Machine Learning Engineer Take-Home

Cofactor AI

Instructions

The purpose of this take-home interview is to get a sense of how you think about solving machine learning problems in the real world. This entire assessment should take you about 4 hours assuming familiarity with some common LLM techniques, but you have 48 hours to make sure you have time to think through things as you need.

The questions are intentionally open-ended, the expectation is that you are able to do a bit of research to sketch a reasonable approach, but you do not generally need to implement it unless asked. Cite your sources (links will suffice, no need for a formal bibliography) where possible. Your responses don't have to be perfect, but try to show your reasoning as much as possible - think of this as the notes you would prepare going into a conversation with a future engineer to build on it.

Task

Ambient scribes are popular software tools for transcribing audio data from doctors as they go through their daily work. You're given a dataset of 10 transcripts from different doctor's visits generated using ambient scribing tools, and your task it to turn them into well structured and high quality medical record notes.

You will find the transcripts of the doctor's visit in `./transcripts`. We will go through three different iterations of turning these transcripts into high quality medical notes, with each iteration increasing in complexity. However you will only be expected to actually implement the first iteration, and subsequent iterations will be in the form of notes/sketches.

Your response should be consist of the following contents:

1. A git repository with a response to section 1 (can be a link to a github repo or a zip file).
2. A PDF or markdown document with written responses to sections 2 - 3. Your responses can include pictures/screenshots, tables, drawings, bullet points, or anything that clearly but simply conveys the answer. If you want to use external whiteboarding tools like Miro or Excalidraw, you can use those and include screenshots.

1 Idealized SOAP Note

The goal of this section is to turn each transcript into a simple, well-structured SOAP note (Subjective, Objective, Assessment, Plan) using an LLM or a chain of LLMs. Feel free to use any LLM tooling you are familiar with, and make sure to document your approach. You can see an example SOAP note in the `./example_notes/Medical Visit SOAP Note.pdf` file. You might find the Prompting Guide helpful for this section.

Implement a pipeline that takes a transcript and outputs a SOAP note. The pipeline should use LLM(s) to extract and organize the relevant information into the four SOAP sections. Document your code and design decisions thoroughly, as if you were convincing a skeptical engineer of your approach. Include comments, docstrings, and a README if appropriate. Clearly describe any prompt engineering, chain design, or post-processing steps you use. If there are parts of the system that are better done without LLMs, describe and justify those choices. Provide a brief discussion of the limitations of your approach and how you might improve it in future iterations. Your implementation should be reproducible and easy to run for someone with basic Python and LLM experience.

2 Basic EMR

In this section, you will design (but not implement) a Retrieval-Augmented Generation (RAG) system that can turn a transcript into a basic electronic medical record (EMR) note. You can see an example EMR note in the `./example_notes/Simple EMR Note.pdf` file. You may find this survey of RAG systems from Gao et. al. (2023) helpful for this section, and feel free to use any other literature you are familiar with as a reference.

Your response should address the following:

1. **Data Sources:** Identify and justify the external data sources (e.g., medical ontologies, clinical guidelines, patient history, etc.) that would be most useful for retrieval in this context.
2. **System Architecture:** Sketch a high-level architecture for your RAG system. Include components for retrieval, LLM generation, and any necessary preprocessing or postprocessing. Diagrams or flowcharts are encouraged. If there are parts of the system that are better done without LLMs (e.g., rule-based extraction, regex, entity recognition, section segmentation), describe and justify those choices.
3. **Evaluation and Optimization:** Propose a process for evaluating the quality of the generated EMR notes. Discuss metrics, human-in-the-loop evaluation, and how you would iteratively improve the system.

3 Modern EMR

In this section, you will design a system to turn a transcript into a much more complex electronic medical record using an agentic approach with tool use. You can see an example EMR note in the `./example_notes/Complex EMR Note.pdf` file. Sign et. al. (2025) provides an overview of the state of the art in agentic RAG systems, feel free to use it or any other literature you are familiar with as a reference.

Your response should address the following:

1. **Tool Selection:** Identify the set of tools (e.g., medical knowledge bases, code execution, search APIs, calculators, structured data extractors) that should be exposed to the LLM agent. Justify your choices.

2. **Agentic System Architecture:** Sketch an architecture for an agentic system, showing the different roles LLMs can play (e.g., orchestrator, retriever, summarizer, validator). Use diagrams or tables if helpful. Consider where fine-tuning or distillation of a larger model may be appropriate like in the Chen et. al (2024) paper, and discuss the tradeoffs between different components (LLMs vs. non-LLM tools). Be thoughtful about the division of labor between LLMs and traditional software components, and justify your design decisions.

3. **Evaluation and Optimization:** Propose a process for evaluating and optimizing the agentic system. Discuss both automatic and human-in-the-loop evaluation, and how you would use feedback to improve the system.