Course Project Report

Course: Foundations of Machine Learning

Topic: Multidocument Summarization

Team Members:
Aniket Yadav (22M2158)
Raj Gothi (22M2160)
Rahul Kumar (22M1163)

MULTI DOCUMENT SUMMARIZATION

Aniket Yadav, Raj Gothi, Rahul Kumar
IIT Bombay

Abstract: There can be many methods for single document summarization but Multi-Document summarization is considered to be a difficult task. Here we have proposed a architecture for generating summary of similar data taken from different sources. One of the challenge for Multidocument summarization is dataset to be used. There are very little source of dataset for Multidocument Summarization. One we used was multinews dataset from hugging face. Another challenge was to get away from redundant sentences. As we are working with multidocument containing similar content there is high chance for having similar redundant sentences in the summary. Our method took this fact in account and reported final summary by removing redundant sentences.
Keywords: Summarization, Natural Language Processing, K Means Clustering.

INTRODUCTION:

Nowadays, there are many sources of information which provide similar information on a given topic. For example: For news there are various sources and many of them have similar content. Therefore there is a requirement of summarising content from all the sources to reduce time. Our objective was to create a document summarizer for multiple documents on similar topics taken together, such that content from all the sources could be summarised to a single precise summary. The broad view of characteristics of our work are as follows:-

- **INPUT AND OUTPUT**:
  Dataset: Hugging face Multi-news https://huggingface.co/datasets/multi_news
  Multi-News, contains news articles and human-written summaries of these articles.

  There are two features of Dataset:
  - document: text of news articles separated by special token "|||||". Which is the set of multiple documents.
  - summary: Actual news summary.
    Our input output were as follows:-
    Input: Multiple documents (more than 1 document will be passed to input)
    Output: Model gives summary of that Input multiple document.

- **EVALUATION METRIC**:
  - **BERT (Bidirectional Encoder Representations from Transformers) Score**: It calculates precision, recall, and F1 measure. Which is useful for determining the similarity between actual summary and predicted summary using contextual embedding from the BERT model.

- **Rouge (Recall-Oriented Understudy for Gisting Evaluation) Score**: It is used to compare the data generated summary against the given summary. It uses Ngram model for getting comparison scores.

RELATED WORK:

- The sentences of the documents are represented using the Tf.idf method and finding the similarity between a pair of sentences using cosine similarity. where frequent sentences are considered as the centroid of the cluster. Where each cluster's centroid is included in the final summary. But It does not give any intuition about the semantic meaning of each word in sentences.

- If we solve the above problem using representing each sentence as word vector embedding and run the K-mean centroid algorithm then It solved the problem of semantic meaning but It might contain redundant sentences in the summary. Here we always select the centroid sentence of the cluster that might be an unimportant cluster sentence for the summary.

  To overcome these problems we use K-mean, centroid-based method and MMR(Maximal Marginal Relevance) with sentence position.

METHODOLOGY:

1) Extractive summarization:

Extractive summarization identifies the most important sentences from the documents and merges all those sentences together and generates the summary. Here we used K-mean, centroid-based method and MMR(Maximal Marginal Relevance) with sentence position to do Extractive summarization.

2) Abstractive summarization:

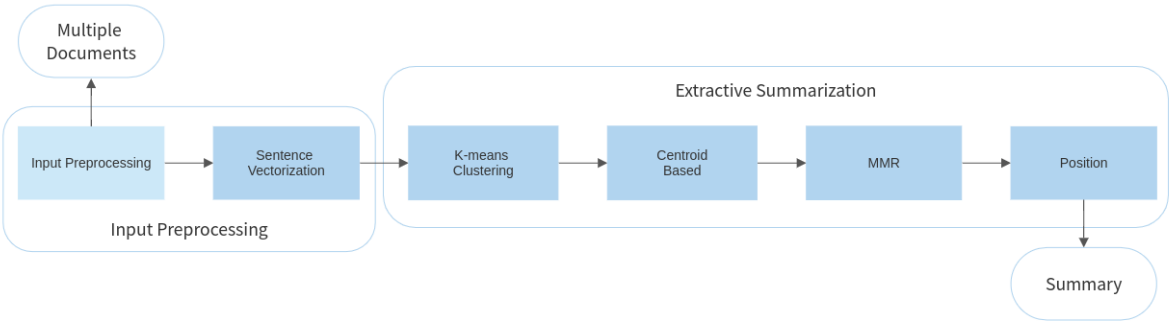
Abstractive summarization generates the summary through understanding the documents semantically and rewriting the novel sentence by either rephrasing or adding new words or adding the important sentences. This summarization can be done using Encoder and decoder Deep learning model.

