A comparative study of Natural Language Processing Models

Vikas Rajashekar

TU Kaiserslautern 409206

Abstract. Natural Language Processing(NLP) is a sub-field of Artificial Intelligence that aims to enable the computers to interpret human language and deduce the contextual meaning. Before the introduction of transfer learning, most of the architectures in NLP followed classical approaches of using the combination of per-trained word embeddings and task-specific architecture trained on a single task-specific data set. There was an acceleration in progress in the NLP community when transfer learning was used. In Transfer learning the hierarchical contextualized representations are learned by using large web-scale data sets in an unsupervised manner called language models, which is later used in downstream tasks. In this paper, we introduce the concept of language models and survey the various state of the art architectures. First, the Transformers model is introduced as it is the building block of the rest of the models in the survey. Then, we introduce the BERT model, whose architecture is a multi-layer bidirectional Transformer encoder based on the original implementation Transformers architecture which was introduced previously. At last, we talk about XLNET which overcomes the disadvantages of BERT and improved the results in 20 NLP tasks in which BERT was state of art.
. . .

Keywords: Natural language processing, language modelling, Transofmers, BERT, XLNET

1 Introduction

NLP being part of Artificial Intelligence mainly deals with providing the computers the ability to understand the human language statements. NLP-based systems have a wide range of applications. Google's powerful search engine, as well as voice assistants, like Alexa by Amazon, Siri by Apple are all NLP based systems. NLP plays a key role in complex natural language-related tasks like machine translation and dialogue generation. NLP is a computer science discipline focused to develop user-friendly systems that can converse with people with just everyday language. It is related to human-computer interaction. NLP integrates anything that a computer needs to understand the normal language and also to construct natural language.

Most of the NLP problems can be defined in terms of semantic representation of the data set. A successful model finds a suitable representation of the data that could be used for the given task. Neural Networks(NN) are known for extracting flexible representations of the data that could be used for a wide range of tasks. Surprisingly, A NN trained for one task can be used for a completely different task. Intuitively, we can create a universal language representation with a similar approach.

NLP algorithms are machine learning-based. NLP algorithms analyze the data set consisting of a large corpus of sentences and make a statistical inference. With this, NLP learns the useful representation of data instead of coding large sets of rules. This approach tackles hard NLP problems such as — automatic summarization, machine translation, named entity recognition, relationship extraction, sentiment analysis, speech recognition, and topic segmentation, etc.

1.1 Language Model

Language Modeling(LM) being the major part of NLP, plays a key role in various NLP application areas like Machine Translation, Spell Speech Recognition, Summarization, Question Answering, Sentiment Analysis, etc. LM can be defined as the task of assigning the probability distribution for the sequence of words that suits the language's distribution. More formally, a LM estimates the likelihood of the term concerning the given context.

The need for learning the probability distribution can be understood with the following example. Machine translation deals with the task of translating a sentence in one language to another language. An ambiguity arises when the system predicts different ordering of words for the given sentence. So, the probability is an absolute necessity to know which order to pick. In this case, the distribution with maximum probability as shown in Equation 1.

$$p(\text{the cat is small}) > p(\text{small cat the is}) \tag{1}$$

NLP related tasks get their strength from the model's ability to model the language and its laws as a probability distribution.

1.2 Word Embeddings

Word embedding is an effective tool to model the semantic relationship between words. Word embedding takes into account the surround words for a given word in training corpus and models the distribution of words and represents the statistics in low dimensional vectors.

There are two major types of word embedding. First is the classical approach called Static word embedding, where the context is not considered in the representation. Static word embedding generates the same embedding vector for all the contexts, failing to capture polysemy. The second approach is Contextual (Dynamic) embedding words which take into account the context of the word. This approach captures the meaning of the same word in different contexts and hence generates different embedding to the same word based on the context, thus capturing the polysemy. In a static word embedding approach, a shallow model is used for training and only the output from the model is considered for further processing, not the model itself. The Contextual approach uses both the model and the generated vectors for further processing which is the reason for better performance.

1.3 Transfer Learning

It is efficient to first pre-train a model on large data set and then fine-tune this pre-trained model for different downstream NLP tasks. This approach is termed as Transfer learning. It is inefficient to train a model from scratch because most of the NLP task-related data sets are small in size, if trained from scratch the model usually over-fits and generalizes poorly. Researchers in the field of NLP first train a Language Model on a large corpus of sentences and then fine-tune it for the different downstream NLP tasks. This approach helps to achieve good performance because the semantic knowledge of the language from the pre-training can be effectively used for downstream tasks during fine-tuning.

