Chair of Digital Libraries and Web Information Systems

5981P TEXT MINING PROJECT

PROF. DR. Markus Endres, DR. Jelena Mitrovic

Classification Using BERT Language Model

Solomon Raj Usthela

Contents

1 Abstract

Natural Language Processing (NLP)is a diversified field with several distinct tasks, in which most data sets contain only a few hundreds or thousands human-labeled training examples. However, modern day NLP models see benefits from such huge amounts of data, but the major changes occurred when the google community introduced a new language model called ”BERT” abbreviated as Bidirectional Encoder Representations from Transformers. This language model is based on transformer learning and attention mechanism. The main idea for this project to use Bert language model is for classification purpose. Prior to BERT model, numerous transformer learning architectures was built for multiple NLP task such as machine translation. In our case, We have used BERT language model for classification. We are building features for a BERT model in a way that BERT model was built on it. After that, we have passed these features to BERT and assigning activation function to get an output. We will see our research question and proposed solution for classification in a further section.

2 Introduction

As the internet and web are growing exponentially, the amount of text data being generated is huge and it is expected to grow more in the coming years. To extract information or knowledge we need to structure the unstructured text data. We know that there are many kinds of data available such as images, audios, texts, videos. These data can be used to perform certain tasks like discovering useful patterns for example pattern recognition, discovering knowledge accurately and extracting information efficiently without the need of manual work. All these tasks led to the rapid expansion in research and technologies such as data mining, machine learning, natural language processing, information extraction.

The Main objective of a language modelling is language understanding and it requires modeling complex language phenomena to deal with challenging language understanding problems such as sentiment analysis, translation, question answering. A key feature of language modelling is that, it aims to predict the next word given a previous sequence of words. It is able to do this because the language models are typically trained on very large data sets in an unsupervised manner [2]. For instance to predict the next word in the sentence the model needs to know lot of world knowledge and language. The full form of BERT is as follows.

BIDIRECTIONAL- This model reads from both direction, left as well as the right to gain the understanding of the text and can make a better decision.
ENCODER- Encoder-Decoder has been used for quite some time into the AI indus-

try. The architecture deploys already well known an encoder-decoder mechanism for carrying out NLP task(like seq2seq).
REPRESENTATION-Encoder Decoder architecture is represented using a transformer, Traditionally it has been represented with RNN.
TRANSFORMER- BERT makes use of Transformer, an attention mechanism that learns contextual relations between words in a text.Transformer includes two separate mechanisms, first an encoder that reads the text input and second a decoder that produces a prediction for the task. Since BERT model is built on using three features 1-Token embedding, 2-Sentence embedding, 3-Transformers positional embedding as mentioned in Fig. 4 [1]. BERT overall goal is to generate a language model. On the other hand,for classification, we are using supervised label data set. In supervised label data set, the input-output pair will be there. We will divide our data set in train and test set so that we can evaluate our BERT model [3].

There is a challenge when training language models to define a predictive objective. Many models predict a sequence of the next word (for example ”Yesterday i ate ”), a directional strategy that inherently limits learning of the context. BERT utilizes two training approaches to solve this challenge: ’Masked Language model’(ML) and ’Next sentence prediction’(NSP).

3 Motivation

Numerous researchers from various domains have done a lot of efforts studying the shortage of training data in NLP which is one of the biggest challenges. Today’s top performing systems in NLP use large neural architectures to obtain token embedding. Things get worse when using fine tuned models as it requires a lot of memory and time but BERT model overcomes these issues with its effective techniques adding to it, this language model which is bidirectionally trained will have a deeper sense of language context and flow, than with single direction language models which motivated us to implement this topic.

4 Related Work

Since there are many approaches which are defining language representation, so here we will talk about widely-used methods or models in this section.

