INSTALLING TRANSFORMER

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

INSTALLING SPACY AND SPACY TRANSFORMERS

```
!pip install https://github.com/explosion/spacy-models/releases/download/en_core_web_trf-3.2.
```

```
Collecting en-core-web-trf==3.2.0
  Downloading https://github.com/explosion/spacy-models/releases/download/en_core_we
     |████████████████████████████████| 460.2 MB 30 kB/s
Collecting spacy<3.3.0,>=3.2.0
  Downloading spacy-3.2.4-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
     |████████████████████████████████| 6.0 MB 7.3 MB/s
Collecting spacy-transformers<1.2.0,>=1.1.2
  Downloading spacy_transformers-1.1.5-py2.py3-none-any.whl (51 kB)
     |████████████████████████████████| 51 kB 167 kB/s
Collecting catalogue<2.1.0,>=2.0.6
  Downloading catalogue-2.0.7-py3-none-any.whl (17 kB)
Collecting pathy>=0.3.5
  Downloading pathy-0.6.1-py3-none-any.whl (42 kB)
     |████████████████████████████████| 42 kB 1.4 MB/s
Collecting pydantic!=1.8,!=1.8.1,<1.9.0,>=1.7.4
  Downloading pydantic-1.8.2-cp37-cp37m-manylinux2014_x86_64.whl (10.1 MB)
     |████████████████████████████████| 10.1 MB 57.2 MB/s
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.7/d
Requirement already satisfied: jinja2 in /usr/local/lib/python3.7/dist-packages (fro
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.7/dis
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.7/dist-
Collecting typer<0.5.0,>=0.3.0
  Downloading typer-0.4.1-py3-none-any.whl (27 kB)
Collecting spacy-loggers<2.0.0,>=1.0.0
  Downloading spacy_loggers-1.0.2-py3-none-any.whl (7.2 kB)
Collecting srsly<3.0.0,>=2.4.1
  Downloading srsly-2.4.3-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
     |████████████████████████████████| 457 kB 51.2 MB/s
Collecting thinc<8.1.0,>=8.0.12
  Downloading thinc-8.0.15-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
     |████████████████████████████████| 653 kB 47.3 MB/s
Requirement already satisfied: wasabi<1.1.0,>=0.8.1 in /usr/local/lib/python3.7/dist
Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: numpy>=1.15.0 in /usr/local/lib/python3.7/dist-packag
Collecting typing-extensions<4.0.0.0,>=3.7.4
  Downloading typing_extensions-3.10.0.2-py3-none-any.whl (26 kB)
Collecting langcodes<4.0.0,>=3.2.0
  Downloading langcodes-3.3.0-py3-none-any.whl (181 kB)
     |████████████████████████████████| 181 kB 57.5 MB/s
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.7
Collecting spacy-legacy<3.1.0,>=3.0.8
```

```
  Downloading spacy_legacy-3.0.9-py2.py3-none-any.whl (20 kB)
Requirement already satisfied: click<8.1.0 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: blis<0.8.0,>=0.4.0 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/
Collecting smart-open<6.0.0,>=5.0.0
  Downloading smart_open-5.2.1-py3-none-any.whl (58 kB)
     |████████████████████████████████| 58 kB 6.6 MB/s
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: torch>=1.6.0 in /usr/local/lib/python3.7/dist-package
```

INSTALLING EN_CORE_WEB_SM FOR ENGLISH MODEL PIPLINE OPTIMIZE FOR CPU

```
!python -m spacy download en_core_web_sm
```

```
Collecting en-core-web-sm==3.2.0
  Downloading https://github.com/explosion/spacy-models/releases/download/en_core_web_sm
     |████████████████████████████████| 13.9 MB 5.1 MB/s
Requirement already satisfied: spacy<3.3.0,>=3.2.0 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: srsly<3.0.0,>=2.4.1 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: typer<0.5.0,>=0.3.0 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: typing-extensions<4.0.0.0,>=3.7.4 in /usr/local/lib/pytho
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<1.9.0,>=1.7.4 in /usr/local/lib/py
Requirement already satisfied: pathy>=0.3.5 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/lib/python3.7/d
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.8 in /usr/local/lib/python3.7/di
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: thinc<8.1.0,>=8.0.12 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.7/dis
Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-packages (fro
Requirement already satisfied: numpy>=1.15.0 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: jinja2 in /usr/local/lib/python3.7/dist-packages (from sp
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: wasabi<1.1.0,>=0.8.1 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: blis<0.8.0,>=0.4.0 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: click<8.1.0 in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist
Requirement already satisfied: smart-open<6.0.0,>=5.0.0 in /usr/local/lib/python3.7/dist
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: MarkupSafe>=0.23 in /usr/local/lib/python3.7/dist-package
Installing collected packages: en-core-web-sm
```

```
  Attempting uninstall: en-core-web-sm
    Found existing installation: en-core-web-sm 2.2.5
    Uninstalling en-core-web-sm-2.2.5:
      Successfully uninstalled en-core-web-sm-2.2.5
Successfully installed en-core-web-sm-3.2.0
✓ Download and installation successful
You can now load the package via spacy.load('en_core_web_sm')
```

