NLP HW3

110550108 施柏江

1. Describe how you implement your model, including your choice of packages, model architectures, model input, loss functions, hyperparameters, etc.

Ans:

The model was implemented using the PyTorch and Transformers libraries. I employed BERT as the backbone architecture for this sequence classification task. To create meaningful input sequences, the utterances, situations, and responses were combined into a single text input. The inputs are tokenized using the AutoTokenizer from the Transformers library. Input sequences were truncated or padded to a maximum length of 512 tokens, as BERT's architecture has this inherent limitation.

For training, the cross-entropy loss function was used to optimize the binary classification task. The optimizer chosen was AdamW. The model was trained with a batch size of 32, a learning rate of 5e-5, a weight decay of 5e-4 for regularization, and up to 10 epochs. Training was conducted with gradient clipping to stabilize updates and prevent exploding gradients. Validation was performed after each epoch, and the model with the highest validation accuracy was saved.

2. What processing did you do with the data? Is there an improvement in predictive accuracy when utilizing both situations and utterances for prediction, compared to solely relying on utterances? Why or why not?

Ans:

The raw data consisted of three main components: utterances, situations, and responses. Each component underwent preprocessing to ensure compatibility with the BERT tokenizer. The first one is text concatenation. Utterances, situations, and responses were combined into a unified format for input to the model. The second one is padding and truncation. Input sequences were adjusted to the fixed maximum token length to maintain uniformity. The last one is label conversion. For training data, the quality labels were converted into integer format for compatibility with the loss function.

Incorporating the situations alongside the utterances provided a substantial improvement in predictive accuracy. When relying solely on utterances, the model often struggled to capture the broader context of the conversation. By including situations, the model had access to additional context about the environment or background, which helped it better understand the relevance and appropriateness of the response. The improvement can be attributed to reduced ambiguity. Some responses may be appropriate in one context but not in another. Situations help resolve such ambiguities, leading to better predictions.

3. **Compare all the methods you have tried and use a table to display their respective performances. Which method performed the best, and why?**

Ans:

To evaluate the performance of various approaches in modeling the task, I explored multiple methods that focused on different model architectures and configurations while maintaining the same input structure for fair comparison.

Traditional machine learning models like Random Forest and Logistic Regression rely on sparse, fixed feature representations. These methods are not equipped to handle nuanced relationships within the text, resulting in substantially lower performance.

While Bi-LSTM with pre-trained embeddings can capture sequential relationships, it lacks the deep contextual understanding that BERT's transformer architecture provides.

Fine-tuning BERT achieved the highest validation accuracy of **76.5%**. BERT's ability to capture contextual information across input sequences made it particularly effective for this task. BERT's bidirectional attention mechanism enables it to process the entire input sequence contextually, capturing intricate dependencies between tokens. Fine-tuning allows the model to adapt to domain-specific nuances, making it highly effective compared to other approaches that either lack pre-training or rely on fixed feature representations.

Method	Logistic Regression	Random Forest	Bi-LSTM with Pre-trained Embeddings	Fine-tuning BERT
Validation Accuracy	60.8%	63.5%	70.3%	76.5%