4.1 OpenAI GPT-2

GPT-2 is trained to predict the next word from the 40GB of text data. This framework is also based on a transformer-based approach. Due to concern related to

malicious application the full version of this model has not been released. Since GPT-2 is a large transformer-based model with 1,5 billion parameters and trained on a data set of 8 billion web pages. Normally this model is trained to predict next word in the text. GPT-2 shows a wide range of capacities, including the capacity to produce unprecedented quality conditional synthetic text samples. Furthermore, GPT-2 can outperform the language models even without using the domain-specific training data set. when the raw data is processed in the GPT-2 model then it can perform tasks like answering the questions, reading understanding, summarizing and translating the text[5].Moreover, this model is capable of generating samples from a variety of texts. However, we have noted different failure as well in this model, such as repetitive text, word modelling failure (model sometimes gives meaningless results for e.g 'An old apple sat down once more'.) Because of these uncertainties, the GPT model was knocked off by BERT. One reason for the failure of GPT was because it was pre-trained using traditional language modeling. BERT, on the other hand, is pre-trained using masked language modeling, which is more like fill in the blanks i.e predicting missing ['masked']words by given the words that appear before and after[6].

4.2 ELMo

ELMo is a deep contextualized representation of words that models both (1) complex word-use characteristics (e.g., syntax and semantics) and (2) how these uses differ across linguistic contexts[7].Elmo uses two LSTM's, one LSTM goes in left to right and another right to left. A single LSTM takes in an input sequence one by one E1, E2,,,En, it produces hidden states at each step that is the result of the previous hidden state and the current token. These hidden states are now the embedding of the token(E1, E2,,,En). The word vectors are just one vector per word so they are not in isolation anymore, basically, we need an entire sequence to compute the word vector as a result of this LSTM. This is more powerful because it can give individual words, each word is kind of unique and depending on the surrounding word.Given word would have similar embedding or a similar word vector all across the language. However, we can fine-tune it to particular sentences and we can completely change its meaning if its a kind of a word that has completely new meaning in that sentence.The fundamental limitation here is, in ELMo model we have information from the left LSTM and from the right LSTM. However, in open AI GPT it is very shallow because it is a concatenation of left-facing LSTM and a concatenation of right-facing LSTM. On the other side, ELMo is still sub optimal. Ultimately what we want is a single model to output word vectors to interpret a language that can look at both left and the right at the same time then incorporate information simultaneously and this makes us implement BERT model, which is an

attention based model with positional encoding to represent word positions.

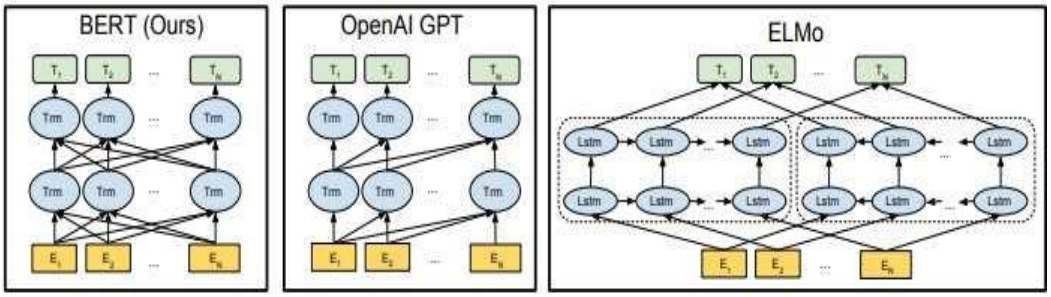

Figure 1: Illustration of pre-training models architecture. BERT uses a bi-directional transformer. OpenAI GPT uses a left-to-right Transformer and ELMo uses Left to right, Right to left LSTM. Among those BERT and OpenAI GPT are fine-tuning approach where ELMo is feature-based approach(image source-Original paper[1])

5 BERT

We will include detailed implementation of Bert in this section. There are two main steps in this framework: Pre-training and Fine-tuning. Throughout pre-training, the model is trained on unlabeled data over completely different pre-training tasks. For fine-tuning, the BERT model is first initialized with the pre-trained parameters and all of the parameters are fine-tuned using labelled data from the downstream tasks[1] such as Masked LM and Next sentence prediction. Now we will discuss two main steps of BERT language model.

