From Word2Vec to BERT: A Review on Language Representations

Anshul Vashisth*[1], Vedpal[2] and Piyush Gupta[3]

[1]Department of Computer Engineering, [2]Department of Computer Applications, [3]Department of Information Technology

[1,2,3]J.C. Bose University of Science & Technology, YMCA, Faridabad, Haryana, India

Abstract: *Transfer learning in the field of language generation is the fundamental idea that comprises pre-training calibrating with fine-tuning of the tasks basis on a particular model. Traditional models were based on a Word embedding such as word2vec and GloVe. These models were used for downstream Natural Language Processing (NLP) tasks. The major limitation of these models was that there was a limit to the amount of information they could capture and didn't take the context of word into account that result in losing valuable information. These limitations encourage in developing new language generation tasks. A new language depiction model known as Bidirectional Encoder Representations from Transformers (BERT) was introduced in 2018 which is a deep bidirectional model which gives state-of-the-art results to the NLP community. In this paper, we review the convolution models and new bidirectional models which do a revolutionary change in the NLP.*

Keywords: Text Summarization, Word embedding, BERT, NLP.

1. Introduction

In the modern era propelled by the modern technological innovations, an explosion occurred in the quantum of text data from diversity of sources such as Google, Yahoo, Bing and so on. Due to this unrivalled evolvement of blogs, reports, news articles, government officials search engines etc. To shorten these longer texts automatically the need to develop machine learning algorithms arises. Basically, text summarization is the technique which selects certain important sentences from a document and creates a fluent summary of that document.[1] This technique helps to provide a good overview of the entire document.

Having pinpoint essence, the text content can be easily processed, digested, and retrieved in such a manner to accomplish desired result efficiently.[2] Machine learning models are ordinarily trained to perceive documents and useful information is distilled before producing the required summarized texts of any document.

The task of text summarization has two main techniques: One is Extractive Summarization and other is Abstractive Summarization. Extractive summarization involves pulling crucial sentences as well as phrases from any text article or group of papers. These sorted sentences are then combined to formalize the summary whereas extraction is made with substantial which is present within input source. Any substantial outside of input document is not present in this kind of summary. In contrast, summarization using an Abstractive approach requires deeper evolution of the text.

[*] Corresponding Author

Lloret et al. [13] generated abstracts of bio-medical papers using both extractive and abstractive technique and analyzed their appropriateness. First, the author extracts the most relevant sentence from the corpus and then integrates information compression and fusion with the result of extraction. Then this integration results in an abstractive oriented summary. Both approaches generate the summary but performing abstraction on extractive summary gives a new face to the summary.

The algorithms of abstract summarization originate new turn of phrases as well as sentences which relay the valuable information of original text. This improves the center of attention of summary, decreases the level of redundancy and also good compression rate is maintained [4]. A fast abstractive summarization technique is also proposed by Chen and Bansal [17] to summarize large documents. Also an open source library for abstractive text summarization i.e. Neural Abstractive Text Summarizer (NATS) toolkit is developed by Shi et al. [18]. But, still extraction is used more popular than abstraction because the complexity of developing abstraction algorithms is more.

Mirani and Sasi [12] generated two level summaries by using extractive summarization approach. To generate summaries the authors took data online from news articles which covered data related to politics science, sports data, science and health related data, movie reviews data of multiple countries such as Fox News information from USA, NZ Herald newspaper reviews from New Zealand, Hindustan Times paper information from India, views by BBC from UK, etc. Consequently, they generate summary of every article using sentiment analysis at first level, then summary of related articles is combined and then again generate summary of related articles at second level. This achieves a better result in summarization.

Traditional models were restricted to word embedding such as Glove or word2vec, where every single word was linked to a vector, which used to represent some feature of its meaning. Ibrahim [11] used word2vec model as feature weighting technique during his research for clustering Arabic text which results in enhancing the accuracy.

Recently, some pre-training methods namely BERT, ELMo, and OpenAI GPT have achieved state-of-the-art precision in various language understanding tasks and attracted too much attention with this in natural language processing. From these pre-training methods that are discussed above, BERT is the most vital one because of its bidirectional encoder representations.

