BERT as Text Generator and Question Answering Model

Andrea Bertolini
Politecnico di Torino
s286314@studenti.polito.it

Annalisa Deiana
Politecnico di Torino
s292286@studenti.polito.it

Stefano Giraudo
Politecnico di Torino
s292332@studenti.polito.it

Abstract

BERT is one of the most efficient language representation models introduced in last years. Its transformer-based architecture permits to create state-of-art models for a wide range of tasks by easily fine-tuning the pre-trained BERT. Conceptually simple and powerful, BERT is the strong basis for solving many tasks, especially the question answering one included in the functioning of search engines. Furthermore, BERT is shown to be a high-quality, fluent text generator too. In this work, first its structure is analyzed and its results as a text generator are compared to other existing models. Finally the power of BERT is extended in the specific field of question answering task.

1. Introduction

BERT (Bidirectional Encoder Representations from Transformers) [4] is one of the most recent and powerful language models used to create state-of-art models for a wide range of language tasks. It derives from the encoder structure of Transformer, an architecture introduced in 2017 based just on attention mechanisms. BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a consequence, it can be fine-tuned with just one additional output layer. The pre-training is composed by two objectives, the Masked Language Model and the Next Sentence Prediction pre-training objectives. Once pre-training is complete, BERT has some notion of language. With fine-tuning, it can be used for wide variety of tasks, especially Question Answering. BERT is applied to improve nowadays dominant search engines, considering this last task as a key point for web research. A further strength of BERT regards text generation. Text generation is done through sentence sampling followed by a ranking phase, to choose the best sentence in the context.

The power of Transformer architecture and the bidirectionality of BERT are exploited to get new state-of-the-art language models, such as XLNet. A comparison using Texygen metrics shows this.

2. Transformers

Transformer model was introduced in 2017 to overcome the sequential computation constraints of previous sequence transduction models. It relies entirely on attention mechanisms [11] to draw global dependencies between input and output, allowing in this way more parallelization and reaching a new state of the art in many tasks.

2.1. Model Architecture and Self-Attention

Like earlier models, the transformer adopts an encoder-decoder architecture, shown in the left and right halves of Figure 1, respectively. [1]

The encoder consists of identical encoding layers that process the input iteratively one layer after another. The function of each encoder layer is to generate encodings that contain information about which parts of the inputs are relevant to each other. It passes its encodings to the next encoder layer as inputs. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a fully connected feed-forward neural network. The self-attention mechanism accepts input encodings from the previous encoder and weighs their relevance to each other to generate output encodings. The feed-forward neural network further processes each output encoding individually. These output encodings are then passed to the next encoder as its input, as well as to the decoders. The first encoder takes positional information and embeddings of the input sequence as its input, rather than encodings. After each of these two sub-layers, a layer normalization is considered.

On the other hand, the decoder consists of identical decoding layers that do the opposite of the encoder, taking all the encodings and using their incorporated contextual information to generate an output sequence. Each decoder layer consists of three major components: a self-attention mechanism, an attention mechanism over the encodings, and a feed-forward neural network (with respect to encodings, an additional attention mechanism is inserted which draws relevant information from the encodings generated by the encoders). Like the first encoder, the first decoder takes posi-

[1] Image Credits to: Vaswani *et al*. [11]

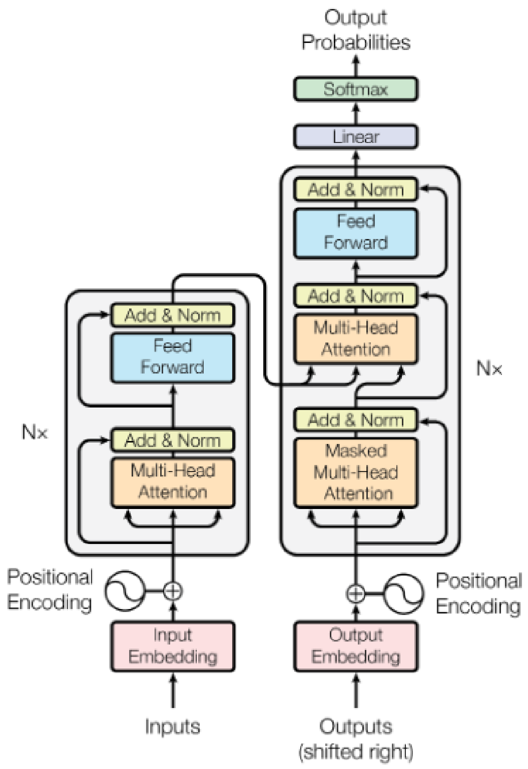

Figure 1. Transformer Model architecture. On the right-hand side the encoder. Decoder on the left part.

tional information and embeddings of the output sequence as its input, rather than encodings. The last decoder is followed by a final linear transformation and softmax layer, to produce the output probabilities over the vocabulary.