5.1 Pre-training BERT

We will not use traditional left to right and right to left language model to pre-train BERT. Instead, we will train the BERT model with two unsupervised tasks, such as Masked LM and Next sentence prediction, which will be discussed in detail in further sections.

5.1.1 Masked Language ML

It is reasonable to believe that deep bidirectional models are better than traditional language models which perform either shallow concatenation of left to right and right to left(openAI GPT) model or the LSTM based approach model(ELMo). The function of this approch is: In the sequence of the sentence, 15% of words will be replaced by Mask tokens, and the model will attempt to predict the original

vocabulary ID based only on the context provided words in sequence. Prediction of the Output words requires to predict the next word in the sequence, a classification layer will be added to the encoder output. Then Multiplying the output vectors by the embedding matrix, converting them into the vocabulary dimension, and then finally, it will calculate the probability of every word in the vocabulary. However, it only predicts the masked values and refuses to consider non-masked words. The downside of this approach is we are creating a mismatch between pre-training and fine-tuning since the [MASK] token does not appear during fine-tuning[1].

5.1.2 Next Sentence Prediction(NSP)

Bert model is also pre-trained for next sentence prediction. This allows us to use fine-tuned BERT on downstream tasks like intent detection, question answering, sentiment classification[8], which is based on understanding the relationship between two-sentence, which language model can not perform directly. In order to do this, we have to train our language model which can understand sentence relationship.In the training phase, the model receives couple of sentences as input and it will learn whether second sentence in the pair is the subsequent sentence in the original document. consequently, 50% of the second sentence is the subsequent sentence in the document, and for the rest, 50% is the random sentence from the corpus is chosen as a second sentence. For example, when choosing P and Q, 50% of the time Q will be the actual next sentence that follows P(labeled as Isnext). and 50% of time it is a random sentence from the corpus (labeled as Notnext). By this random sentence will be disconnected from the first one. Advantage of Bidirectional context is,no model could pre-train the model with a profound bidirectional context before BERT. In the traditional models, it uses word to predict the next word where in BERT we are applying the same idea but in both directions.

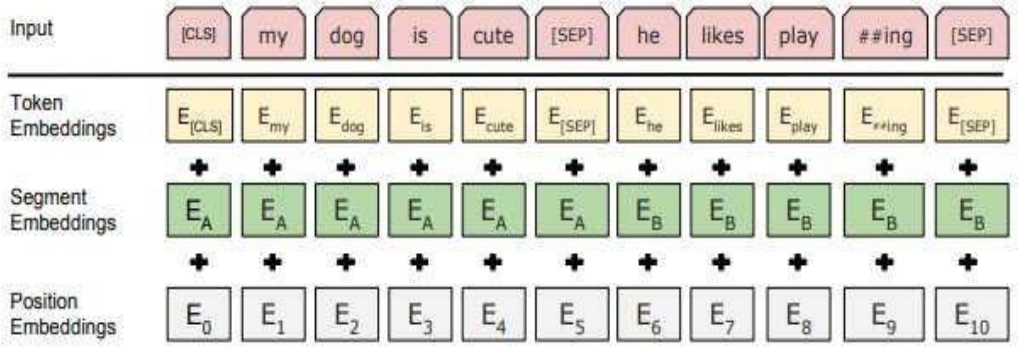

Figure 2: Classification using BERT Language Model[1].

5.2 Fine Tuning BERT

Next strategy after pre-training BERT model is Fine-tuning. The easiest task to fine-tune BERT is a sequence level task where the model takes a pair of sequences and pools the representation of the first token in the sequence. The initial BERT model was trained for a masked language model and next-sentence prediction tasks, which includes layers for language model decoding and classification. These layers will not be used for fine-tuning the sentence pair classification[10]. The easiest task to fine-tune BERT is a sequence level task where we basically have seen layers will not be used for fine-tuning the sentence pair classification, For example, there are more sophisticated tasks that we can supervise the data for. We input two sentences (as illustrated in figure:3 below) which means it will take two-sentence as input, that input sequence has one or two segments that the first token of the sequence is always [CLS] which contains the classification embedding and another special token [SEP] is used for separating segments[11].The input embeddings will be then passed through BERT model and from there we will get the layer of class label corresponding to the first token.

