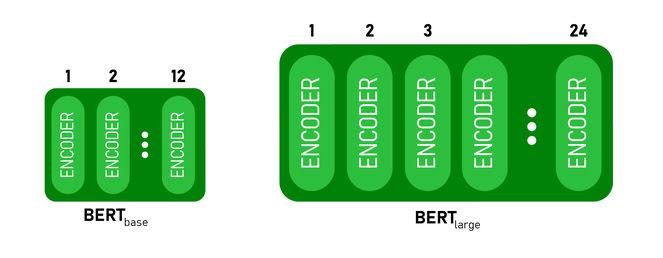
**Assignment 07**

**1. Explain the architecture of BERT ?**

**Ans:** BERT (Bidirectional Encoder Representations from Transformers) is a Natural Language Processing Model proposed by researchers at Google Research in 2018.

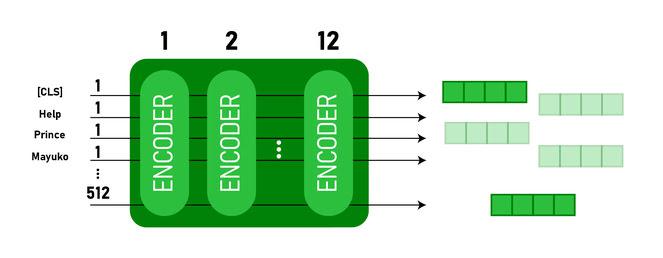
BERT is released in two sizes **BERTBASE** and **BERTLARGE**. The BASE model is used to measure the performance of the architecture comparable to another architecture and the LARGE model produces state-of-the-art results that were reported in the research paper.

BERT is basically an Encoder stack of transformer architecture. A transformer architecture is an encoder-decoder network that uses self-attention on the encoder side and attention on the decoder side. BERTBASE has 1*2 layers in the Encoder stack* while BERTLARGE has *24 layers in the Encoder stack*. These are more than the Transformer architecture described in the original paper (*6 encoder layers*). [BERT](https://www.geeksforgeeks.org/understanding-bert-nlp/) architectures (BASE and LARGE) also have larger feedforward networks (768 and 1024 hidden units respectively), and *more attention heads (12 and 16 respectively)* than the Transformer architecture suggested in the original paper. It contains *512 hidden units and 8 attention heads*. BERTBASE contains 110M parameters while BERTLARGE has 340M parameters.



*BERTBASEand BERT LARGE architecture.*

This model takes the **CLS**token as input first, then it is followed by a sequence of words as input. Here CLS is a classification token. It then passes the input to the above layers. Each layer applies [self-attention](https://www.geeksforgeeks.org/self-attention-in-nlp/) and passes the result through a feedforward network after then it hands off to the next encoder. The model outputs a vector of hidden size (*768*for BERT BASE). If we want to output a classifier from this model we can take the output corresponding to the CLS token.



*BERT output as Embeddings*

Now, this trained vector can be used to perform a number of tasks such as classification, translation, etc

**2. Explain Masked Language Modeling (MLM) ?**

**Ans:** Masked language modeling is an example of autoencoding language modeling (the output is reconstructed from corrupted input) - we typically mask one or more of words in a sentence and have the model predict those masked words given the other words in sentence. By training the model with such an objective, it can essentially learn certain (but not all) statistical properties of word sequences.

BERT is a model that is trained on a masked language modeling objective.

Language modeling approaches shown in figure below.

A picture containing text, screenshot, menu, document

Description automatically generated

Language modeling approaches - Autoregressive approach (e.g. left to right prediction, right to left prediction). Masked language approach - Using prediction of a word using all other words in a sentence (the words in red in BERT case is masked out - replaced with a special token [MASK]). BERT masks about 15% of words in a sentence and using the context words to predict it.

Masked language modeling is useful when trying to learn deep representations (that is learning multiple representations of a word using a deep model - these representations have shown to improve performance in downstream tasks. For example lower layer representations of certain models being useful for syntactic tasks whereas higher layer representations for semantic tasks) for a word using words from either side of a word in a sentence (deep and bidirectional representations).

A masked language model is particularly useful for learning deep bidirectional representations because the standard language modeling approach (autoregressive modeling) wont work in a deep model with bidirectional context - the prediction of a word would indirectly see itself making the prediction trivial as shown below (the word “times” can be used in its own prediction from layer 2 onwards. BERT addresses this problem by replacing the word being predicted with a mask token) . However, we could also learn deep bidirectional representations without having to resort to masked language modeling by using the permutation approach of a more recent model - XLNet.