2. The Need of Review

The aim of review of these approaches is to improve the current work. The previous work on automatic summarization that was done either uses convolutional methods which are directional models or doesn’t predict the accurate sentence as a result in extractive summarization as well as in abstractive summarization. Here, we will have a look on those convolution directional models and the new language generation tools that caused a huge revolution change in the NLP community as bidirectional models.

3. Related Work

Transfer learning has a lot of work which uses natural language processing tasks. The automated summarization is significantly different from traditional summarization. We will discuss these summarization techniques in this section.

3.1 Word Embedding

Before proceeding further first we will discuss traditional approaches such as word embeddings or word2vec [6]. Basically, there is a word2vec model for vector representation of words called “word embeddings”. Conceptually, the text has learned representation where words which have the same meaning will definitely have similar vector representation. In deep learning approaches for NLP tasks, word embeddings are taken as input and are usually trained together with neural network parameters. All the information is provided as input of word embedding. The performance of the learning task highly impacts the quality of embedding. The need for word embedding arises to deal with words or key phrases of any text corpus because computers can’t understand words naturally. Word embedding basically maps the text data with numeric data by encoding text in a numeric form. This numeric data is learned by neural network by applying mathematical rules and performing mathematical operations to them.
Let’s take a look at this example. Here,” Vector representation of a particular word” is a sentence and this sentence has multiple words like vector, particular, word etc. Dictionary consists of all these words of a sentence in a list like- [“Vector”, “representation”, “of”, “a”, “particular”, “word”]. For these words one hot encoding vector is generated where position value will be 1 when word exists otherwise value will be 0 everywhere. According to this one hot encoding vector are: Vector = [100000], Representation = [010000], of = [001000], a = [000100], particular = [000010], word = [000001]. Mandelbaum and Shalev [9] discussed the need of word embedding and some methods to create these embeddings. They also compared these embeddings to image embedding to see how they can be combined to perform various tasks.
Basically, Word embeddings are grouped into two categories: One is Frequency based embedding and other is prediction based embedding.

3.1.1 Frequency Based Embedding: Here, word frequency information is taken into account. Frequency embeddings are based on the frequency of words. The three common metrics representing word frequency information are raw counts (when a word appears many times in the whole corpus), ranking (i.e., rank 0 for the most common word) and frequency classes [7]. In this category three vectors that are encountered are: Count Vector (sometimes called as one hot encoding), TF-IDF Vector and Co-Occurrence Vector.

a) Count Vector

Count vector results a matrix of token counts of the word content of our text documents. This creates vectors that have a dimensionality which is equal to our vocabulary size, and if text data features that vocab word, it simply put a one in that dimension. Every time we encounter that word again, it will increase the count. If it did not find the word even once it leaves 0s everywhere. One issue with simple counts is meaningful in the encoded vectors. To overcome these types of problems a new acronym TF-IDF comes into consideration.

b) TF-IDF

TF-IDF stands for term frequency inverse document frequency. TF scores the quantity of words that occur in a document whereas the IDF scores how rare the word is present in a document. Ideally, these are more important but there is a limitation with these methods is that the vocabulary gets too large. By this, for encoding documents, it requires large vectors.

To process these large vectors there is a large memory requirement which can slow down the algorithms.

c) Co-Occurrence Matrix

Here, similar words occur together and this matrix simply counts how these similar words occur in the documents and captures the words relationships. It also captures the same semantic connection between words. Such as male and female is much closer than male and car. The main issue with this is large memory space is required to store this co-occurrence matrix. Another method of word embedding is Predictions based word embedding.

