Review

Not peer-reviewed version

A Review on BERT: Language Understanding for Different Types of NLP Task

Md Saiful Islam and Long Zhang *

Posted Date: 26 January 2024

doi: 10.20944/preprints202401.1857.v1

Keywords: Natural Language Processing (NLP); BERT; Language Model; Transfer Learning; Transformers

Review

A Review on BERT: Language Understanding for Different Types of NLP Task

Md Saiful Islam [1,2] and Long Zhang [1,2,*]

[1] Henan Key Laboratory on Public Opinion Intelligent Analysis, Zhengzhou, China
[2] School of Computer Science, Zhongyuan University of Technology, Zhengzhou, China
* Correspondence: Author: zhanglong@zut.edu.cn

Abstract: In this review paper, we discuss the use of BERT, one of the most well-liked deep learning-based language models. The model's operation mechanism, key areas of applicability to text analytics tasks, comparisons with related models for each activity, and a description of certain proprietary models are all covered in this work. The data from several dozen original scientific studies that have been published in the last few years and have garnered the greatest interest from the scientific community were systematized in order to prepare this review. All researchers and students who wish to learn about the most recent developments in the field of natural language text analysis may find this survey helpful. Bidirectional Encoder Representations from Transformers, or BERT for short, is the subject of a thorough investigation that we present. The area of natural language processing holds significant importance in the creation of intelligent systems. To do a variety of activities, one must comprehend the sentence's correct meaning in order to produce the desired result. Computers have a hard time understanding languages because of how context constantly changes. The main challenge in natural language processing tasks is getting computers to understand text context, and BERT is seen as a revolution in this regard. It picks up the language and its meaning in a manner that is quite similar to how the human brain processes meaning from sentences. Its capacity to identify a word from both the left and right contexts of a sentence makes it special. A new era in the perception and comprehension of natural languages has been ushered in with the development of BERT, which could help computers understand natural languages more fully. This study aims to provide readers with a deeper knowledge of the BERT language model and how it is applied to different NLP tasks.

Keywords: natural language processing (NLP); BERT; language model; transfer learning; transformers

1. Introduction

Automatic natural language processing revolves around the task of identifying universal representations of text. A significant advancement in this field has been the creation of pretrained text attachments, including GloVe [1] and word2vec [2]. Supervised models have consistently outperformed unsupervised models in recent years [3]. However, in recent years, models based on learning without a teacher have proliferated because they can use massive corpora of texts that are already available or automatically generated, allowing them to learn on a much larger sample and fully utilize deep learning without the need for the preparation of a specially labeled dataset.

Natural language processing problems have a wide range of applications, including conversational bots, machine translation[4], voice assistants, and online speech translation. This industry has grown rapidly in recent years, both numerically and qualitatively, in terms of the number of market applications and goods as well as the effectiveness of the latest models and their near-human language understanding.

Transformers represented a significant breakthrough in NLP tasks. Transformers are a neural network-based design with two key components: an encoder and a decoder[5].

 doi:10.20944/preprints202401.1857.v1

Transformers used as language models in GPT typically consist of a decoder stack, which means that many decoders are layered one on top of the other, with each decoder receiving the output of the previous one as an input. They use feedforward neural networks at both the encoder and decoder, as well as self-attention[6]. This combination enables the algorithm to derive meaningful context from words at the sentence level.

Text representation is an important topic in natural language processing. Text representation is a set of rules that translate natural language input information into machine-readable data[3]. A representation can also be thought of as a computer encoding of text; however, in the context of applied machine learning problems, representations that reflect the underlying content and conceptual structure of the text are more beneficial.

When every word is represented as a vector filled with zeros everywhere but one location that corresponds to the word's dictionary number, the simplest textual representations are called categorical encoding. The early days of the industry made advantage of this idea. It's conceptually quite simple, easy to implement, and doesn't require a lot of processing power[7]. But because each word is represented by a vector whose dimension is equal to the total number of words in the dictionary, such a representation ignores the semantic characteristics of words and is extremely large and redundant.

The popular bag of words approach has a similar perspective. The complete text is represented by this model as a vector with the dimension of a vocabulary, where each component denotes the quantity of a particular word that appears in the text. The categorization of texts is one task that this relatively simple model—which ignores word semantics—applies to pretty successfully.