A picture containing diagram, screenshot, line, design

Description automatically generated

**3. Explain Next Sentence Prediction (NSP) ?**

### **Ans:** Next sentence prediction (NSP) is one-half of the training process behind the BERT model (the other being masked-language modeling — MLM). Where MLM teaches BERT to understand relationships between words — NSP teaches BERT to understand longer-term dependencies across sentences. Next Sentence Prediction Using BERT

BERT is fine-tuned on 3 methods for the next sentence prediction task:

* In the first type, we have sentences as input and there is only one class label output, such as for the following task:
  + **MNLI**(Multi-Genre Natural Language Inference)**:** It is a large-scale classification task. In this task, we have given a pair of sentences. The goal is to identify whether the second sentence is entailment, contradiction, or neutral with respect to the first sentence.
  + **QQP**(Quora Question Pairs): In this dataset, the goal is to determine whether two questions are semantically equal.
  + **QNLI** (Question Natural Language Inference): In this task, the model needs to determine whether the second sentence is the answer to the question asked in the first sentence.
  + **SWAG** (Situations With Adversarial Generations): This dataset contains 113k sentence classifications. The task is to determine whether the second sentence is the continuation of the first or not.

A picture containing text, screenshot, diagram, rectangle

Description automatically generated

*BERT architecture first type*

* In the second type, we have only one sentence as input, but the output is similar to the next class label. Following are the task/datasets used for it:
  + **SST-2**(The Stanford Sentiment Treebank): It is a binary sentence classification task consisting of sentences extracted from movie reviews with annotations of their sentiment representing in the sentence. BERT generated state-of-the-art results on SST-2.
  + **CoLA:**(Corpus of Linguistic Acceptability): is the binary classification task. The goal of this task to predict whether an English sentence that is provided is linguistically acceptable or not.

A picture containing text, screenshot, diagram, rectangle

Description automatically generated

*BERT architecture second type*

* In the third type of next sentence, prediction, we have been provided with a question and paragraph and outputs a sentence from the paragraph that is the answer to that question. It is performed on SQuAD (Stanford Question Answer D) v1.1 and 2.0 datasets.

A diagram of a question and answer sentence

Description automatically generated with low confidence

*BERT architecture 3rd type.*

In the above architecture, the [CLS] token is the first token in the input. This means an input sentence is coming, the [SEP] represents the separation between the different inputs. Here, the inputs sentence are tokenized according to BERT vocab, and output is also tokenized.

**4. What is Matthews evaluation?**

**Ans:** Matthews defined VM, known as the Matthews coefficient, as the crystal volume per unit of protein molecular weight, and showed that VM bears a straightforward relationship to the fractional volume of solvent in the crystal.

**5. What is Matthews Correlation Coefficient (MCC)?**

**Ans:** Matthew’s correlation coefficient, also abbreviated as MCC was invented by Brian Matthews in 1975. MCC is a statistical tool used for model evaluation. Its job is to gauge or measure the difference between the predicted values and actual values and is equivalent to chi-square statistics for a 2 x 2 contingency table.

**6. Explain Semantic Role Labeling ?**

**Ans:** In natural language processing, semantic role labeling (also called shallow semantic parsing or slot-filling) is the process that assigns labels to words or phrases in a sentence that indicates their semantic role in the sentence, such as that of an agent, goal, or result.

It serves to find the meaning of the sentence. To do this, it detects the arguments associated with the predicate or verb of a sentence and how they are classified into their specific roles. A common example is the sentence "Mary sold the book to John." The agent is "Mary," the predicate is "sold" (or rather, "to sell,") the theme is "the book," and the recipient is "John." Another example is how "the book belongs to me" would need two labels such as "possessed" and "possessor" and "the book was sold to John" would need two other labels such as theme and recipient, despite these two clauses being similar to "subject" and "object" functions.

A picture containing text, font, screenshot, line

Description automatically generated

**7. Why Fine-tuning a BERT model takes less time than pretraining ?**

**Ans:** During pre-training, the model is trained on unlabeled data over different pre-training tasks. For fine-tuning, the BERT model is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks.

**8. Recognizing Textual Entailment (RTE) ?**

**Ans:** Textual Entailment Recognition has been proposed recently as a generic task that captures major semantic inference needs across many NLP applications, such as Question Answering, Information Retrieval, Information Extraction, and Text Summarization. This task requires to recognize, given two text fragments, whether the meaning of one text is entailed (can be inferred) from the other text.