2 Tranformers

In the paper 'Attention Is All You Need'[1], transformers architecture was introduced . As the name indicates, Transformers is all about attention mechanism used at all possible places. Transformer architecture avoids the time-consuming recurrence and completely depends on the attention mechanism which is relatively faster to learn the dependencies between the input and output. The transformer converts one sequence into another just like machine translation using two sections i.e Encoder and Decoder. Unlike sequence-to-sequence models, the encoder and decoder of the transformer do not use any recurrence. Transformers are the first transduction model to use self-attention alone for translating the sequence instead of sequence-aligned RNNs or convolution.

In the Figure 1 encoder unit is placed on the left side and decoder unit on the right side. Both encoder and decoder consist of stacks of modules that are placed on each other, defined by Nx within the figure. Each module unit consists of Multi-head attention and Feedforward layers. As the strings cannot be used directly, the input and output sentences are first converted to its numerical embeddings. Since there is no recurrence used in the Transformer it is important to keep track of the order of words that are fed into the model. So, along with the word embedding, positional embedding is used to encode the relative position of the word in the sentence.

2.1 Self-Attention

Tranformers use Scaled Dot-Product Attention as shown in Figure 2. Queries and keys of dimension d_k , and values of dimension d_v are the inputs. First, dot products of the query with all the keys are calculated and then divided by

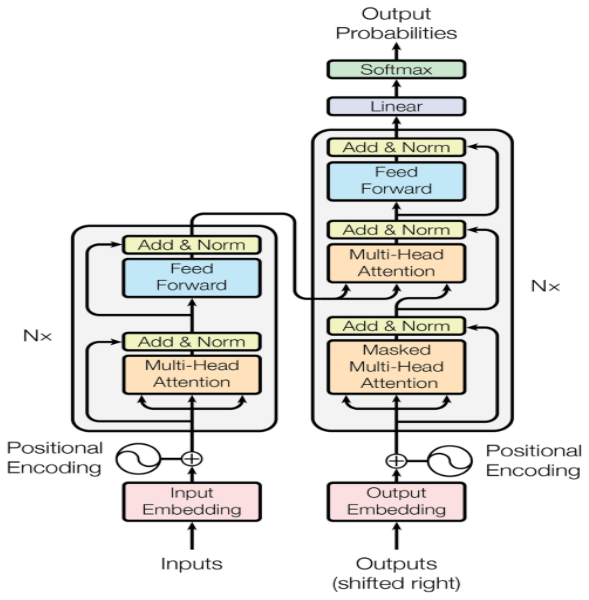


Fig. 1. The Transformer model architecture taken from [1].

$\sqrt{d_k}$, and then softmax function is applied to obtain the attention weights. For computational efficiency, the queries, keys ,and values are packed together into matrix Q, K ,and V respectively. The attention equation is shown in Equation 2.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2}$$

Multi-head attention consisting of multiple self-attention layers helps to attend to the information from different representation sub-spaces at different positions rather than attention to one sub-space. The transformer uses this attention in three ways:

1. In encoder-decoder attention layers, the keys and values are composed of encoder layers' output and queries from the previous decoder layer, which allows each decoder position to attend to encoder output.
2. In encoder attention layers, the keys, values, and queries come from the previous encoder layer, which allows each encoder position to attend to itself.
3. In decoder attention allows the decoder position to attend all the positions that are up to and including itself. It is necessary to stop the decoder's attention to look at the position further than itself. This is handled by masking illegal positions by self-attention.

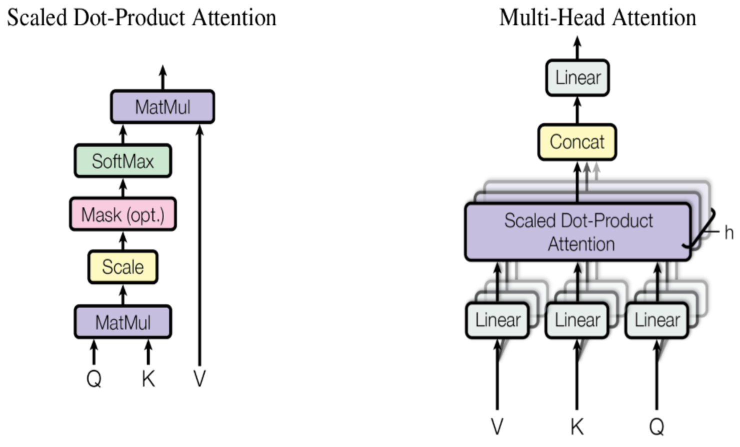


Fig. 2. Attention used in Transformers, taken from [1].

