Design Project

Natural Question Answer Research

Guide: Prof. D V L N Somayajulu

Group: Ashutosh Hathidara (116CS0007)
Nischay Pandey (116CS0021)
Akash Dhayal (116CS0037)

TABLE OF CONTENTS

1. Introduction

A main objective in building artificial intelligence systems is to build systems that can read the web pages, and then answer complex questions about any topic in that web page. This question-answering (QA) system could have a big impact on the way that we access information. Moreover, natural question answering is a fundamental task in the development of AI, since understanding text using natural language understanding and being able to answer questions about it through critical reasoning is something that we generally associate with intelligence.

If we give a question to natural question answering (QA) system, then it should be able to respond to it in context. QA systems imitate how people tend to look for information by reading the web to return answers to common questions. We can use ML and NLP concept to improve accuracy of these answers.

Currently available NLP models have been concentrated on extracting answers from a short passages rather than reading a whole page of text for proper context. As a result, the responses can be complicated or lengthy. A good answer will be both succinct and relevant.

The objective is to predict short and long answers to questions about Wikipedia articles. The dataset used for this project is taken from various resources like Jian Zhang's SQuAD 2016, Google AI's NQA benchmark Dataset 2019 but contains its own unique private test set. A visualization of dataset illustrates long and short answers wherever available.

This is a very popular research problem and still, there is a lot of scope for improvements. The dataset was taken from Google AI's research which is enhancement of SQuAD 2016. There are many models existing for Natural Language Understanding task among which BERT i.e. Bidirectional Encoder Representations from Transformers is a very popular model that works on the sliding window LSTM training.

The objective of the project is not just to query the data from the lake of texts, but it is also to provide the computer an ability to critically think about the question and to find the answer to those questions.

The model must be evaluated on the micro F1-score parameter. The reason for this is that if the model doesn’t give the answer then it is acceptable but it should not give the wrong answer. Thus, we are evaluating the model on F1-score rather than accuracy or error.

2. Dataset

To help development in natural question answering, Google AI has built the Natural Questions (NQ) dataset, and also a challenge website based on this dataset. The NQA dataset consists of questions from organic users, and it requires QA systems to read and get contexts from a whole Wikipedia article that may or may not contain the answer to the question asked. The addition of organic user questions, and the necessity that solutions should read a complete page to find the answer, because NQA to be a more pragmatic and non-trivial task than prior QA datasets.

Each example consists of Wikipedia article, corresponding question, and the long answers. The training data also gives the correct long and short answers or answers for the sample, if available.

For every article and question pair, we are predicting/selecting long and short answers to the question extracted directly from the article given. A long answer would be a longer section of text that answers the question some sentences or a paragraph. A short answer might be a sentence or phrase, or even in some cases a True/False or Yes/No. It is an assumption here that the short answers are always contained within / a subset of one of the long answers. A provided Wikipedia page can allow for both long and short answers, based on the question.

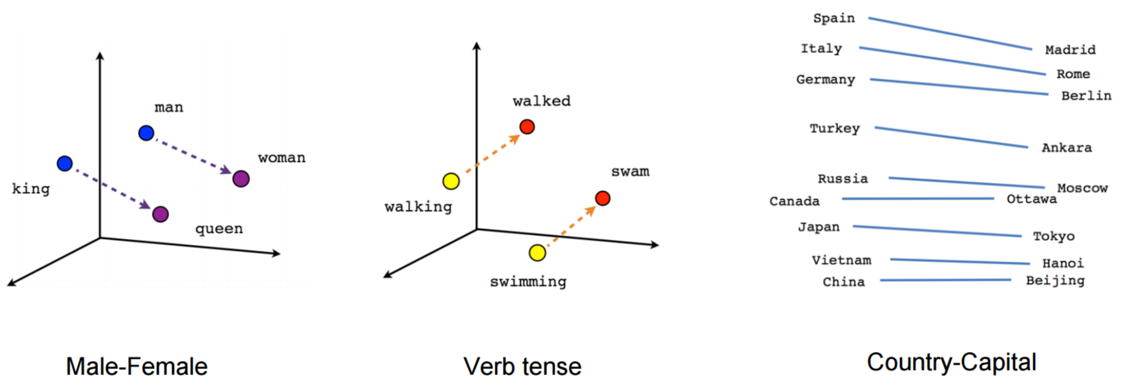
There is a lot more information about the dataset and what we are predicting are available on the Github page for the Natural Questions and answers dataset. Note that we are using the simple text type/version of the data - most of the HTML tags have already been removed, and only necessary to be used paragraphs/sections are included.

Data fields in the data are explained below :

- **document_text** - the content of the article in question (with some HTML tags to provide document structure). The content can be tokenized and cleaned by stemming whitespace.
- **question_text** - the question based on article we want to answer.
- **plausible_long_answers** - a JSON array consisting of every long answers.
- **annotations_ranges** - a JSON array consisting all of the correct long and short answers.
- **document_url** - the URL for the full article.This is NOT the simplified version of the article so indices from this cannot be used directly. It is also possible that the text may also not match the HTML used to generate document_text.
- **example_id** - a unique ID for the sample.

