Bidirectional Encoder Representations from Transformers (BERT)

May Hammad

Previous Work

Recently obtaining a low dimensional representation for text has gained an increasing interest ,and lots of research efforts have been invested since the first appearance of word representations .
Text representation aims to project words ,which represent a categorical discrete variable onto a continuous numerical space .Since having words as numerical vectors enabled performing vectorized operations on text ,and have made using deep learning methods to NLP applications possible ,thereupon opening the door for the development of contextualized word and sentence representations .

The Bag Of Words (BOW) approach was initially used as the main text representation method.Its working principle depended on giving each word in the vocab a unique high-dimensional sparse vector representation , disregarding the context or similarity between words .Also it failed to give task-specific representations,as the don't rely on any supervision.

The work by (Bengio et al. 2003) was the first to refer to word embeddings as distributed representations of words. They trained the word embeddings jointly with the model's parameters in a language model using a neural network .
Thereafter ,(Collobert and Weston 2008) conceivably were the first

to prove the utility of pre-trained word representations (embeddings) ,additionally showing that they could be utilized in different NLP downstream tasks .

The word2vec concept was then introduced in (Mikolov et al. 2013) with proposing Continuous Bag of Words(CBOW) , which is trained with criteria for predicting the middle word given a window of words around the middle word . Also the skip grams models ,which was trained with a similar criteria as the CBOW , but instead of predicting the current word based on the context, it tries to maximize classification of a word based on another word in the same sentence. These word2vec models have succeeded to capture fine-grained semantic and syntactic regularities additionally ,later enabled the training and the use of pretrained word embeddings .
In 2014 (Pennington, Socher, and Manning 2014) proposed the Global Vectors for Word Representation(Glove) ,which produces word vector representations with meaningful semantic sub-structures by learning to construct concurrence matrix .Also they released a pre-trained version ,that produced state of the art results on various down stream tasks.

As a matter of fact , every feed-forward neural network that takes as input words from the vocabulary and embeds them as vectors into a dense lower dimensional space, which it then fine-tunes through backpropagation by definition outputs word embeddings as the weights of the first layer, which is commonly referred to as the Embedding Layer. The main distinction between the above architecture and methods like word2vec is its computational complexity .
Training shallow models like Glove and word2vec also is different from training a neural network in its objective ,as they both aim at producing word embeddings that encode general semantic relationships and can provide benefit in many downstream tasks.
While regular neural networks, in comparison, generally yield task-

specific embeddings ,which usually limit there use else where .

Word embedding models are quite closely linked to language models. The quality of language models is measured by their ability to learn a probability distribution over words in vocab V .
As a matter of fact, many state-of-the-art word embedding models try to predict the next word in a sequence to some extent. Additionally, word embedding models are often evaluated using perplexity, a cross-entropy based measure borrowed from language modelling.

One of the biggest challenges in natural language processing and language modeling is the shortage of parallel labelled training data for supervised learning tasks .
To alleviate the limitations of shortage many approaches tried to utilize existing pre-trained language representations and use them as features for different downstream supervised learning tasks
Past efforts mainly used static pre-trained representations of words and text sequences as in (Pennington, Socher, and Manning 2014) ,(Mikolov et al. 2013) or various NLP tasks .Traditionally, these word representations were referred to as static ,as each word is represented by a single vector, regardless of context.
In fact ,static word representations could only leverage off the vectorized outputs from unsupervised models being trained on downstream tasks not the unsupervised models them selves .Also they are mostly shallow models to begin with and were often discarded after training ,for instance word2vec and Glove models .
These models only incorporate previous knowledge in the first layer of the model. The rest of the network still needs to be trained from scratch for a new target task.
They fail to capture higher-level information that might be even more useful. Word embeddings are useful in only capturing semantic meanings of words but we also need to understand higher level concepts like anaphora, long-term dependencies, agreement, nega-

tion .

In general Static pre-trained embeddings had many limitations ,being used for many NLP tasks that involves text context understanding such as "Word Sense disambiguation" ,"Question answering", "Next Sentence Prediction NSP" .Those limitation arise from the fact that all the senses of a polysemous word share the same representation regardless of the context they were used in .

While contextualized language models in (Melamud, Goldberger, and Dagan 2016) , (Fedus, Goodfellow, and A. Dai 2018) and (Devlin et al. 2018), have a different vector representations for similar but semantically irrelevant words which makes it powerful for NLP tasks. Their is a long history of previous work on using pre-trained language representations.Traditionally there are two existing strategies for using pre-trained language representations in several downstream learning tasks either unsupervised feature based approaches or fine tuning approaches .

- **Unsupervised feature based approaches :**

  Unsupervised feature based models basically uses a pre-trained neural network that was previously trained on large unsupervised text corpus.
  Those pre-trained networks are used to produce embedding vectors that represents words,sentences or even paragraphs.Embedding vectors are in-turn used as features for NLP models in various learning tasks .
  To pre-train feature based sentence embedding left-to-right generation objectives have been used for instance generation of a future sentence ,given the feature representation of the previous sentence as in (Kiros et al. 2015).

Some models such as ELMo proposed in (Peters et al. 2017) outperformed static traditional word embeddings.Their approach uses task specific architectures that include pre-trained representations as additional features.It depends on forming a contextual representation of the text ,by concatenating features from both left-to-right and right-to-left language models .

- **Fine tuning approaches**

  As for Fine-tuning Approaches ,they are usually either based on supervised or unsupervised pre-trained models.
  Encoders that encodes sentences or documents to a low-dimensional contextual token representation are examples of unsupervised approaches pre-trained from unlabeled text.That are further fine-tuned to be used for a supervised downstream task (A. M. Dai and Le 2015);(Howard and Ruder 2018), (Radford 2018).
  To pre-train an unsupervised encoder model a left-to-right language model is used .The main advantage of this approach is that it reduces the number of parameters to be trained and decreases the training complexity .

There has also been work that showed the effectiveness of transfer learning from supervised tasks trained on large datasets, for instance natural language inference (Conneau et al. 2017) as well as machine translation (McCann et al. 2017)

Problem Statement

Language Modeling is one of the most important elements for many Natural Language Processing applications.For certain emerging NLP tasks ;understanding and modeling the context of the text is crucial for the model's performance .
Former methods either used heavily engineered task-specific models

or utilized pre-trained models for such tasks . However ,task specific models are computationally expensive and inefficient.
They are inefficient as often each task has only a few thousand samples of labelled data, which is not enough for training a good model. Also the existing strategies to apply pre-trained language representation to downstream tasks discussed earlier has shown to have many short comings as well.
Those short comings are introduced as they use standard uni-directional language representations ,for instance a token embedding model as it only takes into account the previous token to give a representation.This clearly limits the choice of the models that can be pre-trained .Also the pre-trained models doesn't perform as it should if further fine-tuned to be used on token-level tasks that needs to consider the context in both forward and backward directions such as "question answering tasks" .

Being able to use models pre-trained using unlabelled data on language modelling tasks and fine-tune it for specific NLP tasks would reduce the need for the heavily engineered task specific architectures or expensive training. Also bidirectional pre-training for language representation that attends context from both left and right directions would help improve the existing language models for several NLP tasks.

Benefits Of Innovation

Bidirectional Encoder Representation from Transformers (BERT) model offers a unified architecture across different tasks which means it needs minimal changes to use the pre-trained model for several downstream tasks.
Bert also overcomes the limitations of the standard pretrained unidirectional language models, as it trains a bi-directional deep language

representation model conditioned of both left and right context.
Unlike static word embeddings BERT produces contextualized word embeddings,which means that BERT produces multiple embeddings of a word, each representing the context around the word. As well as improving the existing fine-tuning approaches being fine-tuned for multiple downstream tasks with just one additional output layer .
In addition to BERT being the first fine-tuning based representation model that achieves state-of-the-art performance on a large suite of sentence-level and token-level tasks, outperforming many task-specific architectures.
Last but not least, it advances the state of the art for eleven NLP tasks.