IMPORT SPACY

```
import spacy
from transformers import *
```

```
# sample text from Wikipedia
text = """Rabindranath Tagore FRAS (Bengali: রবীন্দ্রনাথ ঠাকুর, /rəˈbɪndrənɑːt tæˈɡɔːr/ (listen);

A Bengali Brahmin from Calcutta with ancestral gentry roots in Burdwan district[9] and Jessor
```

USING NER PIPELINE FOR CALLING THE MODEL

```
# load BERT model fine-tuned for Named Entity Recognition (NER)
ner = pipeline("ner", model="dslim/bert-base-NER")
```

```
loading configuration file https://huggingface.co/dslim/bert-base-NER/resolve/main/c
Model config BertConfig {
  "_name_or_path": "dslim/bert-base-NER",
  "_num_labels": 9,
  "architectures": [
    "BertForTokenClassification"
  ],
  "attention_probs_dropout_prob": 0.1,
  "classifier_dropout": null,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "id2label": {
    "0": "O",
    "1": "B-MISC",
    "2": "I-MISC",
    "3": "B-PER",
    "4": "I-PER",
    "5": "B-ORG",
    "6": "I-ORG",
    "7": "B-LOC",
    "8": "I-LOC"
  },
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "label2id": {
    "B-LOC": 7,
    "B-MISC": 1,
    "B-ORG": 5,
    "B-PER": 3,
    "I-LOC": 8,
    "I-MISC": 2,
    "I-ORG": 6,
    "I-PER": 4,
    "O": 0
  },
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "output_past": true,
  "pad_token_id": 0,
  "position_embedding_type": "absolute",
  "transformers_version": "4.17.0",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 28996
}

loading configuration file https://huggingface.co/dslim/bert-base-NER/resolve/main/c
Model config BertConfig {
  "_name_or_path": "dslim/bert-base-NER",
  "_num_labels": 9,
  "architectures": [
    "BertForTokenClassification"
```

EXTRACTING ENTITIES FROM TEXT

```
# perform inference on the transformer model
doc_ner = ner(text)
# print the output
doc_ner
```

```
[{'end': 2,
  'entity': 'B-PER',
  'index': 1,
  'score': 0.99941504,
  'start': 0,
  'word': 'Ra'},
 {'end': 5,
  'entity': 'B-PER',
  'index': 2,
  'score': 0.9878808,
  'start': 2,
  'word': '##bin'},
 {'end': 8,
  'entity': 'I-PER',
  'index': 3,
  'score': 0.8254768,
  'start': 5,
  'word': '##dra'},
 {'end': 12,
  'entity': 'I-PER',
  'index': 4,
  'score': 0.9565929,
  'start': 8,
  'word': '##nath'},
 {'end': 16,
  'entity': 'I-PER',
  'index': 5,
  'score': 0.99897206,
  'start': 13,
  'word': 'Tag'},
 {'end': 19,
  'entity': 'I-PER',
  'index': 6,
  'score': 0.9970107,
  'start': 16,
  'word': '##ore'},
 {'end': 33,
  'entity': 'B-MISC',
  'index': 11,
  'score': 0.99661094,
  'start': 26,
  'word': 'Bengali'},
 {'end': 36,
  'entity': 'B-PER',
  'index': 13,
  'score': 0.9385123,
  'start': 35,
  'word': 'র'},
 {'end': 41,
  'entity': 'I-PER',
  'index': 18,
  'score': 0.49223137,
  'start': 40,
  'word': '##দ'},
 {'end': 129,
  'entity': 'B-MISC',
  'index': 61,
  'score': 0.9996644
```

