

NLP Final Project: LLM Application (llama-2) For Jewish Rabbi Question Answering

Tomer shimshi

tomershimshi@gmail.com

Amit Damri

amitdamri@tau.ac.il

Abstract

In recent years, there has been an increase in the popularity of large language models (LLMs) and their ability to perform tasks that were previously considered beyond the capabilities of computers, such as writing reports or planning trips abroad. In the following paper, we examined the generalization ability of a LLM to solve a very unique and specific task, answering a Jewish Rabbi questions, a domain that most likely it wasn't exposed to during its training processes. We apply and compare two state of the art techniques, fine-tuning an LLM and Retrieval-Augmented Generation (RAG), and use an open source LLM to solve this problem. The reason we choose this specific case is due to its niche field, which is a good way to show that if we succeed on this task we can apply the results we would receive into many other fields that no one tried to apply LLMs on before. In our project, we found that for our specific task the RAG pipeline was proved as more capable than the classic the fine-tuning approach, but over all both gave descent results.

1 Introduction

In the following paper we will try to use an LLM model for solving a specific task, that it has yet to see before, namely we will try to test whether a LLM is capable of answering questions from a very unique domain - Rabbi questions. For that we will first define the basic terms of the specific use case we tested in our project: A Rabbi Question is question asked by a Jewish person to his Rabbi, Which is a Jewish man who studied the "Halacha" laws for many years and is considered a authority in the matter. The Rabbi needs to answer the given question according to the Jewish tradition laws. The questions can be about anything. In the Jewish tradition it is a custom to seek guidance from your Rabbi about any life question you may have. If this project will succeed it can help many remote orthodox Jews that do not have a frequent

access to a nearby Rabbi but do require answers for their immediate questions, and since it is much more scale-able to use a LLM to answer all these questions, rather than a real human Rabbi it can revolutionize the entire Jewish world. We know that LLMs are good with questions answering but it is not so trivial to answer Rabbi questions and not many people are qualified to do so. In order to achieve this goal we will try two different approaches and check which one is better:

1. Model Fine tuning: fine tune an LLM on the specific task of answering Rabbi questions. For that we first fine tuned the model on [Kitzur Shulha Aruch](#) which is a book that summaries most of the Jewish halacka laws and then perform a second fine tuning using a dataset we collected from the "[Ask a Rav](#)" website to try and teach the model how to properly answer this type of questions.
2. Retrieval-Augmented Generation (RAG): We used this approach in order to supply our LLMs the most meaningful part from "[Kitzur Shulha Aruch](#)" so with that as an additional prompt it will be able to answer any question being asked according to the "Halacka" laws specified in the retrieval dataset.

In our project we choose to use "Meta"s "[llama-2 7b](#)" model, which is one of the state of the art open source LLMs.

2 Related work

Ever since Meta released they're open source llama-2 model there has been a lot of people from the [Hugging face community](#)) that used this open source model and fine tuned it on they're own dataset, examples for that can be found in the following guides: [guide No.1](#) or [guide No.2](#). And in general in the recent years fine tuning a LLM has became a lot more common. following that is what made

us decide on fine tuning our llama-2 model to be able to answer Rav questions. Although fine tuning LLMs for specific tasks have become very common in the recent years we have yet to witness a model that was trained for the specific task of answering Rav questions.

In the of Retrieval-Augmented Generation (RAG) case, recently there has been an increase in the popularity of using RAG in order to have an LLM capable of answering questions on areas it have not seen in its initial training. given that it has access to a specific dataset provided by the user. There are many examples and guides on how to use RAG for a specific LLM like for example [RAG guide No.1](#) or [RAG guide No.2](#). even tough that RAG is increasingly more popular these days we have yet to see any example of using a RAG pipeline combined with Jewish laws in order to create a Jewish Rav capable of answering question according to the Jewish laws, so from use case point of view our work has never been tested before .

Table 1: Dataset example

question	answer
question from the site	answer from the site

3 Training and Data collection

3.1 Data collection and prompt preparation

Since all of the Rav answers are based on the Jewish laws (the halcka) we couldn't just use a question and answer dataset (as we initially taught) so we understand that the first step in creating our dataset booth for RAG and for fine tuning a pre-trained model is to first have a dataset containing the majority of the Jewish laws for that we have collected the English version of the [Kitzur Shulhan Aruch](#) book since it contains most of the Jewish laws and customs while still being a relatively short text so that we can perform RAG fine tuning on it. The second step for the LLM fine tuning to answer Rav question is to find the relevant dataset containing real Rav questions and answers and since that there is no known dataset for Rav questions we had to create a dataset ourselves. For this task we had to find a relevant site to perform data scraping from, and since most the open source models are best suited for the English language rather than the Hebrew language we performed the data scraping from an

English site namely [Ask a Rav](#) site which is full of real questions and answers in English. From this site we comprised our dataset of real Rav Questions and answers as can be seen on table 1. Since some of the questions answers on the site were asked using some Hebrew words inside them (usually old Hebrew words) and since we know that the model do not handle Hebrew well we have decided to remove all these questions/ answers from the dataset, leaving us with a dataset of 1160 only English Q&A. Here is an example of a one question and answer used in our dataset.

