Bhuvi Ghosh
60009210191

ACL Experiment-5

```
!pip install torch transformers
```

```
Requirement already satisfied: torch in /usr/local/lib/python3.10/dist-packages (2.4.1+cu121)
Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-packages (4.44.2)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from torch) (3.16.1)
Requirement already satisfied: typing-extensions>=4.8.0 in /usr/local/lib/python3.10/dist-packages (from torch) (4.12.2)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages (from torch) (1.13.3)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch) (3.3)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch) (3.1.4)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from torch) (2024.6.1)
Requirement already satisfied: huggingface-hub<1.0,>=0.23.2 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.24.7)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (1.26.4)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from transformers) (24.1)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (2024.9.11)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from transformers) (2.32.3)
Requirement already satisfied: safetensors>=0.4.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.4.5)
Requirement already satisfied: tokenizers<0.20,>=0.19 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.19.1)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-packages (from transformers) (4.66.5)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch) (2.1.5)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2024.8.30)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy->torch) (1.3.0)
```

```
[2] from transformers import BertModel, BertTokenizer, BertConfig
    import torch
    import torch.nn as nn
```

```
[3] model_name = 'bert-base-uncased'
    tokenizer = BertTokenizer.from_pretrained(model_name)
    bert_config = BertConfig.from_pretrained(model_name)
    bert_model = BertModel.from_pretrained(model_name)
```

```
[4] sentence = "Hello, how are you?"
    inputs = tokenizer(sentence, return_tensors="pt", padding=True, truncation=True)
```

```
[5] with torch.no_grad():
        encoder_outputs = bert_model(**inputs)
```

```
[6] encoder_hidden_states = encoder_outputs.last_hidden_state
```

```
[7] class SimpleDecoder(nn.Module):
        def __init__(self, embed_size, vocab_size, hidden_size):
            super(SimpleDecoder, self).__init__()
            self.embedding = nn.Embedding(vocab_size, embed_size)
            self.rnn = nn.GRU(embed_size, hidden_size, batch_first=True)
            self.fc_out = nn.Linear(hidden_size, vocab_size)

        def forward(self, target_sequence, encoder_output):
            embedded = self.embedding(target_sequence)
            output, hidden = self.rnn(embedded)
            output = self.fc_out(output)
            return output
```

```
[8] vocab_size = tokenizer.vocab_size
    hidden_size = 768
    embed_size = 512
```

```
[9] decoder = SimpleDecoder(embed_size, vocab_size, hidden_size)
```

```
target_sequence = torch.tensor([[101, 7592, 2054, 2024, 2017, 102]])
```

```
[9] decoder = SimpleDecoder(embed_size, vocab_size, hidden_size)
```

```
[10] target_sequence = torch.tensor([[101, 7592, 2054, 2024, 2017, 102]])
```

```
[11] decoder_output = decoder(target_sequence, encoder_hidden_states)
```

```
[12] predicted_token_ids = torch.argmax(decoder_output, dim=-1)
```

```
[13] translated_sentence = tokenizer.decode(predicted_token_ids[0], skip_special_tokens=True)
```

```
print(f"Translated sentence: {translated_sentence}")
```

```
Translated sentence: ##vino dalton reforms startedthesis initials
```