BERT

Google BERT is a pre-training method for natural language understanding that performs various NLP tasks better than ever before

BERT works in two steps, First, it uses a large amount of unlabeled data to learn a language representation in an unsupervised fashion called **pre-training**. Then, the pre-trained model can be **fine-tuned** in a supervised fashion using a small amount of labeled trained data to perform various supervised tasks. Pre-training machine learning models have already seen success in various domains including image processing and natural language processing (NLP).

BERT stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. It is based on the transformer architecture (released by Google in 2017). The general transformer uses an encoder and a decoder network, however, as BERT is a pre-training model, it only uses the encoder to learn a latent representation of the input text.

Technology

BERT stacks multiple transformer encoders on top of each other. The transformer is based on the famous multi-head attention module which has shown substantial success in both vision and language tasks.

BERT's state-of-the-art performance is based on two things. First, novel pre-training tasks called ***Masked Language Model(MLM)*** *and* ***Next Sentense Prediction (NSP)****. Second, a lot of data and compute power to train BERT.*

MLM makes it possible to perform bidirectional learning from the text, i.e. it allows the model to learn the context of each word from the words appearing both *before and after it*. This was not possible earlier! The previous state-of-the-art methods called Generative Pre-training used left-to-right training and ELMo used shallow bidirectionality.

The MLM pre-training task converts the text into tokens and uses the token representation as an input and output for the training. A random subset of the tokens (15%) are masked, i.e. hidden during the training, and the objective function is to predict the correct identities of the tokens. This is in contrast to traditional training methodologies which used either unidirectional prediction as the objective or used both left-to-right and right-to-left training to approximate bidirectionality. The NSP task allows BERT to learn relationships between sentences by predicting if the next sentence in a pair is the true next or not. For this 50% correct pairs are supplemented with 50% random pairs and the model trained. BERT trains both MLM and NSP objectives simultaneously.

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)

# Use inputs and outputs to construct a final model

model = tf.keras.Model(inputs=[text_input], outputs = [l])

# Compile and fit

model model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
model.fit(X, y, epochs=epochs)
```