Question:

May I put a frozen bag of vegetables etc. on my body on Shabbos as it might melt from my body heat? If the answer is yes, may I cool myself off with these packets of frozen milk knowing and benefiting from the fact that they will melt? If neither of these are acceptable, may I use a cold cloth placed in the freezer?

Answer:

1) One may handle ice packs to cool off with, since only a small amount of water is melted, there is no intention of melting them and the melted water is going to waste. Melting ice is problematic when one actively melts it, or even when no action is taken if this his intention, for example if wants to melt it save the melted water – which is not applicable in this case. However, frozen food items like meat and vegetables should not be used since they are muktzah.

2) Since you want to melt the milk, you should not purposely use it to cool off so as to proactively melt it.

3) Cold towels that are not wet and will not become squeezed is permissible to use.

After we finished the data collection we needed to create the model prompt, in order for it to fit to the task at hand we created the following prompt in order to feed it into the the model :

you are a jewish Rav, please answer the following question according to the Halakha (Jewish law) .

Question:

Answer:

EOS TOKEN

Where the EOS TOKEN stands for the token which the tokenizer was trained on to mark the end of sentence. After finishing the data collection and prompt creation we loaded the model and started

the model training.

3.2 Model loading

The model which have tested in this project is [meta llama-2 7B](#) (where the B stands for the models number of parameters in billion) As such it is a vary large model that consume a lot of RAM on the GPU in order to still be able to load and fine tune them on the provided GPU we have used [bitsandbytes](#) python library which essentially enables accessible large language models via k-bit quantization for PyTorch. bitsandbytes provides three main features for dramatically reducing memory consumption for inference and training:

1. 8-bit optimizers uses block-wise quantization to maintain 32-bit performance at a small fraction of the memory cost.
2. LLM.Int() or 8-bit quantization enables large language model inference with only half the required memory and without any performance degradation. This method is based on vector-wise quantization to quantize most features to 8-bits and separately treating outliers with 16-bit matrix multiplication.
3. QLoRA or 4-bit quantization enables large language model training with several memory-saving techniques that don't compromise performance. This method quantizes a model to 4-bits and inserts a small set of trainable low-rank adaptation (LoRA) weights to allow training.

We loaded the model using 4 bit quantization which enabled us to load the quantized model and fine tune just small set of trainable low-rank adaptation ([LoRA](#)) weights. by doing so we manged to speed up each training step and use a lesser GPU in order to load the model. We used the following parameters for the model loading:

```
BitsAndBytesConfig(  
    load_in_4bit=True,  
    bnb_4bit_quant_type="nf4",  
    bnb_4bit_compute_dtype=torch.float16,  
    bnb_4bit_use_double_quant=True,  
)
```

3.3 Fine Tuning

As explained in the previous subsection after defining the Quantization profile for the model loading we defined the trainable low-rank adaptation

([LoRA](#)) weights. The advantages of using [LoRA](#) are as follows: LoRA enhances the training and adaptation efficiency of large language models like OpenAI's GPT-3 and Meta's LLaMA. Traditional fine-tuning methods require updating all model parameters, which is computationally intensive. LoRA, instead, introduces low-rank matrices that only modify a subset of the original model's weights. These matrices are small compared to the full set of parameters, enabling more efficient updates. The approach focuses on altering the weight matrices in the transformer layers of the model, specifically targeting the most impact-full parameters. This selective updating streamlines the adaptation process, making it significantly quicker and more efficient. It allows the model to adapt to new tasks or datasets without the need to extensively retrain the entire model. By doing so we were able to fine our llama model using a single GPU and in a matter of a few hours (as oppose to model re-training which according to meta the training of llama-2 7B took [184320 GPU hours](#)) We used the following QLoRA values:

```
LoraConfig(  
    lora_alpha = 32,  
    lora_dropout=0.1,  
    r=8,  
    bias="none",  
    task_type="CAUSAL_LM",  
)
```

Where the main parameters shown above stanfs for:

- r: the rank of the update matrices, expressed in int. Lower rank results in smaller update matrices with fewer trainable parameters.
- lora_alpha: LoRA scaling factor.
- bias: Specifies if the bias parameters should be trained. Can be 'none', 'all' or 'lora_only'.
- lora_dropout (float) — The dropout probability for Lora layers.