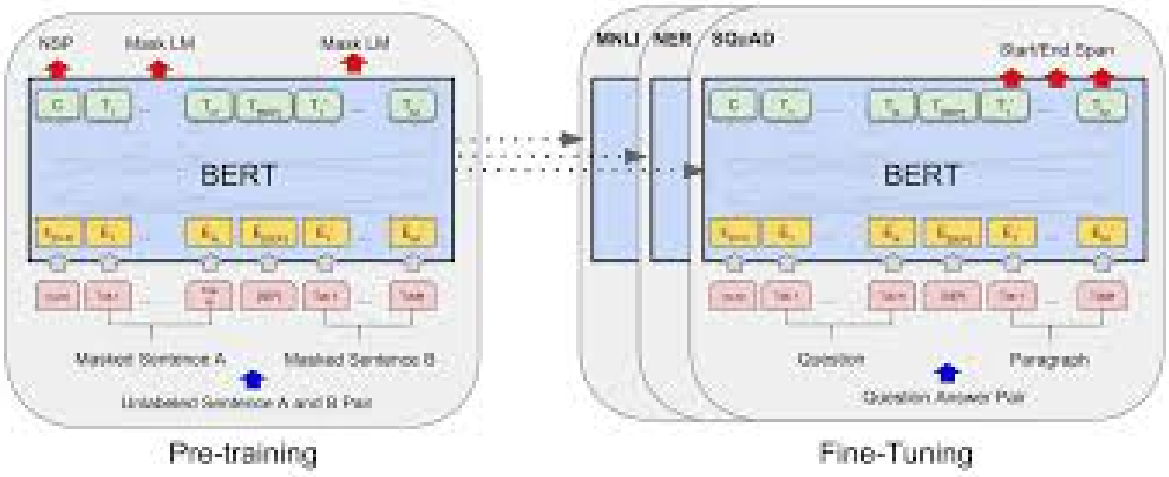

Figure 1: This figure represents the overall pre-training and fine-tuning procedures for BERT.It shows that the sane architecture is used in both stages pre-taining and fine-tuning .Also the same pre-trained initialized parameters are used for different downstream tasks

Outline Of Solution

This section will introduce BERT and its working principle as well as detailed implementation .There are two main steps in BERT pro-

posed framework ,which are pre-training and fine tuning .

Throughout the pre-training stage the model is trained using unlabeled data over various pre-training tasks . For the fine tuning stage ,the BERT model is firstly ,initialized with the parameters initially learned during the pre-training stage , all these parameters are then finetuned on labeled data from the downstream tasks .
As a matter of fact , for every downstream task there exists a certain fine-tuned model ,that should be specific to this task ,despite all being initialized with the same pre-trained parameters.
The multi-layer bidirectional transformer model described in (Vaswani et al. 2017) represents the backbone of the BERT model architecture , as it heavily relies on it and almost had similar model structure as the original .

Two main architectures where proposed in (Devlin et al. 2018) ,the first is the $BERT_{BASE}$ and the second is called $BERT_{LARGE}$.
The architecture presented as $BERT_{BASE}$ in (ibid.) was chosen to resemble OpenAI GPT in size for comparison reasons . Notably , while ,BERT transformer uses bidirectional self attention which, attends to both directions , GPT transformer relies on constrained self-attention where every token can only attend to context to its left.

The $BERT_{BASE}$ model consisted of 12 transformer blocks (L) ,12 self attention heads (A) ,786 hidden dimension (H) ,and had a total number of parameters = 110 .
The $BERT_{LARGE}$ architecture however, had 24 transformer blocks (L) ,16 attention heads (A) , 1024 hidden dimension (H) and a total number of parameters = 340

The BERT model is usually inputted a sequence of tokens , that could represent a single sentence or two sentences packed together in a pair .

The BERT implementation in (Devlin et al. 2018) uses WordPiece embeddings described in (Wu et al. 2016) with a 30,000 token vocabulary.
Usually ,the first token of every input sequence is represented by a classification token $[CLS]$,and the final hidden state vector denoted as C,with $C \in R^H$, of this token is used as a representative to the entire sequence in classification tasks .
Since sentences are packed together to form a single sequence , they used a separator token $[SEP]$ is added after each sentence to mark the boundaries between sentences as well as adding a learned embedding to every token to indicate whether it belongs to sentence A or sentence B . They used E to refer to input embeddings ,and T_i to denote d the final hidden vector for the i^{th} input token ,with $T_i \in R^H$
.