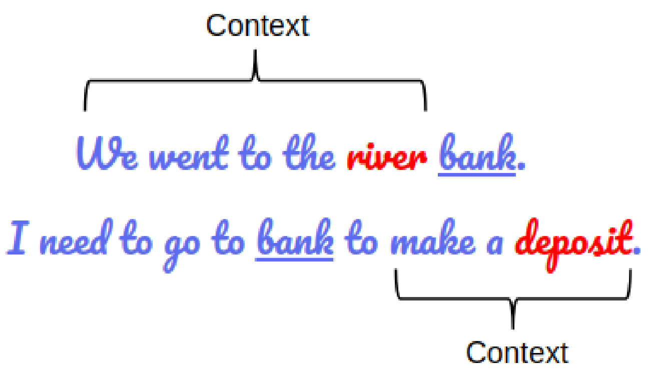
3. Preprocessing

We are dealing with text here and the problem with text is that we can not use it directly into the model we are building. Therefore, there are some ways in which we can refine the text and then we can vectorize it.



Ref. Blog AV 2019

We can use TFIdfVectorizer which directly converts the word into vector using Word2Vec pretrained model. There is a limitation of Word2Vec model. I.e. it generates the embedding of the word same irrespective of the context in which the word is used. For example,


Context
We went to the river bank.
I need to go to bank to make a deposit.
Context

Ref. Blog AV 2019

In the above example, we can observe that the word **bank** denotes the shore of the river in the first sentence whereas it denotes the financial organisation where we withdraw and deposit money. Since, the spelling of both words are the same, If we only define context based on the spelling of that particular word only, then it will be improper. Therefore, Word2Vec will definitely fail in one of the above two cases and it will not be able to generate proper context of words.

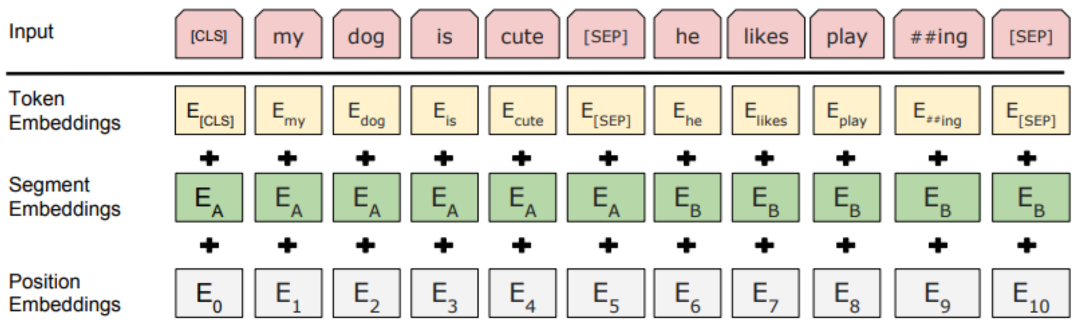
Instead, we should vectorize the given word w.r.t the context in which it is appearing. It means that we have to vectorize the words based on the words appearing before and after it in the sentence. This has led to the birth of BERT model to train the words such that it can create embeddings of words based on the context in which it appears. Note that if we use the same word in two different sentences then both these same words may have different vector embeddings based on the context of the sentences.

We are using **FullTokenizer** class of **tokenization** package to tokenize the document text. The vectors are initialized by random values and will update the weights during back propagation of training phase.

The vocabulary size taken for this project has **30522** words. We are using the equivalent techniques as Word2Vec but here the training part is different and more complex because the model that we have used here is deep bidirectional RNN. The length of embeddings used is default BERT embedding length.

Cleaning of TEXT

The creators of BERT model included a particular policies which can denote the inputted document string data. These designs are very robust and particularly flexible to work with.



Ref. Blog Analytics Vidya 2019

To fundamentally describe, each input embedding consists of three distinctive layers embedding:

1. **Spatial based Embeddings**: BERT model uses this position based embedding in order showing the spatiality each embedding contains in the given text corpus. We are including them to prevent the problem occurring with Transformer which, not as a Recurrent Neural Network, which was unable to find order patterns related features and sequence related feature in the text corpus.
2. **Relation based Embeddings**: BERT model used for this purpose to find in the given two sentences if there exist a relation between EA(embedding A) and EB(embedding B).It just gives output as IsNextSentence otherwise NotIsNextSentence if there is no contextual relationship between A and B
3. **Badge based Embeddings**: We have built vocabulary consisting of token words and we have used that vocabulary to train these model. These embeddings are very critical to parameter selection and training part is very sensitive.

> *For a given token, its input representation is constructed by summing the corresponding token, segment, and position embeddings.*

Such complex embedding technique proves to be meaningful to the BERT model and make it different from other models.

