SUPaHOT: Universally Scalable and Private Method to Demystify FHIR Health Records

Stanford CS224N Custom Project

Michael Brockman
Department of Symbolic Systems
Stanford University
`mikebrockman@stanford.edu`

Nina Boord
Department of Computer Science
Stanford University
`nboord@stanford.edu`

Hamed Hemkat
Department of Symbolic Systems
Stanford University
`hhemkat@stanford.edu`

1 Abstract

The evolving landscape of the standardized healthcare data format Fast Healthcare Interoperability Resources (FHIR) and the integration with Large Language Models (LLMs) present a promising tool for increasing healthcare literacy. In this paper, we explore using LLMs to allow patients to query and understand their personal medical records. While existing research, notably Schmiedmayer et al. (2023), have applied OpenAI's GPT-4 to querying FHIR data, challenges of cloud-based API usage of 3rd party models such as scalability costs and privacy concerns remain. To address these concerns, we assess 3 models on 3 tasks pertaining to the filtering and summarizion of FHIR data based on patient queries, comparing them to the performance of our "oracle", GPT3.5 Turbo. We used LLaMA2 7B Chat (a general-purpose baseline), Meditron (a 7B parameter LLaMA model pre-trained on a large medical corpus), and SUPaHOT (a LLaMA2 7B model fine-tuned using our custom datasets built for our specific tasks). We do not advocate for our model to be used for these tasks in a real-world context. Rather, we seek to demonstrate the potential of the use of smaller "expert" models that are fine-tuned on a specific tasks to enable scalability and privacy for healthcare data interaction with LLMs. We note that whether or not our fine-tuned SUPaHOT models outperform our baseline and/or Meditron in key metrics is dependent on the specific nature of the task, demonstrating the relative merits and importance (depending on the context) of both task-specific fine-tuning and increasing general domain knowledge through broader pretraining.

2 Key Information

- Mentor: Bessie Zhang
- External mentors: Philipp Zagar, Oliver Aalami, Vishnu Ravi, Paul Schmiedmayer
- Sharing project: n/a
- Team contributions: Hamed developed the preprocessing, query generation, LLaMA2 generation, and evalutation scripts, along with conducting the fine-tuning of SUPaHOT and attempting to draft a fine-tuning script for Meditron. Michael figured out how to use FastChat (surprisingly complicated), developed the oracle and Meditron generation scripts, and attempted to use Meditron's built in fine-tuning process. Nina repurposed our scripts to operate asynchronously, wrote scripts to process data for fine-tuning, helped conduct the fine-tuning of SUPaHOT, and communicated with our external mentors for guidance in designing our system. We all worked on the paper and poster.

3 Introduction

In the United States healthcare system, patient medical records are fragmented across a variety of providers, preventing patients from accessing their data. The implementation of electronic records has done little to solve this problem, and patients often jump through hoops to gain access to their health data. Emerging tailwinds, however, such as the Anti-Information Blocking section of the Cures Act, effective September of 2023, require healthcare systems to give patients easy access to their own electronic medical records. New regulation mandates the data standard FHIR be used for all medical records, and new Apple Health infrastructure enables users to access these records. Put together, society is heading towards a future of democratized healthcare data. Access, however, is null without understanding. Health records are verbose, disorganized, and dense with medical jargon–often unintellegable for the average patient. LLMs have the potential to unlock power of these records, allowing patients to inquire about and understand their own data - allowing both patients and physicians to make more informed treatment decisions.

4 Related Work

The only paper we use for inspiration is Schmiedmayer et al. (2023), as using LLMs on FHIR data is a considerably new field of research. The closest adjacent research to Schmiedmayer et al. (2023) are papers evaluating the performance of LLMs in the task of translating text data into FHIR format, which is not relevant to our work.

LLMonFHIR demonstrates an LLM's ability to query and understand one's own medical data. For instance, a patient might ask the LLM to list out all of his or her medications as well as what substances might react with such medications. They might also ask the LLM to provide information on a past surgery or medical procedure that is relevant to their next doctor's appointment. Follow-up questions allow the user to have a deeper understanding of their data. LLMonFHIR uses expert ratings from doctors to evaluate the LLM on three abilities: its ability to provide accurate answers given the resources, the responses' relevancy to the patient query and given resources, and the understandably of the responses by the average person. The paper found that GPT4 preformed well across the board, but performed better on some questions than others, sometimes providing out-of-context information. Furthermore, the research points out an inconsistency in answers provided even for the same prompt. The paper further describes the limitation of GPT4 in terms of both scalability and privacy, inspiring SUPaHOT.

5 Approach

We have selected these models because they are large enough to handle the task at hand, yet small enough to potentially be hosted locally. In our experiments, we compare and contrast a baseline (LLaMA 2 7B Chat from Touvron et al. (2023)) with the same model pre-trained on domain-specific knowledge (Meditron) versus the same model fine-tuned on each of our tasks (SUPaHOT). We chose to work with variations of LLaMA 2 models because we understood them to be among the more efficient LLMs available by leveraging sparse attention mechanisms. More specifically, we learned about grouped query attention (GQA), striking a balance between multi-headed attention (1-to-1 matching of keys to queries) that prioritizes accuracy and multi-query attention (only 1 key shared by all queries) that prioritizes efficient inference (Ainslie et al. (2023)). Additionally, LLaMA 2 7B provided us with the perfect combination of appropriate size and ease of accessibility/use, with the added convenience of discovering Meditron as an OpenSource medically pretrained version. Meditron 7B (Chen et al. (2023)) is a LLaMA 2 model further pretrained on a large corpus of medical documents using the Meditron team's custom Megatron-LLM distributed library. Due to the large number of tokens in the training corpus (nearly 50B), Megatron-LLM utilizes data parallelism by processing different batch subsets across different GPUs, pipeline parallelism by processing different layers on different GPUs, and tensor parallelism by storing different subtensors on different GPUs, all to maximize efficiency. We selected Meditron 7B because the Meditron team reports that their largest model outperforms GPT3.5 and can compete with GPT4, and Meditron 7B in particular on average bests all similarly sized models, so we were curious about its ability to generalize to the very specific set of tasks required by the LLMonFHIR application. As for SUPaHOT, we'll discuss our fine-tuning methodology (hyperparameters, etc) in the Experiment Details section. We'll be referring to these respective models as baseline, Meditron, and SUPaHOT throughout our report. We'd assume that Meditron fine-tuned on each specific task would outperform both Meditron and SUPaHOT, so our experiments instead deal with understanding the differences in performance between a model with "domain-specific knowledge" versus one with "task-specific training".

FHIR documents cannot be directly queried by LLMs as their token sizes are far too large for most models. As a result, after considerable deliberation and discussion with the team from Schmiedmayer et al. (2023), we developed a custom pipeline to mirror the steps taken by the backend of the LLMonFHIR application (visually represented in Figure 1).

- Preprocessing: Our custom script conducts prefiltering in a deterministic manner similar to that of LLMonFHIR, only storing resources with labels that may be potentially relevant ('allergyIntolerance', 'Condition', 'Encounter', 'Immunization', 'MedicationRequest', 'Observation', 'Procedure') and converting each JSON snippet for a given resource (thousands of characters including many codes and other information that isn't immediately interpretable) into a simple resource label in natural language form (e.g. Condition Cardiac Arrest 06-19-2018).
- Task 1 (Filtering): A binary classification task where the model, presented with a patient query and a potential resource label, is prompted to return True or False for whether that resource could contain relevant information for answering the patient's query.
- Task 2 (Summarization): A summarization task where the model, presented with a FHIR Resource snippet in its raw JSON format (corresponding to a resource determined to be relevant by Task 1), is prompted to provide a short summary of the relevant resource.
- Task 3 (Response): A combined QnA and summarization task where the model, presented with a list of summaries of relevant resources from Task 2, is prompted to answer the patient's query using the provided information.