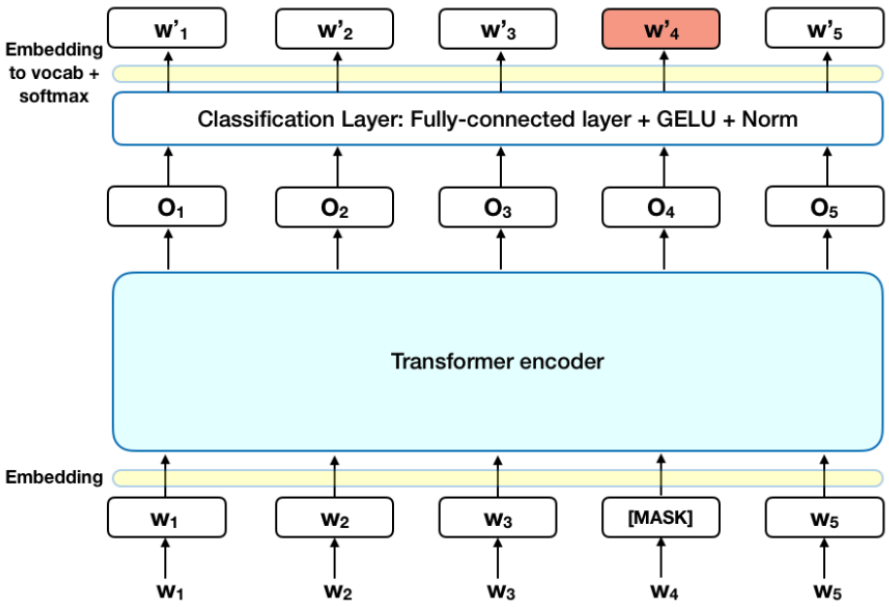

Figure 2: The figure illustrates the working mechanism of BERT Masked Language Model,where the model only learns to predict the masked words.

Pre-training Stage

The author's approach to pre-train the BERT , is using two unsupervised tasks , the Masked Language Model (MLM) and the Next Sentence Prediction (NSP) .

MLM

The Masked LM approach first defined in (Taylor 1953) as Cloze task was deployed in (Devlin et al. 2018) in order to train a bidirectional representation .
Their approach masks 15% of the WordPiece tokens in each sequence (being represented as a mask token $[MASK]$) , chosen at random , and then aims to predict the masked word depending on surrounding context of tokens .

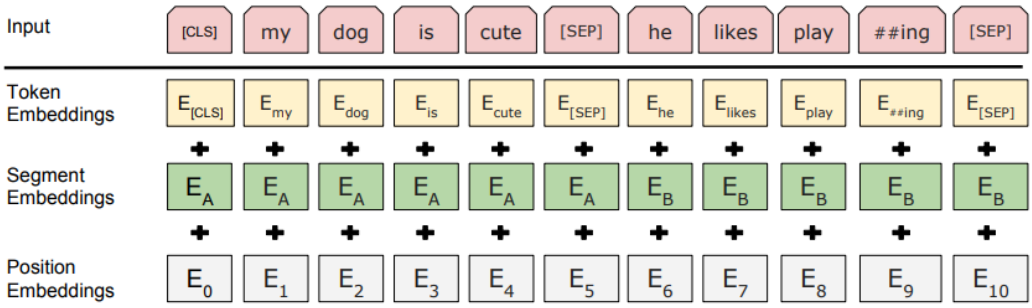

Figure 3: This figure illustrates the BERT input representation . It shows that the input embeddings are represented as the summation of the token embeddings, the segmentation embeddings and the position embeddings .

NSP

Since understanding the relationships between two sentences is crucial to many downstream tasks such as Question Answering (QA) and Natural Language Inference (NLI) , the authors pre-trained the BERT to perform binarized Next Sentence Prediction(NSP) .
Throughout this task ,and during the pre-training stage , the model

receives a pair of sentences packed together in a sequence ($sentenceA$, $sentenceB$) , and aims to learn to predict if $sentenceB$ follows $sentenceA$ in the original text (given classification tag [$isNext/notNext$]).
Notably , the strategy the authors used when choosing $sentenceA$ and $sentenceB$ for each training example ,was that 50% of the time $sentenceB$ is chosen not to follow $sentenceA$ in the original text (labeled as $notNext$) and 50% following (labeled as $isNext$) .

Fine-tuning Stage

Pre-trained BERT can be fine-tuned to model various specific NLP tasks. It uses self attention mechanism which makes it easy for BERT to model many downstream tasks . Fine-tuning BERT for a certain task involves first adding a single output layer to the core model ,then using the task specific data as inputs and outputs for the model and fine-tuning all the parameters end to end. In the fine-tuning stage, most hyper-parameters stay usually the same as in BERT pre-training

- In the Question Answering task evaluated on the (SQuAD v1.1) benchmark.The input to the model is a question regarding a text sequence and the model is required to mark the answer in the sequence .

  For token level-task such as "Question Answering" .The Core BERT model outputs a token level representations for the input data. Subsequently at the output level the token representations from previous step are fed into the custom output layers .The output is the answer marked at the beginning and the end.

