Project

NLP with Deep Learning
CSSE15

BACHELOR OF TECHNOLOGY
ELECTRONICS & COMMUNICATION ENGINEERING

Submitted by:
NAME: SARTHAK VISHNU
Roll No.: 20240

SCHOOL OF ELECTRONICS

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, UNA
HIMACHAL PRADESH
2020-24

1. Aim:

Customizing a Pre-trained BERT Model for Question-Answering Task.

Model used: distilbert-base-uncased-finetuned-squad

2. GitHub Repository Link:

https://github.com/Sarthak-Vishnu/20240-CSSE15-NLP-with-Deep-Learning.git

3. Theory:

Transformers represent a breakthrough in natural language processing (NLP) and machine learning. Introduced in the seminal paper "Attention is All You Need" by Vaswani et al. in 2017, Transformers have revolutionized various NLP tasks, including translation, summarization, and question-answering.

Unlike traditional sequence-to-sequence models, Transformers rely solely on self-attention mechanisms to weigh the importance of different words in a sequence, enabling parallel processing of input tokens and capturing long-range dependencies more effectively. This architecture not only improves computational efficiency but also enhances model performance.

Transformers have become the cornerstone of many state-of-the-art NLP models, including BERT (Bidirectional Encoder Representations from Transformers), GPT (Generative Pre-trained Transformer), and T5 (Text-to-Text Transfer Transformer). Their versatility and scalability have made them indispensable tools for researchers and practitioners in the field of machine learning.

In this report, we delve into the customization of a pre-trained BERT model for a specific question-answering task.

4. Methodology:

It's Smart-Question Answering System on short as well as long texts. It can automatically find answers to matching questions directly from context provided. The deep learning language model converts the questions and documents to semantic vectors to find the matching answer. Then converts back the logits obtained from the model to tokens followed by converting them to natural language texts for human interpretation.

5. Code and Implementation:

In [81]:
```
from transformers import AutoTokenizer, TFAutoModelForQuestionAnswering
```

In [82]:
```
import numpy as np
```

In [ ]:
```
# !pip install tf-keras
```

In [83]:
```
checkpoint = "Rocketknight1/distilbert-base-uncased-finetuned-squad"
model = TFAutoModelForQuestionAnswering.from_pretrained(checkpoint)
tokenizer = AutoTokenizer.from_pretrained(checkpoint)
```

```
Some layers from the model checkpoint at Rocketknight1/distilbert-base-uncased-finetuned-squad were not used
when initializing TFDistilBertForQuestionAnswering: ['dropout_19']
- This IS expected if you are initializing TFDistilBertForQuestionAnswering from the checkpoint of a model tr
ained on another task or with another architecture (e.g. initializing a BertForSequenceClassification model f
rom a BertForPreTraining model).
- This IS NOT expected if you are initializing TFDistilBertForQuestionAnswering from the checkpoint of a mode
l that you expect to be exactly identical (initializing a BertForSequenceClassification model from a BertForS
equenceClassification model).
Some layers of TFDistilBertForQuestionAnswering were not initialized from the model checkpoint at Rocketknigh
t1/distilbert-base-uncased-finetuned-squad and are newly initialized: ['dropout_59']
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inferenc
e.
```

Testing Que-Ans abilities

Example 01

In [84]:
```
context = """The dominant sequence transduction models are based on complex recurrent or convolutional
neural networks in an encoder-decoder configuration. The best performing models also connect the encoder
and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer,
based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on
two machine translation tasks show these models to be superior in quality while being more parallelizable
and requiring significantly less time to train."""

question = "What kind of mechanisms is Transformer based on?"

inputs = tokenizer([context], [question], return_tensors="np")

outputs = model(inputs)

start_position = np.argmax(outputs.start_logits[0])
end_position = np.argmax(outputs.end_logits[0])
print(start_position)
print(end_position)

# Extract this substring from the inputs
answer = inputs["input_ids"][0, start_position: end_position + 1]
print(answer)
print("\n")
print("Que:", question)
print("Ans:", tokenizer.decode(answer))
```

```
64
65
[ 3086 10595]


Que: What kind of mechanisms is Transformer based on?
Ans: attention mechanisms
```

Example 02

In [85]:
```
context = """In a small cafe on a busy street, the smell of fresh coffee fills the air.
People chat and laugh while soft music plays in the background. The baristas
```

```
make each coffee with care, and everyone enjoys their time here, whether chatting
with friends or enjoying a good book. It's a calm place in the middle of a
busy city, where people can relax and enjoy simple moments."""

question = "Where is this Cafe?"

inputs = tokenizer([context], [question], return_tensors="np")

outputs = model(inputs)

start_position = np.argmax(outputs.start_logits[0])
end_position = np.argmax(outputs.end_logits[0])
print(start_position)
print(end_position)

# Extract this substring from the inputs
answer = inputs["input_ids"][0, start_position: end_position + 1]
print(answer)
print("\n")
print("Que:", question)
print("Ans:", tokenizer.decode(answer))
```

```
58
70
[2009 1005 1055 1037 5475 2173 1999 1996 2690 1997 1037 5697 2103]


Que: Where is this Cafe?
Ans: it's a calm place in the middle of a busy city
```

In [ ]: