FALL 2025 | DATA 266 | Homework -3

Deadline – 11.59 PM – 11/13/2025

20 Points

Question # 1 10 pts:

Fine-Tuning Decoder-Only Language Models for Tabular Classification using the Titanic Dataset

In this task, you will fine-tune two recently released decoder-only models to perform binary classification on the classic Titanic dataset, predicting whether a passenger survived or not. Although the Titanic dataset is small and tabular, your goal is to treat this problem as a text-based classification task, simulating how language models can learn from structured data when properly formatted as text prompts. Apply a stratified split (80%: 20% between train and test) on the label of the dataset to maintain the class ratio.

(2 pts) Data Formatting (Prompt-Response Format)

Convert each training record into a natural-language training example using a function (not LLMs), for example:

Input: Passenger details - Pclass: 3, Sex: male, Age: 22, SibSp: 1, Parch: 0, Fare: 7.25, Embarked: S.

Output: Survived: No

- During inference, only the input part is given (no "Output" line), and the model must generate "Survived: Yes" or "Survived: No."
- Show at least five samples before and after transformation.

Model Selection

- Use at least two recent decoder-only models (released in 2024 or later).
- Suggested examples (you may choose others if available):
  - **microsoft/Phi-3-mini-4k-instruct**
  - **google/gemma-2b**
  - **mistralai/Mistral-7B-v0.3**
  - **meta-llama/Meta-Llama-3-8B**

(6 pts) Fine-Tuning Requirements

- Use Hugging Face Transformers with the Trainer API.
- Apply response-only masking (compute loss only on the "Output" text).

- Train for minimum 5 epochs on the training split and evaluate on the test split.

(2 pts) Evaluation

- Compare model predictions on the test set.
- Evaluation metrics: Accuracy, Precision, Recall, and F1-score.

BOUNS: Test performance: Accuracy above 84% → 2 pts; Accuracy above 88% → 4 pts; Accuracy above 92% → 6 pts;

Question # 2 (10 pts):

Building a Scientific Literature Assistant using Retrieval-Augmented Generation (RAG)

You will build a RAG-based question-answering system that can answer free-form research queries by retrieving and reasoning over scientific paper abstracts from the arXiv dataset.

The goal is to compare two retrieval strategies:

- Naive RAG – basic dense retrieval using vector similarity.
- Re-ranking RAG – retrieval followed by an LLM-based or cross-encoder re-ranking stage.

Dataset Setup

```
from datasets import load_dataset

dataset = load_dataset("CShorten/ML-ArXiv-Papers")
```

Each record includes: title and summary (abstract text)

- **(1 point) Document Creation -** Combine each paper's metadata into a single paragraph for retrieval. Show at least two samples.

Generate a response for the following query using Naïve RAG and Re-ranking RAG:

"What are the current limitations of diffusion models in handling long-sequence text generation tasks?"

(3 points) ***Naive RAG Pipeline***

- Encode each document using a dense retriever such as sentence-transformers (all-MiniLM-L6-v2).
- Build a FAISS index.
- Given a research query, retrieve the top 3 most relevant papers and pass them to a generator model (e.g., mistralai/Mistral-7B-Instructor meta-llama/Meta-Llama-3-8B-Instruct).
- Generate a concise academic answer.

(3 points) ***Re-ranking RAG Pipeline***

- Retrieve **top 10** papers using the same retriever.
- Re-rank them using either
  - a cross-encoder model (e.g., cross-encoder/ms-marco-MiniLM-L-6-v2)
- Select the top 3 after re-ranking and generate an improved response.

(3 points) ***Queries for Evaluation***

Use the following 10 sample queries:

1. What are the latest methods for multimodal learning?
2. Which papers explain reinforcement learning with human feedback?
3. How are transformers applied in computer vision?
4. What are recent advances in self-supervised learning?
5. Which models address few-shot text classification?

Metric	Description	Scale
Relevance	How well the generated response answers the query	1–5
Factuality	Is the information consistent with the retrieved abstracts?	1–5
Fluency	Clarity and readability of the generated text	1–5

Compare the **Naive RAG** and **Re-ranking RAG** outputs for all queries using these metrics. Create two tables using human judge and LLM Judge like below. Mention the prompt used for LLM judge. Finally, compare the performance of human judge vs. LLM judge.

Query	Naive RAG Relevance	Re-ranked RAG Relevance	Naïve RAG Fluency	Re-Ranking RAG Factuality	Naïve RAG Factuality	Re-Ranking RAG Fluency	Observation
Q1							Re-ranking improved factual grounding
Q2							No major change

You are required to follow:

1. Submit **one** MS/PDF documents in terms of report. Failing to submit a report containing the following will be penalized from 2-5 points:

- Include all the steps of your calculations (if there is any).
- **Include all discussion in this report.**
- Include the summary of the model.
- Attach screenshots – showing first few epochs of model training.
- Attach screenshots of the important code outputs such as confusion matrices, learning curves, and classification reports.

2. Source code:

   a. Python (Jupyter Notebook)

   b. Ensure it is well-organized with comments and proper indentation.

- **Failure to submit the source code will result in a deduction of full/partial points.**
- Format your filenames as follows: "your_last_name_HW2.pdf" for the document and "your_last_name_HW2_source_code.ipynb" for the source code.
- Before submitting the source code, please double-check that it runs without any errors.
- Must submit the files separately.
- Do not compress into a zip file.
- HW submitted more than 24 hours late will not be accepted for credit.