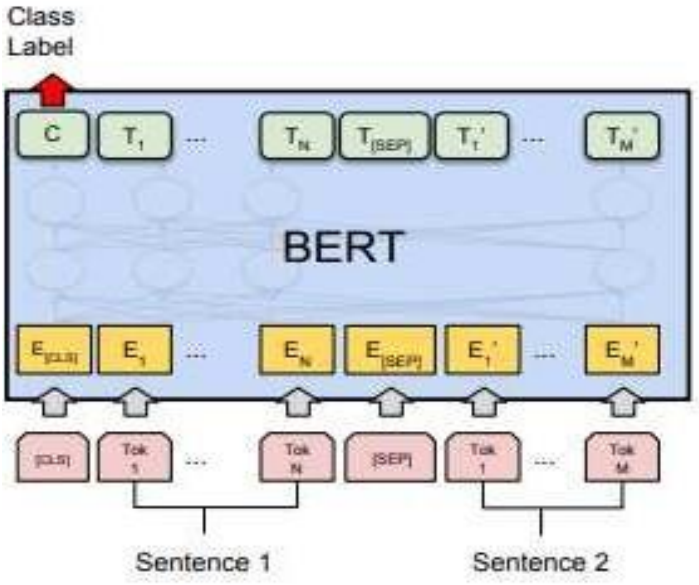

Figure 3: Sentence pair classification task.[1]

6 Methodology

In this project, we are using IMDB data set to classify text. In our dataset, there are two columns (1) Reviews and (2) Sentiment. Since there are 50K reviews and

sentiments. We will work with 2K reviews (just to keep computation small). To perform the classification we have downloaded pre-trained BERT model from google and we set the initial parameters of the model same as suggested in the original algorithm. As we have two types of classes (Positive and Negative), we are encoding those sentiments into 0 and 1, which means the review whose sentiment is positive will be labelled as 1 and negative as 0. To perform such a task we are splitting our data into train-test split, in which we will perform training on 90% of the data and testing on rest 10% of the data.

We merge our newly trained vocabulary and configuration files with original files downloaded from the BERT algorithm. The code we use to train our model comes from Google's predefined BERT model. Meanwhile, we train our model to perform the sentiment analysis on the reviews provided to the trained model. We are adding input-ids, input mask, segment id, label id on the created feature set. The features will be represented by class input features.
Input_ids: list of numerical ids for the tokenized text.
input_mask: it will be encoded to 1 for real tokens and 0 for the padding tokens.
segment_ids: it will segment two different sentences while encoding 0 to the first sentence and 1 to the next sentence.
label_ids: encoded labels for the text.

Fig. 4 shows an overview of the work-flow of our proposed system. Each of the phases shown in the figure are elaborated in details in the section that follow.

6.1 Data Acquisition

The role of acquiring data is highly important as it impacts any further processing to be performed on collection data. We have taken IMDB review dataset as a data which has movie reviews and corresponding binary sentiments as positive or negative from a well known website Stanford.edu[9]. It consists of two columns with 50K movie reviews and their sentiments which is overall 50 percent positive reviews and 50 percent negative reviews.

6.2 Feature Engineering

The act of feature engineering is to create features that help the BERT model work by using the data acquired from dataset. It extracts relevent features from the acquired data and generates features for further processing. The produced features are then given as input for embedding layers for further processes.