The most useful in today's business world is text representation in the form of "attachments," which are mechanisms for encoding words as vectors with distinct semantic meanings assigned to each position. Attachments with hundreds of coordinates are most frequently used[8]. Certain words in a text can have their semantic meaning captured by attachments, which can then show that meaning as a coordinate in multidimensional space. The capacity to carry out vector operations in this semantic latent space is a well-known example of this.

Unsupervised model training on extensive text corpora is the most common method used for nesting. Filling in the blanks in the text and judging the significance of sentences are two examples of such tasks. The task of learning textual representations requires a lot of processing power. The majority of general-purpose text analysis problems are solved using pre-trained, ready-made representations[9].

2. Besic concepts of BERT

Today, the most advanced text models use transformers to teach how to represent text. Transformers are a type of neural network that are increasingly finding their use in various branches of machine learning, most often in sequence transduction problems, that is, such problems when both the input and output information is a sequence[10]. Such models use a combination of recurrent and convolutional neural networks. Regular recurrent networks perform rather poorly with long-range context[11]. However, in natural text, the representation of the token can be influenced by the context through several words and even sentences from the token itself. To take into account the long-range influence, LSTMs are used in conjunction with the attention mechanism to improve the learning efficiency, taking into account the influence of distant tokens[12].

At the end of 2018, a group of scientists from the Google AI Language laboratory under the leadership of J. Devlin presented a new linguistic model called BERT[13]. This model is intended for deep preliminary learning of bidirectional text representation for subsequent use in machine learning models. The advantage of this model is its ease of use, which involves adding just one output layer to the existing neural architecture to obtain text models that surpass the inaccuracy of all existing ones in several natural text processing problems[14][15].

There are two categories of natural text processing tasks: holistic, operating with text at the sentence level, and tokenized ones, such as answering a question and attribution of entities, which produce more detailed output at the level of individual text elements. Both categories of problems

 doi:10.20944/preprints202401.1857.v1

have recently been using pretrained models, which can significantly reduce the time for designing and training private models while maintaining a high level of efficiency[16],[12].

Regarding the approaches to pre-training models of deep text representation, two approaches are also distinguished: feature extraction and fine-tuning of models. Both approaches use the same pre-learning objective functions and unidirectional text analysis. The first, implemented, for example, in the ELMo model[17], is to use models with a task-specific architecture to train a certain representation, which will then be used as additional features in applied models. The second involves the construction of models that do not use parameters specific to a particular task but training the model, which is then supposed to be retrained by adjusting all the internal parameters of the model. This approach is used, for example, in the OpenAI GPT model[18].

The authors of the BERT model point out that a significant limitation of existing approaches is their focus, which narrows the choice of possible neural network architectures. For example, the GPT model uses left-to-right text scans so that the representation of each element of text (token) can only take into account the previous, but not subsequent tokens[11]. This may not be optimal for holistic text analysis, but for tokenized analysis, this approach significantly reduces the semantic power of the model, since the meaning of a word can depend on its context, both left and right.

BERT's design consists of many encoders layered on top of one another in a feed-forward manner, where the output from one encoder is fed as the input for the following one. Two model sizes were used to assess the results: BERTLARGE, which had 24 encoder blocks, 1024 hidden units, and 16 attention heads, and BERTBASE, which had 12 layers of transformer blocks, 768 hidden units, and 12 attention heads[13]. The study suggests two frameworks for BERT implementation: pre-training and fine-tuning.

Data used for training was composed of plain Wikipedia text corpus and Google's BookCorpus. BERT's design consists of many encoders layered on top of one another in a feed-forward manner, where the output from one encoder is fed as the input for the following one. Two model sizes were used to assess the results: BERTLARGE, which had 24 encoder blocks, 1024 hidden units, and 16 attention heads, and BERTBASE, which had 12 layers of transformer blocks, 768 hidden units, and 12 attention heads [13]. The study suggests two frameworks for BERT implementation: pre-training and fine-tuning. This is known as the Masked Language Model and BERT has a probability of each word or token in an input sequence to be masked as 15%. Besides, to avoid any mismatch between the two frameworks, it replaces the token to be masked with a [MASK] 80% of times and by a random token rest of the time[13]. This helps the model to comprehend the meaning of a word when it appears in more than one context, thus making it capable of keeping up with ever-changing nature of languages.