By attention mechanism it is meant that, for each input, attention weighs the relevance of every other input and draws from them to produce the output. By self-attention we mean attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence. The basis of self-attention is to use attention functions inside the model. An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values. Two main types of attention mechanisms are Scaled Dot-Product Attention and Multi-Head Attention. The last one applies several attention layers running in parallel. Multi-Head Attention is chosen for the Transformer, bringing to better results.

The 3 desiderata that motivated the use of self-attention in the architecture are:

1 Total computational complexity per layer.

2 Amount of computation that can be parallelized (as measured by the minimum number of sequential operations required).

3 Path length between long-range dependencies in the network (learning long-range dependencies is a key challenge in many sequence transduction tasks).

2.2. Examples

Inspired by Transformers, many variants and improved models have been developed.

GPT (Generative Pre-trained Transformer) [9] was introduced in 2018 as a proof of the effective improvement of natural language models comprising a wide range of diverse tasks when pre-trained on a large unlabeled text before a fine-tuning phase for a specific task. It reaches better performance than Transformers in many tasks such as question answering.

Instead, **Transformer-XL** is a variant of Transformer introduced in 2019 [3] that aimed at reaching better results after a slight modification to the architecture that enables learning dependency beyond a fixed length without disrupting temporal coherence (the originary problem of Transformer is the fixed-length context in the setting of language modeling). Some of the interesting applications of Transformer-XL is the text generation task.

Some studies made it possible to model not only unidirectional contexts, but also bidirectional ones (a model

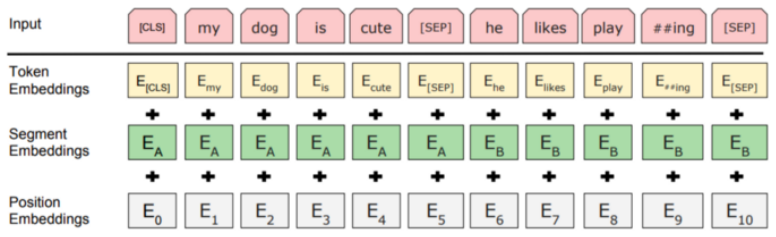
Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Figure 2. Input representation via 3 embeddings types: token embeddings, segment embeddings, position embeddings.

in this setting is for example BERT). With this capability, denoising autoencoding based pre-training achieves better performances than pre-training approaches based on autoregressive language modeling (such as Transformer-XL). On the other hand, they suffer from a pretrain-finetune discrepancy. For solving this problem, the **XLNet** model is introduced [12]. It is a generalized autoregressive pre-training method that enables learning bidirectional contexts by maximizing the expected likelihood over all permutations of the factorization order and overcomes the limitations of BERT thanks to its autoregressive formulation. Furthermore, XLNet integrates ideas from Transformer-XL, the state-of-art autoregressive model, into pre-training.

2.3. From Transformer to BERT

After GPT, in 2019 a new state-of-the-art NLP approach was introduced, **BERT** (Bidirectional Encoder Representations from Transformers). The further improvement of BERT over GPT is that even though both use transformer architecture to learn the text representations, the most recent method uses bidirectional transformer (both left-to-right and right-to-left direction) rather than left-to-right direction. Thanks to this deep bidirectional training structure, pre-trained BERT can be easily fine-tuned to create state-of-the-art models for a wide variety of tasks.

3. BERT

BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers.

As a consequence, it can be fine-tuned with just one additional output layer to create state-of-art models for many tasks, included the question answering task.

Furthermore, BERT is shown to be very useful to generate sentences.

3.1. Model Architecture and Pre-Training

BERT was introduced to improve fine-tuned based approaches by exploting bidirectionality, infact previous models had the limitation of one single direction. Since 2 steps in the BERT framework are pre-training and fine tuning, this is done in the first part using 2 unsupervised tasks, MLM (Masked-Language-Model Task) and NSP (Next Sentence Prediction Task).

