

▾ Imports and Installation

```
1 #transformer models from hugging face open source website
2 !pip install transformers
```



```
Collecting transformers
  Downloading transformers-4.33.1-py3-none-any.whl (7.6 MB)
    7.6/7.6 MB 16.9 MB/s eta 0:00:00
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from transformers) (3.12.2)
Collecting huggingface-hub<1.0,>=0.15.1 (from transformers)
  Downloading huggingface_hub-0.17.1-py3-none-any.whl (294 kB)
    294.8/294.8 kB 24.1 MB/s eta 0:00:00
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (1.23.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from transformers) (23.1)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (6.0.1)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (2023.6.3)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from transformers) (2.31.0)
Collecting tokenizers!=0.11.3,<0.14,>=0.11.1 (from transformers)
  Downloading tokenizers-0.13.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (7.8 MB)
    7.8/7.8 MB 33.6 MB/s eta 0:00:00
Collecting safetensors>=0.3.1 (from transformers)
  Downloading safetensors-0.3.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.3 MB)
    1.3/1.3 MB 25.4 MB/s eta 0:00:00
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-packages (from transformers) (4.66.1)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.15.1->transformers) (2023.12.1)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.15.1->transformers) (4.5.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.2.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2023.7.22)
Installing collected packages: tokenizers, safetensors, huggingface-hub, transformers
Successfully installed huggingface-hub-0.17.1 safetensors-0.3.3 tokenizers-0.13.3 transformers-4.33.1
```

```
1 #import libraries
2 import pandas as pd
3 import numpy as np
4 import tensorflow as tf
5 from tensorflow.keras.layers import Input, Dense
6 from tensorflow.keras.models import Model
7 from transformers import TFAutoModel, AutoTokenizer
8 from sklearn.model_selection import train_test_split
```

▾ Exploring the Dataset

```
1 # Load the data
2 train_url = 'https://raw.githubusercontent.com/google-research-datasets/query-wellformedness/master/train.tsv'
3 test_url = 'https://raw.githubusercontent.com/google-research-datasets/query-wellformedness/master/test.tsv'
4 dev_url = 'https://raw.githubusercontent.com/google-research-datasets/query-wellformedness/master/dev.tsv'
5 #Formatting the data into query and well_formed columns
6 df_train = pd.read_table(train_url, sep='\t',names=['query', 'well_formed'])
7 df_test = pd.read_table(test_url, sep='\t',names=['query', 'well_formed'])
8 df_dev = pd.read_table(dev_url, sep='\t',names=['query', 'well_formed'])
9 #combine datasets into one dataframe
10 df_combine = df_train.append(df_test,ignore_index = True)
11 df = df_combine.append(df_dev,ignore_index = True)
12 df
```

```
<ipython-input-3-4e2683372d69>:10: FutureWarning: The frame.append method is deprecated
df_combine = df_train.append(df_test,ignore_index = True)
<ipython-input-3-4e2683372d69>:11: FutureWarning: The frame.append method is deprecated
df = df_combine.append(df_dev,ignore_index = True)
```

	query	well_formed	
0	The European Union includes how many ?	0.2	
1	What are Mia Hamms accomplishment ?	0.4	
2	Which form of government is still in place in ...	1.0	
3	When was the canal de panama built ?	0.8	

Preprocessing the Data

```
#####
1 # Preprocess the data
2 tokenizer = AutoTokenizer.from_pretrained('bert-base-uncased')
3 max_len = 128
4
5 X = []
6 for query in df['query']:
7     encoded = tokenizer.encode_plus(
8         query,
9         add_special_tokens=True,
10        max_length=max_len,
11        padding='max_length',
12        truncation=True,
13        return_tensors='tf'
14    )
15    X.append(encoded['input_ids'][0])
16 X = np.array(X)
17
18 Y = df['well_formed'].values
```

```
Downloading 28.0/28.0 [00:00<00:00,
(...)okenizer_config.json: 100% 730B/s]
Downloading 570/570 [00:00<00:00,
(...)lve/main/config.json: 100% 19.4kB/s]
Downloading 232k/232k [00:00<00:00
```

Splitting the data

```
1 #splitting into training and validation sets
2 X_train, X_val, Y_train, Y_val = train_test_split(X, Y, test_size=0.2, random_state=42)
```

Transformer model

```
1 #building transformer model with linear activation, mean square error loss and mean absolute error metrics
2 def build_model(transformer, max_len):
3     input_ids = Input(shape=(max_len,), dtype=tf.int32, name="input_ids")
4     sequence_output = transformer(input_ids)[0]
5     cls_token = sequence_output[:, 0, :]
6     out = Dense(1, activation='linear')(cls_token)
7     model = Model(inputs=input_ids, outputs=out)
8     optimizer = tf.keras.optimizers.Adam(learning_rate=1e-5)
9     model.compile(optimizer=optimizer, loss='mse', metrics=['mae'])
10    return model
```

Fine-tuning the model

```
1 #fine-tuning language recognizing model bert-base-uncased
2 transformer_model = TFAutoModel.from_pretrained('bert-base-uncased')
3 model = build_model(transformer_model, max_len)
4
```

```

5 # Train the model
6 batch_size = 32
7 epochs = 10
8
9 model.fit(
10     X_train,
11     Y_train,
12     validation_data=(X_val, Y_val),
13     batch_size=batch_size,
14     epochs=epochs
15 )

```

```

Downloading model.safetensors: 440M/440M [00:02<00:00,
100% 197MB/s]

```

Some weights of the PyTorch model were not used when initializing the TF 2.0 model
- This IS expected if you are initializing TFBertModel from a PyTorch model trained
- This IS NOT expected if you are initializing TFBertModel from a PyTorch model that
All the weights of TFBertModel were initialized from the PyTorch model.

If your task is similar to the task the model of the checkpoint was trained on, you
Epoch 1/10

```

WARNING:tensorflow:Gradients do not exist for variables ['tf_bert_model/bert/pooler
WARNING:tensorflow:Gradients do not exist for variables ['tf_bert_model/bert/pooler
WARNING:tensorflow:Gradients do not exist for variables ['tf_bert_model/bert/pooler
WARNING:tensorflow:Gradients do not exist for variables ['tf_bert_model/bert/pooler
628/628 [=====] - 635s 936ms/step - loss: 0.1164 - mae: 0
Epoch 2/10
628/628 [=====] - 586s 934ms/step - loss: 0.0666 - mae: 0
Epoch 3/10
628/628 [=====] - 586s 933ms/step - loss: 0.0533 - mae: 0
Epoch 4/10
628/628 [=====] - 586s 934ms/step - loss: 0.0443 - mae: 0
Epoch 5/10
628/628 [=====] - 547s 871ms/step - loss: 0.0365 - mae: 0
Epoch 6/10
628/628 [=====] - 548s 872ms/step - loss: 0.0305 - mae: 0
Epoch 7/10
628/628 [=====] - 547s 872ms/step - loss: 0.0254 - mae: 0
Epoch 8/10
628/628 [=====] - 587s 935ms/step - loss: 0.0225 - mae: 0
Epoch 9/10
628/628 [=====] - 587s 934ms/step - loss: 0.0189 - mae: 0
Epoch 10/10
628/628 [=====] - 586s 934ms/step - loss: 0.0168 - mae: 0

```

▼ Testing the results

```

1 queries = ["What were the reasons for everyone to leave the company?", "tell me way city to the", "What is the capital of France?", "what i
2
3 results = []
4
5 for query in queries:
6     encoded = tokenizer.encode_plus(
7         query,
8         add_special_tokens=True,
9         max_length=max_len,
10        padding='max_length',
11        truncation=True,
12        return_tensors='tf'
13    )
14    X_test = encoded['input_ids']
15
16    prediction = model.predict(X_test)[0][0]
17
18    results.append((query, prediction))
19
20 for query, prediction in results:
21     print(f"The predicted well-formedness score for query '{query}' is: {prediction}")
22

```

```

1/1 [=====] - 4s 4s/step
1/1 [=====] - 0s 119ms/step
1/1 [=====] - 0s 80ms/step
1/1 [=====] - 0s 95ms/step
The predicted well-formedness score for query 'What were the reasons for everyone to leave the company?' is: 0.9441385269165039
The predicted well-formedness score for query 'tell me way city to the' is: -0.07047920674085617

```

The predicted well-formedness score for query 'What is the capital of France?' is: 0.9906370639801025
The predicted well-formedness score for query 'what was the reasons for everyone to leave the company' is: 0.5781500339508057

▾ Comparing with another pre-trained model

```
1 #testing another trained model from the forums https://huggingface.co/salesken/query\_wellformedness\_score
2 import torch
3 from transformers import AutoTokenizer, AutoModelForSequenceClassification
4 tokenizer = AutoTokenizer.from_pretrained("salesken/query_wellformedness_score")
5 model = AutoModelForSequenceClassification.from_pretrained("salesken/query_wellformedness_score")
6
```

```
Downloading pytorch_model.bin: 501M/501M [00:28<00:00,
100% 18.6MB/s]
Some weights of the model checkpoint at salesken/query_wellformedness_score were not use
```

```
1 sentences = ["What were the reasons for everyone to leave the company?", "tell me way city to the", "What is the capital of France?", "wha
2
3 features = tokenizer(sentences, padding=True, truncation=True, return_tensors="pt")
4 model.eval()
5 with torch.no_grad():
6     scores = model(**features).logits
7 print(scores)

tensor([[0.9530],
        [0.0323],
        [1.0022],
        [0.2357]])
```

Reference

@InProceedings{FaruquiDas2018, title = {{Identifying Well-formed Natural Language Questions}}, author = {Faruqui, Manaal and Das, Dipanjan},
booktitle = {Proc. of EMNLP}, year = {2018} }