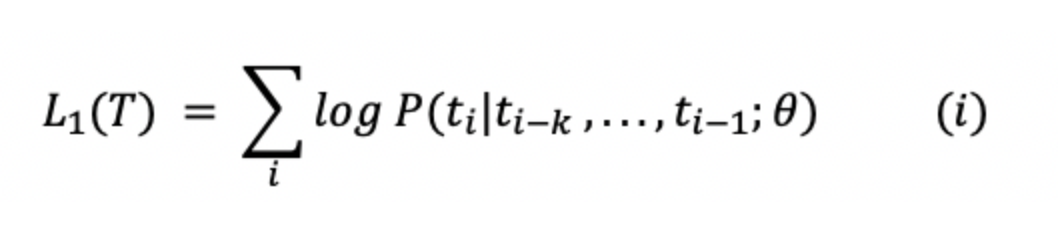
**9. Explain the decoder stack of GPT models ?**

# Ans: GPT-1

Open AI came up with the first iteration of Generative Pre-Training-1 (GPT-1). It was trained on books Corpus data set which has 7000 un-published books. GPT model was based on Transformer architecture. It was made of decoders stacked on top of each other (12 decoders). These models were same as BERT as they were also based on Transformer architecture. The difference in architecture with BERT is that it used stacked encoder layers. GPT model works on a principle called autoregressive which is similar to one used in RNN. It is a technique where the previous output becomes current input.

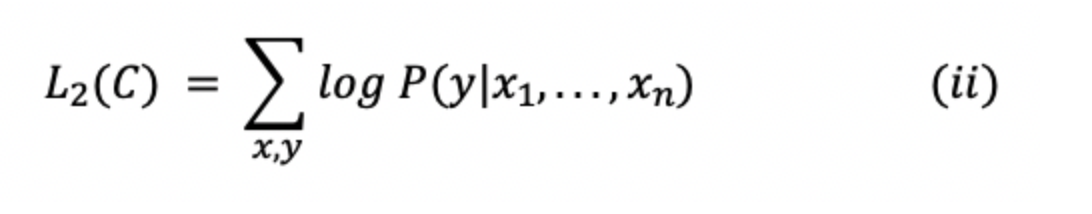
The semi-supervised learning which include first performing unsupervised pre-training and then supervised fine-tuning.

a. *Unsupervised Pre-training*: For unsupervised learning, normal (standard) language model objective was used.

.

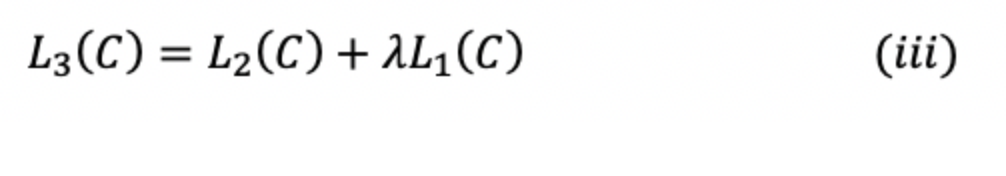
In the equation above, T is tokens in the data, k is contex window size and theta is parameters of the model.

b. *Supervised Fine-Tuning*: This equation maximizes the likelihood of label y (given features or tokens i.e., x1, x2,...,xn).



In the equation above, C is the labeled dataset made of training examples.

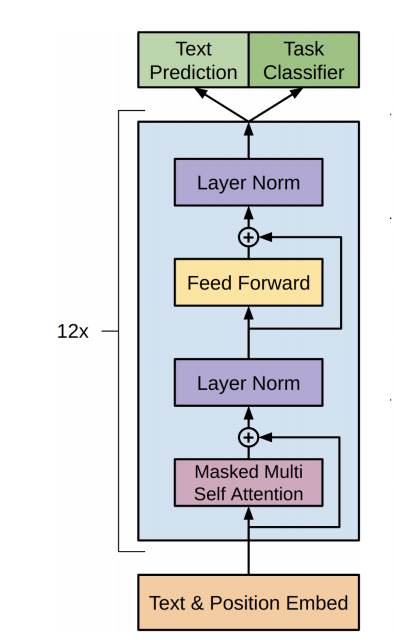
After combining equation 1 and 2, we'll get combined training objective;



where lambda is the weight.

c. *Task Specific Transformations at the input*: In order to make minimal changes to the GPT model during the phase of fine tuning, inputs to the certain tasks were converted to ordered sequences.

1. Tokens such as start and end were added to the input sequence.
2. A token called delimiter was added between parts of the input so that it can be taken as an ordered sequence.



The architecture of the GPT is shown above. It shows that it uses 12 layers of decoder with 12 attention heads in each self-attention layer. It contains masked self attention which is used for training the model. The architecture was very similar to the original transformer architecture. Masking helps where the model doesn't have access to the words to the right side of the current word.