Next, let's make a function that uses spaCy to visualize this Python dictionary:

```
def get_entities_html(text, ner_result, title=None):
  """Visualize NER with the help of SpaCy"""
  ents = []
  for ent in ner_result:
    e = {}
    # add the start and end positions of the entity
```

```
      e["start"] = ent["start"]
      e["end"] = ent["end"]
      # add the score if you want in the label
      # e["label"] = f"{ent["entity"]}-{ent['score']:.2f}"
      e["label"] = ent["entity"]
      if ents and -1 <= ent["start"] - ents[-1]["end"] <= 1 and ents[-1]["label"] == e["label"]
        # if the current entity is shared with previous entity
        # simply extend the entity end position instead of adding a new one
        ents[-1]["end"] = e["end"]
        continue
      ents.append(e)
    # construct data required for displacy.render() method
    render_data = [
      {
        "text": text,
        "ents": ents,
        "title": title,
      }
    ]
    spacy.displacy.render(render_data, style="ent", manual=True, jupyter=True)
```

```
# get HTML representation of NER of our text
get_entities_html(text, doc_ner)
```

```
 Rabin B-PER dranath Tagore I-PER FRAS ( Bengali B-MISC : র B-PER বীন্ দ I-PER ্রনাথ
May 1861 – 7 August 1941) was a Bengali B-MISC polymath who worked as a poet, writer, playwright, co
painter. He reshaped Bengali B-MISC literature and music as well as Indian B-MISC art with Con B
the late 19th and early 20th centuries. Author of the "profoundly sensitive, fresh and beautiful" poetry of Gita
1913 the first non- European B-MISC and the first lyricist to win the Nobel B-MISC Prize in Literature
poetic songs were viewed as spiritual and mercurial; however, his "elegant prose and magical poetry" remain
. He was a fellow of the Royal B-ORG Asiatic Society I-ORG . Referred to as "the Bard of Bengal I-
known by sobriquets: Guru B-PER de I-PER v, Ko B-PER bi I-LOC gu I-PER ru, Bis B-L

A Bengali B-MISC B I-MISC rahmin from Calcutta B-LOC with ancestral gentry roots in Bur B-L
 B-LOC ore I-LOC , Tag B-PER ore I-PER wrote poetry as an eight-year-old. At the age of sixtee
under the pseudonym B B-LOC hānusiṃha (" Sun B-MISC Lion I-MISC "), which were seized upo
classics. By 1877 he graduated to his first short stories and dramas, published under his real name. As a hun
```

O: Outside of a named entity. B-MIS: Beginning of a miscellaneous entity right after another miscellaneous entity. I-MIS: Miscellaneous entity. B-PER: Beginning of a person’s name right after another person's name. I-PER: Person's name. B-ORG: The beginning of an organization right after another organization. I-ORG: Organization. B-LOC: Beginning of a location right after another location. I-LOC: Location.

INSTALLING ROBERTA A BETTER MODEL TO CHECK

```
# load roberta-large model
ner2 = pipeline("ner", model="xlm-roberta-large-finetuned-conll03-english")
```

```
loading configuration file https://huggingface.co/xlm-roberta-large-finetuned-conll0
Model config XLMRobertaConfig {
  "_name_or_path": "xlm-roberta-large-finetuned-conll03-english",
  "_num_labels": 8,
  "architectures": [
    "XLMRobertaForTokenClassification"
  ],
  "attention_probs_dropout_prob": 0.1,
  "bos_token_id": 0,
  "classifier_dropout": null,
  "eos_token_id": 2,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 1024,
  "id2label": {
    "0": "B-LOC",
    "1": "B-MISC",
    "2": "B-ORG",
    "3": "I-LOC",
    "4": "I-MISC",
    "5": "I-ORG",
    "6": "I-PER",
    "7": "O"
  },
  "initializer_range": 0.02,
  "intermediate_size": 4096,
  "label2id": {
    "B-LOC": 0,
    "B-MISC": 1,
    "B-ORG": 2,
    "I-LOC": 3,
    "I-MISC": 4,
    "I-ORG": 5,
    "I-PER": 6,
    "O": 7
  },
  "layer_norm_eps": 1e-05,
  "max_position_embeddings": 514,
  "model_type": "xlm-roberta",
  "num_attention_heads": 16,
  "num_hidden_layers": 24,
  "output_past": true,
  "pad_token_id": 1,
  "position_embedding_type": "absolute",
  "transformers_version": "4.17.0",
  "type_vocab_size": 1,
  "use_cache": true,
  "vocab_size": 250002
}

loading configuration file https://huggingface.co/xlm-roberta-large-finetuned-conll0
Model config XLMRobertaConfig {
  "_name_or_path": "xlm-roberta-large-finetuned-conll03-english",
  "_num_labels": 8,
  "architectures": [
    "XLMRobertaForTokenClassification"
```

```
# perform inference on this model
doc_ner2 = ner2(text)


# get HTML representation of NER of our text
get_entities_html(text, doc_ner2)
```

Rabindranath Tagore FRAS **I-PER** (Bengali **I-MISC** : রবীন্দ্রনাথ ঠাকুর **I-PER** , / rə **I-PER** ˈbɪndr

1861 – 7 August 1941) was a Bengali **I-MISC** polymath who worked as a poet, writer, playwright, compos

He reshaped Bengali **I-MISC** literature and music as well as Indian **I-MISC** art with Contextual Mode

20th centuries. Author of the "profoundly sensitive, fresh and beautiful" poetry of Gitanjali **I-PER** , he beca

MISC and the first lyricist to win the Nobel Prize in Literature **I-MISC** . Tagore **I-PER** 's poetic songs w

however, his "elegant prose and magical poetry" remain largely unknown outside Bengal **I-LOC** . He was