2.2 Transformers Results

On the WMT 2014 English-to-French translation task and the WMT 2014 English-to-German translation task, the transformer model established a state-of-the-art BLEU score of 41.0 and 28.4 respectively. On the English-to-French translation task, the new BLEU score was achieved at a training cost of 1/4 of the previous best model.

3 BERT

Bidirectional Encoder Representations from Transformers (BERT) [2], has four major features:

1. Bidirectional: Bidirectional representations of unlabeled text are pre-trained by conditioning them together in both the left and right direction in all layers.
2. Generalizable: BERT can be fine-tuned easily for downstream tasks.
3. High-performance: Achieves state-of-the-art results for many NLP tasks.
4. Universal: BERT is trained on Wikipedia and book corpus with no requirement of a special data set.

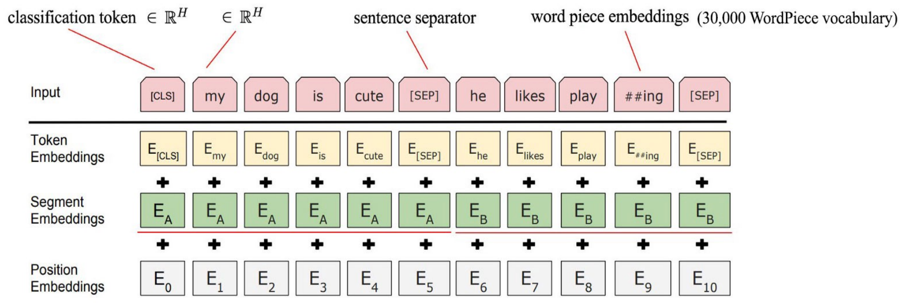
3.1 Model architecture

Based on the original implementation described in [1], BERT's model architecture is a multi- layered bidirectional transformer encoder. The number of layers, also known as the Transformer blocks, is denoted as L, the hidden size as H, and the number of self-attention heads as A. The results are reported on models with different sizes :

1. BERT BASE :L(Transformer blocks)=12, H(Hidden size)=768, A(Self-attention heads)=12, Total Parameters=110M
2. BERT LARGE :L(Transformer blocks)=24, H(Hidden size)=1024,A(Self-attention heads)=16, Total Parameters=340M

3.2 Input/Output Representations

To handle the different spectrum of NLP tasks, the model requires one or two sequences of words as input. All the inputs must contain a special classification token i.e., [CLS] in the beginning. Also, if the input consists of two sequences then a separator token i.e. [SEP] has to be placed between two sequences. BERT generates a series of outputs as shown in fig. 4. For classification purposes, only the output C is to be used. For non-classifications tasks, the series of outputs excluding C is used. For each token, the input representation is creating by adding its token, segment and positional embeddings as shown in Figure 3.



classification token $\in \mathbb{R}^H$
$\in \mathbb{R}^H$
sentence separator
word piece embeddings (30,000 WordPiece vocabulary)
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}

Fig. 3. Input/Output representations in BERT taken from [2].

BERT consists of two major phases.

1. Pre-Training: During pre-training, various pre-training tasks are used to train the model on unlabeled data.
2. Fine-Tuning: The model is initialized to pre-trained parameters in this step, and is fine-tuned using downstream tasks data. Each downstream tasks will have its own fine-tuned model though initialized with same pre-trained parameters

BERT uses 2 NLP tasks to pre-train the model:

1. Masked Language Model(MLM): In this model, to train a deep bidirectional representation, a procedure known as ”Masked Language Model”(MLM) is used, popularly referred to as Cloze task in the literature [3]. According to the procedure, 15% of the input tokens are masked randomly and the masked tokens are then predicted by the model. 80% of the masked Word-piece will be replaced with a [MASK] token, 10% with a random token and 10% will remain unchanged. In this approach, the loss is used to calculate how well the model predicts missing words not the reconstruction of the whole input sequence.

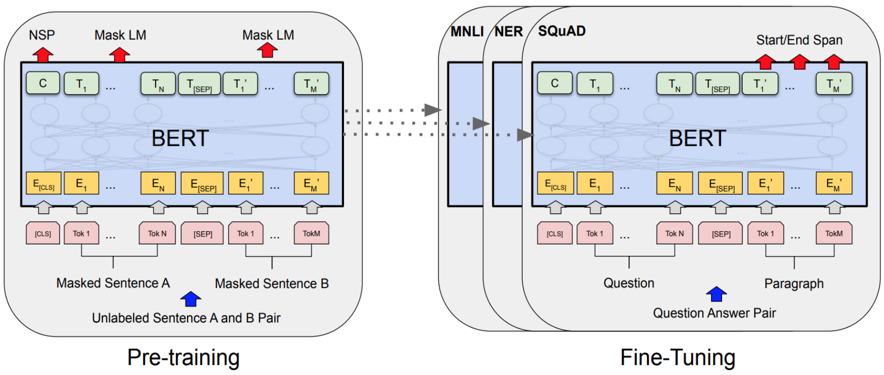


Fig. 4. Overall pre-training and fine-tuning procedures for BERT taken from [2].