Breaking down the process of FHIR querying into the aforementioned three tasks circumvents the issue of token length. We initially hypothesized that SUPaHOT's task-specific fine-tuning would yield similar competency to the oracle output, or at least outperform the baseline and Meditron. We thus finetuned it on each task separately and in isolation, allowing us to determine where in the pipeline that the model might have a lapse in performance and so that we may potentially address weaknesses with further fine-tuning. In addition, for both Meditron and SUPaHOT, we address the challenge of inconsistency of responses outlined by Schmiedmayer et al. (2023) by decreasing the temperature of the model and giving specific instructions in our system prompts and conversation templates.

(Note: Schmiedmayer et al. (2023) uses GPT4. However, using GPT4 would be expensive for this project, and we believe the gap in model size between the 7B parameter models and GPT3.5 Turbo still demonstrates the feasibility of utilizing smaller models for the specified tasks, while the creation of training data from GPT3.5 Turbo demonstrates the ability of larger cloud language models to assist with fine-tuning of smaller models.)

The architecture of these various tasks is done to circumvent the problem of token size as well as decrease the chances of irrelevant information being used to answer the query. In breaking up the architecture by each of these tasks, we can understand where in the pipeline the accuracy of our models lapse and continue to fine-tune accordingly. Furthermore, in our design, each task is evaluated independent of each other. For instance, if a model poorly chooses relevant resources in Task 1 for a particular query, this will not affect its evaluation downstream in Tasks 2 or 3 for that particular query. In doing this, we ensure each operation is independently operating as intended.

6 Experiments

6.1 Data

Leveraging 6 mock patient FHIR data JSON files generated using Synthea (Jason Walonoski (2016)), we crafted (as mentioned before) our own custom data preprocessing and generation pipeline to mirror the backend of the LLMonFHIR architecture, starting with the initial filtering and cleaning of the FHIR resources. For each patient, we presented GPT 3.5 with a random subset of 20 FHIR resources from their records and prompted it to generate a query about one or more of them, creating

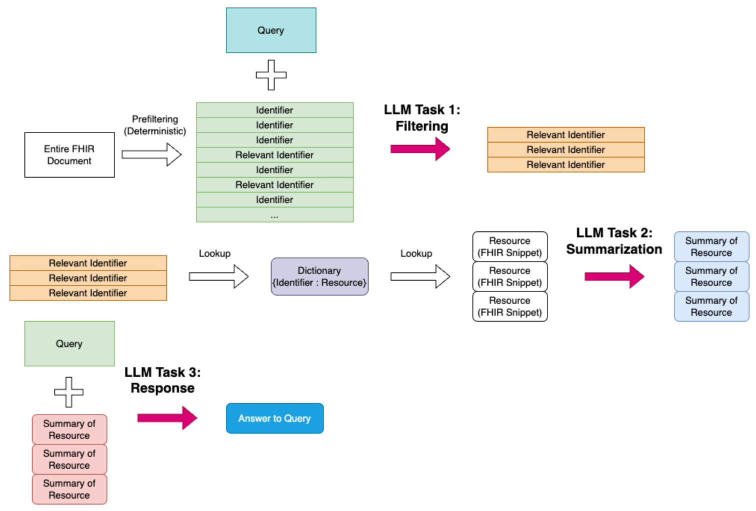

Figure 1: A flow chart of the system pipeline and its corresponding tasks.

50 queries with this method. We experimented with the level of detail until this prompted was generating satisfactory and appropriate queries: "Here is a subset of data points from a patient's medical record: ... Pretend you are a patient curious about an aspect of your medical history. Come up with a query that this patient might have regarding their medical data. At least one or more medical data points from the provided list should be sufficient to answer the query. Make the question realistic, simple, and non-technical. For example, 'What are my current medicines?' or 'When was my last shot?' or 'What were the complications of my last heart procedure?'." See Table 1 for examples of generated queries. We used these 300 queries (and 20 associated resources for each) as the 6000 example prompts for Task 1. For Task 2, our dataset consisted of the output of our oracle on Task 1, meaning for each query, we had the list of resources from the initial 20 that GPT 3.5 had determined to be potentially relevant along with the corresponding resource in its original JSON format. Finally, for Task 3, we used the oracle output from Task 2 along with the corresponding initial query to have GPT3.5 generate final answers, leading to a dataset of 300 query-response pairs. For all datasets, we used an 80-10-10 split for training-validation-test. Using our constructed datasets, we formatted a

ID	**Question**
Q1	"When was my last prescription for tacrolimus filled?"
Q2	"What blood clotting medications am I on?"
Q3	"What are my current medical conditions related to my heart surgery?"
Q4	"When was my last flu shot?"
Q5	"When was the last time I had a general examination?"

Table 1: Examples of Queries from our Training and Evaluation sets

subset into small finetuning datasets for each task with the proper LLaMA 2 prompt formatting. For Task 1, we used 400 query-resource pairs (from our test and validation sets) with the corresponding True or False label, splitting our dataset into 100 examples of True and 300 examples of False. The split was intended to mirror the trends we observed in the selectiveness of our oracle's outputs on this task, and we opted for a smaller dataset due to the relative simplicity of the classification task. For Task 2, we used 900 example JSON FHIR snippets and corresponding summaries, expanding the dataset size to help SUPaHOT gain comfortability interacting with the data in the raw JSON format. Finally, for Task 3, we used the 240 training queries and their corresponding summaries from the oracle.

6.2 Evaluation Methods

For all models and tasks, we used our oracle as a reference point upon which we computed our evaluation metrics. As Task 1 is a simple binary classification task (whether the resource is relevant to the query), we used precision, recall, and F1 scores by comparing the list of resources determined to be relevant by a given model to that of the oracle. Tasks 2 and 3 are both summarization tasks, so we used more traditional text generation evaluation methods. To robustly capture syntactic overlap, we used both BLEU scores from nltk.translate and ROUGE scores from PyRouge. Though BLEU scores are traditionally used for machine translation tasks, conciseness of response is critical to our tasks. For Task 2, brief summaries are much easier to use as inputs for the next step, and for Task 3, brevity in response can be instrumental for accessibility in terms of medical literacy. Since BLEU score penalizes outputs that are too long compared to the reference outputs, we decided to include average BLEU score across all sentences in the test set. For ROUGE, we used ROUGE-2 and ROUGE-L to look at precision, recall, and F1 scores for bigrams and longest common subsequences in particular. We believe considering bigrams is an effective balance between the granular nature of unigrams and the ability of longer n-grams (e.g. 4-grams) to also capture cohesiveness/fluency. As for longest common subsequence, since ROUGE-L is agnostic of word order, it balances examining syntactic and semantic overlap. Perhaps most importantly, we also utilized BERTScore (Zhang et al. (2020)) to more accutely assess semantic overlap as we believe that's most relevant for determining effective summarization. Due to the specific nature of our tasks, we value precision slightly more heavily in some and recall slightly more heavily in others (to be further discussed in the Results section), but for simplicity, we'll stick with F1 scores where relevant in the aforementioned metrics. In future work, determining distinct β values for each task to contextually reflect the relative importance of precision vs recall in our holistic $F\beta$ scores would be ideal.

6.3 Experimental Details

Repurposing a colab notebook to be able to train on Google colab GPUs Labonne (n/a), we finetuned the baseline LLaMA 2 7B Chat on the fine-tuning datasets for each of our 3 tasks. We were curious about whether our design choices in our datasets would reflect in the performance of our finetuned models, so for all 3 tasks we kept hyperparameters and training configuration constant with the following values:

- Epochs = 1 - We were using relatively small datasets, so we didn't want to train for multiple epochs to limit the model's ability to memorize specific patterns, preventing overfitting.
- Learning rate = $2 * 10^{-4}$ - Low to gradually approach minima, preventing overfitting
- Optimizer = AdamW - Leverages momentum, adaptive learning rates, and weight decay
- Weight decay = 0.001 - Lower to prevent over-reliance on specific features in dataset, again preventing overfitting
- Learning rate scheduler = cosine - to promote stable training and convergence by exploring more quickly early on and adapting learning rate throughout the training process

Additionally, we finetuned using LORA (Low Rank Adaptation) from the PEFT library for parameter-efficient fine-tuning, allowing us to train efficiently by only adapting the parameters of targeted layers such as the feedforward and attention layers.