ORG . Referred to as "the Bard **I-PER** of Bengal **I-LOC** ", Tagore **I-PER** was known by sobriquets

, Biswakobi **I-PER** .[a]

A Bengali Brah **I-MISC** min from Calcutta **I-LOC** with ancestral gentry roots in Burdwan **I-LOC** dis

PER wrote poetry as an eight-year-old. At the age of sixteen, he released his first substantial poems under t

Sun Lion **I-PER** "), which were seized upon by literary authorities as long-lost classics. By 1877 he gradu

As you can see, now it's improved, naming Rabindranath Tagore as a single entity and also the district Jessore.

```
#There are a lot of other models that were fine-tuned on the same dataset. Here's yet another
```

```
# load yet another roberta-large model
ner3 = pipeline("ner", model="Jean-Baptiste/roberta-large-ner-english")
# perform inference on this model
doc_ner3 = ner3(text)
# get HTML representation of NER of our text
get_entities_html(text, doc_ner3)
```

```
loading configuration file https://huggingface.co/Jean-Baptiste/roberta-large-ner-engli
Model config RobertaConfig {
  "_name_or_path": "Jean-Baptiste/roberta-large-ner-english",
  "architectures": [
    "RobertaForTokenClassification"
  ],
  "attention_probs_dropout_prob": 0.1,
  "bos_token_id": 0,
  "classifier_dropout": null,
  "eos token id": 2.
```

This model, however, only has PER, MISC, LOC, and ORG entities. SpaCy automatically colors the familiar entities.

To perform NER using SpaCy, we must first load the model using spacy.load() function:

```
# load the English CPU-optimized pipeline
nlp = spacy.load("en_core_web_sm")

#We're loading the model we've downloaded. Make sure you download the model you want to use b

# predict the entities
doc = nlp(text)

        "O": 0,

# display the doc with jupyter mode
spacy.displacy.render(doc, style="ent", jupyter=True)
```

```
  "hidden_dropout_prob": 0.1,
```

This one looks much better, and there are a lot more entities (18) than the previous ones, namely CARDINAL, DATE, EVENT, FAC, GPE, LANGUAGE, LAW, LOC, MONEY, NORP, ORDINAL, ORG, PERCENT, PERSON, PRODUCT, QUANTITY, TIME, WORK_OF_ART

Author of the "profoundly sensitive, fresh and beautiful" poetry of Gitanjali PERSON he became in 191

```
#However, Calcutta was mistakenly labeled as an product, so let's use the Transformer model t

# load the English transformer pipeline (roberta-base) using spaCy
nlp_trf = spacy.load('en_core_web_trf')
```

```
loading configuration file /tmp/tmpmempe9u1/config.json
Model config RobertaConfig {
  "_name_or_path": "/tmp/tmpmempe9u1/config.json",
  "architectures": [
    "RobertaForMaskedLM"
  ],
  "attention_probs_dropout_prob": 0.1,
  "bos_token_id": 0,
  "classifier_dropout": null,
  "eos_token_id": 2,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "layer_norm_eps": 1e-05,
  "max_position_embeddings": 514,
  "model_type": "roberta",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "pad_token_id": 1,
  "position_embedding_type": "absolute",
  "transformers_version": "4.17.0",
  "type_vocab_size": 1,
  "use_cache": true,
  "vocab_size": 50265
}

Didn't find file /tmp/tmpmempe9u1/added_tokens.json. We won't load it.
loading file /tmp/tmpmempe9u1/vocab.json
loading file /tmp/tmpmempe9u1/merges.txt
loading file /tmp/tmpmempe9u1/tokenizer.json
loading file None
loading file /tmp/tmpmempe9u1/special_tokens_map.json
loading file /tmp/tmpmempe9u1/tokenizer_config.json
```

```
#Let's perform inference and visualize the text:

# perform inference on the model
doc_trf = nlp_trf(text)
```

```
# display the doc with jupyter mode
spacy.displacy.render(doc_trf, style="ent", jupyter=True)
```

```
/usr/local/lib/python3.7/dist-packages/torch/autocast_mode.py:162: UserWarning: User pro
  warnings.warn('User provided device_type of \'cuda\', but CUDA is not available. Disab
```

Rabindranath Tagore **PERSON** FRAS (Bengali **LANGUAGE** : রবীন্দ্রনাথ ঠাকুর, /rəˈbɪndrənɑːt tæˈɡɔːr/ (l

1941) was a Bengali **NORP** polymath who worked as a poet, writer, playwright, composer, philosopher, sc

Bengali **NORP** literature and music as well as Indian **NORP** art with Contextual Modernism in the la

Author of the "profoundly sensitive, fresh and beautiful" poetry of Gitanjali **PERSON** , he became in 191