The next task considered while pre training was next sentence prediction (NSP) where given two sentences as input, the model was to output whether the second sentence follows the first one or not. The purpose was to capture the relationship between the input text[19]. Because of the profound bi-directional characteristic, it is possible to correctly interpret the context of individual utterances, which is crucial for this. Taking this a step further, BERT applies bi-directional cross-attention on the embedded input after combining the two phrases using specific tokens at the necessary locations[20][21].

It uses Word Piece embeddings along with three special tokens– [CLS], [MASK] and [SEP]. The [CLS] token stands for classification and is considered in classification tasks. The [MASK] token is used for masking the words randomly in the input, whereas the [SEP] token marks the end of first sentence and start of second in NSP task.

Fine tuning of BERT is relatively easy as the pre trained model remains as it is but only the required parameters as per the task are extracted and set accordingly[22]. After pre-training, the specific output is fed to the output layer of the specific task. That is, the [CLS] representation is fed to output layer for classification task, similarly question answering task is provided with token representations.

2.1. BERT Model Architecture

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 January 2024 doi:10.20944/preprints202401.1857.v1

A multi-layer bidirectional Transformer encoder makes up the architecture of BERT[16]. Stated otherwise, BERT is made up of an L-layer stack of identical Transformer encoders. Two kinds of sub-layers are present in every encoder layer. First, when encoding a particular word, a multi-head self-attention mechanism helps the brain look at other words in the sequence. The second uses a straightforward, position-wise fully connected feed-forward network (FFN) made up of two linear transformations that are applied to each position independently and in the same way.

$(W_1 \in \mathbb{R}^{d_{model} \times d_{ff}}, b_1 \in \mathbb{R}^{d_{ff}}),\ (W_2 \in \mathbb{R}^{d_{ff} \times d_{model}}, b_2 \in \mathbb{R}^{d_{model}})$ such that

$$FFN(x) = \max(0, xW_1 + b_1)\, W_2 + b_2$$

The dimensionality of input and output is dmodel and the inner-layer has dimensionality $d_{ff} = 4d_{model}$. The feed-forward network also uses a GELU activation [18], defined as

$$GELU(x) = 0.5x(1 + \tanh\left(\sqrt{\frac{2}{\pi}}(x + 0.044715x^3)\right))$$

which was shown to work better than the standard ReLU[23] within a Transformer encoder. In addition, an encoder layer employs a residual connection[17] around each of the two sublayers, followed by a layer normalization[24] such that the output of each sublayer is

$$\text{LayerNorm}(x + \text{Sublayer(x)})$$

where Sublayer(x) represents the function implemented by the sublayer itself. To facilitate these residual connections, all sublayers in the model produce outputs of the same dimension d_{model}.

As shown in Figure 3.7, BERT comes in two versions:

• BERT-base: $L = 12$, $d_{model} = 768$, $h = 12$, $d_{ff} = 3072$ (110M total parameters).

• BERT-large: $L = 24$, $d_{model} = 1024$, $h = 16$, $d_{ff} = 4096$ (340M total parameters)

Here, L denotes the number of layers, d_{model} the dimensionality of input and output of each layer, h the number of attentions heads in a self-attention sublayer, and d_{ff} the number of hidden units in a feed-forward sublayer.

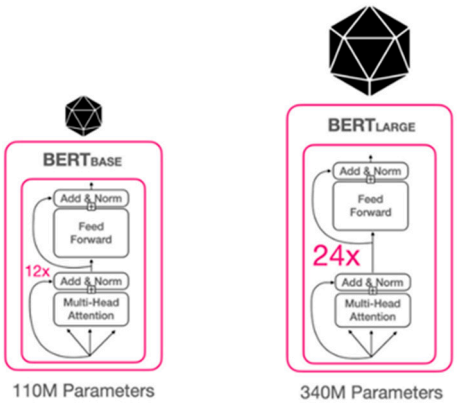

Figure 1. BERT model size and Architecture.

