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]**

You have a pre-trained large model which is trained on general text corpus from sources like the internet etc. It contains correct and incorrect data both. So, after pre-training the model generalizes well on generating tokens based on some context. But there is no ground truth. To tailor this pre-trained model on specific applications, we need to further train it on specific datasets with ground truth (QnA, Chatbot etc) to give new behaviours. This is achieved by freezing the weights of the model and only updating final few layers of the model (Transfer learning). Below are some methods:

- Supervised Fine tuning to update all the weights of the model (like the example in Q3)
- Transfer learning – freeze model weights and update only the final few layers.
- Parameter Efficient Fine Tuning – using methods like LoRA or adapters

2. **While using the RL policy update method for fine-tuning, why and how is human data used? [2 points]**

In reinforcement learning fine-tuning, human data is used to guide the model toward preferred behaviors. This is done by collecting human feedback (ranking model responses) to train a reward model, which the RL policy then optimizes.

This approach helps align the model's outputs with human preferences, especially in open-ended tasks like dialogue.

3. **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. [6 points]**

The model as worked on in the class with 1000 samples gave an **accuracy of 77%**

> Who did the Panthers beat to become the NFC champs?
>
> Ans: Arizona Cardinals
>
> **Incorrect (Predicted: cam newton, Given: the Arizona Cardinals)**

Experiment 1: Increased batch size to 2000, eval size 200. Epocs = 5. Batch size 16.

Accuracy: 80%

Experiment 2: Let's first increase the dataset size from 2000 to 5000 and eval size from 200 to 500. Also, increased the number of epochs from 3 to 5 and batch size from 8 to 16. It took 30 mins to run. Below are the results:

The training loss decreased significantly. It shows the model learned well.

Step	Training Loss
500	1.072300
1000	0.393500
1500	0.130500

Accuracy improved to 82%.

> Who did the Panthers beat to become the NFC champs?
>
> Ans: Arizona Cardinals
>
> **Predicted: arizona cardinals, Given: the Arizona Cardinals**

Training on more samples and increasing the number of epochs allowed the model to learn more features. I was afraid looking at the model's training loss decrease and was thinking that it may be starting to overfit on data. But that does not seem to be the case. I tried random examples and found that even for incorrect predictions, many answers are wrong because of missing article ('a', 'an', 'the') etc. But the answer is correct. The accuracy of 82% is misleading here, it's much more in reality.