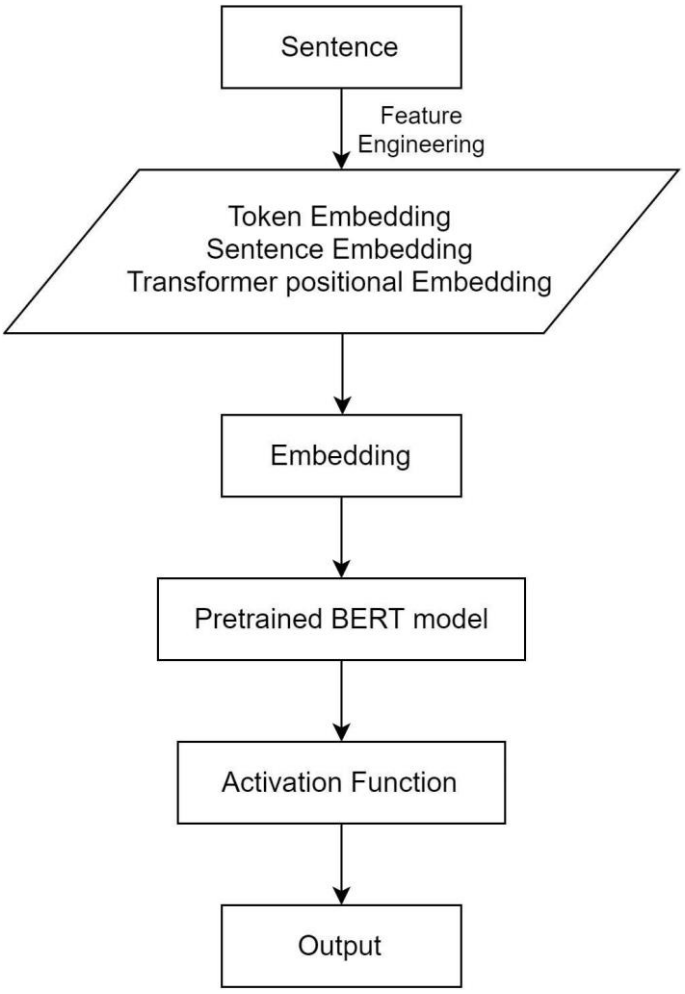

Figure 4: Work flow of Classification using BERT Language Model

6.3 Token Embedding

The token embedding layer transforms the words from the sentence acquired into vector representations of fixed dimensions. Tokenisation of the input text is done before passing it to the token embedding layer. Extra token are additionally added to the start and end of the sentence. For classification tasks, these tokens serve as an input representation. These tokens also separate a pair of input texts respectively.

6.4 Sentence Embedding

Given a pair of input texts in text classification, BERT solves the NLP tasks. BERT uses sentence embedding to distinguish the inputs in a given pair. It uses two vector

representation(index 0 and index 1) for the pair of sentences provided.An example of such a problem is classifying whether two pieces of text are semantically similar. The pair of input texts are simply concatenated and passed to the model.

6.5 Transformer positional Embedding

Input sequences of length up to 512 can be determined by BERT model. Each position is sequentially represented through a vector by incorporating sequential nature of input sequences and BERT model having learnt it.The first word in the input sequence will have identical position embedding similarly so on with other words in the input sequence with similar positioning.

6.6 Embedding

The above three steps after being processed i.e Token Embedding, Sentence Embedding and Transformer Positional Embedding, we would merge all the three embeddings to process it further.The output per token from every layer above, will be used as word embedding. This makes our model more flexible in further dimensions without losing any information.

6.7 Pretrained BERT model

The Pre-trained BERT model already has a lot of encoded information about our language model, which results in taking much less time to train fine-tuned models. This model helps in enhancing the network as the bottom layers of the model have already been trained and only needs to be gently tuned while using output as features in classification task. Overall Pretrained models could easily be fine tuned with less data and less compute time to produce the results.

6.8 Activation Function

We apply a Activation function so that the output signal would not be a linear function, let's suppose we do the sum of products of inputs(A) and their Weights(B) and apply a Activation function f(x) to it to get the output of that layer which we feed it as an input to the succeeding layer, also we can consider these output values to be 0 and 1. Activation function solves something complicated ,high dimensional,non-linear -huge data sets, when the model has a architecture which is very complicated and has lots of hidden layers, through this we can extract data/knowledge from such big data sets.

