# Natural Language Processing with Transformers

## Detailed Step-by-Step Solution

### 1. Introduction

This assignment involves using a pre-trained BERT model from Hugging Face to compute sentence embeddings, measure cosine similarity between sentence pairs, and predict whether sentences are semantically similar.

### Key Tasks:

Apply BERT for sentence encoding.

Extract token-level embeddings.

Compute cosine similarity between embeddings.

Predict similarity based on a threshold (0.7).

Evaluate accuracy against manual labels.

```
!pip install --upgrade transformers
```

```
Requirement already satisfied: transformers in /usr/local/lib/python3.11/dist
Collecting transformers
  Downloading transformers-4.53.3-py3-none-any.whl.metadata (40 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 40.9/40.9 kB 2.9 MB/s eta 0:00:0
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-pacl
Requirement already satisfied: huggingface-hub<1.0,>=0.30.0 in /usr/local/lib/
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.11/dist-p
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/di
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.11/dist-p
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.11/
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-pacl
Requirement already satisfied: tokenizers<0.22,>=0.21 in /usr/local/lib/python
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.11
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.11/dist-pa
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.11/d
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/py
Requirement already satisfied: hf-xet<2.0.0,>=1.1.2 in /usr/local/lib/python3
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/pyth
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11
Downloading transformers-4.53.3-py3-none-any.whl (10.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 10.8/10.8 MB 99.7 MB/s eta 0:00:00
Installing collected packages: transformers
  Attempting uninstall: transformers
    Found existing installation: transformers 4.53.2
    Uninstalling transformers-4.53.2:
      Successfully uninstalled transformers-4.53.2
Successfully installed transformers-4.53.3
```

```
!pip install tf-keras
```

```
Requirement already satisfied: tf-keras in /usr/local/lib/python3.11/dist-pacl
Requirement already satisfied: tensorflow<2.19,>=2.18 in /usr/local/lib/pythor
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.11/dis
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.11/
Requirement already satisfied: flatbuffers>=24.3.25 in /usr/local/lib/python3
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in /usr/lo
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.1
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.11/d
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.11/
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-pac
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.1
Requirement already satisfied: setuptools in /usr/local/lib/python3.11/dist-pa
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.11/dist-p
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.11/d
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/pytl
Requirement already satisfied: wrapt>=1.11.0 in /usr/local/lib/python3.11/dist
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.1
Requirement already satisfied: tensorboard<2.19,>=2.18 in /usr/local/lib/pythc
Requirement already satisfied: keras>=3.5.0 in /usr/local/lib/python3.11/dist
Requirement already satisfied: numpy<2.1.0,>=1.26.0 in /usr/local/lib/python3
Requirement already satisfied: h5py>=3.11.0 in /usr/local/lib/python3.11/dist
Requirement already satisfied: ml-dtypes<0.5.0,>=0.4.0 in /usr/local/lib/pythc
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/lo
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.11
Requirement already satisfied: rich in /usr/local/lib/python3.11/dist-packages
Requirement already satisfied: namex in /usr/local/lib/python3.11/dist-package
Requirement already satisfied: optree in /usr/local/lib/python3.11/dist-packa
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/pytl
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.11/di
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/l
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.11/di
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.11/
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/pythc
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-pa
```

## 2. Task Completion

### Step 1: Import Libraries & Load Pre-trained BERT

We use transformers for BERT, tensorflow for model execution, and sklearn for cosine similarity.

```
from transformers import BertTokenizer, TFBertModel
import tensorflow as tf
import numpy as np
from sklearn.metrics.pairwise import cosine_similarity
```

```
# Load BERT tokenizer and model
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
bert_model = TFBertModel.from_pretrained('bert-base-uncased')
```

```
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/_auth.py:94: Use
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access pu
  warnings.warn(
tokenizer_config.json: 100%  48.0/48.0 [00:00<00:00, 1.98kB/s]
vocab.txt: 100%  232k/232k [00:00<00:00, 2.43MB/s]
tokenizer.json: 100%  466k/466k [00:00<00:00, 2.82MB/s]
config.json: 100%  570/570 [00:00<00:00, 25.9kB/s]
model.safetensors: 100%  440M/440M [00:11<00:00, 42.8MB/s]
TensorFlow and JAX classes are deprecated and will be removed in Transformers
Some weights of the PyTorch model were not used when initializing the TF 2.0 m
- This IS expected if you are initializing TFBertModel from a PyTorch model t
- This IS NOT expected if you are initializing TFBertModel from a PyTorch mode
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
```