3.1.2 Prediction Based Embedding: This type of embedding helps to accurately predict the word in a given context and ensures to be state-of-the-art. Basically, these types of embedding are done by combining two techniques. One of them is Continuous Bag of Words (CBoW) and Skip Gram Model another. Both of these techniques map words to target and learn weights. The CBOW model tends to predict the current word and learns the embedding from its context (surrounding words). While another model, Skip Gram model, attempts to predict the context of word. Mikolov et al. [10] used skip gram model and presented several extensions to that which results in improving quality of the vectors and its training speed. The authors also described negative sampling which is an alternative to hierarchical softmax.
All approaches that we studied above have some limitations such as these are difficult to train on large datasets. These can't be fine-tuned as they are trained from scratch on a neural network with one hidden layer. To overcome such problems a different approach which has bidirectional training of transformers is used which produce state-of-the-art results.

4. BERT

In year 2018, **BERT** which means **B**idirectional **E**ncoder **R**epresentations from **T**ransformers was proposed by Google AI researchers [14]. This gives a new twist into language representation model with the use of bidirectional transformers. These transformers include two distinct mechanisms: first, an encoder and other is decoder. Encoder takes and reads the text as input whereas Decoder generates a prediction for task.
These NLP tasks can be solved using two steps. One is Pre-tuning and another is Fine tuning [15]. During pre-training, different pre-training tasks are performed to train model over the unlabeled data. Whereas for fine tuning, first, the pre-trained parameters are initialized with BERT model, then by using labeled data these parameters are fine-tuned through downstream tasks. Every downstream task is associated with separate fine-tuned model, although same pre-training parameters are initialized with them [8].
Some benefits of using Bert over other convolutional methods are:

- Unlike old and conventional language models which used previous tokens to predict the next token, BERT takes both the next and previous token for prediction.
- BERT is also specialized for next sentence prediction which makes it an appropriate choice for tasks such as question-answering or in sentence comparison.
- The transformer architecture of encoder and decoder for encoding sentences is used by BERT, and with wide variations in parameters, its performance is remarkable even on small datasets.

4.1 Directional and Bidirectional Models

In this paper, we are repeatedly using terms directional and bidirectional models. These directional models take the text as input and read it sequentially from left to right. In contrast, the text data is read from both directions (left to right as well as right-to-left) in Bidirectional models during training phase. The directional models uses word embedding approach in which context of word is not possible and are difficult to train on large datasets. You also couldn't fine tune them. But in bidirectional, transformers read the entire sequence of word at once and after pre-training these can be fine-tuned on large datasets easily.

4.2 Training Strategies

The language model is trained by predicting a goal. Predicting that goal is the main challenging task in NLP. It predicts the word or predicts the probability of occurring a word in particular context. Let's take an example: "Children are _____ in park". Here language model may predict the blank word as "playing" with 90% probability and may tell it as "studying" with 10% probability. To predict that word, BERT uses two training strategies: One approach is **Pre Training** a neural network **and** other is **Fine Tuning.**

4.2.1 Pre Training of Bert: The main reason of success of Bert is the pre-training methodology because the model is trained on large text corpus and the model start learning and understanding of how language works. This learning is the main key factor of success of Bert which is useful for many NLP tasks. This language learning also helps in fine-tuning the model in any natural language processing task. Basically, there are two tasks on which BERT has been pre-trained i.e.; MLM means masked language model other is Next Sequence Prediction (NSP).

a) Masked Language Model:

The Masked Language Model did not feed word sequences directly into BERT. These words from the input are replaced with tokens, and then attempts to make prediction of the original vocabulary of masked words on the basis of its context given by other. A different from other directional (left to right) language pre-training model, this Masked LM objective assents the representation to synthesize the left-right context that allows deep bidirectional Transformer to pre-train [8]. Consecutively to train a model, some of the input words as well as tokens are masked in each sequence and then predict the masked token. The probability of each word in vocabulary is calculated in terms to predict the output with Softmax. Here a function, Known as BERT loss function, is taken into consideration to predict the Masked values and ignores the non-masked values. It means sonly masked words are predicted instead of reconstructing the entire input.

b) Next Sentence Prediction:

Several tasks namely Question-Answering task [19] eke Natural Language Inference depends on relationship of two sentences. Next Sentence Prediction helps in training a model which deems the sentence relationship. Here, a pair (X,Y) of sentences is considered as input. There is a 50% chance that Y sentence follows X, and 50% of time can be random sentence from the corpus. The fig-1 illustrates the input representation of BERT. Two tokens ([CLS], [SEP]) are inserted in the input. The [CLS] token represent the beginning of the first sentence and [SEP] token is inserted after end of each sentence. When token embeddings, segment embedding together with position embeddings are combined then they results in input embedding.

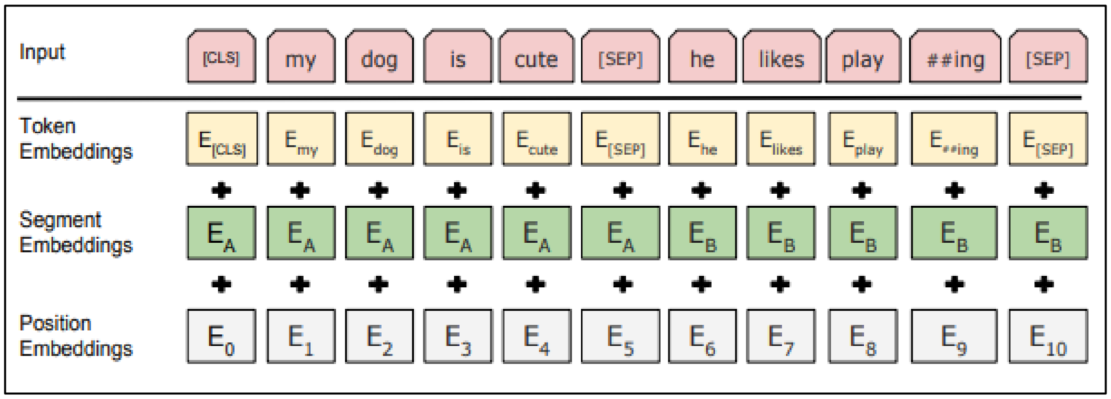

Figure 1. Representation of BERT Input [8]

To minimize this deficit of both tasks, the Masked LM along with Next Sentence Prediction is trained together in BERT Model. Inspired from the great achievement of Bert, Song et al. [16] proposed a new model, Masked Sequence to Sequence Pre-training (MASS) model for generation of language based on the framework of encoder and decoder.

4.2.2 Fine-Tuning: Using Bert is relatively straightforward for a specific task or for a wide variety of language tasks either they include single text or involve text pairs. Instead of using scratch every time for training a new network, lower layers of a model (pre-trained network) with generalized image features could be copied and transferred for use in another network with a different task because pre-trained layer already encode a lot of information about the language.

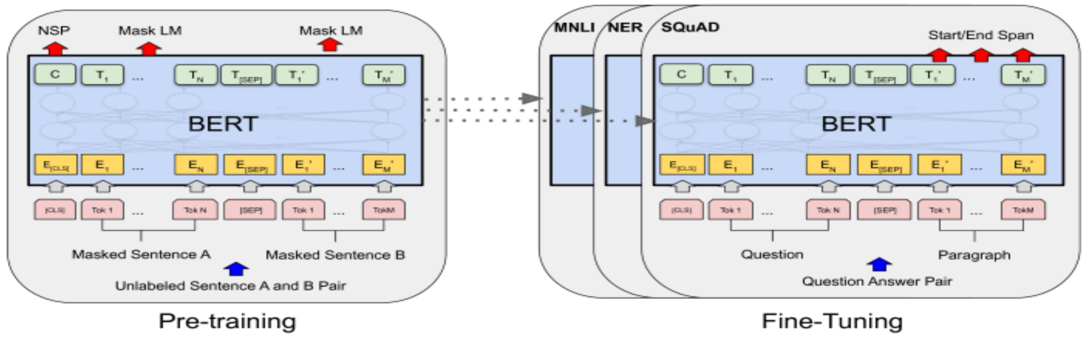

Figure 2. Pre-training and Fine-tuning Procedure of BERT [8]

A proper strategy for fine-tuning is required when we adapt BERT for a specific NLP task. Specifically, a pre-trained BERT model is taken into consideration and an untrained layer is added to it which helps to train a new model for our classification task. The pre-trained model takes less time to train new fine-tuned model because output of the pre-trained model is considered as feature of classification task during fine-tuning. Some layers are added in fine-tuning of NLP tasks such as: classification task, Question answering task (SQuAD) and Named Entity recognition. Not only BERT but many other language models (ELMO, ULMFIT, OPEN GPT etc.) have a drastic change in transfer learning in NLP.