To be able to interact with our oracle and baseline models, we leveraged OpenAI and TogetherAI APIs respectively. For Meditron, we utilized FastChat, an OpenSource repository for LLM chatbots Zheng et al. (2023), which took a significant amount of time to set up at first but then made it straightforward to prompt Meditron and collect its output. Finally, for SUPaHOT, we ran inference pipelines through Hugging Face's transformers library.

For all 3 tasks, we provided the same basic system prompt ("You are a helpful medical assistant, users ask you questions pertaining to their health care information. You will help and be as concise and clear as possible."), attached a task-specific, and provided the appropriate user prompt with the relevant data. We experimented a bit with our oracle as to the specific wording of the prompts as well as which component we included in the system prompt vs user prompt, and we found the following task-specific system prompts to be the most effective:

- Task 1: "Given a query and a resource from a patient's medical record, your job is to determine if the resource could potentially be relevant to providing an answer to the patient's query about their medical history. Respond only with 'True' if the resource may be relevant, or 'False' if the resource would not be helpful at all in providing the patient an answer to their question."

- Task 2: "Given an excerpt of a JSON object corresponding to a resource from a patient's FHIR medical records, your job is to provide a brief (1 to 2 sentence) natural language summary of the JSON. Don't explicitly mention that it's a JSON."
- Task 3: "You will be given a query from a patient who is inquiring about their medical records and a list of summaries (seperated by commas) of medical resources from the patient's medical record. Answer the patient's query using relevant information from the summaries."

FastChat in particular also had a place to input "conversation templates", allowing us to provide sample prompts and responses for each task.

To address the issue of response inconsistency noted in Schmiedmayer et al. (2023), we lowered the temperature of our models to 0.01 to promote nearly deterministic behavior with the positive side effect of limiting the possibility of hallucination.

6.4 Results

6.4.1 Overview

At the top level, we're most interested in F1 score as a holistic way to capture both precision and recall. As demonstrated by Table 2, after running our evaluation script for each task, we observe that SUPaHOT demonstrated an improved performance compared to both our baseline and Meditron in Tasks 1 and 2. However, while it demonstrated a improvement over the baseline, Meditron still outperformed SUPaHOT for Task 3. We'll now explore these results along with discussing precision and recall scores within the context of each task's role in the overall system behind LLMonFHIR.

Table 2: Overall F1 Comparison

	Baseline	Meditron	SUPaHOT
Task 1 F1	0.33	0.298	0.401
Task 2 ROUGE-2 F1	0.0223	0.0264	0.0333
Task 2 ROUGE-L F1	0.0298	0.0427	0.0441
Task 2 BERTScore	0.709	0.777	0.865
Task 3 ROUGE-2 F1	0.0175	0.0423	0.0228
Task 3 ROUGE-L F1	0.0244	0.0711	0.0314
Task 3 BERTScore	0.7011	0.759	0.747

6.4.2 Task 1

For Task 1, we believe precision is slightly more important as with the goal being to increase accessibility for medical literacy, it's critical that in downstream tasks irrelevant or incorrect information isn't returned to the patient/user. Additionally, we found Task 1 to be a rate limiting step as the LLM has to be prompted about all prefiltered resources (usually 100-300, we artificially restricted to 20) individually to determine relevance, so precision is key to minimize unnecessary work in Task 2. We still value recall as ideally all relevant information should be included in the final output of Task 3, but we believe as a user it's easier to reprompt the LLM to further inquire than to discern between important versus irrelevant information individually. Our results (Table 3) indicate that fine-tuning on Task 1 led to a significant increase in precision while still maintaining moderate recall, perhaps by exposing the LLaMA 2 architecture to the specific format of the FHIR resource labels. However, we believe the decrease in recall stemmed from overfitting to our dataset (as we used a 1:3 ratio of True to False), so in future experiments, we'd use a larger more balanced dataset along with slightly modifying some hyperparameters to prevent this from happening.