PREPROCESSING:

Here we process the inputs by removing the stop words, doing word stemming and converting each sentence to vector. Vectorization of the sentences can be done many ways.
The 1st simple approach is using the BoW(Bag of words) model but the problem with this model is that it does not contain the information about the importance of the word in documents. To overcome this problem we can use the tf.idf method (term

frequency-inverse document frequency). But it does not contain the semantic meaning of the sentences. So we have used the Word Embedding representation of a particular word.It can be used to find the semantic relation between other words.

EXTRACTIVE SUMMARIZATION:



CENTROID BASED ALGORITHM:

The K-mean algorithm returns the output as a set of sentences together of the same cluster. This clustered set of sentences can be used to find the most relevant sentences from each cluster using Centroid based algorithm.

Now A sentence vector is represented using the tf.idf method of words in the sentences. A word is centroid which has a tf.idf value greater than some threshold. The sentence containing multiple centroid words will be included in the summary.

Sudo Algorithm:

- Define the centroid vector c: (Aw_1,Aw_2,Aw_3,.....,Aw_n) where n=vocabulary size.

Where each Aw_i = sum($tf.idf_{w,s}$) (Awi is the sum of tf.idf of word w in a set of sentences s from the input documents)

- Calculate the similarity between the sentences vector s and the centroid vector c.

Sim(s,c) = ((1 - cosine(s,c))+1 / 2)

MAXIMAL MARGINAL RELEVANCE:

MMR is used to eliminate the information overlapping among sentences in the summary.

$$MMR \overset{def}{=} Arg \max_{D_i \in C \setminus \{S,Q\}} \left[\lambda \left(Sim_1(D_i, Q) - (1-\lambda) \max_{D_j \in S} Sim_2(D_i, D_j) \right) \right]$$

C is the sentential set that was selected from the previous algorithm, Q is selected from the set C that is the sentence that best described the main idea of input documents, S is the sentential set that included in the summary,

$$Sim_1(u,v) = Sim_2(u,v) = \frac{\sum_{w \in v} tf_{w,u} tf_{w,v} (idf_w)^2}{\sqrt{\sum_{w \in u} (tf_{w,u} idf_w)^2}}$$

where u, v are two sentences that we need to calculate the similarity, $tf_{w,u}$ is the term frequency of the word w in the sentence u, idf_w is the inverse document frequency of the word w.

The final summary is generated with a reasonable order based on sentence positions.

IMPLEMENTATION DETAILS:

We have implemented the extractive method of text summarization. Here we have taken multiple documents as input and produced a summary. The APIs/libraries used by us are as follows:

- TfidfTransformer from sklearn
- Cosine from scipy
- word2Vec from gensim.models
- KMeans from sklearn.cluster
- MMR from mmr_summarizer
- CentroidBow from centroid_summarizer