3. BERT Application Implementation

Transformers are used by the most sophisticated text models available today to teach text representation. Neural networks known as transformers are becoming more and more common in many areas of machine learning. Sequence transduction problems—that is, problems where the input and output data are both sequences—are the most common use for transformers[6]. Recurrent and convolutional neural networks are combined in these models. Long-range context deters regular recurrent networks from performing well. In real text, on the other hand, the context might affect the

 doi:10.20944/preprints202401.1857.v1

token's representation by influencing many words or even sentences from the token itself. To take into account the long-range influence, LSTMs are used in conjunction with the attention mechanism to improve the learning efficiency, taking into account the influence of distant tokens[25].

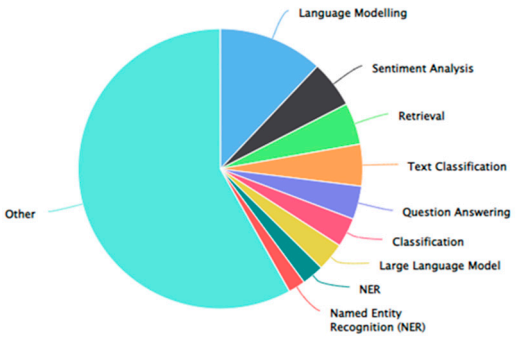

Figure 2. Different types of NLP Task.

3.1. Aspect-base Sentiment Analysis

BERT was trained on Wikipedia and BookCorpus, it frequently fails to describe or analyze reviews of electronic devices like refrigerators and washing machines due to homophones or polysemies. Homonymy is where same words convey different meanings depending on the context whereas polysemy is when a word with same meaning conveys different sentiments. Moreover, neural networks using attention prioritize prominent signal words like “good” or “awful” more than words like “crowded” that grant more sentiments[13].

Aspect sentiment classification (ASC), an AE follow-up task, is to categorize the sentiment polarity (positive, negative, or neutral) stated on an aspect that was taken from a review sentence. An aspect and a review sentence referencing that aspect are the two inputs for ASC[26]. Because the review is only a review sentence and the question is only about an aspect, ASC is therefore similar to RRC. However, instead of needing to produce a textual span, ASC only needs to output a class of polarity.

Let x = ([CLS], q_1,…, q_m, [SEP], d_1,…., d_n, [SEP]), where q_1,…, q_m, now is an aspect (with m tokens) and d_1,…., d_n is a review sentence containing that aspect. After h = BERT(x), we leverage the representations of [CLS] $h_{[CLS]}$, which is the aspect-aware representation of the whole input. The distribution of polarity is predicted as l_4 = softmax($W_4 \cdot h_{[CLS]} + b_4$), where $\in W_4 \in \mathbb{R}^{3*r_h}$ and $b_4 \in \mathbb{R}^3$ (3 is the number of polarities). Softmax is applied along the dimension of labels on [CLS]: $l_4 \in [0; 1]^3$. Training loss is the cross entropy on the polarities.

3.2. Text Classification

The process of giving free-text documents predetermined classifications is known as text classification. It has significant practical uses and can offer philosophical perspectives on document collections[27]. Text classification makes it easier to locate the information you need or streamline certain tasks by applying predetermined categories to a document. For example, an application of text classification is spam filtering in email[23].

BERT can be applied to text classification tasks by using a labeled dataset to fine-tune the pre-trained model[28][29]. The procedure is summarized as follows in general.

Prepare the textual data. Tasks like lowercasing, tokenization, and stop word removal may fall under this category.

Transform text data into numerical input features: Since BERT relies on numerical input data to function, text data must be transformed into numerical form. Techniques like sentence or word embeddings can be used for this.

 doi:10.20944/preprints202401.1857.v1

After pre-training the BERT model, load it and include a classification layer: A classification layer can be applied on top of the BERT model, which can be loaded from a checkpoint. The ultimate forecast will be made by the classification layer.

After pre-training the BERT model, load it and include a classification layer: A classification layer can be applied on top of the BERT model, which can be loaded from a checkpoint. The ultimate forecast will be made by the classification layer.

Utilizing the labeled dataset, adjust the model: Gradient descent can be used to fine-tune the model by changing the weights of the pre-trained layers and the classification layer. A small labeled dataset plus a labeled text classification dataset can be used for this.

Analyze the model using a set of tests: The model can be tested on a test set to see how well it performs once it has been adjusted. The performance of the model can be assessed using performance indicators like accuracy and the F1 score.