6.4.3 Task 2

For Task 2, we believe recall is slightly more important to ensure Task 3 has access to all relevant information from the FHIR resource, and we anticipated that our baseline and Meditron would

Table 3: Task 1 Comparison

Metric	Llama2 7B Base	Meditron	SUPaHOT
Precision	0.217	0.198	0.547
Recall	0.951	0.837	0.444

struggle to process the raw JSON snippet in string form. As expected, task-specific fine-tuning triumphed (see Table 4), with SUPaHOT demonstrating increased performance across all metrics. Most notably, Meditron's performance actually dropped in terms of BERTScore Recall which is arguably the most important metric on this summarization task to ensure all semantic information is included for Task 3, while SUPaHOT increased significantly, perhaps due to its exposure to the specific format of the FHIR snippets during its fine-tuning process.

Table 4: Task 2 Comparison

Metric	Llama2 7B Base	Meditron	SUPaHOT
Avg Sentence BLEU	0.0071	0.0106	0.0112
ROUGE-2 Precision	0.0113	0.0134	0.017
ROUGE-2 Recall	0.8959	0.7716	0.942
ROUGE-L Precision	0.0152	0.0219	0.0225
ROUGE-L Recall	0.977	0.9606	0.9935
BERTScore Precision	0.654	0.783	0.851
BERTScore Recall	0.778	0.775	0.88

6.4.4 Task 3

For Task 3, similar to Task 1, we believe precision is slightly important when providing an answer directly to a patient, as the user can always ask follow up questions but may not be able to discern incorrect information on their own. We observe (as seen in Table 5) that while SUPaHOT demonstrated a boost over the baseline, Meditron just barely wins at this task on precision metrics. As all our models are chat-ready, we believe this is because fine-tuning didn't add much additional information to the baseline except for exposing it to some queries and medical information for the QnA. As a result, the tie-breaker is domain knowledge, with Meditron having a much richer trove of medical expertise to ensure precision in its responses. Additionally, for the sake of accessibility, we mentioned the importance of conciseness in the final answer/summary to promote readability for users, so the brevity penalty aspect of BLEU scores captures how Meditron was best for this purpose as well.

Table 5: Task 3 Comparison

Metric	Llama2 7B Base	Meditron	SUPaHOT
Avg Sentence BLEU	0.0063	0.0185	0.0081
ROUGE-2 Precision	0.0088	0.0218	0.0115
ROUGE-2 Recall	0.915	0.7273	0.9226
ROUGE-L Precision	0.0124	0.037	0.0159
ROUGE-L Recall	0.9756	0.9319	0.9876
BERTScore Precision	0.637	0.788	0.702
BERTScore Recall	0.785	0.743	0.806

7 Analysis

To dive deeper into understanding the various models' behavior on each of the tasks, we note some qualitative observations and examine some examples. First, before diving into the task-specific analysis, we'd like to note the importance of the conversation templates we provided for Meditron. Before them, Meditron would simply spew information about random medical literature, especially early on before we fully fleshed out our prompts.

For Task 1, as further evidence of SUPaHOT's overfitting, we noted that some examples would print the special start and stop tokens as part of the model output. Additionally, we noted all models struggled with determining which resources would be relevant in the context of compound questions such as "When was my last examination and what medication was prescribed to me in 2008?" In these cases, instead of reporting True for resources pertaining to either question, the models would often say False to nearly all. Finally, some errors may be the artifact of the specific formatting of the distilled FHIR resource labels. For example, in queries about past medications or prescriptions, models sometimes struggled to note that the resource type "MedicationRequest" should be relevant. Perhaps slightly adjusting the preprocessing to change those labels to "Medication Request" or simply "Medication" would combat this problem.

For Task 2, when inspecting the summaries from each model, we noted that Meditron would be the most to the point, while SUPaHOT would sometimes continue to ramble even after including all relevant information from the FHIR resource. For example, in a query about height, Meditron simply responded with the patient's height, while SUPaHOT both provided the height and then continued to note "The record is a final observation and was taken during a specific encounter", text that doesn't add to the quality of the summary. Generally speaking, we deemed this acceptable as the added text didn't detract from the quality of the summary either, and we valued recall slightly more than precision on this task.