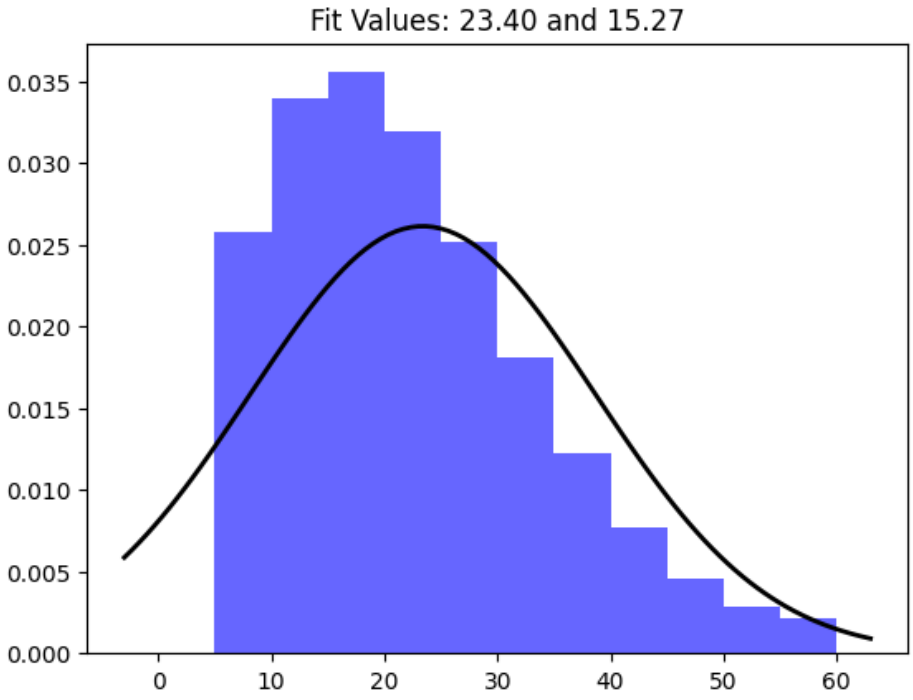
We have used above mentioned libraries to create following functions:

- find _position: It is used to generate an index where every sentence is ending and a new sentence is starting.
- Kmean_summarizer: It is using kmeans from sklearn to create clusters of sentences and from every cluster center sentences are chosen for further processing.
- Centroid_summarizer: it takes the output of kmean_summarizer and gives indexes of the centroid sentences.
- Mmr_summarizer: It uses the MMR algorithm to find redundant sentences in the summary generated by Kmean_summarizer and remove those sentences.
- MMRScore: used to calculate MMR score
- TFs: Used to calculate term frequency.
- IDFs:Used to calculate Inverse document frequency.
- TF_IDF: Used to calculate TF_IDF of a sentence using TFs and IDFs.
- Summary: combines all the above functions in a sequence to generate the summary.

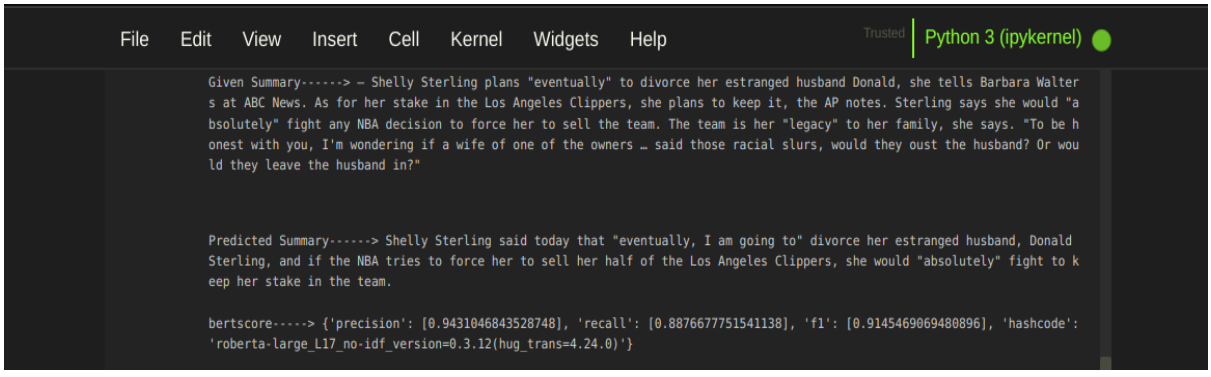
We have also used Matplotlib for plotting some results and to do exploratory data analysis.

RESULTS:

Some data analysis showed us that mean length of sentences is 23.40 and standard deviation is 15.27 (in graph x axis contain no. of sentences and y contains probability). It helped us to estimate the no. of clusters for best results.