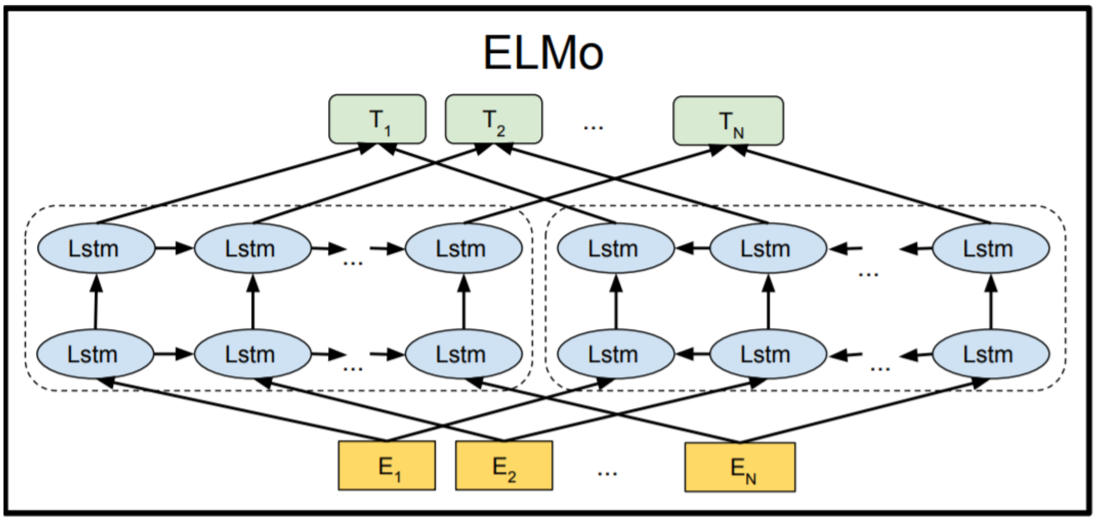
These types of steps in text preprocess make BERT model very robust. This shows that without making a lot of changes in the model we can train the model for n number of different NLP problems.

4. Methodology

Since we know the research community related to Natural Language Processing (NLP) has many non trivial problems because there is very less available text data for training and test purpose. The deficiency of dataset is there because many NLP tasks have only few thousand text corpus data to train model and since the NLP research is very specialized kind of research, we need more data. Whereas, we know that deep learning is the technology in which the amount of data it uses is very huge. The data must consistently improvising when we train it on a very large dataset,These example are through hand engineering. To fill this deficiency in shortage of new text corpuses, researchers have found a new paradigm and technology for training natural language processing models utilizing very volumetric large data which not annotated (not hand crafted), textual data which is taken from websites which is also known as pre training. These already trained models can now be used for different tasks. We can tune these models on very tiny datasets related to sentiment detection and NQA dataset, which results in some good accuracy results while comparing them to training as compared to training them again

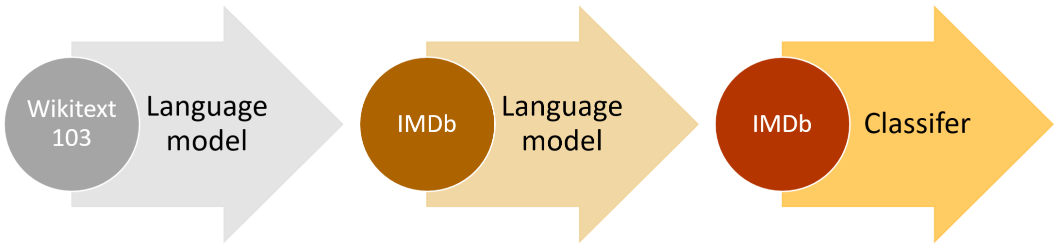
ELMO and ULMFiT

Methew Peters' ELMo 2018 was response to challenge of Polysemy by Natural Language Processing community. The Polysemy is general term which is very common. We can give example of some words which can be used in different places and there, they may have different meaning. This is called Polysemy. We can train data on simple small number of layers neural networks but that is inefficient. That’s why we have chosen to train model on deep LSTM Network. Also, LSTM layers are bidirectional. This shows that different words can be represented by distinct embeddings generated using ELMO which are used in different contexts.



Ref. Blog Analytics Vidya 2019

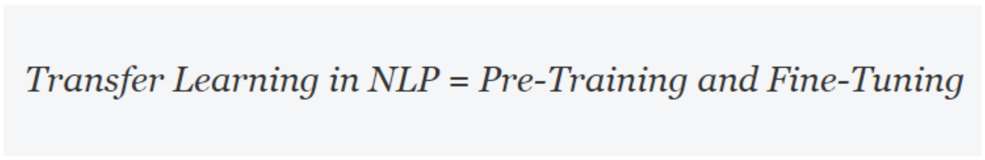
That is the phase of research when we began to use pre training models for training mechanism. The process is shown as below.



Ref. Blog Analytics Vidya 2019

Jeremy Howard's ULMFiT 2018 has made further advancement in this. These models are very robust and complex at the same time. They can be used with very less amount of dataset to train on and they can tune themselves to understand that data very well. Since we know that earlier there was nothing like transfer learning for NLU and NLP tasks. But

ULMFiT has made significant enhancement in this. The time when the ULMFiT model was discovered, it was the supreme time of using pre training and other advanced techniques in NLP tasks.