5. Conclusion

Text summarization is very common and interesting field in machine learning. As research in this area continues many models such as BERT, ELMo, OpenAI GPT etc. is absolutely a rift in machine learning community. These breakthroughs result in fluent and accurate shortening of long document. In this paper, we started discussion from word embeddings or directional models and ends with BERT or bi-directional model. A study on how these word embeddings are replaced with language models is done. Here, a deep bidirectional architecture model BERT and its training strategies for summarization of a large corpus are discussed that result in quickly and high quality model with minimal effort and training time. Particularly, even low-resource tasks take the benefit of these outcomes.

REFERENCES

[1] Ranganathan, V., Raja, A., Ravichandran, A., & Viswanathan, N. (2018). Text Highlighting–A Machine Learning Approach. International Research Journal of Engineering and Technology, 5(8), 1603-1606.

[2] Shi, T., Keneshloo, Y., Ramakrishnan, N., & Reddy, C. K. (2018). Neural abstractive text summarization with sequence-to-sequence models. arXiv preprint arXiv:1812.02303.

[3] Khan, A., & Salim, N. (2014). A review on abstractive summarization methods. Journal of Theoretical and Applied Information Technology, 59(1), 64-72.

[4] Gaikwad, D. K., & Mahender, C. N. (2016). A review paper on text summarization. International Journal of Advanced Research in Computer and Communication Engineering, 5(3), 154-160.

[5] Peters, M. E., Neumann, M., Zettlemoyer, L., & Yih, W. T. (2018). Dissecting contextual word embeddings: Architecture and representation. arXiv preprint arXiv:1808.08949.

[6] Gong, C., He, D., Tan, X., Qin, T., Wang, L., & Liu, T. Y. (2018). Frage: Frequency-agnostic word representation. In Advances in neural information processing systems (pp. 1334-1345).

[7] Bollegala, D., Hayashi, K., & Kawarabayashi, K. I. (2017). Learning linear transformations between counting-based and prediction-based word embeddings. PloS one, 12(9), e0184544.

[8] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[9] Mandelbaum, A., & Shalev, A. (2016). Word embeddings and their use in sentence classification tasks. arXiv preprint arXiv:1610.08229.

[10] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).

[11] Ibrahim, N. M. A. R. (2019). A New Model for Arabic Text Clustering by Word Embedding and Arabic Word Net.

[12] Mirani, T. B., & Sasi, S. (2017, July). Two-level text summarization from online news sources with sentiment analysis. In 2017 International Conference on Networks & Advances in Computational Technologies (NetACT) (pp. 19-24). IEEE.

[13] Lloret, E., Romá-Ferri, M. T., & Palomar, M. (2013). COMPENDIUM: A text summarization system for generating abstracts of research papers. Data & Knowledge Engineering, 88, 164-175.

[14] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008).

[15] Sun, C., Qiu, X., Xu, Y., & Huang, X. (2019, October). How to fine-tune bert for text classification?. In China National Conference on Chinese Computational Linguistics (pp. 194-206). Springer, Cham.

[16] Song, K., Tan, X., Qin, T., Lu, J., & Liu, T. Y. (2019). Mass: Masked sequence to sequence pre-training for language generation. arXiv preprint arXiv:1905.02450.

[17] Chen, Y. C., & Bansal, M. (2018). Fast abstractive summarization with reinforce-selected sentence rewriting. arXiv preprint arXiv:1805.11080.

[18] Shi, T., Keneshloo, Y., Ramakrishnan, N., & Reddy, C. K. (2018). Neural abstractive text summarization with sequence-to-sequence models. arXiv preprint arXiv:1812.02303.

[19] Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016). Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250.