Finally, for Task 3, we noted that the baseline's responses to the patient's query were often not in a conversational form, first restating the information before then adding "Therefore, the answer to the patient's query is: ..." As noted in Task 2, Meditron was also more concise and to the point in this final QnA task as well, sounding much more conversational and responding to the patient in the second person (e.g. using "you" and "your"). Though SUPaHOT's responses did better than the baseline at incorporating these key characteristics, it was far more inconsistent at doing so than Meditron.

8 Conclusion

Our experiments demonstrate great potential in the value of using smaller specialized models for the 3 tasks performed by GPT4 on the backend of the LLMonFHIR application, indicating that such models could be hosted locally to address concerns regarding patient privacy and security. We'd like to note that though results were promising and exceeded our expectations in many ways, there's still a clear tradeoff in performance and efficiency. Additionally, we note that both pretraining and fine-tuning provide valuable boosts in model performance when applied to the appropriate context (dependent on the nature of the task).

Throughout this process, we increased our knowledge of the underlying architecture of LLaMA 2, the differences between pretraining and fine-tuning along with the effect of different hyperparameters and design choices for each context, and the distinctions between each of our evaluation methods. We also developed a greater understanding of how to intentionally approach complex pipelines where the performance of each step/task deeply influences downstream performance. Through conversations with the team from Schmiedmayer et al. (2023), we dedicated a lot of effort into understanding the application's backend and approximately recreating it in Python. We had to make some key assumptions in the process due to limited compute resources and the timeframe of the project, accounting for some of our limitations such as only examining a subset of resources for Task 1, also incorporating asynchronous processing for efficiency. From a more practical standpoint, this project was an excellent exercise in knowing when to pivot, as we encountered several roadblocks that took up an extensive amount of our time simply trying to set up and interact with each of our models, especially Meditron. We spent a lot of energy trying to finetune Meditron on our specific tasks, though we were unable to satisfactorily do so and include the results in this report. Attempting to use their provided supervised finetuning script at first, we encountered seemingly endless dependency issues as their script relied on several other OpenSource codebases, so we pivoted to drafting our own custom finetuning script during which we encountered lots of dimensionality issues. We had to pivot to focusing on our LLaMA finetuning after the sunk cost.

For future work to address some of our limitations and build on our results, we'd first seek to evaluate a model that's both pretrained and fine-tuned, along with evaluating the performance of these models end-to-end (e.g. using their own outputs from each task as the input to the next rather than using the oracle outputs) to more realistically assess how they'd fare within the context of LLMonFHIR. Additionally, we'd want to explore having a single model fine-tuned on all 3 tasks to see if there's a dip in performance compared to having a team of models specialized for each task. Finally, as mentioned above, we'd hope to explore medium-sized models (whose performance after pretraining and/or fine-tuning could approach GPT4) to be hosted on the cloud but not reliant on a 3rd party API, prioritizing the security of patient data.

References

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. 2023. Gqa: Training generalized multi-query transformer models from multi-head checkpoints.

Zeming Chen, Alejandro Hernández-Cano, Angelika Romanou, Antoine Bonnet, Kyle Matoba, Francesco Salvi, Matteo Pagliardini, Simin Fan, Andreas Köpf, Amirkeivan Mohtashami, Alexandre Sallinen, Alireza Sakhaeirad, Vinitra Swamy, Igor Krawczuk, Deniz Bayazit, Axel Marmet, Syrielle Montariol, Mary-Anne Hartley, Martin Jaggi, and Antoine Bosselut. 2023. Meditron-70b: Scaling medical pretraining for large language models.

Joseph Nichols Andre Quina Chris Moesel Dylan Hall Carlton Duffett Kudakwashe Dube Thomas Gallagher Scott McLachlan Jason Walonoski, Mark Kramer. 2016. Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record. Journal of the American Medical Informatics Association. [Online; accessed 12 Feburary 2024].

Maxime Labonne. n/a. Fine-tune llama 2 in google colab. `https://colab.research.google.com/drive/1PEQyJO1-f6j0S_XJ8DV50NkpzasXkrzd?usp=sharing`.

Paul Schmiedmayer, Vishnu Ravi, and Oliver Aalami. 2023. Stanford llm on fhir. [Online; accessed 12 Feburary 2024].

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena.