Analysis of feature vectors generated by the Contextualized Neural Language Model

Student: Keshav, Padiyar Manuru
kespa139@student.liu.se

Supervisor: Ali, Basirat
ali.basirat@liu.se

January 18, 2022

Abstract

Linguistically it is assumed that the abstract words need more contextual information for the human brain to process them. Whereas concrete words are processed based on sensory information. Analysis are performed to check how a contextualized language model such as BERT can distinguish between concrete and abstract words. The method we propose is to group the feature vectors generated by BERT into clusters of appropriate contextual sense. These clusters are used to analyze the relationship between the distribution of feature vectors and word concreteness/abstractness.

1 Acknowledgement

I would like to express my special thanks of gratitude to Dr.Ali Basirat for his invaluable support, time and guidance throughout this research work.

2 Introduction

Natural language words can be classified on the basis of abstractness/concreteness. Linguistically, it is assumed that the abstract words need more contextual information for the human brain to process them. On the other hand concrete words are processed based on sensory information. For example, in the sample sentence: *"The speaker was reflecting on the importance of global institutional experiences"*, the word "*Speaker*" is a concrete word as it refers to a person and we could visually see a person. But the word "*Experience*" is an abstract word as we cannot visualize it without context. In the past contextualized language models such as BERT have had tremendous success in generating vector representations for natural language words. The vectors that represent natural language words are also referred as feature vectors. It would be interesting to study these vector representations for concrete/abstract words. Therefore in this paper analysis is performed to see how these vector representations can distinguish between concrete/abstract words. In a contextualized learning model a word has different vector representation across different contexts [9]. We would naturally expect that an abstract word will have multiple dissimilar vector representations depending on the context. Thus, a clustering method would identify more clusters in the feature vectors of an abstract word than in a concrete word. The aim of the paper is to verify whether this hypothesis is validated by using a mixture of Gaussian clustering method on the feature vectors generated by BERT. Also, an analysis is performed to check if there is any linear relation between the distribution of those feature vectors and the concreteness/abstractness of a word.

3 Data

For the purpose of analysis an English corpus consisting of around 23,300 random meaningful sentences is used as input data set. In addition, we are using a data set having 37058 English words and its concreteness/abstractness ratings. The word concreteness is measured by asking subjects to rate words on a numerical scale [6]. Concreteness ratings are presented for 37,058 English words obtained from over 4,000 participants employing a norming study using internet crowdsourcing for data collection [1]. The concreteness ratings used in this analysis are obtained from a subset of a comprehensive list of English lemmas and contain all lemmas known by at least 85 % of the raters. Words with higher ratings are concrete words and those are with lower ratings are abstract words.

4 Theory

All the relevant theory, algorithms and model evaluation metrics used for the analysis are described briefly in this section.

4.1 BERT

For the purpose of this paper we need not deep dive into the specifics of this architecture and why it works. However it is important to know its basic concept and usage.
BERT [5] stands for Bidirectional Encoder Representations of Transformers. It makes use of Transformers [10] that learns contextual relations between words in a text. It's model architecture is a multi-layer bidirectional encoder, as it represents a word based on both previous and next context making it bidirectional. An encoder block [10] maps an input sequence of symbol representations $(x_1, x_2, ..x_n)$to a sequence of continuous representation $z = (z_1, z_2, ...z_n)$. It is pre-trained on a large corpus of English data in a self-supervised fashion. This means it was trained only on the raw texts and no labeling was involved. It has an auto process to generate labels for the input text. Based on the original paper many variants of BERT have been developed and for our experiment pre-trained *BERT Large* model provided by the hugging face is used. The BERT Large model has 24 transformer blocks which constitutes for 24 intermediate layers [5].
The first layer is the input layer where the input sequence may be a single sentence or two sentences packed together is first broke into individual linguistic units or tokens. The tokens are vectorized using WordPiece Embeddings[12]. To get the position of the token in the sequence a position embedding is used. Lastly, to identify from which sequence the token has emerged segment embedding is used. Thus for a given token its input representation is constructed by combining corresponding token, position, and segment embeddings. This token representation is then fed into transformer segment to train the BERT model to produce the learned feature vectors. Special token [CLS] (classification token) is used for classification predictions, and [SEP] separates input segments[5]. A visualization of this construction can be seen in Figure 1

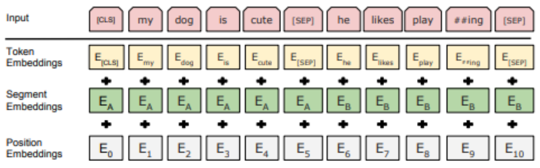
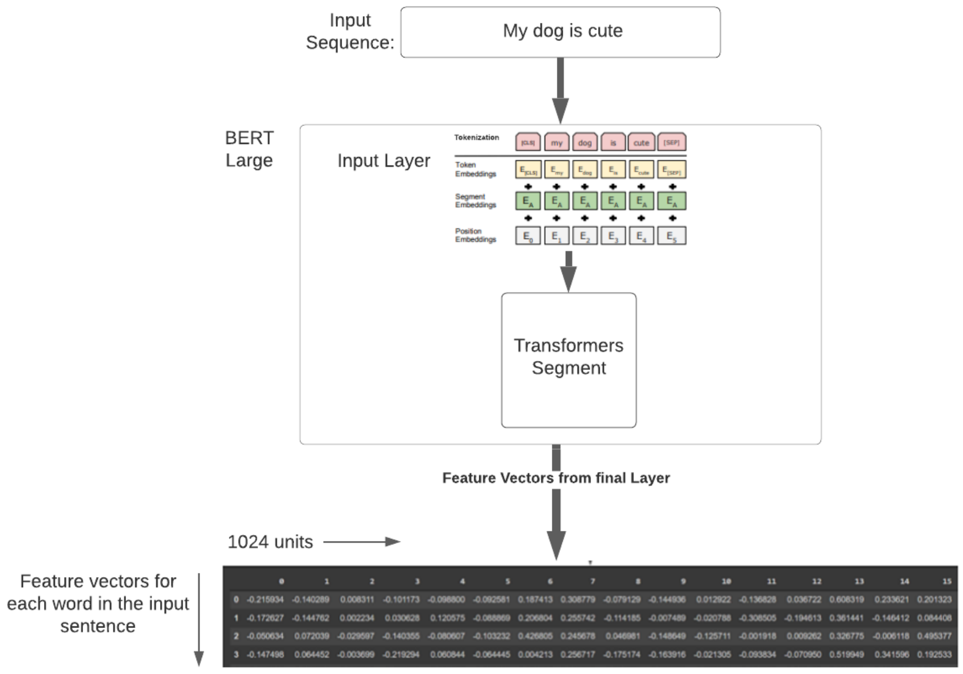

Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8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

Figure 1: BERT input representation [5]

Each transformers in BERT Large have 1024 hidden units that forms the dimensions of the feature vectors [5]. Output values from all hidden units are represented in a vector which represents the feature vectors of length 1024.
The feature vector generation flow can be viewed in Figure 2


Input Sequence:
My dog is cute
BERT Large
Input Layer
Tokenization
[CLS]
my
dog
is
cute
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5
Transformers Segment
Feature Vectors from final Layer
1024 units
Feature vectors for each word in the input sentence

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	-0.215934	-0.140289	0.008311	-0.101173	-0.098800	-0.092581	0.187413	0.308779	-0.079129	-0.144936	0.012922	-0.136828	0.036722	0.608319	0.233621	0.201323
1	-0.172627	-0.144762	0.002234	0.030628	0.120575	-0.088869	0.206804	0.255742	-0.114185	-0.007489	-0.020788	-0.308505	-0.194613	0.361441	-0.146412	0.084408
2	-0.050634	0.072039	-0.029597	-0.140355	-0.080607	-0.103232	0.426805	0.245678	0.046981	-0.148649	-0.125711	-0.001918	0.009262	0.326775	-0.006118	0.495377
3	-0.147498	0.064452	-0.003699	-0.219294	0.060844	-0.064445	0.004213	0.256717	-0.175174	-0.163916	-0.021305	-0.093834	-0.070950	0.519949	0.341596	0.192533

Figure 2: Feature vectors generated by using BERT Large model for the words in an input sentence

4.2 Mixture Of Gaussian (MoG) Clustering Method

As the name implies, a Gaussian mixture involves overlapping or superposition of multiple Gaussian distributions. In our analysis, we assume that the distribution of vectors associated with words is a mixture of Gaussian. The goal is to assign the data points to the appropriate cluster as shown in Figure 3.

Each data sample x_j belongs to one of the k Gaussian distributions. The sets of samples S_i that belong to the distribution $N(p_i, C_i)$ $i = 1...K$ with mean p_i and covariance matrix C_i are identified.