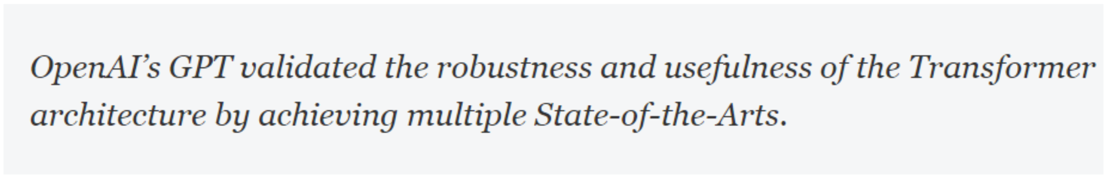


Transfer Learning in NLP = Pre-Training and Fine-Tuning

Most of the NLP research after ULMFiT has followed components of the above equation and gained best benchmarks.

Open AI's GPT

GPT developed by Open AI 2019 expanded the limits of the number of functionalities provided by robust models ELMO and ULMFiT which can be trained before and used in multiple way by fine tuning them. Generative Pre Training especially transformed to paradigm of building transformers like ELMO and ULMFiT from traditional ways of building LSTM networks like Word2Vec. The Generative Pre training models are not just limited to small analysis and classification tasks, they can also understand the complex data like conversations, doctor prescriptions, emotion recognition etc. Generative Pre Training also concentrates on the importance of Transformers which is now has became paradigm of NLP. For the information, these are also preferred because the speed of training is very high for transformers as compared to conventional LSTM based networks. Attentional models are the paradigm used for finding very micro features and learn them on data even if the data is very small. We use these techniques in the Generative Pre Training.



OpenAI's GPT validated the robustness and usefulness of the Transformer architecture by achieving multiple State-of-the-Arts.

And this is how Transformer lead BERT and all other successful models thereafter in NLP research. There also exists many very critical successes in modelling and discovery projects. We are not describing them here because they are out of scope of this project. One of such modelling techniques is semi-supervised sequence learning. We have not mentioned because they are out of scope of this project.

BERT

So, the new approach to solve Natural Language Processing projects became two step process:

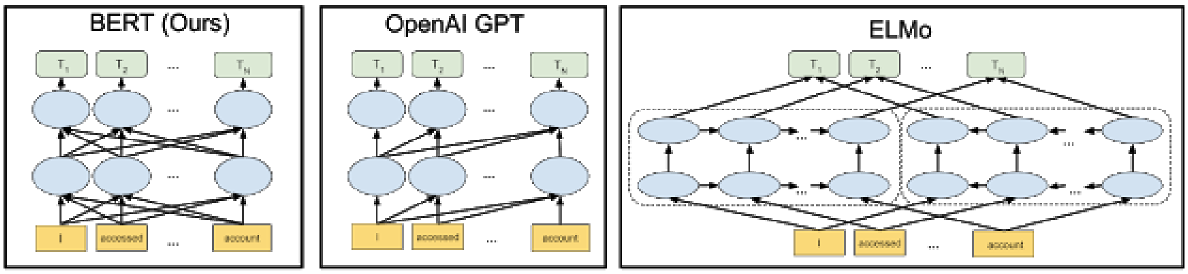
1. Unsupervised or semi supervised technique focuses in this BERT model to train the model on large unlabeled text corpus.
2. Supervised technique focuses on fine tuning the BERT model for the specific task we want to train upon it uses it previous huge knowledge to train on this specific task

Google AI has created a new paradigm for NLP training called BERT which is also described as Bidirectional Encoder Representations from Transformers .By this model, we can train known QA problem on Cloud TPU within few minutes(approx 30 mins), and on our GPU within few hours.We have made the model on Tensorflow and using a lot of pretrained models which are available. In the same paper, they showed model results on several NLP tasks, including one of the famous Stanford Problem of Question Answering on its Dataset (SQuAD version 1.1).

How is BERT differs from other models?

BERT is made up on the past few work in training contextual based representations which contains many already trained existing models such as ULMfit, elmo and Generative pre-training models. Whereas, As compared to these past model only BERT model is the first model which is completely bidirectional in nature and which is being trained on only plain text such as Wikipedia pages.

How it effects? Pre-training can be classified into two parts either it is contextual or non-contextual and further contextual can be classified into two one is unidirectional and other is bidirectional .Context free models such as GLoVe or word2vec forms a single word embedding for each of the text in the given vocabulary.Let us consider an example if we take the word bank in case of Bank account and bank of the river, word2vec which is unidirectional in nature will consider only the first part suppose I accessed the bank account, it will only consider I accessed the as its context but not account . However, if we take BERT model in consideration it will take context from both the end of the sentence it will consider I accessed the {Blank space} as well as account in consideration to find the context of word bank in the sentence. An overview of BERT model as compared to other contextual model is shown below. These lines below denotes the flow of info from one of the bottom part to the next top layer. The green portion indicate contextualized based representation of each of the given input word:



Ref. Google AI Blog on BERT 2019

Pre Processing Tasks

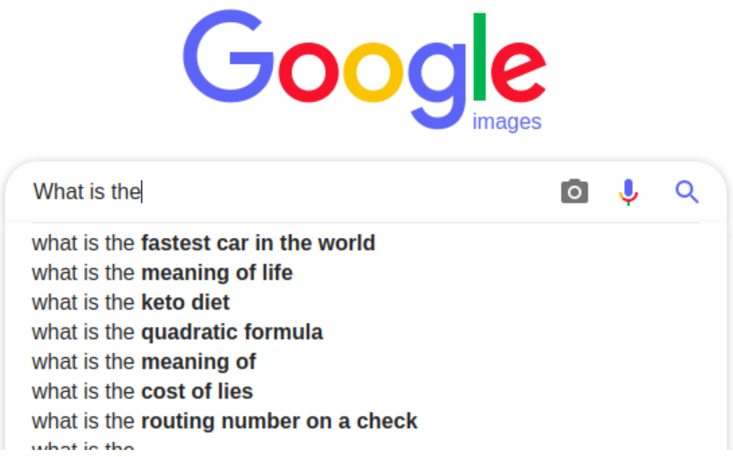
BERT model is being trained on two different types of NLP tasks

- First is MASKing of some words,in which model is trained on removing few words from sentence and training to predict that word in a sentence
- Second is finding the relation between two sentences, given two sentence finding if there exists a relation between two sentences or not.

Mask Language Modelling (Bidirectionality)

BERT model is bidirectional in nature as we have considered the example of bank word in the sentence i deposited rupee in bank account. It will consider the context from both the left and right end of the word to find the true fit in the sentence.

Earlier we have model which can find the context from one direction,either from right to left or left to right but not both. This model can do it easily.



What is the

what is the **fastest car in the world**
what is the **meaning of life**
what is the **keto diet**
what is the **quadratic formula**
what is the **meaning of**
what is the **cost of lies**
what is the **routing number on a check**

Ref. Blog Analytics Vidya 2019

The solution given by ELMO is that we can train two different types of RNN networks. One can be trained from one end to another and another in the opposite direction of the former. Each cell of RNN network is of LSTM blocks. Although it significantly improvised on available modelling paradigms, it is very insufficient.

> *"Intuitively, it is reasonable to believe that a deep bidirectional model is strictly more powerful than either a left-to-right model or the shallow concatenation of a left-to-right and a right-to-left model." – BERT*

BERT model is advanced improvement of GPT and ELMo models.

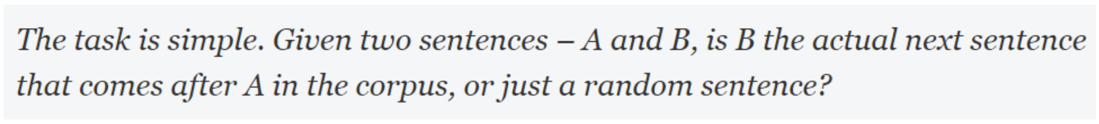
About Masked Language Models

Take an example that we have a text "I like cat pictures given in articles on Wikipedia". Our objective is that we want to train NLP model for this type of data. Rather than we build model to predict following sequence from text, we are able to train model which is able to predict next type of sequence from the text data.

Suppose we change “Wikipedia” to <MASK>. We can call it a badge which is taken in order to specify that that particular word is not there in the sentence. We can then go on with the modelling and training phase which can predict “Wikipedia” as the missing token: “I like cat pictures given in articles on [MASK]”.
Above explained is the gist of MLMs. The researchers who have worked on BERT model have added some modifications so that the model can be improved further.

- The model can choose very particular kinds and position of sequences. So model can be biased. But researchers have tried with the random fraction of masked examples. The data was about 14.9% of masked examples.
- In the masking example task, the sequence which was put in place of maskable sequences actually was never badges made like <UNK> which will not be discovered at the time of training.
- To improve upon that, NLP model creators have started utilizing the following mentioned strategies.
    - In 80% cases, badges corresponding to maskable sequences were put in place of actual sequences.
    - In 10% cases, random sequences were put in place of maskable sequences.
    - In 10% cases, they did not impose any replacement to actual sequence.

Next Sequence Relation Finding
Masked Language technique can learn to find the relation between two contexts. Moreover, BERT model is also trained to find the relation between two given texts it finds if two of the sentences have some context or not in between them If we take an example of two sentences A and B. Below..


The task is simple. Given two sentences – A and B, is B the actual next sentence that comes after A in the corpus, or just a random sentence?

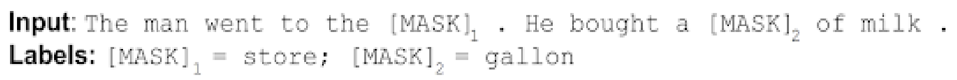
Since this is a simple task we need to divide any data text into two sentences to train the context finding for model.Same as Masked Language Models,. Let’s take this with an example:
Suppose we are taking texts data of 100000 sentences.So we can take two at a time to find the relation between them so finally we will be having 50000 training examples.