## Step 2: Define Sentence Pairs & Labels

We extend the given 5 sentence pairs with 5 new ones and manually label them (1=similar, 0=not similar).

```
sentence_pairs = [
    ("How do I learn Python?", "What is the best way to study Python?"),
    ("What is AI?", "How to cook pasta?"),
    ("How do I bake a chocolate cake?", "Give me a chocolate cake recipe."),
    ("How can I improve my coding skills?", "Tips for becoming better at programm
    ("Where can I buy cheap laptops?", "Best sites to find affordable computers."
    # New pairs
    ("What is the weather today?", "Is it raining outside?"),
    ("How to train a dog?", "Best ways to teach a puppy tricks."),
    ("What is machine learning?", "How does deep learning work?"),
    ("How to make coffee?", "Steps to prepare tea."),
    ("Best restaurants in town?", "Top places to eat nearby.")
]

labels = [1, 0, 1, 1, 1, 1, 1, 1, 0, 1]  # Manual ground truth
```

## Step 3: Define Function to Get BERT Embeddings

BERT generates contextual embeddings. We extract the [CLS] token embedding for sentence-level representation.

```
def get_sentence_embedding(sentence):
    inputs = tokenizer(sentence, return_tensors='tf', padding=True, truncation=Tr
    outputs = bert_model(inputs)
    cls_embedding = outputs.last_hidden_state[:, 0, :]  # [CLS] token embedding
    return cls_embedding.numpy()
```

## Step 4: Compute Cosine Similarity & Predictions

For each pair, we:

Get embeddings.

Compute cosine similarity.

Predict similarity if score > 0.7.

```
predictions = []
for sent1, sent2 in sentence_pairs:
    emb1 = get_sentence_embedding(sent1)
    emb2 = get_sentence_embedding(sent2)
    sim_score = cosine_similarity(emb1, emb2)[0][0]
    pred = 1 if sim_score > 0.7 else 0
    predictions.append(pred)

    print(f"\nSentence 1: {sent1}")
    print(f"Sentence 2: {sent2}")
    print(f"Cosine Similarity: {sim_score:.4f} → Predicted Similar: {pred}")
```

```
TensorFlow and JAX classes are deprecated and will be removed in Transformers

    Sentence 1: How do I learn Python?
    Sentence 2: What is the best way to study Python?
    Cosine Similarity: 0.9743 → Predicted Similar: 1

    Sentence 1: What is AI?
    Sentence 2: How to cook pasta?
    Cosine Similarity: 0.9033 → Predicted Similar: 1

    Sentence 1: How do I bake a chocolate cake?
    Sentence 2: Give me a chocolate cake recipe.
    Cosine Similarity: 0.8938 → Predicted Similar: 1

    Sentence 1: How can I improve my coding skills?
    Sentence 2: Tips for becoming better at programming.
    Cosine Similarity: 0.8633 → Predicted Similar: 1

    Sentence 1: Where can I buy cheap laptops?
    Sentence 2: Best sites to find affordable computers.
    Cosine Similarity: 0.8750 → Predicted Similar: 1

    Sentence 1: What is the weather today?
```

```
Sentence 2: Is it raining outside?
Cosine Similarity: 0.9476 → Predicted Similar: 1

Sentence 1: How to train a dog?
Sentence 2: Best ways to teach a puppy tricks.
Cosine Similarity: 0.9343 → Predicted Similar: 1

Sentence 1: What is machine learning?
Sentence 2: How does deep learning work?
Cosine Similarity: 0.9707 → Predicted Similar: 1

Sentence 1: How to make coffee?
Sentence 2: Steps to prepare tea.
Cosine Similarity: 0.8696 → Predicted Similar: 1

Sentence 1: Best restaurants in town?
Sentence 2: Top places to eat nearby.
Cosine Similarity: 0.8773 → Predicted Similar: 1
```

## Step 5: Evaluate Accuracy

Compare predictions with ground truth labels.

```
correct = sum(1 for i in range(len(predictions)) if predictions[i] == labels[i])
accuracy = correct / len(labels)
print(f"\nAccuracy: {accuracy:.2%}")
```

```
Accuracy: 80.00%
```