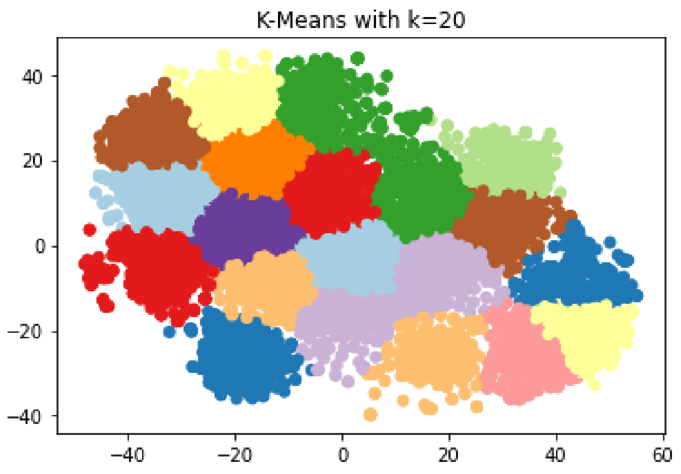

Fig 4.10: Clustering result at k=20 clusters

BERT Summary + Doc2Vec Embedding + t-SNE + K-Means:

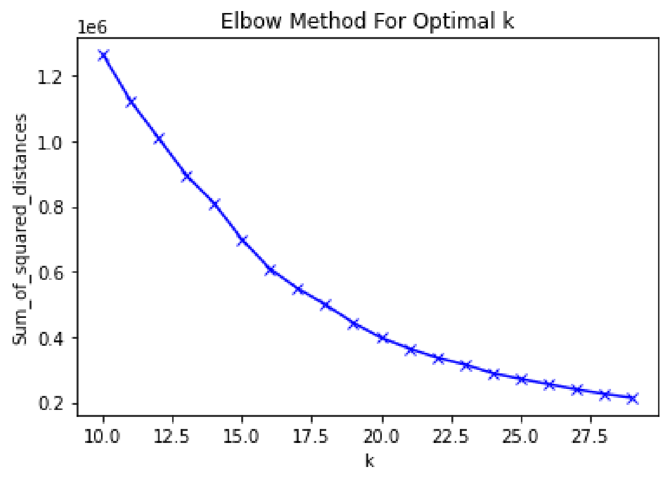

Fig 4.11: Sum of squared distance for various k values

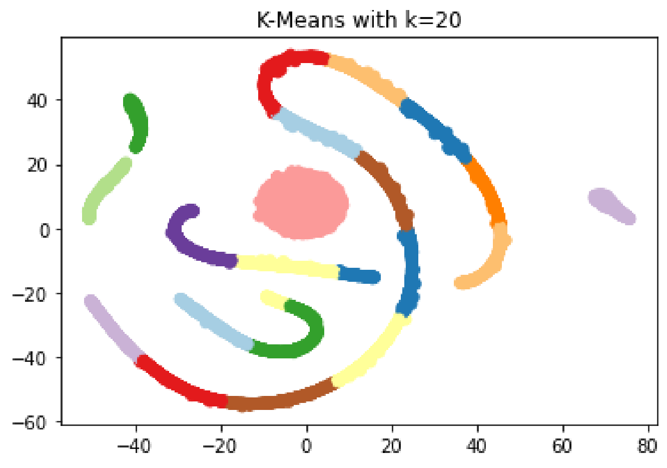

Fig 4.12: Clustering result at k=20 clusters

BERT Summary + BERT Embedding + PCA + DBSCAN:

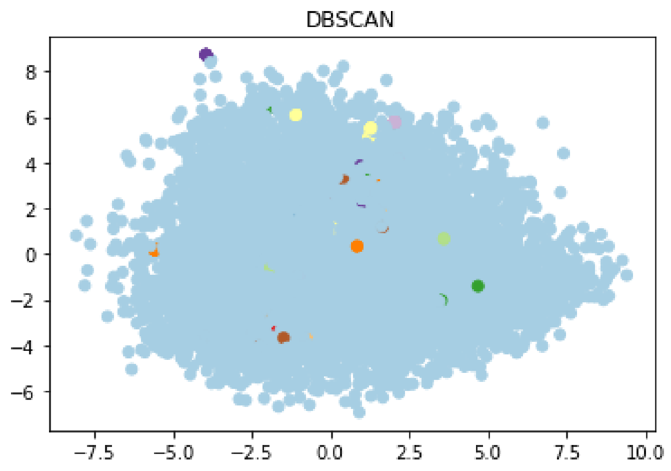

Figure 4.13: DBSCAN Result at eps =0.005, 3, clusters = 15

BERT Summary + Doc2Vec Embedding + PCA + DBSCAN:

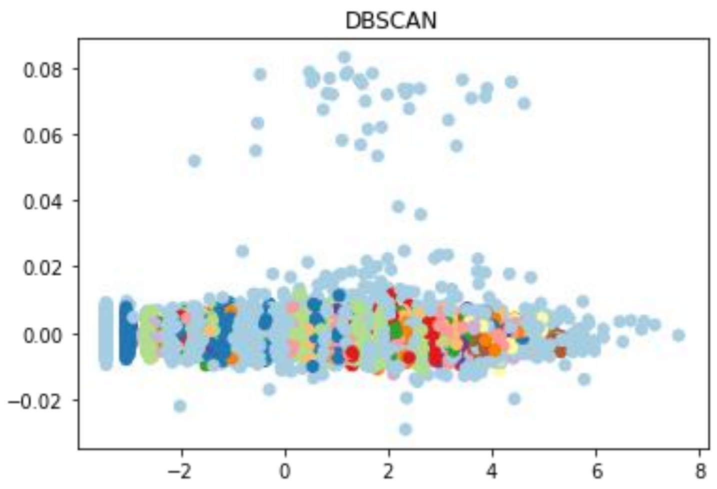

Figure 4.14: DBSCAN result at eps =0.005, 3, clusters = 197

These figures are enough to relate between all the other summaries i.e. GPT-2 and XLNet. As we can see from the figure, DBSCAN is not performing anywhere close to K-Means for our task. So, instead of having all the figures we would provide the result in tabular format for only K-Means clustering.

Stacked Model	Sum of Squared Distance (at k=20)
BERT Summary + BERT Embedding + K-Means	627917.7559826617
BERT Summary + Doc2Vec Embedding + K-Means	138.8381352914954
BERT Summary + BERT Embedding + PCA + K-Means	9708.433495675923
BERT Summary + Doc2Vec Embedding + PCA + K-Means	132.46613718925713
BERT Summary + BERT Embedding + t-SNE + K-Means	384877.6035738
BERT Summary + Doc2Vec Embedding + t-SNE + K-Means	397759.4628152194
GPT-2 Summary + BERT Embedding + K-Means	631419.9000870567
GPT-2 Summary + Doc2Vec Embedding + K-Means	125.06148301365438
GPT-2 Summary + BERT Embedding + PCA + K-Means	10001.23722568379
GPT-2 Summary + Doc2Vec Embedding + PCA + K-Means	114.1691115747464
GPT-2 Summary + BERT Embedding + t-SNE + K-Means	405936.37767738954
GPT-2 Summary + Doc2Vec Embedding + t-SNE + K-Means	374543.4737039017
XLNet Summary + BERT Embedding + K-Means	585212.4296981939
XLNet Summary + Doc2Vec Embedding + K-Means	141.5167325296013
XLNet Summary + BERT Embedding + PCA + K-Means	9582.360340157447
XLNet Summary + Doc2Vec Embedding + PCA + K-Means	133.8402222337754
XLNet Summary + BERT Embedding + t-SNE + K-Means	375852.4768484771
XLNet Summary + Doc2Vec Embedding + t-SNE + K-Means	375280.9844821786

Table 3.2: Result of stacked models experimented

4.5 SUMMARY

This chapter enlisted all the observations and results from the experiments. It results that the combination or stacking of GPT-2 Summary, Doc2Vec Embedding, PCA and K-means clustering outperformed all the other combinations of model resulting in the SSD of 114.17.