The full documentation can be found [here](#).

For the model fine tuning we have split the fine tuning into two. In the first part of the fine tuning we just fine tuned the model on the Kitzur Sulhan Aruch dataset in order to provide the model context regarding the Jewish laws. in this part of the training we used the full Kitzur Sulhan Aruch dataset

without splitting it into a train and test dataset since the model needed to train and see the full book for it to know the Jewish laws, in this part of the train we just fed the raw data into the model without any other instructions to it, we ran this training loop for 2.5 epochs (all the training code is attached to the submission of this paper). Lastly we performed a second fine tuning step this time with the question and answers dataset along with the previously mentioned model prompt since we have used HuggingFace SFTTrainer (Supervised Fine Tuning Trainer) in order to perform our model's fine tuning this we wanted to take leverage of its capability to load the best model at the end (based on its loss on the test dataset) for this reason we have split our Rav questions and answers dataset into 3 groups as follows:

1. train dataset: which is 70% of the dataset, 812 questions and answers, that the model has trained on.
2. evaluation datasetd: which is 15% of the dataset, 174 questions and answers, that model was evaluated on during training.
3. validation datasetd: which is 15% of the , 174 questions and answers, that model was validate on in the finale model assement step

As generally recommended for the training of a deep learning model, like for example [in this guide](#)

We have fine tuned the model on the Q&A dataset for another 2 epochs to try and teach it how to handle Rav Questions Attached is en example of answer provided by our fine tuned llama2 model after this stage:

```
### Question:
Can I eat pork?
### Answer
No.
```

3.4 Retrieval-Augmented Generation (RAG)

For our experiment we have also tested a different approach which is raising in popularity lately which is Retrieval-Augmented Generation (RAG). Which is a technique for enhancing the accuracy and reliability of generative AI models with facts fetched from external sources. as explained [in this site](#) and as can be seen on fig 1 In our case we have taken the Kitzur Shulahn Aruch dataset that we have previously used to fine tune out LLM and this time we provided it into the model as context

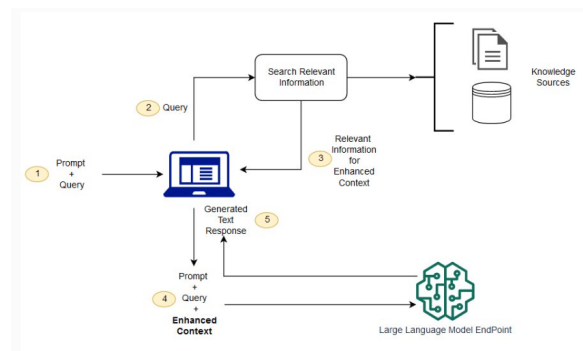


Figure 1: RAG flow

so that it will answer the given question according to it. For this we have used the [langchain](#) python library as explained [in this guide](#) . For the retrieval part we have used the all-mpnet-base-v2 which is a sentence-transformers model that maps sentences and paragraphs to a 768 dimensional dense vector space as explained [in the model page](#) on hugging face . Example of an output given by the model using RAG:

Question:

Can I eat pork?

Answer

According too Jewish Law(Halka),pigs ares considered non kosher because they dont chew cud nor partake lymanica which means its forbidden for Jews to consume them as per bibel injucition Levit-suchas chapter27 verse3-5 furthermore Talmudoy-ersus passage Brachaot Chapter4b rule number6 states Its prohibited totouch any fleischof swine even indirectly thus making consumption impossible under Halkain laws

4 Model Eveluation

4.1 Evaluation pipeline

For the validation of the model we compared booth the RAG and fine tuned model on the validation dataset which is comprised of 15% of the question and answer dataset taken from the [Ask a Rav](#) web site. In order to perform a full evaluation of the answers received from our LLM we used the deepeval [python package](#). In order to separate the models run (RAG / fine-tuned) we performed a preliminary step to the evaluation full. we ran booth versions of the model on the validation dataset to create a csv containing question, model output, expected output as can be seen on [Table 2](#) that way we completely separated the model outputs generation pipeline from the model validation pipeline. For the test

question	model output	expected output
question from the validation dataset	actual model output	real Rav answer to input question

Table 2: Example of the validation csv

that we have decided to use 2 methods for model evaluation:

1. **Answer Relevancy:** The answer relevancy metric measures the quality of our generator by evaluating how relevant the actual_output of our LLM application is compared to the provided input. deepeval's answer relevancy metric is a self-explaining LLM-Eval, meaning it outputs a reason for its metric score. The Answer Relevancy Metric score is calculated according to the following equation:

$$AnswerRelevancy = \frac{NRS}{TNS}$$

Where NRS stands for Number of Relevant Statements and TNS stands for Total Number of Statements. The AnswerRelevancyMetric first uses an LLM to extract all statements made in the actual_output, before using the same LLM to classify whether each statement is relevant to the input.