- Half of the training set, contains in which second sentence is the next sentence to A
- Another half of the training sentence contains in which next sentence is not the next sentence to A.
- And we need to find ‘ISNEXT’ and ‘NOTNEXT’ label for them.

And that is why BERT is as powerful model because it is being trained on both Mask modeling as next sentence prediction model before using for any task.

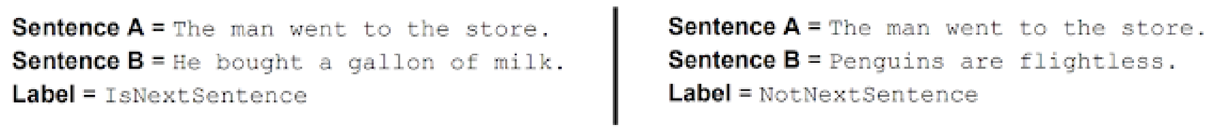
The Strength of Bidirectionality

Ifv we are saying that bidirectional nature is this much powerful then we have not implemented it yet.To understand it, previously unidirectional models well doing well and in low cost on finding next word based on past history . Whereas , if we see the case of bidirectional it is not feasible to train a bidirectional model by just putting some constraints on each word on its last few words *and* next few words within the given phase,because it will stuck in problem to find itself it will start seeing itself to predict itself. To solve the above problem, we use a direct technique of masking out some of the words in the input data and then process each word bidirectionally to predict the masked words.we using the technique of masking leaving out a few words and let it to predict it based on the context of left and right For example:


Input: The man went to the [MASK]$_1$. He bought a [MASK]$_2$ of milk .
Labels: [MASK]$_1$ = store; [MASK]$_2$ = gallon

Ref. Google AI Blog on BERT 2019

Although, we know that pre training and transfer learning was used since very past, but we were not using them for text data. With the creation of BERT model, we have started using these techniques on text data also. BERT model can also learns to find a relationship between two texts using transfer learning for trivial problem irrespective of the place from where the data is produced. Let's suppose, we take two chunks of texts. Our goal is to predict whether one sentence is in any way related to other or can it appear next to the other? See the below image:



Sentence A = The man went to the store.
Sentence B = He bought a gallon of milk.
Label = IsNextSentence

Sentence A = The man went to the store.
Sentence B = Penguins are flightless.
Label = NotNextSentence

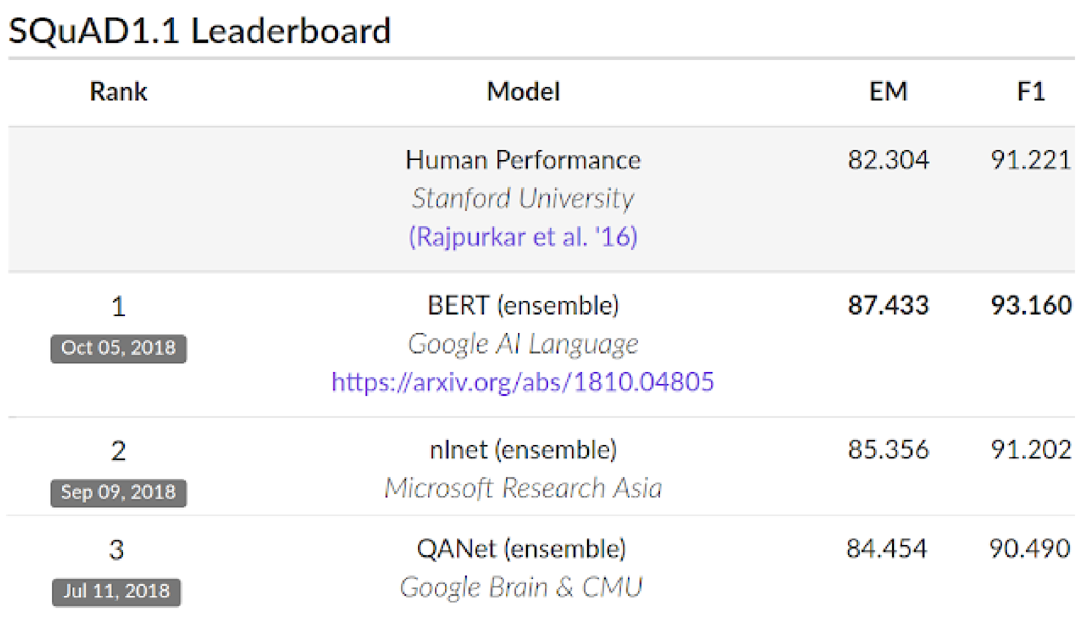
Ref. Google AI Blog on BERT 2019

Modelling and first time Training using faster devices

All these topics that we have explained till now seem to be simple and easy to understand, so what is that missing piece or non trivial thing because of which it performs this much greatly? The fastest devices to train on till now are TPU. These devices provide way and easiness of doing fast tests, to resolve mistakes, or to adjust previous techniques used, because of which we have started thinking beyond transfer learning and PT functionalities. A new paradigm called Design of Transformers, which was built by Google AI team in the year of 2016-17, provide the base that there is necessity of transforming BERT model efficient and sophisticated.