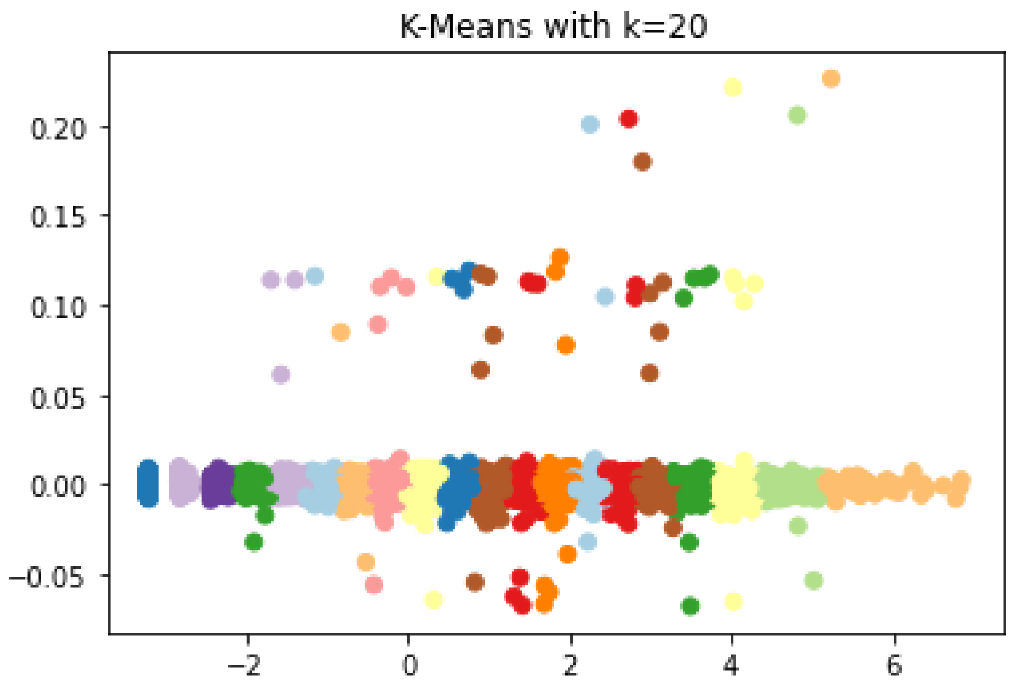

Figure 4.15: Interpretation of the results with stacked GPT-2 Summary, Doc2Vec Embedding, PCA and K-means clustering.

CHAPTER 5

CONCLUSION

We here saw how to leverage the current trending models and combining it with some traditional models gave the result that we wanted. We proposed the stacking of different summaries with different embedding for document clustering with and without different dimensionality reduction techniques. We can clearly establish that always the models which is more complex might not be the best for every task, instead we have to find the models which work for best for our task through various combinations and comparisons.

Future work in this project will include the following:

- I will be adding recommendation system by embedding the summaries together and finding the similarity between various news.
- I will also explore the possibility of using my own pre-trained network for summarization task, but it requires the data to be supervised.
- I would also explore other embedding techniques and combining various embeddings together to see how much improvement can be found by stacking various embeddings together.
- And finally create a full fledge running application with an UI.

REFERENCES

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Proceedings of the 2017 Advances in Neural Information Processing Systems, pages 600–601.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv preprint arXiv:1810.04805.

[3] Yang Liu and Mirella Lapata. 2019. Text Summarization with Pretrained Encoders. arXiv preprint arXiv:1908.08345.

[4] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. arXiv preprint arXiv:1901.02860.

[5] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, Quoc V. Le. 2020. XLNet: Generalized Autoregressive Pretraining for Language Understanding. arXiv preprint arXiv:1906.08237.

[6] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever. Language Models are Unsupervised Multitask Learners. OpenAI blog, 1(8), p.9.

[7] Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, Geoffrey Irving. Fine-Tuning Language Models from Human Preferences. arXiv preprint arXiv:1909.08593v2 8 Jan 2020

[8] Quoc Le, Tomas Mikolov. Distributed Representations of Sentences and Documents. arXiv preprint arXiv:1405.4053 22 May 2014

[9] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. arXiv preprint arXiv:1905.03197.

[10] Sergey Edunov, Alexei Baevski, and Michael Auli. 2019. Pre-trained language model representations for language generation. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4052–4059, Minneapolis, Minnesota.

[11] Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up

abstractive summarization. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 4098–4109, Brussels, Belgium.

[12] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pages 55–60, Baltimore, Maryland.

[13] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. SummaRuNNer: A recurrent neural network based sequence model for extractive summarization of documents. In Proceedings of the 31st AAAI Conference on Artificial Intelligence, pages 3075–3081, San Francisco, California.

[14] Bojanowski, Piotr, et al. 2017. Enriching word vectors with subword information." Transactions of the Association for Computational Linguistics 5, pages 135-146.

[15] Danyang Liu, Ting Bai, Jianxun Lian, Guangzhong Sun, Wayne Xin Zhao, JiRong Wen, and Xing Xie. 2019. News Graph: An Enhanced Knowledge Graph for News Recommendation. In Proceedings of KaRS 2019 Second Workshop on Knowledge-Aware and Conversational Recommender Systems, Beijing, China.

[16] Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010.Word representations: a simple and general method for semi-supervised learning. In Proceedings of the 48th annual meeting of the association for computational linguistics. Association for Computational Linguistics, pp 384–394.

[17] Sebastian Gehrmann Yuntian Deng Alexander M. Rush. 2018. Bottom-Up Abstractive Summarization. arXiv preprint arXiv:1808.10792

[18] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. arXiv preprint arXiv:1607.06450