European **NORP** and the first **ORDINAL** lyricist to win the Nobel Prize in Literature **WORK_OF_ART** .

viewed as spiritual and mercurial; however, his "elegant prose and magical poetry" remain largely unknown o

the Royal Asiatic Society **ORG** . Referred to as " the Bard of Bengal **PERSON** ", Tagore **PERSON**

PERSON , Kobiguru **PERSON** , Biswakobi.[a **PERSON**]

A Bengali Brahmin **NORP** from Calcutta **GPE** with ancestral gentry roots in Burdwan **GPE** district

PERSON wrote poetry as an eight-year-old **DATE** . At the age of sixteen **DATE** , he released his fi

pseudonym Bhānusiṃha **PERSON** ("Sun Lion"), which were seized upon by literary authorities as long-los

TRANSFORMERS

```
!pip install transformers
!pip install torch
```

```
Requirement already satisfied: transformers in /usr/local/lib/python3.7/dist-packages (4
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: sacremoses in /usr/local/lib/python3.7/dist-packages (fro
Requirement already satisfied: tokenizers!=0.11.3,<0.13,>=0.11.1 in /usr/local/lib/pytho
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.7/dist-packages (fro
Requirement already satisfied: filelock in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: huggingface-hub<1.0,>=0.1.0 in /usr/local/lib/python3.7/d
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.7/di
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from sacre
Requirement already satisfied: click in /usr/local/lib/python3.7/dist-packages (from sac
Requirement already satisfied: joblib in /usr/local/lib/python3.7/dist-packages (from sa
Requirement already satisfied: torch in /usr/local/lib/python3.7/dist-packages (1.11.0+c
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packag
```

```
from transformers import BertForQuestionAnswering
from transformers import BertTokenizer
import torch
import numpy as np
```

```
#Step 3: Load pre-trained Bert model
model = BertForQuestionAnswering.from_pretrained('bert-large-uncased-whole-word-masking-finet

tokenizer_for_bert = BertTokenizer.from_pretrained('bert-large-uncased-whole-word-masking-fin
```

```
loading configuration file https://huggingface.co/bert-large-uncased-whole-word-mask
Model config BertConfig {
  "architectures": [
    "BertForQuestionAnswering"
  ],
  "attention_probs_dropout_prob": 0.1,
  "classifier_dropout": null,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 1024,
  "initializer_range": 0.02,
  "intermediate_size": 4096,
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 16,
  "num_hidden_layers": 24,
  "pad_token_id": 0,
  "position_embedding_type": "absolute",
  "transformers_version": "4.17.0",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 30522
}

loading weights file https://huggingface.co/bert-large-uncased-whole-word-masking-fi
All model checkpoint weights were used when initializing BertForQuestionAnswering.

All the weights of BertForQuestionAnswering were initialized from the model checkpoi
If your task is similar to the task the model of the checkpoint was trained on, you
loading file https://huggingface.co/bert-large-uncased-whole-word-masking-finetuned-
loading file https://huggingface.co/bert-large-uncased-whole-word-masking-finetuned-
loading file https://huggingface.co/bert-large-uncased-whole-word-masking-finetuned-
loading file https://huggingface.co/bert-large-uncased-whole-word-masking-finetuned-
loading configuration file https://huggingface.co/bert-large-uncased-whole-word-mask
```

```
Model config BertConfig {
  "_name_or_path": "bert-large-uncased-whole-word-masking-finetuned-squad",
  "architectures": [
    "BertForQuestionAnswering"
  ],
  "attention_probs_dropout_prob": 0.1,
  "classifier_dropout": null,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 1024,
  "initializer_range": 0.02,
  "intermediate_size": 4096,
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 16,
  "num_hidden_layers": 24,
  "pad_token_id": 0,
  "position_embedding_type": "absolute",
  "transformers_version": "4.17.0",
  "type_vocab_size": 2,
```

```
def bert_question_answer(question, passage, max_len=500):

    """
    question: What is the name of YouTube Channel
    passage: Watch complete playlist of Natural Language Processing. Don't forget to like, sh
    """

    #Tokenize input question and passage
    #Add special tokens - [CLS] and [SEP]
    input_ids = tokenizer_for_bert.encode (question, passage,  max_length= max_len, truncatio
    """
    [101, 2054, 2003, 1996, 2171, 1997, 7858, 3149, 102, 3422, 3143, 2377, 9863, 1997, 3019,
