NLP Experiment 10

Aim: Experimenting with an Advanced NLP Problem of Your Choice using Hugging Face Transformers, spaCy, NLTK (Natural Language Toolkit), AllenNLP, and BERTScore for Fine-Tuning Large Language Models (LLMs) for Standard NLP Problems

Theory:

Text classification is a fundamental natural language processing (NLP) task that involves assigning one or more labels or categories to a piece of text. We attempt to classify text using the BERT model.

We have used BERT (Bidirectional Encoder Representations from Transformers) for text classification.

A version of BERT Used: The specific version of BERT is "bert-base-uncased." This version is a base, uncased BERT model pre-trained on a large corpus of text.

Functions:

forward: This method defines the forward pass of the custom classification model.

- optimizer: It configures the AdamW optimizer with a specified learning rate.
- loss_fn: It defines the loss function as CrossEntropyLoss.
- train_loader: It prepares the training data in mini-batches for efficient training.
- Pooling: The code uses the [CLS] token for pooling. In BERT models, the [CLS] token's final hidden state is often used as a fixed-size representation for the entire input sequence.
- Layers: The BERT model consists of several layers, including the embedding layer (word embeddings, position embeddings, token type embeddings), multiple transformer layers (BertLayer), and a pooler layer (BertPooler).

Hyperparameters: Some hyperparameters include the learning rate (lr=1e-5), the number of training epochs (4), batch size (12), and the maximum sequence length (max_length=20).

The steps are:

- Use Bert Tokenizer on the synopses data.
- Encode Labels for the training data.
- Decide attention mask (the most important words in a sentence which need higher attention).
- Fine Tune the BERT model to develop a model that helps classify text.

Libraries and Tools Used:

- Pandas
- transformer