Deep Language Model Representation of Document Clustering

Shakhboz Abdulazizov
B00870379
Dalhousie University
Halifax NS, Canada
sh873370@dal.ca

Bhuvaneshwari Basquarane
B00853122
Dalhousie University
Halifax NS, Canada
bh563857@dal.ca

Rakshit Makan
B00883651
Dalhousie University
Halifax NS, Canada
r.makan@dal.ca

Noah Sealy
B00726289
Dalhousie University
Halifax NS, Canada
noah.sealy@dal.ca

***Abstract*—Powerful document clustering models are important as they are able to efficiently process large sets of documents. These models can be useful in many fields, including general research. Searching through large corpora of publications can be a slow and tedious task; such models can reduce the time of this significantly. We investigated different variations of a pre-trained BERT model to find which is best able to produce word embeddings to represent documents within a larger corpus. These embeddings are reduced in dimensionality using PCA and clustered with K-Means in order to gain insight into which model is able to best differentiate the topics within a corpus. It was found that out of the tested BERT variations, SBERT was the best model for this task.**

1. Introduction

The need for automatic text organization is becoming increasingly more prevalent as more information is passed throughout the Internet each day. An area which this is especially important is in the field of general research. Parsing through large amount of documents in order to find important studies becomes more difficult as new research is constantly being published. A model which is able to organize a large corpus of papers into clusters would be extremely valuable in aiding researchers finding papers efficiently. On top of this, the process could be generalized to be able to organize a corpus beyond just research papers.

It is important that the mentioned corpus is not only organized in some way, but also accurate. There are many options for text clustering models, though it is important that the chosen one is able to represent words appropriately given the context they are presented in.

This project aims to investigate solutions for the two topics discussed. Our goal is to find which deep language model is best fit to represent documents accurately, as well as cluster those documents within a larger corpus.

After investigation of other models, we decided to use the deep language model, BERT [1] and its possible variations to produce contextualized sentence embeddings. In order to organise the documents, we reduced the dimensions of their embedding representation, calculated their cosine similarity, and then finally clustered with a K-Means clustering algorithm.

Since the pretrained BERT [1] model was trained for classification and question answering tasks, its embeddings were found to not work on similarity based tasks like clustering. We also experimented with various methods like MEAN Pooling (taking mean of [CLS] tokens of last layer), concatenating last four layers of encoder, and [CLS] token. The similarity results with these three pooling scheme yielded unsatisfactory results. This encouraged us to use SBERT [2], a variant of BERT, which was able to provide more meaningful results given the scope of the project.

The dataset used for this project is a collection of research papers from the MALNIS lab library. The data format is a CSV file that consists of the following headers: bibliography type, identifier, author, title, journal, month, year, URL, and abstract. This dataset holds 466 academic papers which relate to machine learning, natural language processing, or deep language models. Pre-processed abstract sections of each paper are used as the corpus for the language models.

The following sections will provide the Background, Methodology, and Results of the investigation in detail. We will then present an analysis these results, and the study as a whole in the Discussion section.

2. Related Work

2.1. Language Modelling

Language models are a broad set of functions in the field of natural language processing, which provide solutions to various language related problems. These problems can include text prediction, translation, and audio to text conversion. In the scope of this paper, language models will be used to represent text documents as word embeddings; this process is also known as topic modelling. Some earlier topic modelling techniques will be described before discussing the chosen models for this study, in order to gain insight of the evolution of such models.

2.2. Topic Modelling

An important discussion piece throughout a topic modelling publication by Luhn as early as 1958 was that machines only ever understand words as purely physical objects [3]. Machines are unable to map meaning to words, only count them [3]. This gives rise to an important fundamental concept through many natural language processing models; a machine must learn to handle and process text through numerical representations.

In a summary for topic models by Nenkova & McKeown, they define topic word models as finding significant words within documents which may be used to represent the document as a whole [4].

Luhn's paper also provided the idea that each word in a document has some significance, and this can be measured by the word's presence throughout the document [3]. The presence of an individual word in a document can be measured by its frequency.

Dunning was later able to create a more robust topic model based on the ideas from Luhn [5]. Although Luhn was able to provide this foundation, it was found that significant words were not necessarily common words, and by missing those, the frequency models may not find an accurate summary of the analyzed document. Dunning's model involved statistical analysis to find significance in less common words which the previous models may have missed [5]. This introduces one class of topic models, known as frequency driven topic models.