    2123, 1005, 1056, 5293, 2000, 2066, 1010, 3745, 1998, 4942, 29234, 2026, 3149, 1045, 2290
    """

    #Getting number of tokens in 1st sentence (question) and 2nd sentence (passage that conta
    sep_index = input_ids.index(102)
    len_question = sep_index + 1
    len_passage = len(input_ids)- len_question
    """
    8
    9
    27
    """
    #Need to separate question and passage
    #Segment ids will be 0 for question and 1 for passage
    segment_ids =  [0]*len_question + [1]*(len_passage)
    """
    [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1
    """

    #Converting token ids to tokens
    tokens = tokenizer_for_bert.convert_ids_to_tokens(input_ids)
    """
    tokens = ['[CLS]', 'what', 'is', 'the', 'name', 'of', 'youtube', 'channel', '[SEP]', 'wat
    'play', '##list', 'of', 'natural', 'language', 'processing', '.', 'don', "'", 't', 'forge
    ',', 'share', 'and', 'sub', '##scribe', 'my', 'channel', 'i', '##g', 'tech', 'team', '[SE
    """

    #Getting start and end scores for answer
    #Converting input arrays to torch tensors before passing to the model
    start_token_scores = model(torch.tensor([input_ids]), token_type_ids=torch.tensor([segmen
    end_token_scores = model(torch.tensor([input_ids]), token_type_ids=torch.tensor([segment_
    """
    tensor([[-5.9787, -3.0541, -7.7166, -5.9291, -6.8790, -7.2380, -1.8289, -8.1006,
         -5.9786, -3.9319, -5.6230, -4.1919, -7.2068, -6.7739, -2.3960, -5.9425,
         -5.6828, -8.7007, -4.2650, -8.0987, -8.0837, -7.1799, -7.7863, -5.1605,
         -8.2832, -5.1088, -8.1051, -5.3985, -6.7129, -1.4109, -3.2241,  1.5863,
         -4.9714, -4.1138, -5.9107, -5.9786]], grad_fn=<SqueezeBackward1>)
    tensor([[-2.1025, -2.9121, -5.9192, -6.7459, -6.4667, -5.6418, -1.4504, -3.1943,
         -2.1024, -5.7470, -6.3381, -5.8520, -3.4871, -6.7667, -5.4711, -3.9885,
         -1.2502, -4.0869, -6.4930, -6.3751, -6.1309, -6.9721, -7.5558, -6.4056,
         -6.7456, -5.0527, -7.3854, -7.0440, -4.3720, -3.8936, -2.1085, -5.8211,
         -2.0906, -2.2184,  1.4268, -2.1026]], grad_fn=<SqueezeBackward1>)
    """

    #Converting scores tensors to numpy arrays
    start_token_scores = start_token_scores.detach().numpy().flatten()
    end_token_scores = end_token_scores.detach().numpy().flatten()
    """
    [-5.978666  -3.0541189 -7.7166095 -5.929051  -6.878973  -7.238004
    -1.8289301 -8.10058   -5.9786286 -3.9319289 -5.6229596 -4.191908
    -7.20684   -6.773916  -2.3959794 -5.942456  -5.6827617 -8.700695
    -4.265001  -8.09874   -8.083673  -7.179875  -7.7863474 -5.16046
    -8.283156  -5.108819  -8.1051235 -5.3984528 -6.7128663 -1.4108785
    -3.2240815  1.5863497 -4.9714    -4.113782  -5.9107194 -5.9786243]

    [-2.1025064 -2.912148  -5.9192414 -6.745929  -6.466673  -5.641759
    -1.4504088 -3.1943028 -2.1024144 -5.747039  -6.3380575 -5.852047
    -3.487066  -6.7667046 -5.471078  -3.9884708 -1.2501552 -4.0868535
    -6.4929943 -6.375147  -6.130891  -6.972091  -7.5557766 -6.405638
    -6.7455807 -5.0527067 -7.3854156 -7.043977  -4.37199   -3.8935976
    -2.1084964 -5.8210607 -2.0906193 -2.2184045  1.4268283 -2.1025767]
    """
    #Getting start and end index of answer based on highest scores
    answer_start_index = np.argmax(start_token_scores)
    answer_end_index = np.argmax(end_token_scores)
    """
    31
    34
    """

    #Getting scores for start and end token of the answer
    start_token_score = np.round(start_token_scores[answer_start_index], 2)
    end_token_score = np.round(end_token_scores[answer_end_index], 2)
    """
    1.59
    1.43
    """

    #Combining subwords starting with ## and get full words in output.
    #It is because tokenizer breaks words which are not in its vocab.
    answer = tokens[answer_start_index]
    for i in range(answer_start_index + 1, answer_end_index + 1):
        if tokens[i][0:2] == '##':
            answer += tokens[i][2:]
        else:
            answer += ' ' + tokens[i]
# If the answer didn't find in the passage
    if ( answer_start_index == 0) or (start_token_score < 0 ) or  (answer == '[SEP]') or ( an
        answer = "Sorry!, I could not find an answer in the passage."

    return (answer_start_index, answer_end_index, start_token_score, end_token_score,  answer

#Testing function
bert_question_answer("What is the name of YouTube Channel", "Watch complete playlist of Natur
```

```
 (31, 34, 1.59, 1.43, 'ig tech team')
```

```
def bert_question_answer(question, passage, max_len=500):

    """
    question: What is the name of YouTube Channel
