Siddhesh Thakre (D017)

BIDIRECTIONAL ENCODER REPRESENTATIONS FROM TRANSFORMERS (BERT)

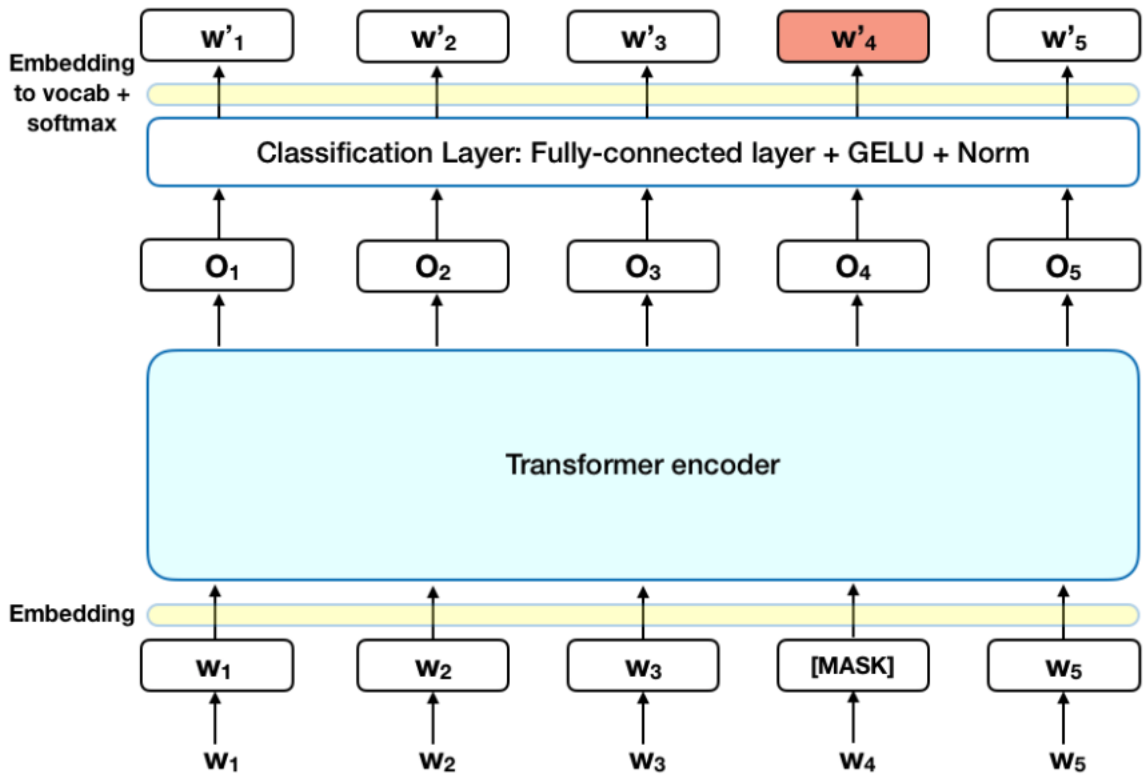
- BERT (Bidirectional Encoder Representations from Transformers) is a recent paper published by researchers at Google AI Language. It has caused a stir in the Machine Learning community by presenting state-of-the-art results in a wide variety of NLP tasks, including Question Answering, Natural Language Inference (MNLI), and others.

- BERT makes use of Transformer, an attention mechanism that learns contextual relations between words (or sub-words) in a text. In its vanilla form, Transformer includes two separate mechanisms — an encoder that reads the text input and a decoder that produces a prediction for the task. Since BERT's goal is to generate a language model, only the encoder mechanism is necessary.

Masked LM (MLM)

Before feeding word sequences into BERT, 15% of the words in each sequence are replaced with a [MASK] token. The model then attempts to predict the original value of the masked words, based on the context provided by the other, non-masked, words in the sequence. In technical terms, the prediction of the output words requires:

- Adding a classification layer on top of the encoder output.
- Multiplying the output vectors by the embedding matrix, transforming them into the vocabulary dimension.
- Calculating the probability of each word in the vocabulary with softmax.



w'1
w'2
w'3
w'4
w'5
Embedding to vocab + softmax
Classification Layer: Fully-connected layer + GELU + Norm
O1
O2
O3
O4
O5
Transformer encoder
Embedding
w1
w2
w3
[MASK]
w5
w1
w2
w3
w4
w5

Code

```
import tensorflow as tf
import tensorflow_hub as hub !pip install tensorflow-text import tensorflow_text as text
```

# Download the BERT preprocessor and encoder for generating the model

```
 bert_preprocess =
hub.KerasLayer("https://tfhub.dev/tensorflow/bert_en_uncased_preprocess/3")

bert_encoder = hub.KerasLayer("https://tfhub.dev/tensorflow/bert_en_uncased_L-12_H-
768_A-12/4")
```

# Bert layers

```
text_input = tf.keras.layers.Input(shape=(), dtype=tf.string, name='text') preprocessed_text
= bert_preprocess(text_input)
outputs = bert_encoder(preprocessed_text)
```

# Neural network layers (binary text classification)

```
l = tf.keras.layers.Dropout(0.1, name="dropout")(outputs['pooled_output']) l =
tf.keras.layers.Dense(1, activation='sigmoid', name="output")(l)
```