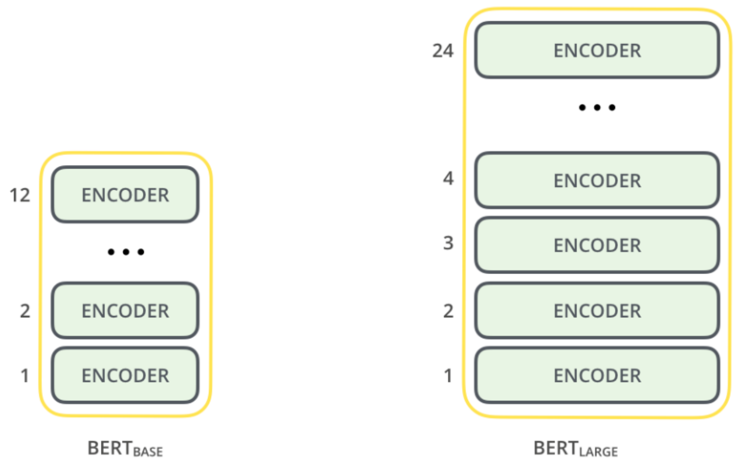
What is BERT?

Bidirectional Encoder Representations from Transformers is abbreviated as BERT. We may infer a lot about what BERT is all about just from the name.

The BERT design is made up of a number of stacked Transformer encoders. A self-attention layer and a feed-forward layer are the two sub-layers that any Transformer encoder contains.

There are two different BERT models:

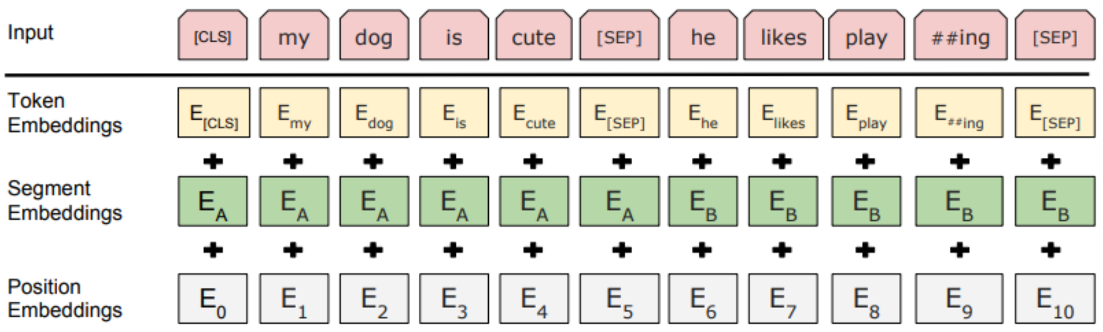
1. BERT base, a BERT model made up of 110M parameters, 768 hidden sizes, 12 attention heads, and 12 layers of Transformer encoders.
2. BERT large, a BERT model with 340 parameters and 24 layers of Transformer encoders, 16 attention heads, and 1024 hidden sizes.



How does it work?

BERT is dependent on a Transformer (the attention mechanism that learns contextual relationships between words in a text). An encoder to read the text input and a decoder to create a prediction for the task make up a basic Transformer. Since the objective of BERT is to produce a language representation model, just the encoder portion is required. A series of tokens that are first transformed into vectors and then processed by the neural network make up the input to the BERT encoder. However, BERT requires the input to be modified and embellished with additional metadata before processing can begin:

1. Token embeddings: A token is added to the input word tokens at the beginning of the first sentence and a token is inserted at the end of each sentence.
2. Segment embeddings: A marker indicating Sentence A or Sentence B is added to each token. This allows the encoder to distinguish between sentences.
3. Positional embeddings: A positional embedding is added to each token to indicate its position in the sentence.


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}

<u>CODE:</u>

```
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_preproc
ess/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)
```