The masked language model randomly masks some of the tokens (generally 15%) from the input, and the objective is to predict the original vocabulary id of the masked word based only on its context. On the other hand, NSP task jointly pre-trains text-pair representations.

Before these two pre-training tasks, all the input words (and special tokens) of the sequence of sentences are translated in embeddings to compute the input representations (Figure 2). These embeddings are useful for the BERT to work and they are built starting from 3 vectors:

1 token embeddings (embeddings of words): for each token (or word) a vector is considered to represent it;

2 segment embeddings: embeddings related to the sentence number in the sequence;

3 position embeddings: embeddings corresponding to the position of a word inside the sentence.

Adding these 3 vectors we obtain an input vector that is used as input to BERT. [2]

In the pre-training, at the output,there is a sequence of tokens, where the first one is used as output for next sentence prediction and the other tokens represent the output words corresponding to masked language modeling.

Finally using the output vectors it is possible to compute the loss (to measure the error) in order to train the model.

Once pre-training is complete, BERT has some notion of language.

Concerning the model architecture, it is a multi-layer bidirectional Transformer encoder based on the original implementation. In particular, the encoder of original Transformer is repeated many times sequentially (Figure 3). Depending on the number of parameters in the model, we have two possible models, BERT Base (110M parameters) and BERT large (340M parameters). [3]

[2]Image Credits to: Devlin *et al*. [4]
[3]Image Credits to: Evtimov *et al*. [5]

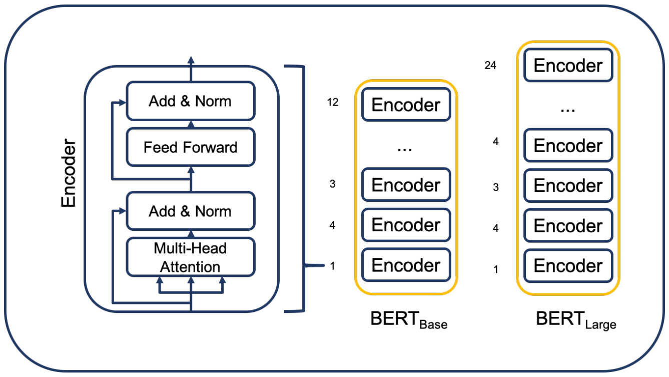

Figure 3. BERT Model Architecture.

3.2. Fine-Tuning and Question Answering Task

Fine tuning is straightforward since the self-attention mechanism in the Transformer allows BERT to model many downstream tasks, reason why, compared to pre-training, fine-tuning is relatively inexpensive. For each task simply plug in the task-specific inputs and outputs into BERT and fine-tune all the parameters end-to-end. Apart from output layers, the same architectures are used in both pre-training and fine-tuning: at the output, the token representations are fed into an output layer for token-level tasks.

An interesting application of these models is the Question Answering Task.

Recent advancements in NLP have unlocked the ability of machines to understand the text and perform different tasks such as answering questions, where the goal is to predict the answer text span in the text [1].

For the Question Answering task, BERT takes the input question and passage as a single packed sequence. Then it introduces a start vector and an end vector. The probability of each word being the start-word is calculated by taking a dot product between the final embedding of the word and the start vector, followed by a softmax over all the words. The word with the highest probability value is considered. A similar process is followed to find the end-word (Figure 4). [4]

An important dataset for fine-tuning BERT for Q&A Task is the SQuAD dataset. The Stanford Question Answering Dataset (SQuAD) is a collection of 100k crowdsourced Q&A pairs.

3.3. Improvements for Question Answering Task

Better achievements in Q&A are obtained considering improvements of BERT as underlying pre-trained models. Furthermore, many multilingual language models are used for this task [10].

[4]Image Credits to: Devlin *et al.* [4]

XLM-R (XLM-RoBERTa) [2] was introduced in 2020 and reaches state-of-art results in Q&A. [5]

By exploting knowledge of **XLM** and **RoBERTa**, it improves previous approaches, finishing to be particularly well performing for low-resource languages and very competitive with monolingual models. XLM is based on cross-lingual pre-training [7], showing to be very effective when dealing with multiple languages. Moreover, RoBERTa [8], an improved recipe for training BERT models, is pretty powerful since it can match or exceed the performance of all the post-BERT methods.

3.4. BERT for Text Generation

Further to being a sequence model used to achieve state-of-the-art results on a wide range of natural language understanding tasks, proving BERT can rank and sample sentences [6] is useful to consider it as a text generator too.