The Gaussian Distribution:

$$f(X|p_i, C_i) = \frac{e^{-0.5(x-p_i)^T C_i^{-1}(x-p_i)}}{(2\pi)^{d/2}|C_i|^{0.5}}$$

where, d = Dimension and $|C_i|$ = determinant of covariance matrix

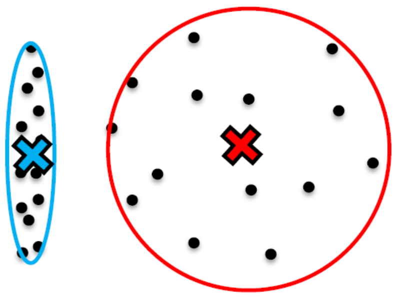


Figure 3: Clustering of Data points into their respective Gaussian distribution [2]

4.3 Linear Regression

Linear Regression is a statistical supervised learning used to predict the quantitative target variable by formulating the linear relationship with its dependent variables. In our analysis linear regression is used to examine word's abstractness/concreteness based on its feature vectors to analyze if they are linearly related to each other.
Given a data set $[y_i; x_i]_{i=1}^n$ where n = total number of observations.
A linear regression model assumes that the relationship between the dependent variable y and regressor x is linear. This relation is modelled with addition of a white Gaussian Noise term ϵ

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i \quad i = 1...n$$

4.4 Mean Silhouette Score

The Mean Silhouette Score is a metric that provides an evaluation of clustering validity, and could be used to select an appropriate number of clusters. Each cluster is represented by a so-called silhouette, which is based on the comparison of its tightness and separation [8]. The silhouette score shows which objects lie well within their cluster and which ones are merely somewhere in between clusters. That is calculated using the mean intra-cluster distance (denoted by 'a' in below equation) and the mean nearest-cluster distance (denoted by 'b' in below equation) for each sample. The function returns mean Silhouette coefficient for all samples [7].

$$Sc = \frac{(b-a)}{max(a,b)}$$

Euclidean distance is used to calculate intra-cluster distance and nearest-cluster distance.
Euclidean distance between two points p and q given by:

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

where q_i and p_i are Euclidean vectors, starting from the origin of space (initial point)

Its value ranges from -1 to 1.

- 1: Clusters are well apart from each other and distinguished.
- 0: Clusters are indifferent, or we can say that the distance between clusters is not significant.
- -1: Clusters are assigned in the wrong way (the data points in the cluster would be more appropriate if it was clustered in its neighboring cluster).

Thus the mean Silhouette score is used in our analysis to select the best number of clusters/k identified using MoG clustering algorithm.

4.5 Root Mean Square Error (RMSE)

This metric is used to measure the quality of the fit by measuring the differences between values predicted ($\hat{y_i}$) by a regression model and the observed values (y_i). It is a quadratic scoring rule that measures the average magnitude of the error. This metric is used in the analysis to measure the goodness of results obtained while examining the word's abstractness/concreteness based on its feature vectors.

$RMSE = \frac{\sum_{i=1}^{n}\sqrt{(y_i - \hat{y_i})^2}}{n}, n > 0$ [11]

The values of this metric can range from 0 to 1 where lower values corresponds to better results.

4.6 Omnibus K-squared normality test

The Omnibus K-squared normality test [4],[3] is used to determine if a data distribution is well-modeled by a normal distribution and to compute how likely it is for a random variable underlying the data distribution to be normally distributed. The test combines the Skewness and Kurtosis of random variable into a single test statistic.

$$K = z(s_k)^2 + z(k)^2$$

The above equation is the sum of squares of the z-score of skewness (sk) and kurtosis(k) of the distribution. This sum in turn forms an approximately Chi-square(2) distribution. The hypothesis is that if the data is normally distributed the p-value of K must be greater than 0.05.

Skewness: It is a measure of asymmetry. A distribution is symmetric if its both left and right looks the same from the center point of the distribution. For Normal distribution the skewness is "0".

The third moment characterizes the asymmetry of a distribution.

$$sk = \frac{\mu_3}{\sigma^3} = \frac{E\{(x - E[x])^3\}}{\sigma^3}$$

Kurtosis: It is a measure of whether the distribution is heavy-tailed/having outliers or light-tailed/not having outliers relative to a normal distribution. High value of kurtosis implies that the distribution has outliers. The Kurtosis for standard normal distribution is 3. It is the ratio of fourth moment and square of the variance.

$$k = \frac{\mu_4}{\sigma^4}, \qquad \mu_4 = E\{(x - E[x])^4\}$$

The test rejects the hypothesis of normality when the p-value is less than or equal to 0.05. And we can state that the data does not belong to normal distribution with a confidence of 95%.
This metric is used to validate if the feature vectors generated by the BERT are normally distributed or not.

5 Distribution of feature vectors

A data distribution is a function or listing which shows the spread of the data. By knowing the distribution of the data we can calculate the quantities such as probability of any one particular observation in the sample space and the likelihoods of observations. The normal distribution is one such know data distribution and it is most important distribution in statistics. Therefore we wanted to analyze whether the feature vectors are normally distributed or not. For this validation we used the Omnibus K-squared normality test and considered the feature vectors of different words and "belief" is one such word. We obtained a p-value = 0 as test result for vectors of "belief" from all layers, this implies that the feature vectors generated by BERT are not normally distributed. In order to visualize the normality we used Quantile-Quantile (QQ) plot which can be seen Figure 4. In the Figure 4 the X-axis represents the quantile values of theoretical normal distribution and the quantiles calculated on the feature vectors are in Y-axis. If the data distribution is normal then all the blue points should have been aligned on the red line and in our case it is not true. Thus the feature vectors generated by BERT Large model are not normally distributed.

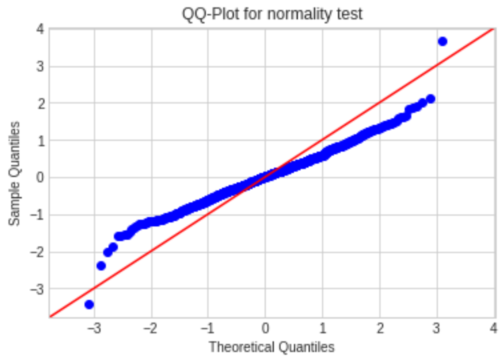


Figure 4: QQ-Plot applied on the feature vectors generated from layer 25

6 Methodology

In our analysis, using BERT the feature vectors are generated at all the layers for a set of most frequently occurring words in a sample English corpus.A maximum Likelihood estimate was applied on vectors associated with each word to analyze if that word has a different context in different sentences.The mixture of the Gaussian clustering method is used to find the number of clusters with clear distinction, where each cluster represents the contextual grouping of vectors corresponding to each word. We hypothesise that the abstract words will be having more number of clusters than the concrete word. Finally, an experiment is done to check that by using the distribution of feature vectors if we could the degree of abstractness/concreteness of the words.The experiment is performed on the vectors obtained at all the layers separately to examine if any layer contributes more towards the cause. The steps are presented in Figure 5

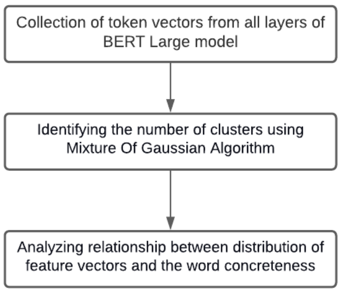

Collection of token vectors from all layers of BERT Large model
Identifying the number of clusters using Mixture Of Gaussian Algorithm
Analyzing relationship between distribution of feature vectors and the word concreteness

Figure 5: Process Flow

6.1 Collection of token vectors from all the layers of the BERT Large model

For every word in a sentence, BERT Large generates token vectors of length 1024. Token vectors were accumulated for every occurrence of the word from all 25 layers. For this experiment, 1000 unique words which are independent from each other and that occurred several times in 23,300 sentences were analyzed.