BERT has demonstrated state-of-the-art performance on several benchmarks and can be optimized for text classification tasks with a small labeled dataset[24].

3.3. Next Sentence Prediction (NSP)

In the BERT training process, the model is given pairs of sentences as input and learns to predict whether the second sentence in the pair is the next sentence in the original text[30]. During training, 50% of the inputs are pairs, with the second sentence being the next sentence in the original text, and the other 50% have a random sentence from the corpus as the second sentence. The presumption is that the random sentence will be separated from the first sentence.

To assist the model in distinguishing between the two sentences during training, the input is handled as follows before entering the model:

- Tokens [CLS] and [SEP] are inserted at the beginning and end of each sentence, respectively.
- Each token has a sentence embedding indicating whether it is Sentence A or B. Sentence embeddings are conceptually similar to token embeddings, but with a vocabulary of two.
- Tokens are embedded with positional information to show their place in the sequence. The Transformer paper[25] presents the notion and implementation of positional embedding.
- To determine if the second statement is connected to the first, perform the following steps:
- The Transformer model processes the full input sequence.
- The [CLS] token output is transformed into a 2×1 vector using a simple classification layer (learned weights and biases).
- Calculating IsNextSequence probability using softmax.

When training the BERT model, Masked LM and Next Sentence Prediction are trained concurrently with the goal of reducing the combined loss function of both techniques.

3.4. Question Answering

BERT has been effectively used in question-answering tasks, such as the Stanford Question Answering Dataset (SQuAD). By fine-tuning BERT on SQuAD, models obtained cutting-edge performance in extracting answers from the provided sections.

The Question Answering System uses BERT to take two parameters, the input question and the passage, as a single packed sequence. The input embeddings are the total of the token and segment embeddings[26].

Token embeddings: A [CLS] token is added to the input word tokens at the start of the question, and a [SEP] token is inserted at the end of both the question and the paragraph.

Segment embeddings: Each token is assigned a tag indicating whether it is Sentence A or Sentence B. This enables the model to distinguish between sentences[31]. BERT has been effectively used in question-answering tasks, such as the Stanford Question Answering Dataset (SQuAD). By

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 January 2024 doi:10.20944/preprints202401.1857.v1

fine-tuning BERT on SQuAD, models obtained cutting-edge performance in extracting answers from the provided sections.

The Question Answering System uses BERT to take two parameters, the input question and the passage, as a single packed sequence. The input embeddings are the total of the token and segment embeddings.

Token embeddings: A [CLS] token is added to the input word tokens at the start of the question, and a [SEP] token is inserted at the end of both the question and the paragraph.

Segment embeddings: Each token is assigned a tag indicating whether it is Sentence A or Sentence B. This enables the model to distinguish between sentences [22].

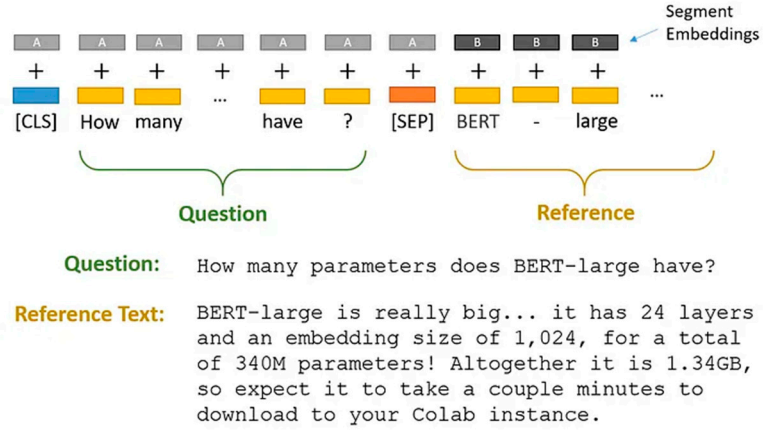

Figure 3. The two pieces of text are separated by the special [SEP] token.

BERT uses "Segment Embeddings" to distinguish the query from the reference text. These are just two embeddings (for segments "A" and "B") that BERT learned and added to the token embeddings before sending them to the input layer.