Results with BERT

To find out the performance of BERT model as compared to other best model we find out that BERT is achieving good performance with not much changes in the neural network architecture. Upon SQuAD dataset v1.1 dataset and BERT got the micro F1 score of 93.2% , overcoming the previous good score of 91.6% and of human accuracy which was 91.2.:

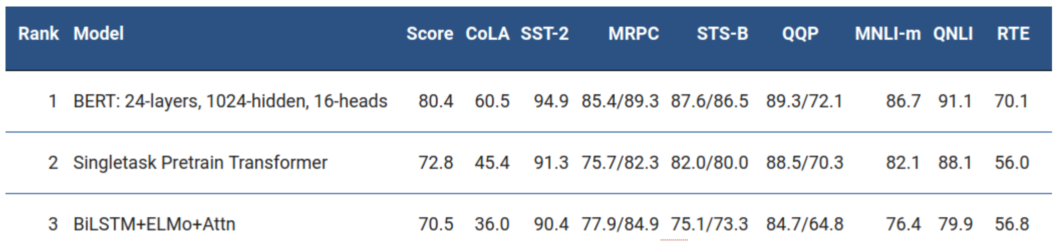


SQuAD1.1 Leaderboard

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar et al. '16)	82.304	91.221
1 Oct 05, 2018	BERT (ensemble) *Google AI Language* https://arxiv.org/abs/1810.04805	**87.433**	**93.160**
2 Sep 09, 2018	nlnet (ensemble) *Microsoft Research Asia*	85.356	91.202
3 Jul 11, 2018	QANet (ensemble) *Google Brain & CMU*	84.454	90.490

Ref. Google AI Blog on BERT 2019

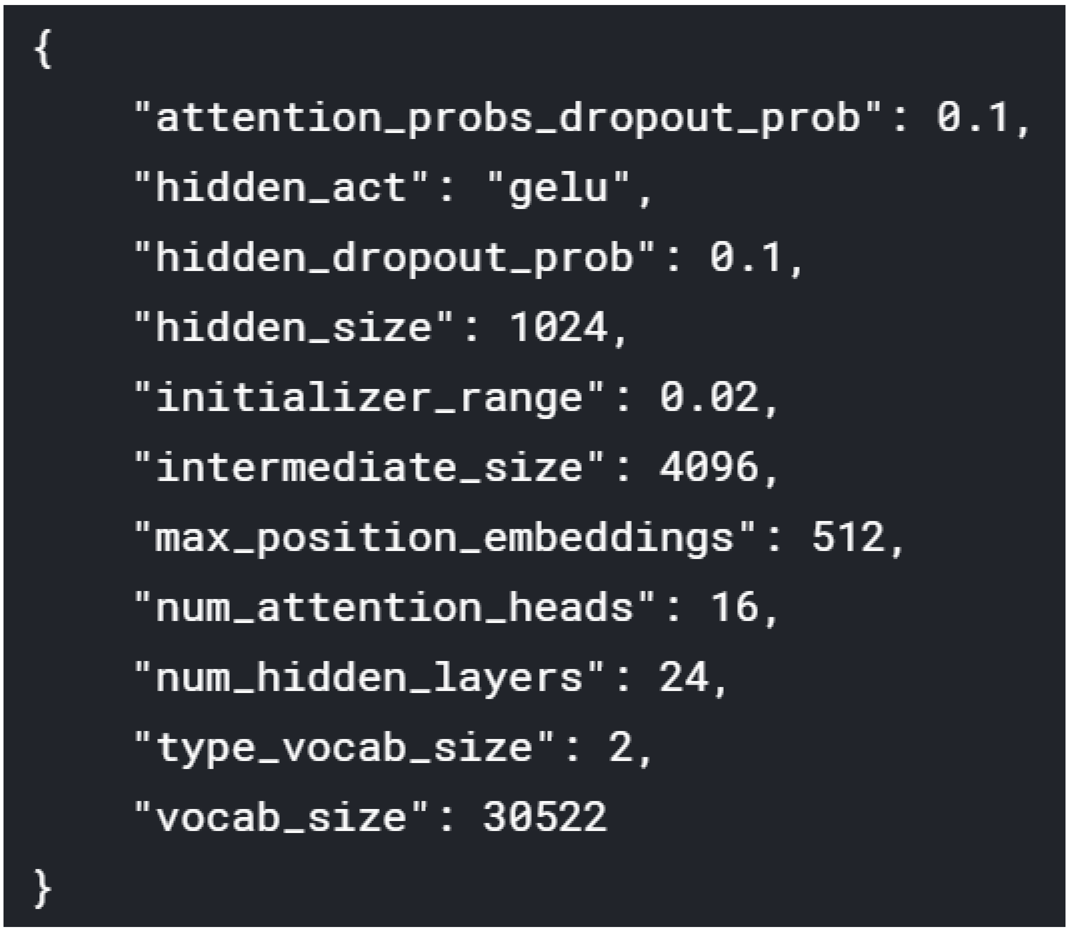
BERT also proved itself as the best existing model by getting a score of 7.6 on a complex GLUE benchmark, a combination of nine different Natural Language Understanding tasks.Human labelled training data was ranging from 2500 to 400000 examples, in those BERT model gives the best accuracy and keep on improving on them.



