NLP HW#4

Student Name:
Mehdi Moosavi

SID:
810102264

June , 2024

Dept. of Computer Engineering

University of Tehran

Contents

1 roberta-large

1.1 Dataset

The Multi-Genre Natural Language Inference (MultiNLI) corpus is a crowd-sourced collection of 433k sentence pairs annotated with textual entailment information. The corpus is modeled on the SNLI corpus, but differs in that covers a range of genres of spoken and written text, and supports a distinctive cross-genre generalization evaluation

1.2 Fine-tuning Approaches

Fine-tuning in deep learning refers to the process of taking a pre-trained model and further training it on a specific task or dataset to improve its performance. There are two traditional approaches to fine-tuning:

1. **Full Fine-tuning**: In this approach, all parameters of the pre-trained model are updated during fine-tuning. This means that every layer of the model is adjusted based on the new task or dataset.

2. **Layer-wise Fine-tuning**: Here, only one or a few layers of the pre-trained model are fine-tuned, while the other layers are frozen. This is particularly useful when the pre-trained model is already well-suited for the task at hand, and only certain parts of it need adaptation.

Now, regarding the LORA method, it proposes a different idea for fine-tuning models. LORA stands for Low-Rank Adaptation, and works by using a smaller set of parameters to achieve similar results. It does this by decomposing the weight matrix of the LLM into two lower-rank matrices. These matrices are then used to transform the input and output of the model in a way that captures the essential information for the fine-tuning task.

1.3 Hard and Soft prompt

After the emergence of Large Language Models (LLMs), research expanded into prompt-based methods. Prompt hard and prompt soft are two such methods:

1. **Prompt Hard**: In this approach, a specific prompt or instruction is provided to the model to generate the desired output. The prompt serves as a rigid constraint, guiding the model to produce results that align with the given instructions. This method requires designing precise prompts tailored to the task at hand, which can sometimes be challenging and may limit the model's flexibility in handling variations in input.

2. **Prompt Soft**: Unlike prompt hard, prompt soft provides a more flexible framework by allowing the model to generate outputs based on softer cues or hints rather than strict prompts. Instead of explicit instructions, prompt soft methods utilize more abstract or implicit guidance, such as example demonstrations or implicit biases embedded in the training data. This approach offers greater adaptability and generalization capabilities, as the model can learn to infer the desired task from subtler cues within the input data. However, designing effective soft prompts and ensuring their consistency across different tasks remain ongoing research challenges.

1.4 Fine Tuning

Now we want to fine-tune the model on our dataset. In the first part, we update all parameters, which amounts to approximately 355 million parameters. The training process took 2 hours and 10 minutes with a learning rate of 2×10^{-5} and a batch size of 4. The model was trained on 3% of the training data. The accuracy on the evaluation set using this approach was 86.45%.

1.5 LORA

In the second part, we used Low-Rank Adaptation (LoRA) to fine-tune the model. The hyperparameters for LoRA were set with $\alpha = 16$, $r = 8$, and a dropout rate of 0.1. Other training hyperparameters remained the same as before, but the number of trainable parameters decreased to 4.6 million, and the training time reduced to 1 hour and 55 minutes. The accuracy on the evaluation set using this approach was 84.45% ,which is a slight drop.

1.6 P-tuning

In the third and final part, we employed P-tuning, a soft prompting method, as mentioned in the question. The count of trainable parameters for this approach was 1.35 million, significantly fewer compared to previous methods and also The training time was notably reduced to just 45 minutes. However there was a substantial drop in accuracy to 33 percent which is accuracy of random classifier over this dataset.

1.7 Why LORA

here is how we can use p-tuning or LORA to address this problem

1.7.1 LORA

To use LORA for multi-task learning, we select a pre-trained base model like RoBERTa, develop task-specific adapters for each task, fine-tune these adapters on task-specific data while keeping the main model parameters frozen, integrate the trained adapters into the model's layers, and during inference, activate the relevant adapter based on the task to obtain task-specific predictions.

