News Embedding and Clustering comparisons

Anuj Kumar Jha   
*M.Tech. Data Analytics,Department of Computer Applications*  
*National Institute of Technology, Trichy*Tiruchirappalli, India  
jha.anuj.2108@gmail.com

Dr. B. Janet   
*Assistant Professor,Department of Computer Applications*  
*National Institute of Technology, Trichy*Tiruchirappalli, India  
janet@nitt.edu

*Abstract*— “The newspaper is a greater treasure to the people than uncounted millions of gold.”  
Henry Ward Beecher. 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 entertainment news. The other major objective of this project is to solve this problem by having document clustering.

Here initially, around 9000 news from various websites were collected and in multiple categories. Different state-of-the-art models like BERT, XL-NET, and GPT-2 were used for summarization of the news. After that, made use of summary extracted from various summarizer along with some embedding techniques and dimensionality reduction techniques to convert words in proper 2-D vector format. And used document clustering algorithms to accumulate similar news article together and making the process of recommendation faster.

Keywords— BERT, XL-NET, GPT-2, summarizer, embedding, dimensionality reduction, 2-D vector, clustering.

# Introduction

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 cannot imagine a life without it. If it were not for news, then we would not 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. This project is aimed 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 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.

Dataset for this project was gathered and collected using various tools, and the reason behind doing so was that to make this project more authentic and real-time for Indian audience. There are lots of dataset for news present, but the problem with them are they were collected from either overseas news agencies or it did not have the format that was required it was supposed to be in.

In total, around 9000 unique news articles were collected 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, the URLs were collected from these websites and used BeautifulSoup, Newspaper3k, and NLTK library to extract articles and other features from the news URLs.

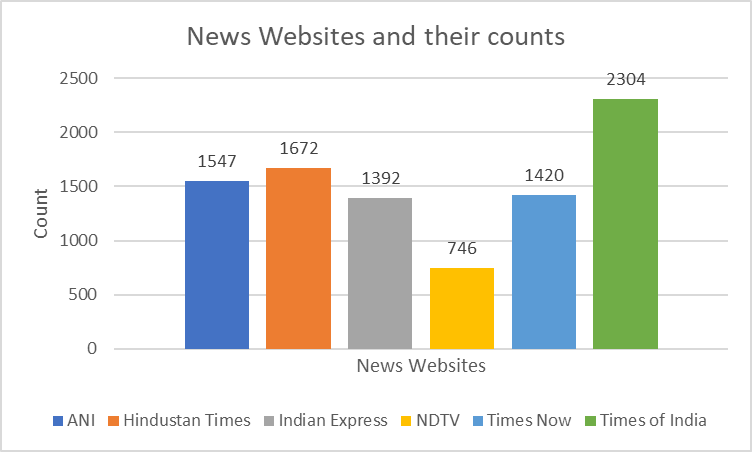


Figure 1.1: Count of news for each website in the dataset

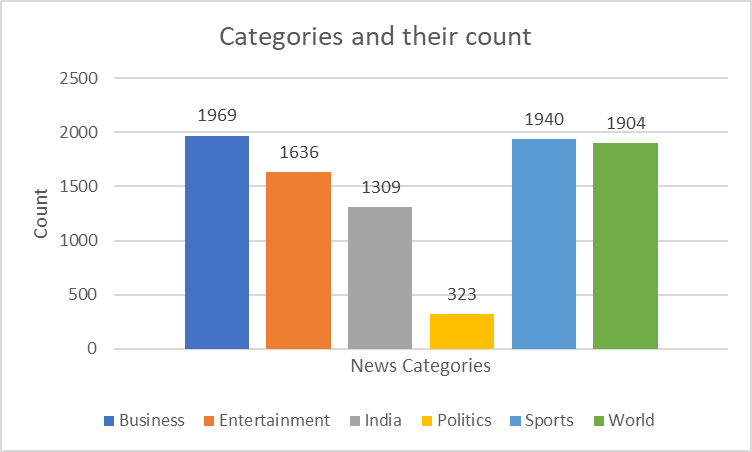


Figure 1.2: Count of news for each category in the dataset

# Methods

## 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. What BERT does is it applies bidirectional training of Transformer to language modelling. And it is quite clear from the result of this that a language model trained bidirectionally can have a better linguistic contextual understanding than a unidirectional language model.

The paper introduces a novel technique named Masked Language Modeling (MLM) which allows bidirectional training of Transformer in language models, which was previously impossible. BERT makes use of Transformer, an attention mechanism that learns contextual relations between words (or sub-words) in a text.