    passage: Watch complete playlist of Ishika Nisha. Don't forget to like, share and subscri
    """
    #Tokenize input question and passage
    #Add special tokens - [CLS] and [SEP]
    input_ids = tokenizer_for_bert.encode (question, passage,  max_length= max_len, truncatio
    """
    [101, 2054, 2003, 1996, 2171, 1997, 7858, 3149, 102, 3422, 3143, 2377, 9863, 1997, 3019,
    2123, 1005, 1056, 5293, 2000, 2066, 1010, 3745, 1998, 4942, 29234, 2026, 3149, 1045, 2290
    """
    #Getting number of tokens in 1st sentence (question) and 2nd sentence (passage that conta
    sep_index = input_ids.index(102)
    len_question = sep_index + 1
    len_passage = len(input_ids)- len_question
    """
    8
```

```
93
27
"""
#Need to separate question and passage
#Segment ids will be 0 for question and 1 for passage
segment_ids =  [0]*len_question + [1]*(len_passage)
"""
[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1
"""
#Converting token ids to tokens
tokens = tokenizer_for_bert.convert_ids_to_tokens(input_ids)
"""
tokens = ['[CLS]', 'what', 'is', 'the', 'name', 'of', 'youtube', 'channel', '[SEP]', 'wat
'play', '##list', 'of', 'ishika', 'nisha', '.', 'don', "'", 't', 'forget', 'to', 'like',
',', 'share', 'and', 'sub', '##scribe', 'my', 'channel', 'i', '##n', 'tech', 'team', '[SE
"""
#Getting start and end scores for answer
#Converting input arrays to torch tensors before passing to the model
start_token_scores = model(torch.tensor([input_ids]), token_type_ids=torch.tensor([segmen
end_token_scores = model(torch.tensor([input_ids]), token_type_ids=torch.tensor([segment_
"""
tensor([[-5.9787, -3.0541, -7.7166, -5.9291, -6.8790, -7.2380, -1.8289, -8.1006,
     -5.9786, -3.9319, -5.6230, -4.1919, -7.2068, -6.7739, -2.3960, -5.9425,
     -5.6828, -8.7007, -4.2650, -8.0987, -8.0837, -7.1799, -7.7863, -5.1605,
     -8.2832, -5.1088, -8.1051, -5.3985, -6.7129, -1.4109, -3.2241,  1.5863,
     -4.9714, -4.1138, -5.9107, -5.9786]], grad_fn=<SqueezeBackward1>)
tensor([[-2.1025, -2.9121, -5.9192, -6.7459, -6.4667, -5.6418, -1.4504, -3.1943,
     -2.1024, -5.7470, -6.3381, -5.8520, -3.4871, -6.7667, -5.4711, -3.9885,
     -1.2502, -4.0869, -6.4930, -6.3751, -6.1309, -6.9721, -7.5558, -6.4056,
     -6.7456, -5.0527, -7.3854, -7.0440, -4.3720, -3.8936, -2.1085, -5.8211,
     -2.0906, -2.2184,  1.4268, -2.1026]], grad_fn=<SqueezeBackward1>)
"""
#Converting scores tensors to numpy arrays
start_token_scores = start_token_scores.detach().numpy().flatten()
end_token_scores = end_token_scores.detach().numpy().flatten()
"""
[-5.978666  -3.0541189 -7.7166095 -5.929051  -6.878973  -7.238004
-1.8289301 -8.10058   -5.9786286 -3.9319289 -5.6229596 -4.191908
-7.20684   -6.773916  -2.3959794 -5.942456  -5.6827617 -8.700695
-4.265001  -8.09874   -8.083673  -7.179875  -7.7863474 -5.16046
-8.283156  -5.108819  -8.1051235 -5.3984528 -6.7128663 -1.4108785
-3.2240815  1.5863497 -4.9714    -4.113782  -5.9107194 -5.9786243]

[-2.1025064 -2.912148  -5.9192414 -6.745929  -6.466673  -5.641759
-1.4504088 -3.1943028 -2.1024144 -5.747039  -6.3380575 -5.852047
-3.487066  -6.7667046 -5.471078  -3.9884708 -1.2501552 -4.0868535
-6.4929943 -6.375147  -6.130891  -6.972091  -7.5557766 -6.405638
-6.7455807 -5.0527067 -7.3854156 -7.043977  -4.37199   -3.8935976
-2.1084964 -5.8210607 -2.0906193 -2.2184045  1.4268283 -2.1025767]
"""
#Getting start and end index of answer based on highest scores
```

```
    answer_start_index = np.argmax(start_token_scores)
    answer_end_index = np.argmax(end_token_scores)
    """
    31
    34
    """

    #Getting scores for start and end token of the answer
    start_token_score = np.round(start_token_scores[answer_start_index], 2)
    end_token_score = np.round(end_token_scores[answer_end_index], 2)
    """
    1.59
    1.43
    """

    #Combining subwords starting with ## and get full words in output.
    #It is because tokenizer breaks words which are not in its vocab.
    answer = tokens[answer_start_index]
    for i in range(answer_start_index + 1, answer_end_index + 1):
        if tokens[i][0:2] == '##':
            answer += tokens[i][2:]
        else:
            answer += ' ' + tokens[i]
    # If the answer didn't find in the passage
    if ( answer_start_index == 0) or (start_token_score < 0 ) or  (answer == '[SEP]') or ( an
        answer = "Sorry!, I could not find an answer in the passage."

    return (answer_start_index, answer_end_index, start_token_score, end_token_score,  answer

    #Testing function
    bert_question_answer("What is the name of YouTube Channel", "Watch complete playlist of I
```