6.2 Identifying the number of clusters

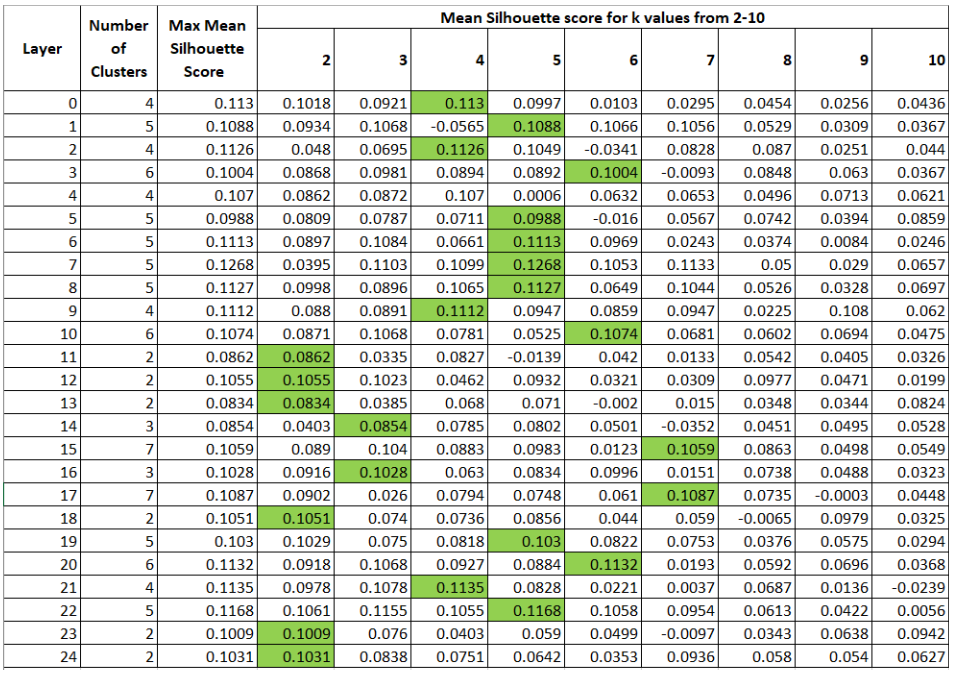
Feature vectors for each words are now given as input to the MoG clustering method. The MoG is executed for K values ranging from 2 to 10 with an expectation to find the optimal number of clusters with clear distinction. Mean silhouette score is used to evaluate the goodness of the clusters. The same process is repeated for all 1000 words and the optimal number of clusters from the feature vector's distribution are identified.

6.3 Analyzing the relationship between distribution of feature vectors and the word concreteness

Linear regression is used to validate if there is any linear relation between the degree of abstractness based on the number of clusters. The regression model is evaluated using root mean square error. In addition, exploratory analysis is done to visualize the relationship that exists between the concreteness and the number of cluster.

7 Results

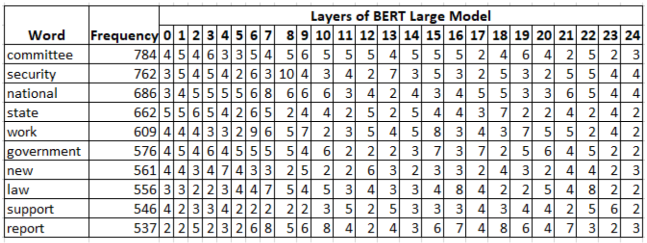
For the purpose of analysis the word 'Government' has been considered. "Government" is considered as an abstract word as it symbolises the system of people, laws, and officials that define and control a place. The said word has occurred 576 times in different sentences. As we could see in the Figure 6 the mean silhouette scores are closer to zero for all layers(similar results are obtained for all other words). This infers that the clusters are identical in nature and we could not discriminate the vectors into different well defined clusters.



Layer	Number of Clusters	Max Mean Silhouette Score	Mean Silhouette score for k values from 2-10								
			2	3	4	5	6	7	8	9	10
0	4	0.113	0.1018	0.0921	0.113	0.0997	0.0103	0.0295	0.0454	0.0256	0.0436
1	5	0.1088	0.0934	0.1068	-0.0565	0.1088	0.1066	0.1056	0.0529	0.0309	0.0367
2	4	0.1126	0.048	0.0695	0.1126	0.1049	-0.0341	0.0828	0.087	0.0251	0.044
3	6	0.1004	0.0868	0.0981	0.0894	0.0892	0.1004	-0.0093	0.0848	0.063	0.0367
4	4	0.107	0.0862	0.0872	0.107	0.0006	0.0632	0.0653	0.0496	0.0713	0.0621
5	5	0.0988	0.0809	0.0787	0.0711	0.0988	-0.016	0.0567	0.0742	0.0394	0.0859
6	5	0.1113	0.0897	0.1084	0.0661	0.1113	0.0969	0.0243	0.0374	0.0084	0.0246
7	5	0.1268	0.0395	0.1103	0.1099	0.1268	0.1053	0.1133	0.05	0.029	0.0657
8	5	0.1127	0.0998	0.0896	0.1065	0.1127	0.0649	0.1044	0.0526	0.0328	0.0697
9	4	0.1112	0.088	0.0891	0.1112	0.0947	0.0859	0.0947	0.0225	0.108	0.062
10	6	0.1074	0.0871	0.1068	0.0781	0.0525	0.1074	0.0681	0.0602	0.0694	0.0475
11	2	0.0862	0.0862	0.0335	0.0827	-0.0139	0.042	0.0133	0.0542	0.0405	0.0326
12	2	0.1055	0.1055	0.1023	0.0462	0.0932	0.0321	0.0309	0.0977	0.0471	0.0199
13	2	0.0834	0.0834	0.0385	0.068	0.071	-0.002	0.015	0.0348	0.0344	0.0824
14	3	0.0854	0.0403	0.0854	0.0785	0.0802	0.0501	-0.0352	0.0451	0.0495	0.0528
15	7	0.1059	0.089	0.104	0.0883	0.0983	0.0123	0.1059	0.0863	0.0498	0.0549
16	3	0.1028	0.0916	0.1028	0.063	0.0834	0.0996	0.0151	0.0738	0.0488	0.0323
17	7	0.1087	0.0902	0.026	0.0794	0.0748	0.061	0.1087	0.0735	-0.0003	0.0448
18	2	0.1051	0.1051	0.074	0.0736	0.0856	0.044	0.059	-0.0065	0.0979	0.0325
19	5	0.103	0.1029	0.075	0.0818	0.103	0.0822	0.0753	0.0376	0.0575	0.0294
20	6	0.1132	0.0918	0.1068	0.0927	0.0884	0.1132	0.0193	0.0592	0.0696	0.0368
21	4	0.1135	0.0978	0.1078	0.1135	0.0828	0.0221	0.0037	0.0687	0.0136	-0.0239
22	5	0.1168	0.1061	0.1155	0.1055	0.1168	0.1058	0.0954	0.0613	0.0422	0.0056
23	2	0.1009	0.1009	0.076	0.0403	0.059	0.0499	-0.0097	0.0343	0.0638	0.0942
24	2	0.1031	0.1031	0.0838	0.0751	0.0642	0.0353	0.0936	0.058	0.054	0.0627

Figure 6: Clusters selection method

Layer wise clusters identified for all words. For the purpose of analysis some sample words are considered which can be observed in Figure 7. The words such as "National","Government","Security","Support" are considered as abstract words.Also the concreteness ratings of these words are 2.24,2.88,2.82, 2.83 respectively. These have occurred more than 500 times in different sentences, and the number of clusters identified at different layers are very less and are not inclining with our initial hypothesis.



Word	Frequency	Layers of BERT Large Model																								
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
committee	784	4	5	4	6	3	3	5	4	5	6	5	5	5	4	5	5	5	2	4	6	4	2	5	2	3
security	762	3	5	4	5	4	2	6	3	10	4	3	4	2	7	3	5	3	2	5	3	2	5	5	4	4
national	686	3	4	5	5	5	5	6	8	6	6	6	3	4	2	4	3	4	5	5	3	3	6	5	4	4
state	662	5	5	6	5	4	2	6	5	2	4	4	2	5	2	5	4	4	3	7	2	2	4	2	4	2
work	609	4	4	4	3	3	2	9	6	5	7	2	3	5	4	5	8	3	4	3	7	5	5	2	4	2
government	576	4	5	4	6	4	5	5	5	5	4	6	2	2	2	3	7	3	7	2	5	6	4	5	2	2
new	561	4	4	3	4	7	4	3	3	2	5	2	2	6	3	2	3	3	2	4	3	2	4	4	2	3
law	556	3	3	2	2	3	4	4	7	5	4	5	3	4	3	3	4	8	4	2	2	5	4	8	2	2
support	546	4	2	3	3	4	2	2	2	2	2	3	5	2	5	3	3	3	4	3	4	4	2	5	6	2
report	537	2	2	5	2	3	2	6	8	5	6	8	4	2	4	3	6	7	4	8	6	4	7	3	2	3

Figure 7: Number of clusters identified at the output of each layer

The graphs is Figure 8 reflects that the regression errors are very high. Which imply that there is no linear relationship identified between the number of clusters and the concreteness scores.

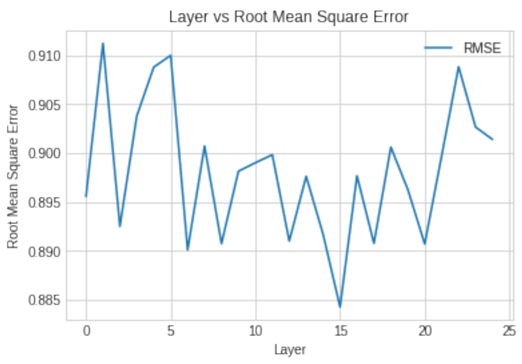


Figure 8: Regression Errors

Figure 8 showcases the regression errors where the Layers are on the X axis and the error rates are on the Y - axis.

In order to validate the regression results we conducted an exploratory analysis which can be visualized in the Figure 9. From the figure it is clear that there is little to no relationship between the concreteness ratings and the number of clusters identified from the vectors obtained from different layers of BERT model. In addition to using clusters from individual layer we took an average of clusters obtained from all layers for each word expecting to see some relation with concreteness ratings. That is visualized in Figure 10, where again the data spread is random.

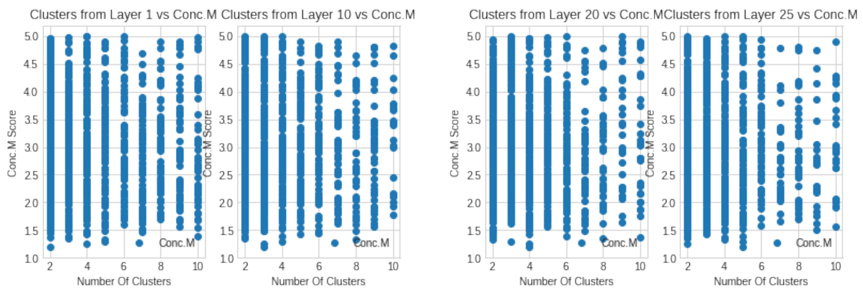


Figure 9: Concreteness ratings vs the clusters obtained at different layers.

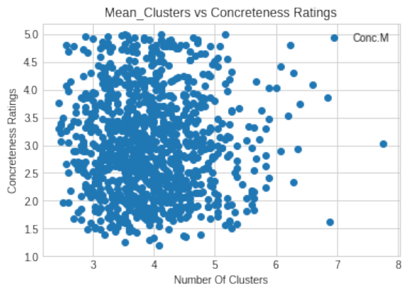


Figure 10: Average of Number clusters vs Concreteness Ratings

8 Discussion and Conclusion

From the results obtained above we notice that the feature vectors generated by BERT at all the layers are not grouped into well defined clusters. The linear regression results had high error values as there is little to no linear relation between the number of clusters and the concreteness ratings. Thus we cannot examine the concreteness ratings for the words based on the number of clusters. From the silhouette scores we could see that the clusters are nearly identical which means that their cluster centers are very close enough. This suggests that the contextual differences in the distribution of feature vectors from individual layers are not very evident enough for MoG algorithm. This implies that the MoG algorithm is not suitable for this exercise which makes our hypothesis fail.

The results obtained in this paper servers as a solid foundation for the future research into other clustering methods such as Hierarchical clustering methods and Density based clustering techniques. It would be interesting to see if these feature vectors generated by BERT can be clustered well using these other clustering methods. In our experiments we have considered the feature vectors from individual layers, instead can combine the feature vectors from certain layers and then perform the analysis. Also it may be the case that the general class of clustering methods are not suitable to sufficiently discriminate between the feature vectors. Another idea can be to generate feature vectors for a set of concrete and abstract words using binary classifiers and try to classify those vectors into respective classes. Even though these approaches does not give any concrete results towards understanding the feature vectors of BERT, it does not mean those feature vectors do not capture enough information about concreteness/abstractness of words.

9 Source Code

The input data, figures, results and source codes can be found here.

References

[1] Jeanette Altarriba, Lisa M Bauer, and Claudia Benvenuto. "Concreteness, context availability, and imageability ratings and word associations for abstract, concrete, and emotion words". In: *Behavior Research Methods, Instruments, & Computers* 31.4 (1999), pp. 578–602.

[2] Magnus Borga. "Lecture 8 Unsupervised Learning –Dimensionality Reduction, Clustering and Auto Encoders". In.

[3] Ralph D'Agostino and E. S. Pearson. "Tests for Departure from Normality. Empirical Results for the Distributions of b2 and b1". In: *Biometrika* 60.3 (1973), pp. 613–622. ISSN: 00063444. URL: http://www.jstor.org/stable/2335012.

[4] Ralph B. D'Agostino. "An Omnibus Test of Normality for Moderate and Large Size Samples". In: *Biometrika* 58.2 (1971), pp. 341–348. ISSN: 00063444. URL: http://www.jstor.org/stable/2334522.

[5] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *CoRR* abs/1810.04805 (2018). arXiv: 1810.04805. URL: http://arxiv.org/abs/1810.04805.

