Task 3 – Dataset Preparation and Fine-Tuning Strategies

Harsh Indoria

Note : PDF is AI augmented not AI generated. Answers are based on my ideas and formatted by ChatGPT for proper reading.

🧠 1. Introduction

Large Language Models (LLMs) like Mistral, GPT-4, or LLaMA perform best when trained on clean, contextual, and well-structured data. This document presents strategies to prepare and refine datasets for fine-tuning a QA chatbot for Bay Wheels and discusses practical fine-tuning methods tailored to real-world deployment.

🔍 2. Key Principles for Dataset Preparation

✅ 2.1 Column Name Refinement from RDBMS Exports

> **"Columns are context."**
> When data is exported from relational databases (RDBMS), column names often use short forms, acronyms, or ambiguous labels (e.g., `st_id`, `rid`, `dur_sec`).

Before using this data for fine-tuning or embedding:

- **Rename columns** into semantically meaningful names:
    - `st_id` → `station_id`,
    - `rid` → `ride_id`,
    - `dur_sec` → `duration_seconds`

- **Why it matters:**
  Models can better understand relationships and generate more accurate answers when the **schema provides rich context**.

🔁 2.2 Data Transformation & Normalization

- Apply **normalization** to unify formats:
  - Dates → `YYYY-MM-DD`
  - Prices → Use consistent currency symbols
  - Units → Standardize time and distance formats
- Perform **data transformation** to:
  - Merge related fields (e.g., first and last name → full name)
  - Split compound columns if needed

Benefits:

- Reduces redundancy
- Saves storage for large datasets
- Improves consistency across different sources

🧪 2.3 Data Validation

- Set up **validation pipelines** to catch:
  - Missing values
  - Invalid types (e.g., text in numeric columns)
  - Out-of-range values (e.g., negative trip durations)

Outcome: Ensures only trusted, clean data goes into fine-tuning or vector indexing.

🧩 2.4 Segmentation & Departmental Structuring

Divide data and questions into **departmental categories** based on:

- Business unit (Marketing, Operations, Support)
- Data source (e.g., trips, stations, membership)
- Use case (e.g., pricing info, error troubleshooting)

Why it matters:

- Fine-tuned models become more **context-aware**.
- You can later pair this with **department-specific routing logic** (Task 2).
- Improves retrieval accuracy and fine-tuning efficiency.

🔧 3. Dataset Curation Methods

✍️ Manual QA Curation

- Real support queries, internal knowledge base, SOP documents
- Reviewed by subject matter experts

🤖 Synthetic QA Generation

- Use GPT-4 or Mistral to auto-generate QA pairs from your documents
- Filter and rephrase for tone and accuracy

🧼 Cleaning & Structuring

- Use JSON or CSV with fields like:
    - `prompt`, `response`, `tags`, `source_doc`, `department`

⚙️ 4. Fine-Tuning Approaches: Comparative Analysis

Method	Pros	Cons	Use Case Fit
Full Fine-Tuning	High performance	High cost + GPU required	Large orgs, cloud setup
Prompt Tuning	Minimal compute needed	Narrow domain	Repetitive queries

I prefer **Prompt Tuning** because:

- Most queries in my use case are **straightforward and domain-specific**, not requiring deep reasoning.
- It allows me to **minimize resource usage** while still getting **reliable responses**.
- It keeps the base model **unchanged**, making deployment and updates easier.

📈 5. Summary & Impact

Best Practices	Impact
Renaming RDBMS columns	Improves schema comprehension
Data normalization & validation	Reduces errors, improves performance
Segmenting data by department/use	Enhances QA relevance & routing

---

🧩 6. Conclusion

Fine-tuning a language model requires more than just feeding data—it requires **engineering discipline**. By focusing on **column clarity, data hygiene, and modular segmentation**, we ensure that models understand both the structure and semantics of our business data.

This approach not only improves answer accuracy but also future-proofs the chatbot system for scale and maintainability.