AILA FIRE TASK 2

Team Spectre Run1 description

The implementation was carried out in Google Colab notebooks (with a Tesla T4 GPU) for training the models built on the Pytorch framework.
Pretrained RoBERTa natural language model (a Robust and optimized BERT pretraining approach) was employed for the task of sentence classification. The transformers library offered by HuggingFace was also used for building the model.

The data was loaded from the raw text files into a pandas dataframe with two separate columns for ‘sentence’ and ‘label’. The labels were then enumerated to an integer (0-7) using a python dictionary.
A pretrained RoBERTa Tokenizer(“roberta-base”) was then loaded and applied for each sentence to get the input ids and the corresponding attention masks.
The resulting training dataset of the input ids, attention masks and labels was converted into a tensor dataset using the TensorDataset function from the torch.utils.data class, and using the same class a Data loader with a batch size of 16 was created.

```
add_special_tokens = True,
max_length = 400,
pad_to_max_length = True,
truncation = True,
return_attention_mask = True,
return_tensors = 'pt',
```

The resulting transformer model (a pre-trained base model from the RobertaForSequenceClassification class) was fine tuned over the training dataset to arrive at the list of parameters shown below. Adam Optimizer was used from the HuggingFace’s AdamW library with the learning rate set to (2e-5) and Adam epsilon set to (1e-8). The seed value was set to 42 and the model was trained for 4 epochs. The norms of the gradients were chipped to 1.0 to help prevent the explosive gradient problem.

```
==== Embedding Layer ====

roberta.embeddings.word_embeddings.weight              (50265, 768)
roberta.embeddings.position_embeddings.weight            (514, 768)
roberta.embeddings.token_type_embeddings.weight            (1, 768)
roberta.embeddings.LayerNorm.weight                     (768,)
roberta.embeddings.LayerNorm.bias                      (768,)

==== First Transformer ====

roberta.encoder.layer.0.attention.self.query.weight       (768, 768)
roberta.encoder.layer.0.attention.self.query.bias           (768,)
roberta.encoder.layer.0.attention.self.key.weight        (768, 768)
roberta.encoder.layer.0.attention.self.key.bias            (768,)
roberta.encoder.layer.0.attention.self.value.weight       (768, 768)
roberta.encoder.layer.0.attention.self.value.bias           (768,)
roberta.encoder.layer.0.attention.output.dense.weight     (768, 768)
roberta.encoder.layer.0.attention.output.dense.bias          (768,)
roberta.encoder.layer.0.attention.output.LayerNorm.weight      (768,)
roberta.encoder.layer.0.attention.output.LayerNorm.bias      (768,)
roberta.encoder.layer.0.intermediate.dense.weight       (3072, 768)
roberta.encoder.layer.0.intermediate.dense.bias            (3072,)
roberta.encoder.layer.0.output.dense.weight            (768, 3072)
roberta.encoder.layer.0.output.dense.bias                (768,)
roberta.encoder.layer.0.output.LayerNorm.weight             (768,)
roberta.encoder.layer.0.output.LayerNorm.bias              (768,)

==== Output Layer ====

classifier.dense.weight                           (768, 768)
classifier.dense.bias                               (768,)
classifier.out_proj.weight                          (7, 768)
classifier.out_proj.bias                              (7,)
```

Finally, the test data was similarly processed and passed through the model with a batch size of 1.