A text generator needs a way to sample generated sentences and in BERT this can be done. It is possible to define such a language model by a graph structure $G = (V, E)$, and a set of potentials $(\theta^c)_{c \in C}$ where C is defined as the set of cliques in graph G. In BERT's case, a sequence of these random variables is considered, modeling a sentence of words, and the random variables form a undirected fully-connected graph where edges indicate dependency. The potential in this case decomposes into an approximated sum of stochastic pseudo log-likelihood. Hence this proves the thesis.

Text generation is obtained with a sampling-ranking approach. Using Markov-Chain Monte Carlo (MCMC) sampling, a sequence of words is sampled, bringing to a sensed-sentence thanks to the stochastic pseudo log-likelihood maximization for finding each time the best word. Ranking

[5]For a specific application of XLM-R for Q&A refer to the code `https://github.com/annalisad98/BERT-Text-Generator-and-QA-Model/blob/main/Google_Search_Platform_XLMROBERTA.ipynb`. A web app for question answering inspired by Google Search Engine using XLM-R can be found at the link `https://d4ephv6bpz2b2gkf.anvil.app/6L3Z3ZV54XCQ2BF4DY7FJQBD`

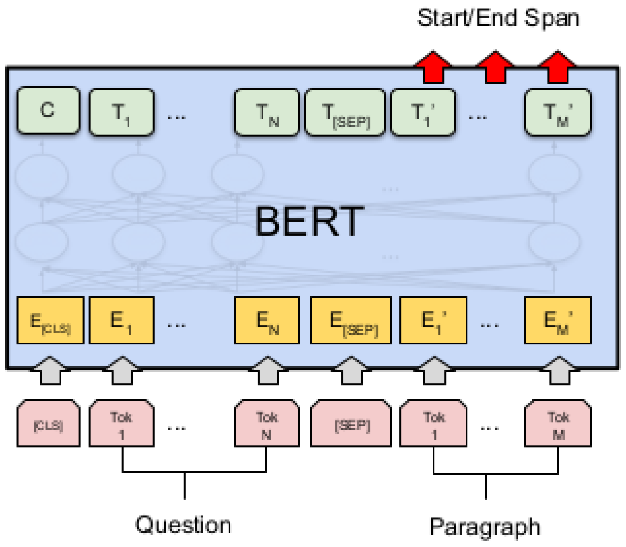

Figure 4. Fine-tuning Pre-Trained BERT for Question Answering Task.

is done by approximating the probability of each sentence in the context and choosing that with higher probability. [6]

4. BERT for Text Generation: Comparison to Existing Models

The goodness of generated sentences using BERT is compared to other models using specific measures. In this analysis Texygen is used, a benchmarking platform for text generation models.

4.1. Texygen

Texygen platform offers support to research on open-domain text generation models. Not only it contains a majority of baseline models (for having a comparison with different types of text generation models), but it has also a variety of metrics that evaluate diversity, quality, consistency, of the generated texts.

Concerning the models of the platform, there are various likelihood-based models. One of them is MLE language model. It adopts an explicit likelihood-based modeling of language. By maximizing the likelihood estimation, MLE manages to have an estimation of the generation procedure.

On the other hand, Texygen implements five text generation metrics, covering different aspects of a generated text: quality metrics, consistency metrics, diversity metrics. An important quality metric is BLEU. It analyzes the similarity between a group of generated sentences against a group of reference sentences. In case the comparison is made between generated sentences, BLEU is called Self-BLEU, a similarity measure. By definition, lower BLEU and Self-BLEU values mean more diversity, so better generated sentences. Another used measure to compare text generation models is the unique grams measure, which measures diversity in the generated text. Higher values mean higher percentage of grams that are unique in the generated text, indicating more diversity in the generation. This metric is somewhat in opposition to BLEU between generation and data, as fewer unique grams implies higher BLEU.

[6]The implemented code for text generation using BERT is shown at link `https://github.com/annalisad98/MLDL2021/blob/main/MLDLproject.ipynb`.

Model	Self-BLEU
BERT(base)	8.49
GPT	38.09
WT103	17.42

Table 1. Self-BLEU score. Compared to the traditional left-to-right language model, BERT generates sentences that are more diverse (lower Self-BLEU). For the WT103, 1000 sentences are sampled from the respective dataset.