2. **G-Eval:** G-Eval is a framework that uses LLMs with chain-of-thoughts (CoT) to evaluate LLM outputs based on ANY custom criteria. The G-Eval metric is the most versatile type of metric deepeval has to offer, and is capable of evaluating almost any use case with human-like accuracy. More specifically we decided to test each models answer correctness when compared to the expected output. We have used the following criteria for this task: criteria="Determine whether the actual output is factually correct based on the expected output."

Since the open source e [mistral_7b model](#) was insufficient model to perform our deep_eval test with it we had to use the paid GPT3.5-Turbo model in order to perform our model evaluation.

Table 3: Evaluation Results

model	Relevancy score	Correctness score
Fine-Tuned	58.1%	22%
RAG	69.9%	26.2%

4.2 Evaluation Results

The json and yaml files containing all the evaluation results are attached to this report

4.2.1 Fine tuned model Evaluation Results

For the fine tuned model we got a accuracy relevancy score of 58.1% and answer correctness score of 22%. Meaning that our fine tuned model did not managed to fully simulate a Rav's answers to the given input question, that is to be expected since it no trivial task to answer these types of questions.

We will now show an example of the deep_eval outputs score for each of the metrics: For the randomly selected test case number 147 we got:

Model input :

I know that one cannot leave the room in middle of Torah reading. How about Haftorah or Megillah (he was already Yotze)?

Model output:

You may do so.

Expected output:

During Haftorah one should ideally not leave, unless absolutely necessary, but it's not forbidden, unless there will not be ten people when he leaves. During Megillah reading, one may leave if there are ten people there, but there is room to be stringent.

For the Answer relevancy score we got a score of 1.0 due to the following reason:

The score is 1.00 because the response directly addresses and provides a relevant answer to the question asked in the input. Great job!

For the Answer Correctness we got a score of 0.195 with the following reason:

The actual output does not align with the expected output in terms of accuracy and completeness.

4.2.2 RAG Evaluation Results

For RAG model we got a accuracy relevancy mean score of 69.9% and answer correctness score

of 26.2% we will now show an example of the deep_eval outputs score for each of the metrics:

For the randomly selected test case number 147 we got: The input and expected is the same as shown in the previous section, the model actual output is:

According halakhic sources, it seems clear than any time during which torat hatorah is not ongoing and therefore leaving after haftora/megila would not violate this prohibition since its no longer "in midst" of tehorah hatoreah therefore i do know

For the Answer relevancy score we got a score of 1.0 due to the following reason:

Great job! The score is 1.00 because the response directly addresses and provides relevant information to the specific questions asked in the input. Keep up the good work! For the Answer Correctness we got a score of 0.254 with the following reason:

The text does not align with the expected input and output comparison and lacks clarity.

5 Conclusions

In conclusion we can see that for this specific case of answering Rav questions it seems that the RAG provides better answers when compared to the fine tuned model. This is surprising to us since we know that it is not enough to just know all the Jewish laws in order to answer a Rav question, some of them can be just related to the Jewish way of life. Which made us initially think that the fine tuned will be better adapted to these types of questions, but since our dataset is quite small (1160 A&A in total) it is possible that there were not many questions of this sort in the validation dataset. We are very pleased with the results we have got since it is not a trivial deal to answer Rav question, since for a human to be able to answer such questions in a relevant way means that this person has to study the Jewish laws and customs for 5-6 years [as explained here](#) and we managed to get decent results with just a few weeks and with little compute power and a relatively small dataset to train/validate on. So we believe that if we had access to a stronger compute power we would have gotten even better results.

We might not have built a perfect Rav to consult with but we believe that for an Orthodox Jew who does not have a Rav to consult with our RAG model might be good enough to answer his im-

mediate relatively easy questions at least until he gets a full validated answer from an official Rav (over the internet or by other means of combination to the Rav). We have uploaded the model to [hugging face](#) and uploaded a file attached to this report called finale_project_Rav_talk.py please do try to ask questions to the Rav and experience our end results. On top of that we have also shared the RAG_llama-2_Rav_talk.py script which enables you to also test our RAG pipeline. And we do encourage to try and use them, we tested it ourselves and both models managed to give a pretty good answer to every question that we managed to come up with. Our experiment can be applied in a similar way on many different specific tasks that one can think of and might need to build a specific chatbot for. We really enjoyed making this project and want to give a special thanks to Maor Ivgy for all the support in making this project.