3.5. Named Entity Recognition (NER)

NER also one of the NLP Task. It is a sub-classification task within Information Extraction (IE) in Natural Language Processing. Every day, many blogs, articles, and other long pieces of information are published on websites, web portals, and social media[32]. NER is the right tool for finding individuals, organizations, places, times, and other information in an article, as well as extracting and categorizing the important points from lengthy descriptions. NER can also be applied to NLP applications such as text summarization, information retrieval, question-answering systems, semantic parsing, and coreference resolution[33].

NER's significance extends beyond labeling items. It is critical in information extraction, a procedure that converts unstructured data into a structured format suitable for further analysis or input into databases. For example, businesses use NER to extract insights from customer feedback, researchers use it to collect data from scientific papers, and news organizations use it to curate content from articles. Any domain that deals with large amounts of textual data can benefit from NER's precision and efficiency[34].

Transformers are a game-changing architecture in the field of natural language processing. While older models handled words or phrases in isolation, transformers transformed this technique by taking into account the full context, resulting in remarkable levels of accuracy in tasks like NER[35]. Their capacity to capture long-range dependencies in text, combined with their natural parallel processing design, has made them the foundation of modern NLP solutions. As we go deeper into fine-tuning BERT, a transformer model for NER, we'll look at how these advanced architectures work together to solve the complex issue of entity recognition.

3.6. Machine Translation

 doi:10.20944/preprints202401.1857.v1

The BERT model is also useful for machine translation, which uses the architecture to convert information from one language to another. We can train the model using specific language inputs to provide the corresponding translated outputs[36]. Models increased translation quality and handled linguistic subtleties better after introducing BERT into the pipeline[37].

Neural Machine Translation (NMT) aims to translate an input sequence from a source language to a target language[38]. An NMT model usually consists of an encoder to map an input sequence to hidden representations, and a decoder to decode hidden representations to generate a sentence in the target language. Given that BERT has achieved great success in language understanding tasks, a question worthy studying is how to incorporate BERT to improve NMT[39]. Due to the computation resource limitation, training a BERT model from scratch is unaffordable for many researchers. Thus, we focus on the setting of leveraging a pre-trained BERT model (instead of training a BERT model from scratch) for NMT.

Given that there are few works leveraging BERT for NMT, our first attempt is to try two previous strategies:

(1) using BERT to initialize downstream models and then fine-tuning the models, and

(2) using BERT as context-aware embeddings for downstream models.

For the first strategy, we initialize the encoder of an NMT model with a pre-trained BERT model, and then finetune the NMT model on the downstream datasets. Unfortunately, we did not observe significant improvement. Using a pre-trained XLM [40] model, a variant of BERT for machine translation, to warm up an NMT model is another choice. XLM has been verified to be helpful for WMT'16 Romanian-to-English translation.

3.7. Word Masking

BERT trains the language model by guessing 15% of the input tokens at random. The tokens are pre-processed as follows: 80% are replaced with a "[MASK]" token, 10% with a random word, and 10% with the original word[25]. The authors' intuition for selecting this strategy is as follows (thanks to Jacob Devlin from Google for the insight):

If we used [MASK] 100% of the time the model wouldn't necessarily produce good token representations for non-masked words. The non-masked tokens were still used for context, but the model was optimized for predicting masked words.

If we used [MASK] 90% of the time and random words 10% of the time, this would teach the model that the observed word is never correct.

If we used [MASK] 90% of the time and kept the same word 10% of the time, then the model could just trivially copy the non-contextual embedding[41].

There was no ablation on the ratios used in this method, and it may have worked better with alternative ones. Furthermore, the model's performance was not tested by simply masking all of the picked tokens.

Conclusion

BERT has transformed the discipline of NLP, allowing for considerable advances in language understanding and representation. Its bidirectional method and transformer architecture have proven to be quite effective for a variety of NLP tasks. With applications ranging from question answering to sentiment analysis, named entity recognition, text categorization, and machine translation, BERT has become an essential tool for data scientists and AI/ML practitioners.

In this paper, we reviewed how BERT can be efficiently fine-tuned to tackle these issues. In this study, we looked at the BERT model and how it works, as well as how it has been implemented in various problem statements. We examined how BERT can improve the capacities of several models used to detect and understand natural languages. With a little fine-tuning, the BERT model can be applied to numerous current frameworks to improve their precision. When examining machine learning capabilities in natural languages, there are several problems to consider, including grammatical semantics and word meanings, as well as the range of natural languages available. In this study, we looked at how BERT can be effectively fine-tuned to address these concerns. It also

deals with homonymy, polysemy, hyponym and many other grammatical semantics of natural languages. However, BRET is a predefined model, so its training is significantly more expensive.