Rank	Model	Score	CoLA	SST-2	MRPC	STS-B	QQP	MNLI-m	QNLI	RTE
1	BERT: 24-layers, 1024-hidden, 16-heads	80.4	60.5	94.9	85.4/89.3	87.6/86.5	89.3/72.1	86.7	91.1	70.1
2	Singletask Pretrain Transformer	72.8	45.4	91.3	75.7/82.3	82.0/80.0	88.5/70.3	82.1	88.1	56.0
3	BiLSTM+ELMo+Attn	70.5	36.0	90.4	77.9/84.9	75.1/73.3	84.7/64.8	76.4	79.9	56.8

Ref. Google AI Blog on BERT 2019

The implementation of deep LSTM network is very brainstorming task because of more than 300 million parameter training is involved. There are also many hyper parameters which helps train parameters during training. Selecting those hyper parameters is very crucial task. We have used hyper parameters as shown below for the training phase.

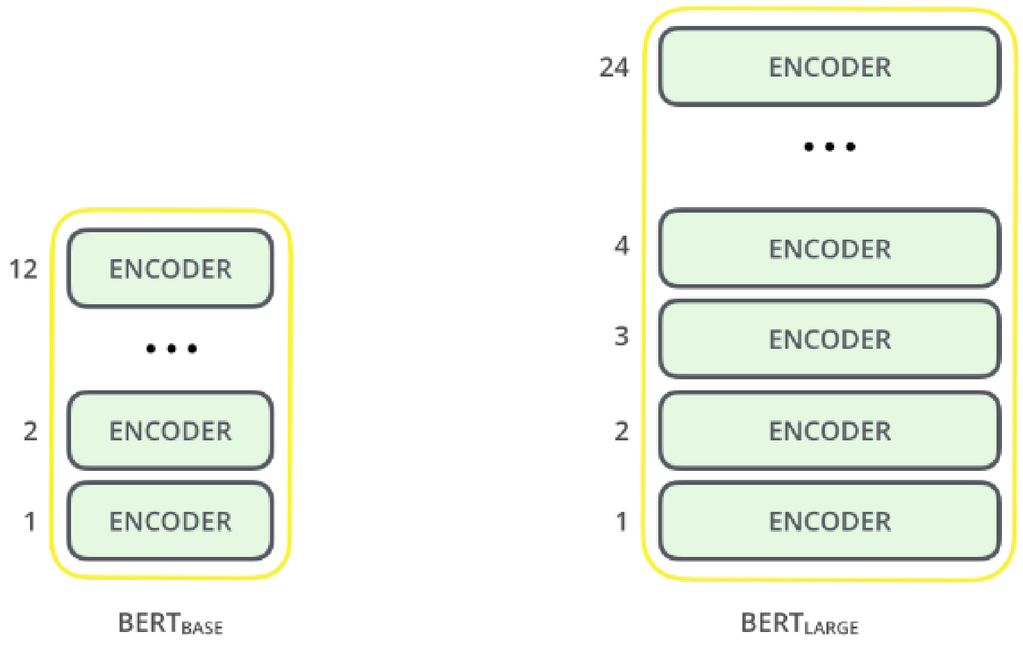


```
{
    "attention_probs_dropout_prob": 0.1,
    "hidden_act": "gelu",
    "hidden_dropout_prob": 0.1,
    "hidden_size": 1024,
    "initializer_range": 0.02,
    "intermediate_size": 4096,
    "max_position_embeddings": 512,
    "num_attention_heads": 16,
    "num_hidden_layers": 24,
    "type_vocab_size": 2,
    "vocab_size": 30522
}
```

Ref. Implementation

5. Implementation

The model is a non trivial and complex deep implementation of LSTM Networks. But the implementation is this model is easy with the help of already known frameworks. There are two different types of BERT models. i.e. Lite model to use in edge devices such as raspberry-pi and mobile devices and large model for cloud applications. The structure of the two models is represented below.



Ref. Blog Analytics Vidya 2019

We have worked on Tensorflow 2.0 for our training which is recently launched in sep 2019 and contains many high level useful functions to process text and contexts in the documents. Many pre-trained models are popular including ELMo, GPT, and BERT.

TensorFlow 2.0

Ref. Tensorflow 2.0

6. Conclusion & Results

The model is very good and accurate and giving a micro F1 score of **0.70**. The model was trained in a semi-supervised way. Supervised training was done in order to learn the context of the words in different positions of the sentences . Whereas, Unsupervised training is done after the model learns the context in order to unlearn if there is high variance. We evaluate and tune the model by asking the model to fill in the blanks to check whether the model has learned the context correctly or not.

7. References

- BERT Github Repo 2019 by Google AI
- Jian Zhang's SQuAD 2016
- Google AI's NQA benchmark Dataset 2019
- Ming-Wei Chang's BERT Paper 2019
- Google AI Blog on BERT 2019
- Effective TensorFlow 2.0 Guide