Task 3 – Dataset Preparation for Fine-Tuning
Name: Lokesh Reddy
Date: 06 July 2025

Introduction

Fine-tuning a language model helps make it more accurate for a specific use case like customer service, legal documents, or medical data. But just having a dataset isn't enough — how we build and clean the dataset really matters. A good dataset teaches the model the right patterns, avoids confusion, and gives better results. In this write-up, I'll explain some key steps I would follow to prepare a good dataset, and also compare a few fine-tuning methods based on what works best in real-world situations.

Techniques for Preparing High-Quality Datasets

1. Start With a Clear Goal

Before gathering any data, it's important to understand what we want the model to do. For example, if the goal is to build a chatbot for college admissions, we should only collect data related to student queries, course details, eligibility, etc. Setting a clear goal helps avoid unnecessary or confusing data.

2. Collect Data From the Right Sources

The quality of data depends on where it comes from. I would try to use sources that are relevant and trustworthy. For example, real chat transcripts, official websites, or PDF handbooks. If we're collecting from the web, we should always double-check and remove anything that doesn't belong.

3. Clean the Data Properly

Raw data is usually messy. It might have grammar issues, strange symbols, duplicate lines, or unrelated content. So, I would clean it by removing noise, fixing spelling, and making the format consistent. This also includes removing personally identifiable or sensitive information if needed.

4. Format the Dataset Clearly

For fine-tuning, the data needs to be in a clear and structured format. A good format would be to have a "question" and an "answer" pair. For example:

json

Copy code

{"prompt": "What is the fee structure?", "completion": "The annual fee is ₹50,000 for undergraduate courses."}

This format helps the model understand what kind of input leads to what kind of output.

5. Split the Data Smartly

Once the data is ready, it should be divided into training, validation, and test sets. Usually, 80% of the data goes into training, 10% for validation (to check during training), and 10% for final testing. This avoids overfitting and helps check how well the model performs on unseen examples.

Comparison of Fine-Tuning Methods

There are a few ways to fine-tune language models depending on the use case and resources:

a. Full Fine-Tuning
This method updates all the weights in the model using the new dataset. It gives strong results but needs a lot of compute power and time. It's usually done for large, high-budget projects.

b. Prompt Tuning
Here, we just train some special tokens (prompts) and keep the rest of the model frozen. It's fast and cheaper, but it may not give great results for complex tasks.

c. LoRA (Low-Rank Adaptation)
LoRA is a smart method that only trains a small number of layers inside the model. It uses less memory and still gives good accuracy. This method is becoming popular because it works well even on small datasets and is easier to manage.

My Preferred Method: LoRA

Out of the three, I would choose LoRA. It's a good balance between cost, speed, and accuracy. Since most student or startup projects don't have access to huge GPUs, LoRA is practical. It allows me to fine-tune the model without needing big infrastructure.

Conclusion

A strong dataset is the foundation of any good fine-tuning project. It’s not just about collecting data, but also about how carefully it’s cleaned, organized, and structured. By using a method like LoRA and preparing the data properly, it’s possible to build models that perform well even with limited resources.