Author Declaration: We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

Reference:

1. J. Pennington, R. Socher, and C. Manning, "Glove: Global Vectors for Word Representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar: Association for Computational Linguistics, 2014, pp. 1532–1543. doi: 10.3115/v1/D14-1162.
2. T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," 2013, doi: 10.48550/ARXIV.1310.4546.
3. X. Ma, Z. Wang, P. Ng, R. Nallapati, and B. Xiang, "Universal Text Representation from BERT: An Empirical Study," 2019, doi: 10.48550/ARXIV.1910.07973.
4. J. Gu, H. Hassan, J. Devlin, and V. O. K. Li, "Universal Neural Machine Translation for Extremely Low Resource Languages," 2018, doi: 10.48550/ARXIV.1802.05368.
5. Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov, "Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context," 2019, doi: 10.48550/ARXIV.1901.02860.
6. A. Graves, "Sequence Transduction with Recurrent Neural Networks," 2012, doi: 10.48550/ARXIV.1211.3711.
7. C. Lo, "MEANT 2.0: Accurate semantic MT evaluation for any output language," in *Proceedings of the Second Conference on Machine Translation*, Copenhagen, Denmark: Association for Computational Linguistics, 2017, pp. 589–597. doi: 10.18653/v1/W17-4767.
8. P. Liu, X. Qiu, and X. Huang, "Recurrent Neural Network for Text Classification with Multi-Task Learning," 2016, doi: 10.48550/ARXIV.1605.05101.
9. J. Schapke, "Evolution of word representations in NLP," Medium. Accessed: Jan. 09, 2024. [Online]. Available: https://towardsdatascience.com/evolution-of-word-representations-in-nlp-d4483fe23e93
10. M. V. Koroteev, "BERT: A Review of Applications in Natural Language Processing and Understanding," 2021, doi: 10.48550/ARXIV.2103.11943.
11. J. Howard and S. Ruder, "Universal Language Model Fine-tuning for Text Classification," 2018, doi: 10.48550/ARXIV.1801.06146.
12. M. E. Peters *et al.*, "Deep contextualized word representations," 2018, doi: 10.48550/ARXIV.1802.05365.
13. V. Chakkarwar, S. Tamane, and A. Thombre, "A Review on BERT and Its Implementation in Various NLP Tasks," in *Proceedings of the International Conference on Applications of Machine Intelligence and Data Analytics (ICAMIDA 2022)*, vol. 105, S. Tamane, S. Ghosh, and S. Deshmukh, Eds., in Advances in Computer Science Research, vol. 105. , Dordrecht: Atlantis Press International BV, 2023, pp. 112–121. doi: 10.2991/978-94-6463-136-4_12.
14. A. Tam, "A Brief Introduction to BERT," MachineLearningMastery.com. Accessed: Jan. 24, 2024. [Online]. Available: https://machinelearningmastery.com/a-brief-introduction-to-bert/
15. N. S. Chauhan, "Google BERT: Understanding the Architecture," The AI dream. Accessed: Jan. 24, 2024. [Online]. Available: https://www.theaidream.com/post/google-bert-understanding-the-architecture
16. A. Vaswani *et al.*, "Attention Is All You Need," 2017, doi: 10.48550/ARXIV.1706.03762.
17. "Improving language understanding with unsupervised learning." Accessed: Jan. 05, 2024. [Online]. Available: https://openai.com/research/language-unsupervised
18. D. Hendrycks and K. Gimpel, "Gaussian Error Linear Units (GELUs)," 2016, doi: 10.48550/ARXIV.1606.08415.
19. M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy, "SpanBERT: Improving Pre-training by Representing and Predicting Spans," 2019, doi: 10.48550/ARXIV.1907.10529.
20. "BERT NLP Model Explained for Complete Beginners," ProjectPro. Accessed: Jan. 24, 2024. [Online]. Available: https://www.projectpro.io/article/bert-nlp-model-explained/558
21. "Understanding BERT." Accessed: Jan. 24, 2024. [Online]. Available: https://www.rws.com/blog/understanding-bert/
22. S. Yu, J. Su, and D. Luo, "Improving BERT-Based Text Classification With Auxiliary Sentence and Domain Knowledge," *IEEE Access*, vol. 7, pp. 176600–176612, 2019, doi: 10.1109/ACCESS.2019.2953990.
23. S. Bag, "Text Summarization using BERT, GPT2, XLNet," Analytics Vidhya. Accessed: Jan. 09, 2024. [Online]. Available: https://medium.com/analytics-vidhya/text-summarization-using-bert-gpt2-xlnet-5ee80608e961