6.9 Output

	Unnamed: 0	review	sentiment	Label	predection
0	14528	This is undeniably the scariest game I've ever...	positive	1	1
1	6750	Why bother to see this movie? It probably rate...	negative	0	0
2	23678	The premise and subject about making a crimina...	negative	0	0
3	30587	I just saw this film in Austin Texas at the Au...	positive	1	1
4	39062	Journey to the Center of the Earth is the stor...	negative	0	1
5	36089	I have to say that this movie was not what i e...	negative	0	0
6	32061	What do you call a horror story without horror...	negative	0	0
7	46866	the plot of this movie revolves around this su...	negative	0	0
8	14352	This is a bit of a first for me, the first tim...	negative	0	0
9	12407	The plot is tight. The acting is flawless. The...	positive	1	1
10	5952	Masters of Horror: Right to Die starts late on...	positive	1	1
11	33718	Here he is. A new horror icon for the new mill...	negative	0	1
12	32433	OK. Who ever invented this film hates humanity...	positive	1	0

Figure 5: Model Output

After training the model we will apply the same method like label_list, seq_length and tokenizer to predict the output of the model. After training the model we will apply the same method like feature prediction, seq length to predict the output of the model. In the result, we will get the prediction of the labels and the result data will be in the form of tensor data. Fig. 5 shows the output of model where Label column shows real value and prediction column shows the predicted value from the model.After predicting the input function we can evaluate the accuracy and F1 score on test data as shown in Table 1.

	Precision	Recall	F1-Score	Support
0	0.85	0.77	0.81	90
1	0.82	0.89	0.86	110
accuracy			0.83	200
macro avg	0.84	0.83	0.83	200
weight avg	0.84	0.83	0.83	200

Table 1: Classification Report

7 Conclusion

As we have already discussed how BERT model performs effectively in text classification, we can now certainly state that using a pre-trained BERT model, a high quality model can be constructed effectively and efficiently within a given short period of time. With enough training data and considering even more training steps, we certainly can ensure higher accuracy or we can certainly improve upon impressive score upon each classification with minimal effort and training time. We also learn that BERT outperforms the other methods of pre-training NLP with its unsupervised deeply bidirectional system.

You can find the code of this system in the following link:
https://github.com/rajadurai1994/Classification-using-BERT-Language-Model

7.1 Software Requirements

- Google Colab
- Python 3.7.2
- Tensorflow

References

[1] Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

[2] Javaid Nabi. Text Classification, Language Modelling using fast.ai. https://towardsdatascience.com/machine-learning-text-classification-language-modelling-using-b1b334f2872d/

[3] Jason Brownlee. Supervised and Unsupervised Machine Learning Algorithms. https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/

[4] Jeremy Howard, Sebastian Ruder. Universal Language Model Fine-tuning for Text Classification.

[5] Alec RadfordJeffrey WuDario AmodeiDaniela AmodeiJack ClarkMiles BrundageIlya Sutskever Better language models and their implications. https://openai.com/blog/better-language-models/

[6] Jesse vig OpenAI GPT-2: Understanding Language Generation through Visualization https://towardsdatascience.com/openai-gpt-2-understanding-language-generation-through-visualization-8252f683b2f8

[7] Peters, Matthew E. and Neumann, Mark and Iyyer, Mohit and Gardner, Matt and Clark, Christopher and Lee, Kenton and Zettlemoyer, Luke Deep contextualized word representations https://allennlp.org/elmo

[8] Ceshine Lee News Topic Similarity Measure using Pretrained BERT Model https://medium.com/the-artificial-impostor/news-topic-similarity-measure-using-pretrained-bert-model-1dbfe6a66f1d

[9] Maas, Andrew L. and Daly, Raymond E. and Pham, Peter T. and Huang, Dan and Ng, Andrew Y. and Potts, Christopher. Learning Word Vectors for Sentiment Analysis. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. June 2011. Portland, Oregon, USA. Association for Computational Linguistics. http://www.aclweb.org/anthology/P11-1015

[10] Devlin, Jacob, et al. “Bert: Pre-training of deep bidirectional transformers for language understanding.” arXiv preprint arXiv:1810.04805 (2018)

[11] Chi Sun, Xipeng Qiu, Yige Xu, Xuanjing Huang How to Fine-Tune BERT for Text Classification? https://arxiv.org/pdf/1905.05583.pdf