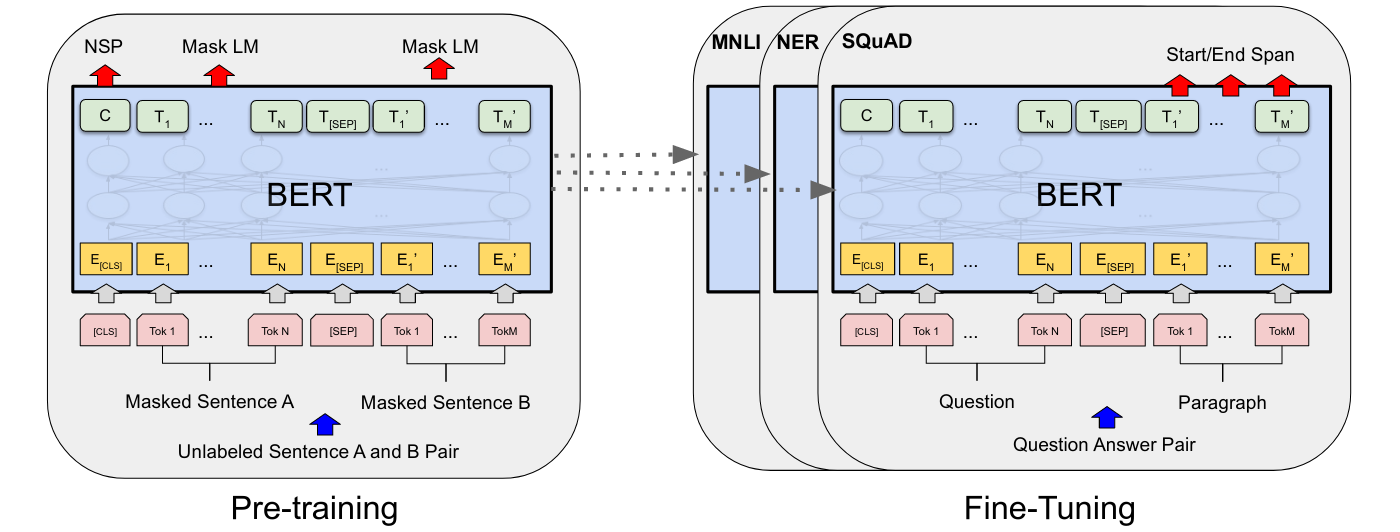


Fig 1.3: BERT Pre-training and Fine Tuning

## GPT-2

The OpenAI GPT-2 showed the noteworthy capacity to compose coherent and passionate essays that surpass what we foreseen current language models could deliver. The GPT-2 was not an especially novel design – its engineering is fundamentally the same as the decoder-only transformer. The GPT2 was a vast transformer-based language model trained with respect to a massive dataset. The GPT-2 is engineered using transformer decoder blocks. GPT2, as customary language models, yields one token at a time. These models work in a way that after a token is generated, that token is merged to the sequence of inputs to become the input to the model for next step. This is called “auto-regression”. One key contrast in the self-attention layer here is that it does not mask future tokens, by changing the word to [mask] like BERT, but by blocking the information that are to the right side of the current position being calculated. A normal self-attention block allows to peek at tokens that are at its right. Masked self-attention does not allow to peek:

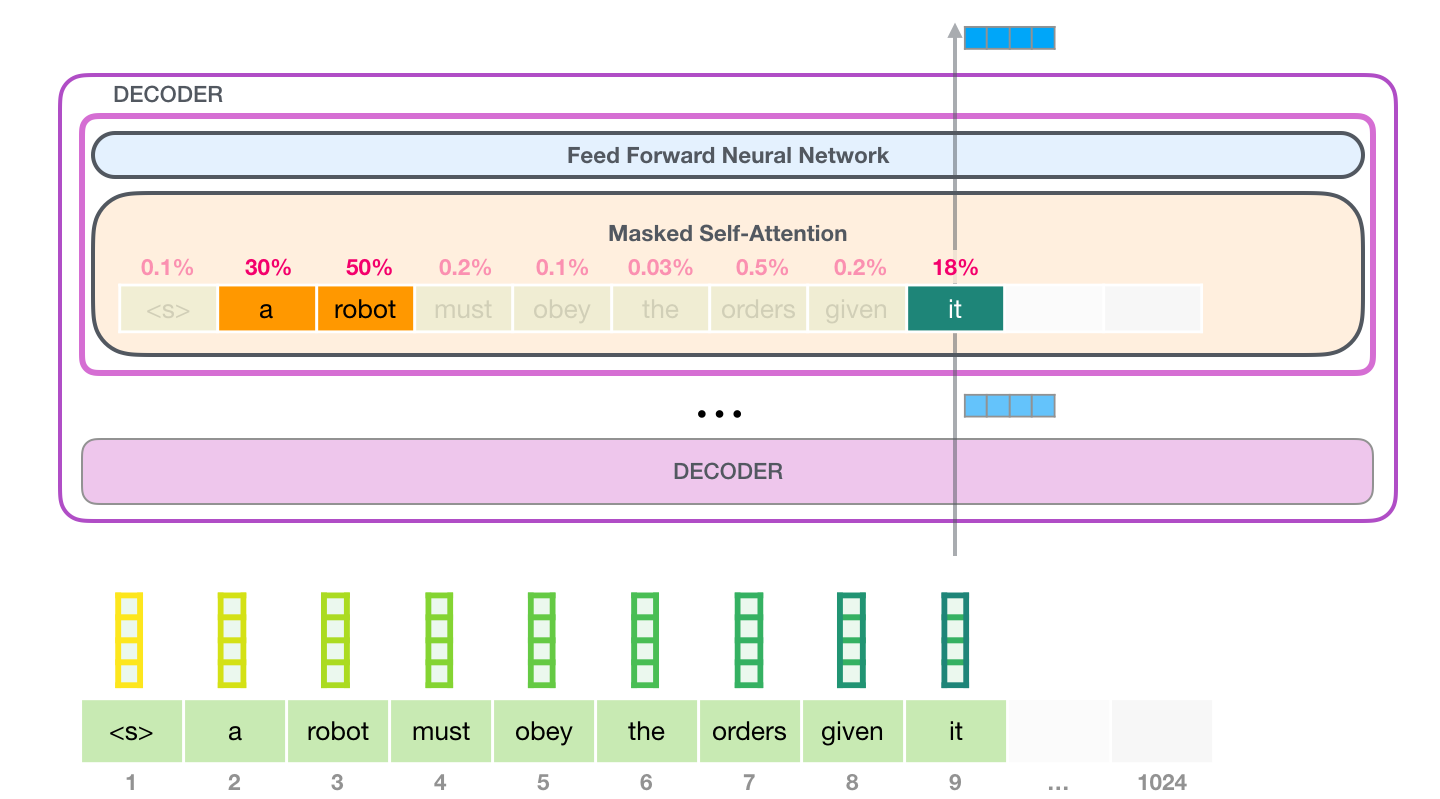


Figure 1.4: A GPT-2 example with Masked Self-Attention

## XLNet

XLNet’s pretraining captures the ideas from Transformer-XL, the state-of-the-art autoregressive model. 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.

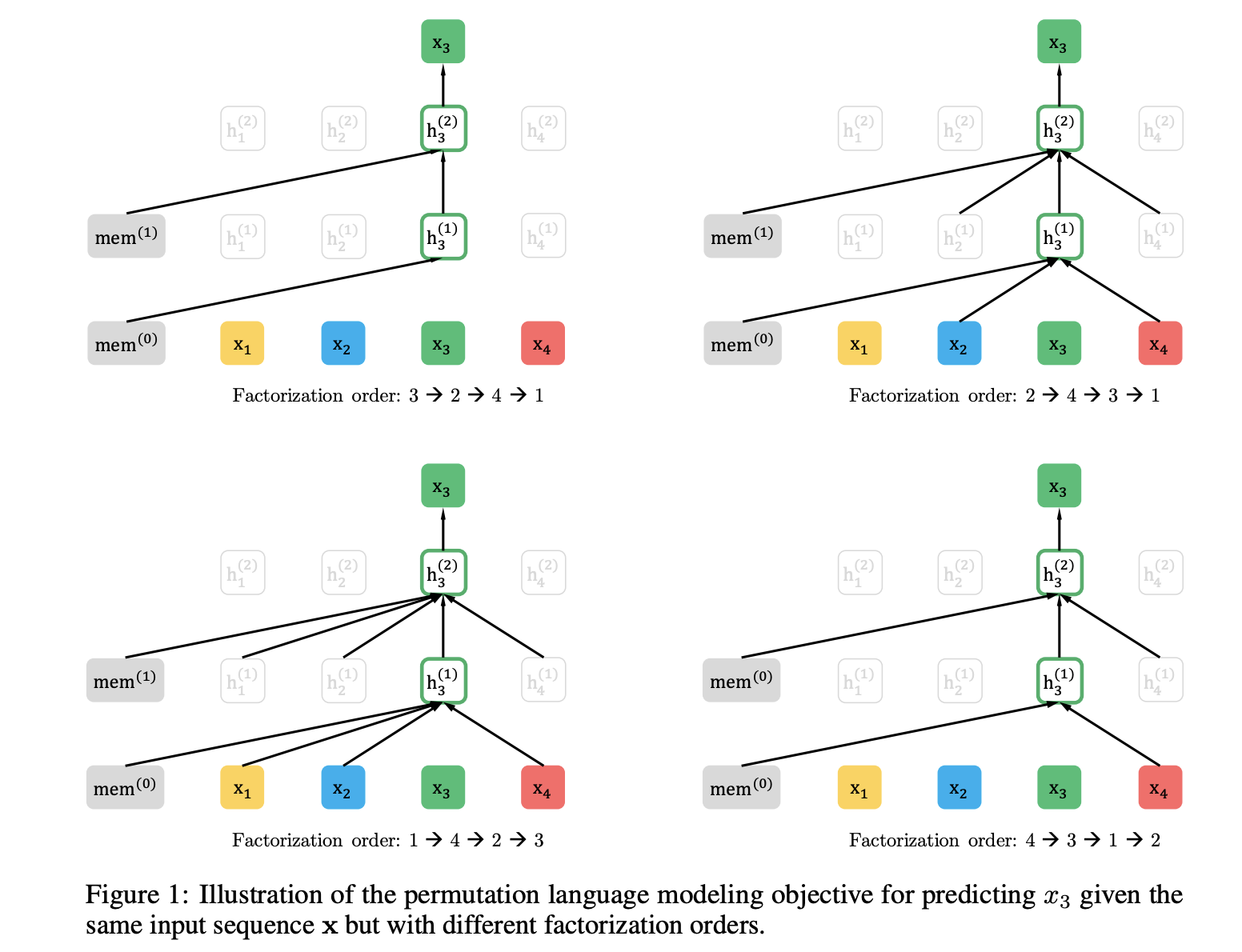


Figure 1.5: Representation of the permutation language modelling objective for anticipating x3 given a similar information arrangement x however with various factorization order in XLNet.

## Model

The whole project is divided into two major parts, first is summarization and second is document clustering.

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. 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.

In the next 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.

Following is the list of experimental setups on the proposed architecture:

|  |  |
| --- | --- |
| **Model** | **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 |

# Result and novelty

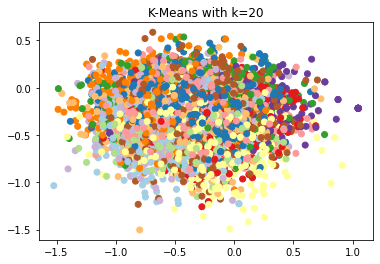
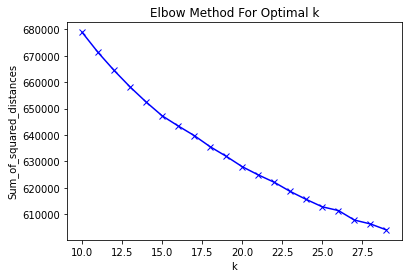
## Result

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 did not 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.

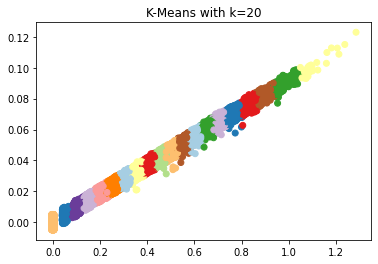
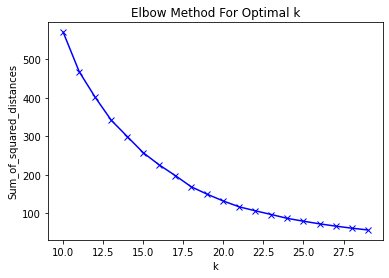
We also used number of dimensions as 2 for dimensionality reduction in both PCA and t-SNE. And for K-Means clustering we had 200 maximum iteration, for K values ranging for 10 to 30, we used elbow method to decide the best k value (which is 20 clusters) and then compared the performance. And for DBSCAN 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.

Following figures indicate the results obtained for different stacked model:

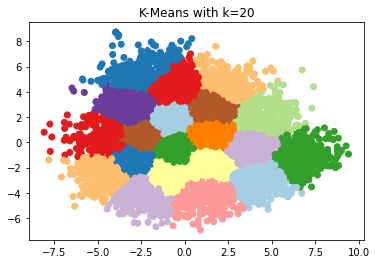
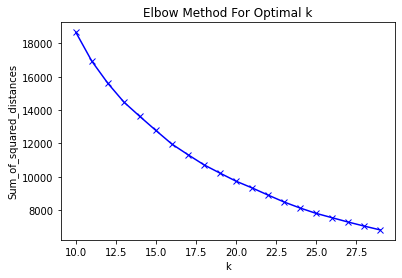
BERT Summary + BERT Embedding + K-Means:



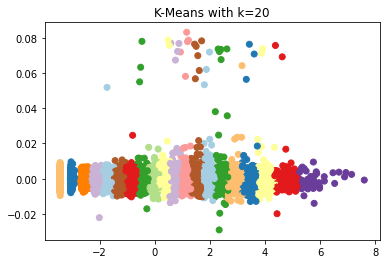
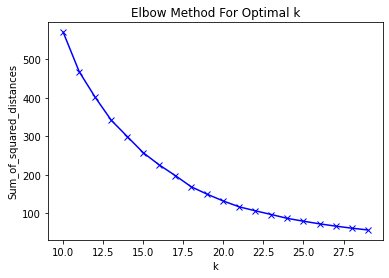
BERT Summary + Doc2Vec Embedding + K-Means:



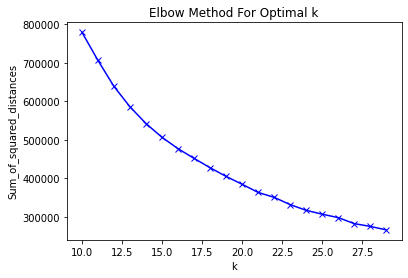
BERT Summary + BERT Embedding + PCA + K-Means:



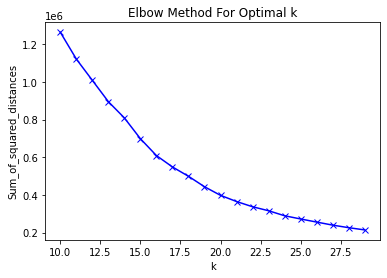
BERT Summary + Doc2Vec Embedding + PCA + K-Means:



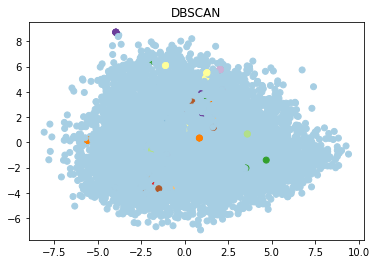
BERT Summary + BERT Embedding + t-SNE + K-Means:



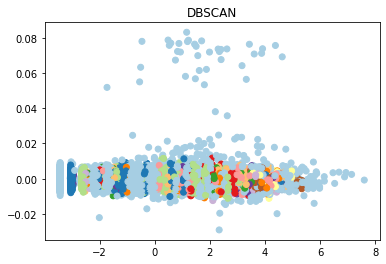
BERT Summary + Doc2Vec Embedding + t-SNE + K-Means:



BERT Summary + BERT Embedding + PCA + DBSCAN:



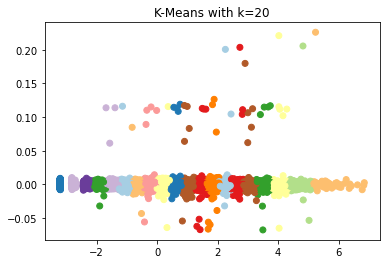
BERT Summary + Doc2Vec Embedding + PCA + DBSCAN:



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 |

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.



## Novelty

Instead of creating individual models from scratch which can take days with several GPUs and TPUs in distributed environment to train, 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. Here the novel approach is to leverage the state-of-the art models which has been trained and tuned with millions and billions of data and parameters and combining them with some traditional models to find the best suitable model stack for this particular use case.

##### Acknowledgment

This project was done under the guidance of Dr. Mrs. B. Janet, whom I am appreciative for providing the opportunity and supervision to lead this project. I am additionally thankful to my peers and my family to demonstrate the help and inspiration expected to build up this undertaking. Furthermore, at long last, I am thankful to National Institute of Technology, Trichy for providing the platform of self-development and righteous.

##### 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.