1.7.2 P-tuning

For P-tuning, we need to prepare task-specific prompts or prompt embeddings, fine-tune these prompt embeddings on task-specific data, use optimization techniques to adjust prompt embeddings for each task, during inference, provide the appropriate prompt for the desired task, and the model utilizes the given prompt to generate task-specific outputs without modifying its core parameters.

By following these integrated steps, we can effectively implement LORA or P-tuning to enable multi-task learning in our model.

2 Lamma3

2.1 ICL

In context learning (ICL) is a technique where the model leverages the context within the input data to make predictions without additional fine-tuning. This approach allows the model to adapt to new tasks by using examples provided within the input, making it highly flexible and efficient for various applications.

2.1.1 Zero-Shot Learning

Zero-shot learning is a machine learning paradigm where a model is trained to recognize and classify instances from classes it has never seen during its training phase. Instead of relying on labeled examples for every possible class, zero-shot learning leverages semantic information about the classes, such as textual descriptions or attributes, to make predictions.

Example prompt is like this:

- **P**remise: A man inspects the uniform of a figure in some East Asian country.
- **H**ypothesis: The man is looking at something.
- **D**oes the hypothesis follow from the premise? Answer (yes, no, maybe):

After giving the examples to the model and inspecting the output, we assigned a label to each response and then calculated accuracy, which in this case was 12 percent. It is worth mentioning that the baseline here is not 33 percent but is 0, as we label the answers that don't have one of the 3 predefined answers as "no answer".

2.1.2 One-Shot Learning

One-shot learning is our next approach, where we provide the model with a single example before asking it to generate the output based on that example. The prompt for this part included the following example:

- **Premise:** "A man inspects the uniform of a figure in some East Asian country."
- **Hypothesis:** "The man is looking at something."
- **Label:** "yes" (indicating entailment)

The reason i used this example is that this is clear ,straightforward, relevant to the subject and representative of the task. Then i add the mentioned example with the prompt that i had in Zero-Shot part to use in this part and apply one shot learning.using this technique accuracy improved to 40 percent which is considerable. In both parts i set the temperature to 0.5 which means there is less randomness in outputs and the output is more precise.I did this as our task was classification and not text generation and we needed more accuracy rather than more variation in output.

2.2 Qlora classification

In the previous section, we trained a model fine-tuned for classification tasks. In this part, we trained a quantized Qlora model with Qlora hyperparameters consistent with our earlier settings: r=64, alpha=32, and dropout=0.05. The total number of trainable parameters remained approximately 55 million. Additionally, the linear layer was positioned as the final layer to effectively map the model's features to three outputs.

On the evaluation set, the accuracy of the model reached 62.8%. We conducted training for 500 steps, which took nearly 45 minutes to complete.

In conclusion, our experimentation with various models and techniques yielded diverse results. We observed distinct outcomes between the models, depending on the techniques employed. Notably, p-tuning and one-shot learning emerged as methods that can be implemented with lower computational power, proving particularly effective for simpler tasks where high accuracy isn't a priority.

However, when the task demands greater precision or complexity, techniques such as quantization and methodologies like LORA become imperative. These approaches allow us to harness the full potential of our models, at the cost of increased computational resources. Thus, the choice of technique must align with the specific requirements of the task at hand, balancing computational efficiency with the desired level of accuracy.

Table 1: Comparison of Model Performance

Model	Trainable Parameters	Time Taken	Accuracy
roberta	355 million	130 minutes	86.45%
roberta lora	4.6 million	115 minutes	84.4%
roberta p-tuning	1.352 million	45 minutes	33%
lamma3 zero shot	0	0	12%
lamma3 one shot	0	0	40%
lamma3 classification	54.6 million	41 minutes	62.8%

note that the numbers are not completely comparable as in lamma3 we limited max steps to 500 or in zero and one shot the baseline accuracy is 0 and not the 33 percent.