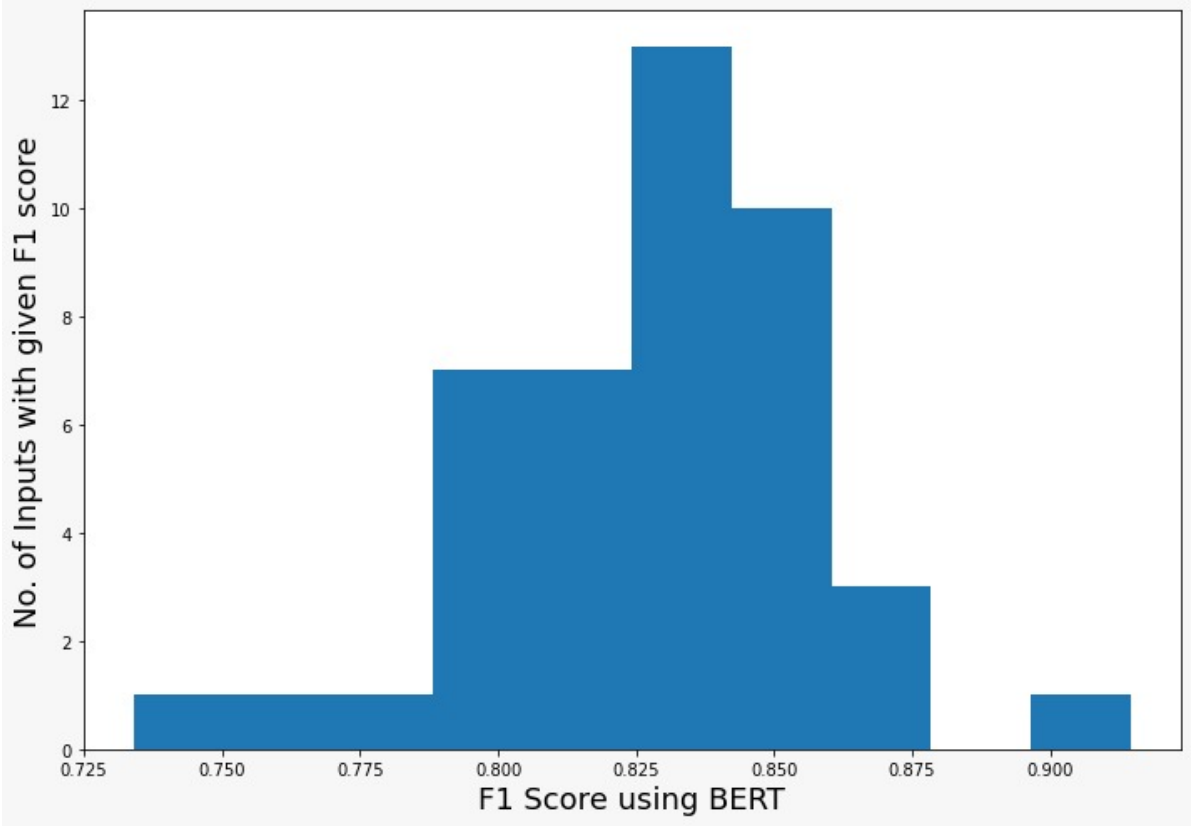


We have till now implemented an extractive method of multi document summarization. We have used F1 score calculating using Bert Score to evaluate over work. Following is the predicted summary and actual summary of some documents for reference to show the output of our model.



F1 Score in this case is 0.9145

We run the code for 50 input points. Following is the graph between F1 score calculated with BERT score and no of input points with that score. The graph shows that most of the inputs have F1 score between 0.825 to 0.850.



REFERENCES:

1. Gaetano Rossiello, Pierpaolo Basile, Giovanni Semeraro. 2017. Centroid-based Text Summarization through Compositionality of Word Embeddings. Proceedings of the MultiLing 2017 Workshop on Summarization and Summary Evaluation Across Source Types and Genres .
2.
3. Harshal J. Jain, M. S. Bewoor and S. H. Patil. 2012. Context Sensitive Text Summarization Using K Means Clustering Algorithm. In International Journal of Soft Computing and Engineering.
4.
5. M R Prathima, H R Divakar. 2018. Automatic Extractive Text Summarization Using K-Means Clustering. In International Journal of Computer Sciences and Engineering
6.
7. Radev, Dragomir R. and Jing, Hongyan and Budzikowska, Malgorzata. 2000. Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In NAACL-ANLP 2000 Workshop: Automatic Summarization.