24. "Text Classification with BERT - Shiksha Online." Accessed: Jan. 09, 2024. [Online]. Available: https://www.shiksha.com/online-courses/articles/text-classification-with-bert/
25. R. Horev, "BERT Explained: State of the art language model for NLP," Medium. Accessed: Jan. 09, 2024. [Online]. Available: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270
26. N. N, "Question Answering System with BERT," Analytics Vidhya. Accessed: Jan. 09, 2024. [Online]. Available: https://medium.com/analytics-vidhya/question-answering-system-with-bert-ebe1130f8def
27. X. Zhang, J. Zhao, and Y. LeCun, "Character-level Convolutional Networks for Text Classification," 2015, doi: 10.48550/ARXIV.1509.01626.
28. "Text Classification model," NVIDIA NeMo. Accessed: Jan. 24, 2024. [Online]. Available: https://nvidia.github.io/NeMo/nlp/text_classification.html
29. "Text classification task guide | MediaPipe," Google for Developers. Accessed: Jan. 24, 2024. [Online]. Available: https://developers.google.com/mediapipe/solutions/text/text_classifier
30. J. Briggs, "BERT For Next Sentence Prediction," Medium. Accessed: Jan. 24, 2024. [Online]. Available: https://towardsdatascience.com/bert-for-next-sentence-prediction-466b67f8226f
31. "Weights & Biases," W&B. Accessed: Jan. 24, 2024. [Online]. Available: https://wandb.ai/mostafaibrahim17/ml-articles/reports/The-Answer-Key-Unlocking-the-Potential-of-Question-Answering-With-NLP--VmlldzozNTcxMDE3
32. "What is Named Entity Recognition (NER)? Methods, Use Cases, and Challenges." Accessed: Jan. 24, 2024. [Online]. Available: https://www.datacamp.com/blog/what-is-named-entity-recognition-ner
33. "A Comprehensive Guide to Named Entity Recognition (NER)." Accessed: Jan. 24, 2024. [Online]. Available: https://www.turing.com/kb/a-comprehensive-guide-to-named-entity-recognition
34. A. M. Kocaman, "Mastering Named Entity Recognition with BERT: A Comprehensive Guide," Medium. Accessed: Jan. 09, 2024. [Online]. Available: https://medium.com/@ahmetmnirkocaman/mastering-named-entity-recognition-with-bert-a-comprehensive-guide-b49f620e50b0
35. "Named Entity Recognition: Challenges and Solutions | ProcessMaker." Accessed: Jan. 24, 2024. [Online]. Available: https://www.processmaker.com/blog/named-entity-recognition-challenges-and-solutions/
36. W. Zhu *et al.*, "Multilingual Machine Translation with Large Language Models: Empirical Results and Analysis," 2023, doi: 10.48550/ARXIV.2304.04675.
37. "Transforming machine translation: a deep learning system reaches news translation quality comparable to human professionals | Nature Communications." Accessed: Jan. 24, 2024. [Online]. Available: https://www.nature.com/articles/s41467-020-18073-9
38. D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," 2014, doi: 10.48550/ARXIV.1409.0473.
39. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," 2018, doi: 10.48550/ARXIV.1810.04805.
40. G. Lample and A. Conneau, "Cross-lingual Language Model Pretraining," 2019, doi: 10.48550/ARXIV.1901.07291.
41. C. Xing, D. Wang, C. Liu, and Y. Lin, "Normalized Word Embedding and Orthogonal Transform for Bilingual Word Translation," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Denver, Colorado: Association for Computational Linguistics, 2015, pp. 1006–1011. doi: 10.3115/v1/N15-1104.