2.3. Frequency Driven Topic Models

Frequency driven models assign weights to words in order to derive representations that best summarize a given document [6]. Words with greater weights are considered more important while summarizing the document. Stop words are once again removed for these models, as they are common yet do not provide much meaning to the overall topic of the document.

Early frequency models were flawed as they had no way of compensating for stop words other than completely removing them. This expands not to just stop words, but also any other word in the analyzed documents which were common, but did not effectively contribute to the overall meaning of the document. Thus a more robust frequency driven model was developed, known as TF-IDF [6]. This model will be briefly discussed.

2.3.1. TF-IDF Model. TF-IDF is a frequency driven topic model as it is able to compensate for the previously discussed common word problem [7]. TF-IDF determines the weights of words based on their frequency throughout a document. As mentioned, the goal of this is to find important words which can be used to summarize the given text. TF-IDF introduces a punishment to weights of words that are common across the corpus being investigated [7]. This means that commonly occurring words, such as stop words, will not have relatively lower weights, despite having a high frequency throughout a given document. Overall, TF-IDF is able to assign weights to words according to their importance, and these important words can be used to represent the given document as a whole [7].

Although TFIDF is able to solve various issues relating to frequency driven models, they still do not create perfect representations of text. A contributing factor to this limitation is that context is lost in these sentences; this alone can make the representations lose a lot of important meanings. The last piece of information provided in this background on language modelling will introduce a mechanism that can retain context with text representations. Models that exhibit this mechanism are known as attention models, and exhibit a critical importance to this study.

2.4. Attention Models

Attention models take a different route of deriving text representation. There is a deep history which will be briefly discussed in order to introduce the Transformer model. A key component of most attention models is the generation of word embeddings which contain more context to the processed text data. In theory, contextualized word embeddings would better represent text compared to models which summarize text simply by its frequency. This is because frequency does not necessarily capture the meaning or context of most text.

The attention model which is most relevant to this study is known as the Transformer. The Google Brain publication known as Attention is All You Need presents this model [8]. Originally proposed to solve an inefficiency observed in recurrent neural networks during sequential text processing, the Transformer has become a foundation for many attention models due to its encoder stack [8]. Though only one part of the model is used, the encoder stack, contains a multi-head self-attention layer, which contributes to its ability to generate contextual word embeddings based on input text [8]. This model plays an important role in the BERT model, as will be discussed throughout the paper.

2.5. BERT

BERT, or Bidirectional Encoder Representations from Transformers, is a model inspired by a recent paper published by researchers at Google AI Language [8]. It caused a stir in the Machine Learning community by presenting state of the art results in a wide variety of NLP tasks, including Question Answering (SQuAD v1.1), Natural Language Inference (MNLI), and others.

BERT's key technical innovation is applying the bidirectional training of a Transformer to language modelling. This is in contrast to previous efforts which looked at a text sequence either from left to right or combined left-to-right and right-to-left training.

2.6. SBERT

SBERT [2] is a modification of the pretrained BERT network that uses siamese and triplet network structures to derive semantically meaningful sentence embeddings. These embeddings can be compared using cosine-similarity methods, in order to organize the corpus [9].

2.7. K-Means

K-Means is one of the most widely known clustering algorithms. The process follows a way to classify a given data set through a certain number of clusters (assuming k clusters). The main idea is to define k centers, one for each cluster. However it is a partitioning algorithm rather than clustering as it partitions the data-set into as many parts as requested through trying to minimize inter-partition distances [9]. The algorithm works in the way to process the training data, the pre-processed dataset starts with a first group of randomly selected centroids, which are used as the starting points for every cluster, and then performs iterative calculations to optimize the positions of the centroids.

3. Methodology

3.1. Preprocessing

As described in previous section the text has been extracted from the PDFs of research paper of MALNIS lab, due to which there are multiple anomalies in the data. We had to clean and pre-process the data before we could pass it through the deep language models. For this task we used the NLTK python library, as well as regular expressions to clean the corpus. The first step in cleaning included dropping the duplicates, removal of numbers and special characters.

3.1.1. Stop Words. Unlike other language models, removal of stop words does not lead to better results with the BERT model variations. Although the pretrained BERT model is provided in a black box, our theory for this is that stop words are not removed during training. This may be because stop words provide additional sentence information, when they are embedded within the context of the sentence. Another study focused on measuring BERT performance scores has found supporting results as they claim that stop words receive as much attention as non-stop words [10].

3.1.2. BERT word limit. BERT has a word limit of 512 words which can be problematic for large size of documents. Fortunately we were dealing with only the Abstract of research papers which had an average of 225 words in all documents as shown in figure 1. There were few documents that exceeded this limit so we had to truncate them to 400 words.

3.2. Deep Language Model

3.2.1. BERT Foundation. As shown in figure 2, there were a lot of building blocks which needed to be understood to fully understand the BERT model. In order to gain this foundation, we started our research from learning about recurrent neural networks, which are most commonly used for sequence or time series data. The limitation of this model was that they were not able to capture the long term dependency, as mentioned in [11]. We then explored LSTM (Long Short Term Memory) [12] networks, which provided the base for seq2seq model [13] and attention models [13]. After covering this groundwork for understanding the BERT, we finally found the Transformer model [8]; which sets up a foundation for the BERT [1] model.

3.2.2. BERT Base. BERT as a model can have multiple use cases in the various tasks of NLP. For our project we were mainly focused on extracting the features of our data set. In order to make natural language (in text) understood by a machine, we had to convert said text into vectors. An available pretrained version of the BERT model was used for converting the text into contextual word embeddings. To create these pre-trained models, BERT's stack of encoders was trained on variety of datasets with two objectives. The first objective consisted of predicting the masked words in the sentences. The other training objective was predicting the next sentence given the first sentence.

As mentioned, to process contextual word embedding from the BERT model, we needed to do some preprocessing of the text before we could pass it to the model. Firstly, we tokenized the given text. These tokens were designed specifically for the BERT model. The BERT input tags the words with an index value that BERT uses to maintain its vocabulary. BERT has a limited vocabulary of 30000 words, which created problems while tokenizing the words as it broke the words down into ngrams. These ngrams were then tagged with a special token, ##. Shown in figure 3, the word *embedding* is divided into multiple meaningful words, which BERT understands. This could be seen as a limitation while processing domain specific text, where most of the words would be new for BERT. Due to this nuance, BERT may be prone to misinterpretation. This limitation is further discussed in the Discussion section.

The output from this tokenization process yields two types of tokens; [CLS] tokens at the beginning of the sentence and [SEP] at the end. These tokens act like flags for the model, because BERT expects all inputs to be of the same length, which should be less than 512. Since, our data set has an average of 200-250 words and maximum of 600 words, shown in figure 1, we truncated the data to 400 words. This truncation minimizes the loss of data. Any text shorter than the required length had their remaining characters set to zero. Along with this matrix we had to also provide an attention vector for each document. This vector marked all the non-zero tokens as 1 and rest as 0. After applying all the processing, we got two matrices of size:

$$\#documents * tokens = 466 * 503$$

In order to obtain the contextualized embeddings a pretrained BERT model, BERT-BASE, was implemented. BERT-BASE is a 12-layer model with 768 hidden layers. To harness the features from these embeddings, the paper [1] provides multiple pooling options. Three options were tested, [CLS] token embeddings, mean of last hidden layer, and concatenation of last 4 hidden layers. The output of the

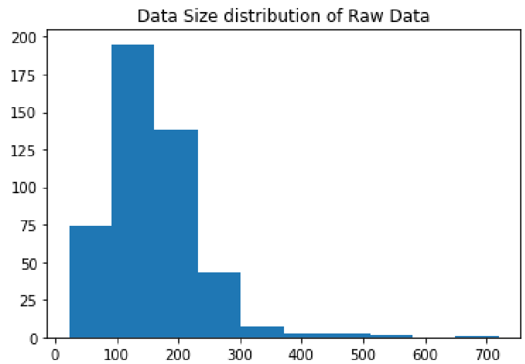


Figure 1: Histogram showing the length of each document in the corpus

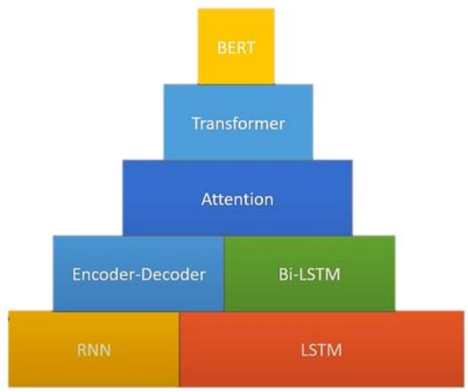


Figure 2: Conceptual building blocks for BERT

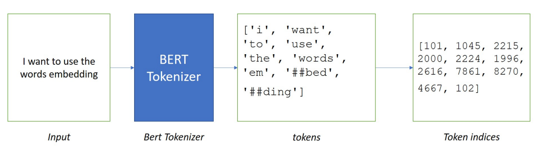


Figure 3: BERT tokenization process

BERT model provided a matrix of the size 503 (number of tokens) x 768, per document in the dataset. This process is shown in figure 4; this will be the output for all the 12 layers of BERT model.

3.2.3. SBERT. SBERT is another model that we tried for this project. It is a modification of the BERT pretrained model designed to produce sentence embeddings which can be used for finding semantic similarity between two sentences. According to the paper [2], the sentence embedding produced by the BERT model variations is not able to perform well on similarity tasks. From [2], three pooling methods were defined for SBERT. From those options, we used the mean pooling as it was proven to outperform the other two [2]. The other two were CLS token pooling and MAX pooling [2].

3.2.4. Cosine Similarity Task. In order to capture the nature of embeddings being produced by the mentioned model, a task was deployed involving comparing embeddings from different texts using cosine similarity. Abstracts were selected by hand into two subsets, A and B, from the corpus. Set A contained two documents which were observed to have similar topics, while set B was contained text with topics that were different. Random sampling occurred within the constraints of those conditions to choose the specific documents. The abstracts were then embedded using the discussed models, then cosine similarity was performed to compare the embeddings of each abstract. In order for this experiment to hold statistical significance, it was performed over multiple trials.

Figure 4: Pretrained BERT output

3.3. Dimensionality Reduction

An issue with many language processing tasks is that word embeddings requires thousands of features to represent a single sentence. This poses some issues, such as slow training times or difficulty in finding optimal solution. This is known as the curse of dimensionality. Dimensionality reduction is the process of decreasing the number of features to the most relevant ones.

Dimensionality reduction methods are convenient for researchers as they will allow for visual analysis of the clusters which are easily readable to the human eye [14]. Another factor of convenience associated with dimensionality reduction techniques is that they will reduce the overall complexity of a given word embedding by a large factor, making clustering a more efficient task [14]. This is crucial as the word embeddings are usually represented in very high dimensions by the deep language models. For our purpose of reducing the dimensions of the word embeddings, we have implemented a technique known as Principal Component Analysis, or PCA.

PCA is a dimensionality reduction technique which aims to find project data onto lower dimensional hyperplanes, while keeping the maximum amount of variation in the data [15]. It is adapative to any type of data presented to it, which makes is very useful for processing word embeddings [15].

3.4. Clustering

In order to cluster each model's word embeddings, K-Means clustering algorithm was used. We compare the results of the clusters from each variant of BERT, in order to find out which deep language model works best for clustering. We first started with finding optimal number of clusters in the data set. With our prior understanding of the data, it was hypothesized that the abstracts could be bucketed into at least 4 reasonable clusters; with each cluster representing a distinct topic.

3.4.1. Optimization/ Hyper-parameter Tuning. During the development process, hyper-parameter tuning occurred after the discussed model selection. The reason why we need to do this step is that finding the right combination of hyper-parameter values in order to achieve maximum performance on the data in a reasonable amount of time. Many algorithm implementations, such as K-means, come with the default hyper-parameters values. Depending on the project and dataset, the model may not always perform optimally using just the default hyperparameters. Due to this, hyperparameter tuning is very important, in order to best optimize the model to obtain its best performance within an efficient training time.

Originally, SBERT, BERT with mean last layer, and BERT with [CLS] tokens yielded embeddings of size 768. The BERT variant which concatenated the last four layers produced embeddings of size 3072. As K-means is a distance based clustering algorithm, these embeddings were represented in too large dimensions to be able to efficiently clustered. Thus, using the PCA technique, we reduced the dimensions of the word embeddings to just two, before clustering.

3.4.2. Elbow method. The elbow method is a useful technique for determining the optimal number of clusters to choose for an algorithm such as K-means. For a tuning starting point, we employed this technique as we had no prior knowledge of the optimal number of clusters in our dataset. The elbow method captures the variation in the dataset and as soon as the number of clusters exceeds the actual number of groups in the data, the added information will drop sharply. This is because at that point, it is just subdividing the actual groups. Assuming this happens, there will be a sharp elbow in the graph of explained variation versus clusters. The elbow plots for all the variations of BERT are shown in figure 6.

4. Results

We first evaluated the different variants of BERT (CLS Token, Mean Pooling and concatenating last four layer) and SBERT (Mean Pooling) model. We assessed them based on the ability to find similarity and dissimilarity between the sentences. The results shows that the SBERT with mean pooling easily surpassed the BERT. We have captured the difference in these models by looking at the color pattern of cosine similarity heat maps as shown in figure 5. There are two things we noticed over here. Firstly, the range of the map and the color density variations in them. SBERT shows the highest range and color intensity changes as compared to all the other models.

4.1. Cosine Similarity

Table 1 shows the average cosine similarity scores for each model during the cosine similarity task discussed in the methodology section. For SBERT, the average score for similarity between embeddings for abstracts in A and B were 94, and 87.

For CLS token variation of BERT the similarity scores for abstracts in A and B were 94, and 87. For the mean last layer variation of BERT the similarity scores for abstracts in A and B were 92, and 88. For the concatenated last layer variation of BERT the similarity scores for abstracts in A and B were 90, and 87.

BERT Model	Similarity Set A	Similarity Set B
SBERT	75	50
CLS Token	94	87
Mean of Last Layer	92	88
Concat Last 4 Layers	90	87

TABLE 1: Cosine Similarity Range for BERT variants

4.2. K-Means Clustering

K-Means clustering was used to cluster the documents together based on the embeddings produced by each implemented model. As mentioned, the number of clusters per model was chosen using the elbow technique, shown in figure 6. The number of clusters for each model variant were found to be relatively low. This may be due to the fact that the dataset contained a high amount of homogeneous paper topics.

The clusters of the lower dimensional word embeddings were naturally unlabelled. Due to this, evaluation of the clustering could not rely on using labels as in supervised learning. In order to compensate, each cluster was analyzed and subjectively summarized with a few main topics by the members of the group. Table 2 shows the results of each subjective analysis, by showing the topics which were thought to best describe the clusters.

In order to show this process, the SBERT clusters from table 2 will be briefly discussed. K-means produced six clusters given the word embeddings from the SBERT model. After further analysis of these clusters, it was evident that each abstract related to one of the topics chosen to represent the cluster. As mentioned, these topics were chosen subjectively by the group members while analyzing the documents contained in each cluster. The rest of the clusters go on in that same fashion. The "Mixed" topic implies that the cluster did not contain document clusters distinct enough to be described simply by a few topics.

5. Discussion

5.1. Frequency Topic Models

The main model implementation in this investigation were various forms of the BERT encoder model. While introducing topic models, a brief summary of frequency topic models was provided, mainly the TF-IDF approach. Although not implemented, these models were briefly discussed in order to further develop the history of topic models thus far. A frequency model was not implemented as it was hypothesized early on that such implementations lose contextual meaning while analyzing text. Due to this hypothesis, we thought that a model such as BERT was a better choice, as it is able to retain contextual meaning while processing text data. In order to further this investigation, future work includes comparing the performance frequency topic models on the tasks provided, with the performance of the various implemented BERT models.

5.2. BERT

As it is clear from table 1, the BERT model implementations struggled while determine when documents were not similar. Given these inaccurate results, BERT still provided an essential benchmark to the study. This is because the SBERT model, which is discussed below, was implemented from the base BERT model. We recognize BERT as an important stepping stone for this project, and within additional studies in this area.

5.3. SBERT

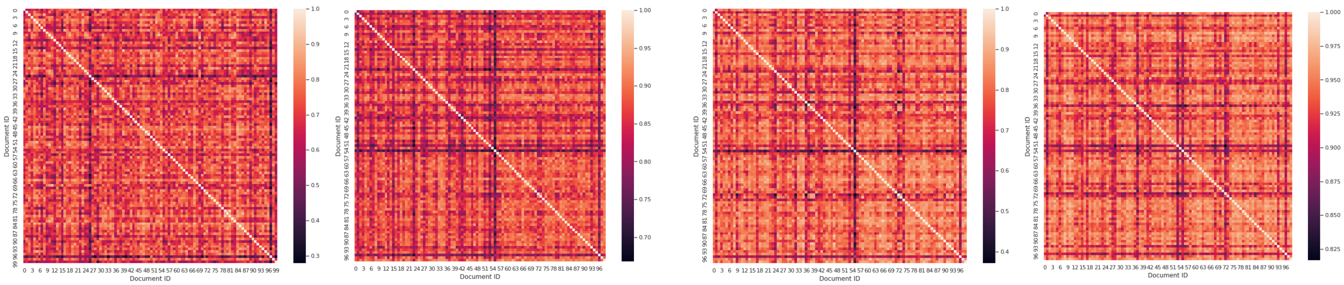
According to table 1, although SBERT typically yielded a lower similarity score compared to the other BERT implementations, it was the only model that accurately captured deviations in the similarities within the set of papers that did not share a topic. This finding shows that the SBERT may be a more competent model in determining if a paper is not actually similar.

Additionally, as mentioned previously, the SBERT model outperformed the BERT variations while clustering documents by topic. This is because during training, SBERT is tuned to capture similarities between documents, using cosine similarities [2]. This indicates that SBERT will naturally have a better performance during semantic similarity tasks.

The results from figure 5 are also supported by the findings shown in [2], as they discuss that both averaging the last layers of the BERT model, and using [CLS] tokens can yield relatively bad sentence embeddings. From this research, it is evident now that these methods are not optimal for a study which focuses on finding robust sentence embeddings.

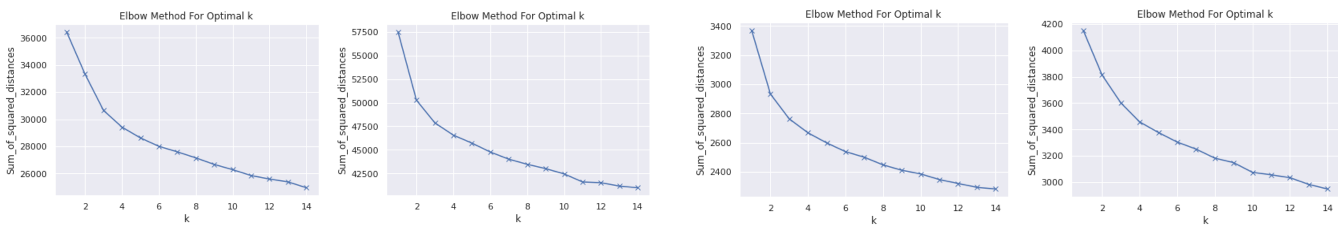
Strong sentence embeddings are important, as this project involves summarizing abstract sections which are made up of at least a few sentences. Our results support this claim, as SBERT was found to have the best performance on the tasks out of all models tested.

These SBERT related findings all contribute to the decision that SBERT is a very robust model for the specific goals of this project.



(a) SBERT (b) Last 4 layers (c) Mean Of Last Layer (d) CLS Token

Figure 5: Cosine similarities Heat Maps



(a) SBERT (b) Last 4 layers (c) Mean Of Last Layer (d) CLS Token

Figure 6: Elbow Plots for selecting # of clusters

Clusters	**Topics in clusters**
	SBERT - 6 Clusters
Cluster 0	Active Learning, Deep Learning, Interactive Analysis
Cluster 1	Radio-Graphs, X-Rays and Healthcare
Cluster 2	Summarizing, Social Media Data and IR
Cluster 3	Deep Learning and Deep Language Models
Cluster 4	Mixture of cluster 0 and 3
Cluster 5	Clinical Data and healthcare
	Last 4 Layers - 3 Clusters
Cluster 0	Mixed
Cluster 1	Mixed
Cluster 2	Mixed
	Mean of Last Layer - 4 Clusters
Cluster 0	Radio-Graphs, X-Rays and Healthcare
Cluster 1	Active Learning, Deep Learning and Deep Language Models
Cluster 2	Mixed of Cluster 0 and 1
Cluster 3	Deep Learning, Interactive Analysis and NLP
	CLS Token - 6 Clusters
Cluster 0	NLP and Deep Language models
Cluster 1	Deep Learning, Interactive Analysis and NLP
Cluster 2	Summarizing, Social Media Data and IR
Cluster 3	BERT, interactive clustering and radiography data
Cluster 4	Same as cluster 3
Cluster 5	Mixed Cluster

TABLE 2: Subjective topics based on the contents of each cluster, for each implemented model

5.4. Limitations

There are a number of limitations to the implementations of this study. This section will present the most pressing limitations to the study.

5.4.1. Limited Input Size. Both BERT models had a limited input size of 512 words. The model would thus not accept input text that was not truncated or summarized, if the original text exceeded this word limit. This is limiting as longer input text is not able to be processed within this study without a lot of preprocessing. Due to the fact that we only used abstracts, we did not experience significant data loss, but this is a potential issue for analysis of larger texts in future work.

5.4.2. Limited Vocabulary. Both BERT models were trained to have only a vocabulary size of around 30000 words. This was limiting especially while processing research papers that referenced domain specific concepts and techniques that were not known by the model.

5.4.3. Slow Processing Time. Specific to the BERT model, there was a high computational requirement in order to run the model. During the experiments, we observed the model to take around 45 minutes to only process around 466 sentences. In order to compensate for this high computational requirement, we had to input sentences individually. This was quite taxing on a general use laptop, which was used for most of the model processing.

5.4.4. Unlabelled Dataset. We were limited to the fields from the dataset that was used throughout the study. This dataset did not include true labels for each individual document. These labels would simply be the overall topic of the given paper, as we have tried to label ourselves from the clusters in table 2. The knowledge of these true labels would help further validate the results and allow us to deploy supervised learning algorithms to compete with the BERT variation performances. Unfortunately, because labels were not initially presented in the data set, developing that field would involve having to manually read through each paper in order to label them.

5.5. Future Work & Implications

The study is constrained to only four model variants. We originally planned on implementing more models, but time limitations reduced that number. Given more time to work on this investigation, more variants and models could be tested. This would broaden the perspective of which language model can perform the best on the tasks presented. Thus, a large part of the future work includes testing other deep language models, such as InferSent, Universal Sentence Encoder, XLNet and Open-GPT3.

The BERT variants were also implemented out of the box. Due to this, there was not a lot of experimentation with the model's hyperparameter during the study. Future work involves tuning these hyperparameters in order to make BERT understand domain specific papers. The general hypothesis for this future work is that BERT may produce better embeddings given the study's dataset, which would directly improve document clustering.

Other future work includes collecting more research papers for the dataset, as well as developing more tasks to test the models. This would further validate each models robustness within the context of the study, as well as obtaining a deeper understanding of each models strengths and weaknesses.

6. Conclusion

The goal of this study was to find which deep language model was able to best produce sentence embeddings on a given research paper corpus so that it could understand semantics. These embeddings would then be clustered in order to assess the overall topics within the corpus. This study is important as quickly processing large corpus's of research papers has the potential to help researchers find papers that match their specific needs during research. This will ultimately increase the efficiency of the background research process for many fields.

Due to limited time, only 4 variants of a pretrained BERT model was implemented, BERT base with mean pooling, BERT base with concatenating last four layers, BERT base with [CLS] tokens, and SBERT. It was found that SBERT was able to best generate sentence embeddings for clustering. Documents were clusters with a K-Means clustering algorithm. Due to various limitations, the future work for this study involves implementing various other deep language models, improving the dataset, and implementing more testing benchmarks for the model.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[2] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," 2019.

[3] H. P. Luhn, "The automatic creation of literature abstracts," *IBM Journal of Research and Development*, 1958.

[4] A. Nenkova and K. McKeown, "A survey of text summarization techniques," *Mining Text Data*, 2012.

[5] T. Dunning, "Accurate methods for the statistics of surprise and coincidence," *Computational Linguistics*, 1994.

[6] M. Allahyari, S. Pouriyeh, M. Assefi, S. Safaei, E. D. Trippe, J. B. Gutierrez, and K. Kochut, "Text summarization techniques: A brief survey," 2017.

[7] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, 1988.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[9] L. McInnes, J. Healy, and S. Astels, "Comparing python clustering algorithms," 2016. [Online]. Available: https://hdbscan.readthedocs.io/en/latest/comparing_clustering_algorithms.html

[10] Y. Qiao, C. Xiong, Z. Liu, and Z. Liu, "Understanding the behaviors of bert in ranking," 2019.

[11] P. S. Y Bengio and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, 1994.

[12] S. Hochreiter and J. Schmidhuber, "Lstm can solve hard long time lag problems," *Advances in neural information processing systems*, 1997.

[13] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," 2014.

[14] E. Sherkat, E. E. Milios, and R. Minghim, "A visual analytics approach for interactive document clustering," vol. 10, no. 1, 2019.

[15] I. Jolliffe and J. Cadima, "Principal component analysis: A review and recent developments," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 2016.