Assignment 8

 # Problem 1:
Why is fine-tuning needed? Given 2-3 reasons. How is it typically achieved? Give 2-3 methods. Write a total of 4-6 sentences. [2 points]
 # Solution 1:
Fine-tuning is needed to adapt pre-trained language models to specific tasks or domains, ensuring they generate more relevant and accurate outputs. It helps in improving performance on tasks like sentiment analysis, question-answering, or domain-specific text generation by providing task-specific knowledge. Fine-tuning also addresses the problem of data distribution mismatch between the general pre-training corpus and the specialized task dataset.

Fine-tuning is typically achieved through methods such as:

1. Supervised Fine-Tuning: Using labeled task-specific data to train the model further on the desired task.
2. Transfer Learning: Adapting a pre-trained model to a new but related task by training it on a smaller dataset related to the new task.
3. Prompt-based Fine-Tuning: Modifying the input prompts to guide the model towards generating more task-specific responses.

 # Problem 2:
While using the RL policy update method for fine-tuning, why and how is human data used? [2 points]
 # Solution 2:
Human data is used in the Reinforcement Learning (RL) policy update method for fine-tuning to ensure that the model's outputs align with human preferences and values. This approach leverages human feedback to guide the model towards generating more desirable and accurate responses. The process typically involves the following steps:

1. Human Feedback Collection: Humans review and rank multiple outputs generated by the model for given prompts. This feedback is used to create a reward signal that reflects human preferences.
2. Reward Modeling: A reward model is trained using the collected human feedback to predict the quality of the model's outputs.
3. RL Fine-Tuning: The language model is fine-tuned using RL techniques like Proximal Policy Optimization (PPO), where the reward model provides the reward signal that the RL algorithm uses to update the model's parameters.

By incorporating human data, the fine-tuning process ensures the model's behavior is more aligned with human expectations and ethical considerations .

 # Problem 3:
Describe two pros and two cons of SLMs compared to LLMs. [2 points]
 # Solution 3:

Pros of SLMs (Smaller Language Models):

1. Resource Efficiency:
    - Pros: SLMs require significantly less computational power and memory compared to LLMs. This makes them more accessible for deployment on devices with limited resources, such as mobile phones and edge devices. They also incur lower costs for training and inference.
    - Cons: This efficiency often comes at the expense of reduced performance and capability, particularly in complex tasks.
2. Faster Inference Time:
    - Pros: Due to their smaller size, SLMs generally have faster inference times. This is beneficial for applications requiring real-time processing and quick responses, such as conversational agents or interactive applications.
    - Cons: Despite the speed, they may struggle with maintaining the same level of fluency and coherence in generated text as LLMs.

Cons of SLMs Compared to LLMs:

1. Limited Understanding and Contextual Awareness:
    - Cons: SLMs typically have a more limited capacity to understand and generate text involving complex language structures and long-range dependencies. They may struggle with maintaining context over longer conversations or documents, leading to less coherent and contextually accurate outputs.
    - Pros: This limitation can sometimes be mitigated with careful prompt engineering and task-specific fine-tuning, though it may never fully match the capabilities of LLMs.
2. Lower Performance on Complex Tasks:
    - Cons: SLMs often underperform on tasks requiring advanced comprehension, such as nuanced language understanding, detailed content generation, and intricate reasoning. Their smaller size restricts their ability to capture and utilize extensive language patterns and knowledge.
    - Pros: However, for simpler or more focused tasks, SLMs can still be highly effective and provide sufficient performance, especially when optimized for specific use cases.

 # Problem 4:
In the class we worked with fine-tuning an LLM to do question-answering task. But we didn't get good accuracy on our test data. We need to improve that. Think about a couple of ideas that

you could try to boost this accuracy. Now do some experimentations. Again, there are going to be some trial-and-error here. Report what you did and what you found. [4 points]

 # Solution 4:

- Modify (increase) Training size
- Increase Iterations/Epochs
- Play with Hyperparameters like batch size, learning rate, scheduler
- Experiment with different LLM models, probably larger ones