(31, 34, 1.59, 1.43, 'in tech team')

```
!pip install torch
```

```
  Requirement already satisfied: torch in /usr/local/lib/python3.7/dist-packages (1.11.0+
  Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packag
```

```
# Let me define another passage
passage= """Rabindranath Tagore FRAS (Bengali: রবীন্দ্রনাথ ঠাকুর, /rəˈbɪndrənɑːt tæˈɡɔːr/ (listen)

A Bengali Brahmin from Calcutta with ancestral gentry roots in Burdwan district[9] and Jessor

print (f'Length of the passage: {len(passage.split())} words')

question ="Who is Rabindranath Tagore"
```

```
print ('\nQuestion 1:\n', question)
_, _ , _ , _, ans  = bert_question_answer( question, passage)
print('\nAnswer from BERT: ', ans ,  '\n')



question ="When was Rabindranath Tagore born"
print ('\nQuestion 7:\n', question)
_, _ , _ , _, ans  = bert_question_answer( question, passage)
print('\nAnswer from BERT: ', ans ,  '\n')
```

```
     Length of the passage: 246 words

     Question 1:
      Who is Rabindranath Tagore

     Answer from BERT:  a bengali polymath


     Question 7:
      When was Rabindranath Tagore born

     Answer from BERT:  7 may 1861
```

```
# Let me define one passage
passage = """Hello, I am Ishika. My friend name is Sakshi. He is the son of Pradip. I spend m
He always call me by my nick name. Sakshi call me programmer. Except Sakshi, my other friend
Amrita is also my friend. """

print (f'Length of the passage: {len(passage.split())} words')

question1 ="What is my name"
print ('\nQuestion 1:\n', question1)
_, _ , _ , _, ans  = bert_question_answer( question1, passage)
print('\nAnswer from BERT: ', ans ,  '\n')


question2 ="Who is the father of Sakshi"
print ('\nQuestion 2:\n', question2)
_, _ , _ , _, ans  = bert_question_answer( question2, passage)
print('\nAnswer from BERT: ', ans ,  '\n')

question3 ="With whom Ishika spend most of the time"
print ('\nQuestion 3:\n', question3)
_, _ , _ , _, ans  = bert_question_answer( question3, passage)
print('\nAnswer from BERT: ', ans ,  '\n')
```

```
     Length of the passage: 51 words

     Question 1:
```

```
 What is my name

Answer from BERT:  ishika


Question 2:
 Who is the father of Sakshi

Answer from BERT:  pradip


Question 3:
 With whom Ishika spend most of the time

Answer from BERT:  sakshi
```

```
#@title Question-Answering Application { vertical-output: true }
#@markdown ---
question= "name of the sons of Rabindranath Tagore" #@param {type:"string"}
3
passage = """Rabindranath Tagore FRAS (Bengali: রবীন্দ্রনাথ ঠাকুর, /rəˈbɪndrənɑːt tæˈɡɔːr/ (lister

A Bengali Brahmin from Calcutta with ancestral gentry roots in Burdwan district[9] and Jessor

#@markdown ---
_, _ , _ , _, ans  = bert_question_answer( question, passage)

#@markdown Answer:
print(ans)
```

Question-Answering Application

question: " name of the sons of Rabi "

Answer:

```
Sorry!, I could not find an answer in th
```

```
#@title Question-Answering Application { vertical-output: true }
#@markdown ---
question= "who is Albert Einstein" #@param {type:"string"}
3
passage = """Albert Einstein was a German-born theoretical physicist, widely acknowledged to
#@markdown ---
_, _ , _ , _, ans  = bert_question_answer( question, passage)

#@markdown Answer:
print(ans)
```

Question-Answering Application

question: " who is Albert Einstein "

Answer:

```
albert einstein was a german - born theo
```