[6] Douglas L Nelson and Thomas A Schreiber. "Word concreteness and word structure as independent determinants of recall". In: *Journal of Memory and Language* 31.2 (1992), pp. 237–260. ISSN: 0749-596X. DOI: https://doi.org/10.1016/0749-596X(92)90013-N. URL: https://www.sciencedirect.com/science/article/pii/0749596X9290013N.

[7] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[8] Peter J. Rousseeuw. "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis". In: *Journal of Computational and Applied Mathematics* 20 (1987), pp. 53–65. ISSN: 0377-0427. DOI: https://doi.org/10.1016/0377-0427(87)90125-7. URL: https://www.sciencedirect.com/science/article/pii/0377042787901257.

[9] Rita Sevastjanova et al. "Explaining Contextualization in Language Models using Visual Analytics". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 2021, pp. 464–476.

[10] Ashish Vaswani et al. "Attention is All You Need". In: 2017. URL: https://arxiv.org/pdf/1706.03762.pdf.

[11] Wikipedia contributors. *Root-mean-square deviation — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Root-mean-square_deviation&oldid=1037360077. [Online; accessed 4-January-2022]. 2021.

[12] Yonghui Wu et al. "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation". In: (Sept. 2016).

10 Appendix

In this section we discuss the implementation details related to memory management and parallel execution.

10.1 Accumulation of feature vectors

For every word in a sentence, BERT Large generates token vectors of length 1024. Token vectors were accumulated for every occurrence of the word from all 25 layers. For this experiment, 1000 unique words which are independent from each other and that occurred several times in 23,300 sentences were analyzed. Accumulating token vectors in memory would take a massive amount of memory after scanning a few thousand sentences. To mitigate memory issues, vectors were collected in memory for each word, up to some number of occurrences (S) and beyond this, these vectors were dumped into a binary pickle file. For every S occurrences, the pickle file is updated for that word.

10.2 Layer-wise grouping of token vectors

The BERT model returns vectors from all layers at once. For example, when a word w1 occurs in a sentence s1 we can collect vectors from all layers at once. Our implementation writes all vectors for every (s) occurrence of the word into a pickle file.

Therefore once all vectors got accumulated, all pickle files pertaining to word w1 are read into memory and then the vectors are grouped based on the layers to analyze the layer-wise vector distribution.

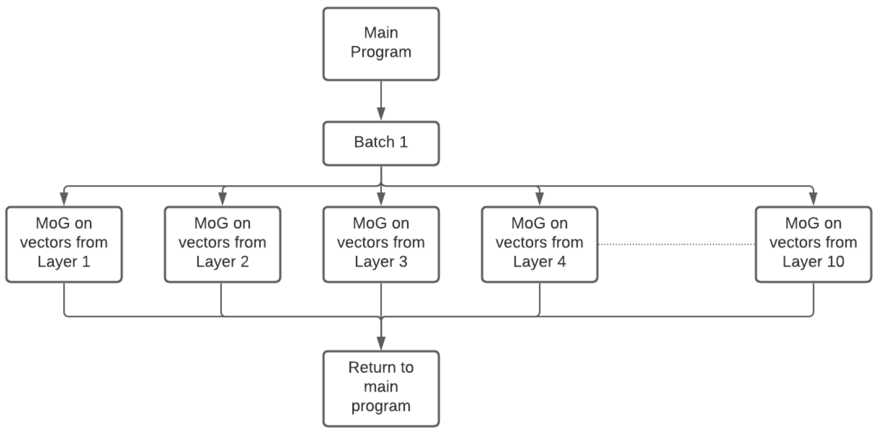

Main Program
Batch 1
MoG on vectors from Layer 1
MoG on vectors from Layer 2
MoG on vectors from Layer 3
MoG on vectors from Layer 4
MoG on vectors from Layer 10
Return to main program

Figure 11: Forking

10.3 Enable parallel execution

The experiment involves application of MoG clustering on the collection of vectors obtained at the output of all 25 layers for 1000 words. This is a time consuming process. Therefore to make the process quicker, dynamic forks (threads) are made with the application of multi-threading. Number of forks are based on the range of layers that are provided for clustering. Since there are 25 layers, three batches of layers(0-9,10-19,20-24) have been created which executes sequentially. Each batch splits into forks equivalent to number of layers that has been provided. Within each forks there executes clustering and evaluation process in parallel as depicted in the Figure 11.