2. Next Sentence Prediction(NSP): The purpose of this method is to make BERT learn the relationship between two consecutive sentences. During training, BERT uses 50% of the times two consecutive sentences and predicts the label 'IsNext' i.e. if the two sentences are consecutive. This task helps BERT to encode the relationship between two sentences in output variable C.

The two training tasks enable BERT to train the Vector representations, which not only captures context but also linguistics information such as semantics and co-reference.

3.3 Results of BERT

BERT outperforms previous best models with new state-of-the-art results on eleven natural language processing tasks, including raising the GLUE score to 80.5% with 7.7% improvement, MultiNLI accuracy of 86.7% with 4.6% improvement, SQuAD v1.1 question answering Test F1 to 93.2 with 1.5 point improvement and SQuAD v2.0 Test F1 to 83.1 with 5.1 point improvement.

4 XLNET

XLNet [4] is a generalized auto-regressive pre-training method that uses the auto-regressive language model. The model predicts the next word using a context word, that is constrained to only two directions, either forward or backward. XLNet improves upon BERT by overcoming the following disadvantages of BERT:

1. At the time of fine-tuning, the artificial symbols like [MASK] used by BERT during pre-training are absent from real data causing the pretrain-finetune discrepancy.

2. Given the unmasked tokens, BERT assumes that the predicted tokens are independent of each other, which is an oversimplified assumption, as high-order, and long-range dependency is predominant in natural language.

A new objective called Permutation language Modeling was proposed by XLNET that makes use permutations of different combinations of words.

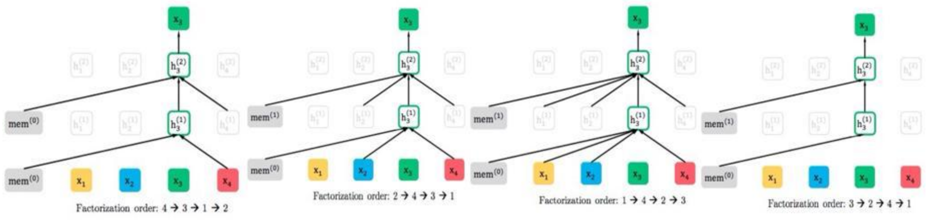

Factorization order: 4 → 3 → 1 → 2
Factorization order: 2 → 4 → 3 → 1
Factorization order: 1 → 4 → 2 → 3
Factorization order: 3 → 2 → 4 → 1

Fig. 5. Illustration of the permutation language modeling objective for predicting x 3 given the same input sequence x but with different factorization orders taken from [4].

For example consider the sequence [x1, x2, x3, x4] consisting of 4 tokens (N=4) in the sentence. So for N=4, there are N! i.e. 24 permutations. Consider the scenario of predicting the token x3, for x3 we observe 4 patterns out of 24 permutations. They are, x3 at the 1st position, 2nd position, 3rd position and 4th position ([x3, xx, xx, xx][xx, x3, xx,xx][xx, xx, x3, xx][xx, xx, xx, x3]).

For predicting x3, the position of x3 is set as the t-th position with t-1 token are used as context words as shown in Figure 5. Since permutations are used, every possible word combination and lengths are tried and eventually, the network learns from both the directions. It is important to note that the permutation order is not the actual order of sequence. The positional encoding is still used by the network to keep track of relative positions.

4.1 XLNET Results

XLNet outperforms BERT on 20 tasks, frequently by means of a big margin, inclusive of question answering, natural language inference, sentiment analysis, and document ranking.

5 Conclusion

In this paper, we provide a survey of the recent state of the art language models like Transformers, BERT, and XLNET. Transformers were the major shift from recurrence to self-attention which helped to parallelize the operations which greatly reduced the training time. BERT used the transformers along with Masked Language Model and was able to obtain a state of art results on various NLP tasks. XLNET overcame the disadvantages of BERT by proposing an Auto-regressive language model which uses Permutation Language Modeling and was able to outperform BERT in 20 NLP tasks.

References

1. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems. (2017) 5998–6008
2. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
3. Taylor, W.L.: "cloze procedure": A new tool for measuring readability. Journalism quarterly **30**(4) (1953) 415–433
4. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: Xlnet: Generalized autoregressive pretraining for language understanding. In: Advances in neural information processing systems. (2019) 5754–5764