- In the classification tasks such as "Sentiment Analysis". The input to the model is text and the output must be discrete la-

bels describing the over all sentiment of the input text.

Using BERT for Classification tasks such as sentiment analysis and "entailment" is very similar to the Next Sentence classification (NSP) task.The output of the core model is a representation for the [CLS] token.A classification layer is added on top of the Transformer output and the [CLS] representation is fed into an output layer for classification .

Requirements For Deployment

Training BERT model:

- Unsupervised pre-training
  - The model was pre-trained on BooksCorpus dataset of estimated (800M words)
  - Followed by training on text only from English Wikipedia of estimated (2,500M words).
- Supervised task-specific fine-tuning Example fine-tuning tasks dataset
  - The General Language Understanding Evaluation (GLUE) benchmark
  - The Stanford Question Answering (SQuAD v1.1)benchmark
  -

Citation examples

References

Bengio, Yoshua et al. (2003). "A Neural Probabilistic Language Model". In: *JOURNAL OF MACHINE LEARNING RESEARCH* 3, pp. 1137–1155.

Collobert, Ronan and Jason Weston (2008). "A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning". In: *Proceedings of the 25th International Conference on Machine Learning*. ICML '08. Helsinki, Finland: ACM, pp. 160–167. ISBN: 978-1-60558-205-4. DOI: 10.1145/1390156.1390177. URL: http://doi.acm.org/10.1145/1390156.1390177.

Conneau, Alexis et al. (2017). "Supervised Learning of Universal Sentence Representations from Natural Language Inference Data". In: *CoRR* abs/1705.02364. arXiv: 1705.02364. URL: http://arxiv.org/abs/1705.02364.

Dai, Andrew M. and Quoc V. Le (2015). "Semi-supervised Sequence Learning". In: *CoRR* abs/1511.01432. arXiv: 1511.01432. URL: http://arxiv.org/abs/1511.01432.

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

Fedus, William, Ian Goodfellow, and Andrew Dai (Jan. 2018). "MaskGAN: Better Text Generation via Filling in the ". In:

Howard, Jeremy and Sebastian Ruder (2018). "Fine-tuned Language Models for Text Classification". In: *CoRR* abs/1801.06146. arXiv: 1801.06146. URL: http://arxiv.org/abs/1801.06146.

Kiros, Ryan et al. (2015). "Skip-Thought Vectors". In: *CoRR* abs/1506.06726. arXiv: 1506.06726. URL: http://arxiv.org/abs/1506.06726.

McCann, Bryan et al. (2017). "Learned in Translation: Contextualized Word Vectors". In: *CoRR* abs/1708.00107. arXiv: 1708.00107. URL: http://arxiv.org/abs/1708.00107.

Melamud, Oren, Jacob Goldberger, and Ido Dagan (Aug. 2016). "context2vec: Learning Generic Context Embedding with Bidirectional LSTM". In: *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. Berlin, Germany: Association for Computational Linguistics, pp. 51–61. DOI: 10.18653/v1/K16-1006. URL: https://www.aclweb.org/anthology/K16-1006.

Mikolov, Tomas et al. (2013). "Efficient Estimation of Word Representations in Vector Space". In: *CoRR* abs/1301.3781. URL: http://dblp.uni-trier.de/db/journals/corr/corr1301.html#abs-1301-3781.

Pennington, Jeffrey, Richard Socher, and Christopher D Manning (2014). "Glove: Global Vectors for Word Representation." In: *EMNLP*. Vol. 14, pp. 1532–1543.

Peters, Matthew E. et al. (2017). "Semi-supervised sequence tagging with bidirectional language models". In: *CoRR* abs/1705.00108. arXiv: 1705.00108. URL: http://arxiv.org/abs/1705.00108.

Radford, Alec (2018). "Improving Language Understanding by Generative Pre-Training". In:

Taylor, Wilson L. (1953). ""Cloze Procedure": A New Tool for Measuring Readability". In: *Journalism Quarterly* 30.4, pp. 415–433. DOI: 10.1177/107769905303000401. eprint: https://doi.org/10.1177/107769905303000401. URL: https://doi.org/10.1177/107769905303000401.

Vaswani, Ashish et al. (2017). "Attention Is All You Need". In: *CoRR* abs/1706.03762. arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762.

Wu, Yonghui et al. (2016). "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation". In: *CoRR* abs/1609.08144. arXiv: 1609.08144. URL: http://arxiv.org/abs/1609.08144.