4.2. Results

As shown in Tables 1 and 2 , compared to GPT, the BERT generations are more diverse. Referring to unique n-grams, Tables 3, 4, 5 indicate BERT has more diverse sentences generated than GPT with respect to WT103, TBC, and their own generations. Further, from Table 3 GPT has lower values than WT103 (while BERT has higher ones), so GPT generations have greater n-grams overlap with this dataset than WT103 has with itself, suggesting that GPT is relying significantly on generic sentences. Moreover, the percentage tends to increase with n.

Using Texygen metrics, BERT is compared to other ex-

Model	BLEU VS WT103	BLEU VS TBC
BERT(base)	8.51	7.04
GPT	11.32	30.02
WT103	14.82	5.79

Table 2. Corpus-BLEU score. BERT is pre-trained using English Wikipedia and TorontoBookCorpus datasets. From these values it is shown that BERT generates sentences that are approximately as diverse from those of WT103 and TBC as from themselves, meaning not a perceptable dependency on the datasets for pre-training.

Model	Unique-2grams VS WT103	Unique-3grams VS WT103	Unique-4grams VS WT103
BERT(base)	59.05	91.80	98.60
GPT	34.11	73.93	91.90
WT103	50.66	85.55	96.80

Table 3. Percentage of unique n-grams for $n = 2, 3, 4$ w.r.t. WikiText103 dataset. GPT has smaller percentage for all the 3 values, meaning BERT has more diverse generated sentences w.r.t. WT103 than GPT.

Model	Unique-2grams VS SELF	Unique-3grams VS SELF	Unique-4grams VS SELF
BERT(base)	63.13	92.38	98.24
GPT	31.83	68.60	88.60
WT103	51.78	85.85	96.75

Table 4. Percentage of unique n-grams for $n = 2, 3, 4$ w.r.t. their own generated sentences. GPT has smaller percentage for all the 3 values, meaning BERT has more diverse generated sentences than GPT.

Model	Unique-2grams VS TBC	Unique-3grams VS TBC	Unique-4grams VS TBC
BERT(base)	62.68	92.53	98.67
GPT	26.05	66.06	89.14
WT103	57.58	89.33	97.94

Table 5. Percentage of unique n-grams for $n = 2, 3, 4$ w.r.t. WikiText103 dataset. GPT has smaller percentage for all the 3 values, meaning BERT has more diverse generated sentences w.r.t. TBC than GPT.

isting models and values are reported in Table 6. BERT generates more diverse sentences than GPT and MLE. Despite this, more recent models improve the results, bringing to smaller values of Self-BLEU. Especially for XLNet, this justifies the trials for improving text generators using features from different models.

5. Conclusions

Arising from transformer encoder structure, BERT is a simple and powerful pre-trained language model, becoming a baseline for state-of-the-art models for many applications such as question answering task. Moreover, being a Markov Random Field Language Model, BERT is a high-quality, fluent text generator too. Its quality outperforms previous generators and is set as basis for the state-of-art text generation models. The complete code is available at https://github.com/annalisad98/BERT-Text-Generator-and-QA-Model.

References

[1] D. Bhageria. Build a smart question answering syste with fine-tuned bert. https://medium.com/saarthi-ai/build-a-smart-question-answering-system-with-fine-tuned-bert-b586e4cfa5f5, June 2020.

[2] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale, 2020.

[3] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context, 2019.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[5] R. Evtimov, M. Falli, and A. Maiwald. Bidirectional encoder representations from transformers (bert). https://humboldt-wi.github.io/blog/research/information_systems_1920/bert_blog_post/, February 2020.

[6] Yacine Jernite, Alexander M. Rush, and David Sontag. A fast variational approach for learning markov random field language models. In *ICML*, pages 2209–2217, 2015.

[7] Guillaume Lample and Alexis Conneau. Cross-lingual language model pretraining, 2019.

[8] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

[9] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf, 2018.

[10] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text, 2016.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[12] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding, 2020.

Model	Corpus-BLEU VS WT103	Self-BLEU	Unique-2grams	Unique-3grams	Unique-4grams
BERT(base)	8.43	16.27	2664.00	3383.60	3410.90
GPT	9.98	49.60	1548.80	2688.70	3132.60
MLE	12.61	25.08	2420.25	4743.94	5390.45
Transformer-XL	11.65	12.26	2805.00	3030.00	2985.00
XLNet	13.71	11.78	2800.00	3057.50	3015.00

Table 6. Comparison between BERT and existing generation models GPT, MLE, Transformer-XL, XLNet, using Texygen metrics.