Language Models for Tax Assistance: An In-Context Learning and Retrieval Augmented Generation Approach

Mahnoor Aftab (maftab)
Luddy School of Informatics
Indiana University

Sadie Askari (saaskari)
Luddy School of Informatics
Indiana University

Ben Lazarine (belazar)
Kelley School of Business
Indiana University

Abdul Rehman (abrehman)
Luddy School of Informatics
Indiana University

Abstract

There are many compelling NLP contexts, such as tax information, that may benefit from the application of language models. However, one of the chief challenges that remains for such specialized contexts is the need for a large amount of well-labeled training data. Developing such data sets can be challenging and costly. Additionally, it may be difficult to actually significantly update the parameters in a Large Language Model (LLM) via fine-tuning alone. In this paper, we illustrate two methods to address the limitations associated with fine-tuning LLMs for specific contexts. First, we leverage In-Context Learning (ICL) to augment the output of a pre-trained LLM and evaluate its performance on tax related questions. We also demonstrate how LLM limitations can be overcome with Retrieval Augmented Generation (RAG), developing a pipeline that allows pre-trained LLMs to enhance their responses with information provided to the LLM as a specialized text corpus. We evaluate our two LLM augmentations with three learning tasks: statutory reasoning, numerical reasoning, and fact retrieval. We find that RAG out-performs ICL; however, RAG is not suited for reasoning tasks while ICL is.

`Github repo:` https://github.com/abrehman94/llama_tax_attorney

1 Introduction

Filing taxes is an annual point of stress for millions across the U.S. Additionally, there are many tax return mistakes that cost both taxpayers and the U.S. government. While there are applications that can help, such as TurboTax, the process remains confusing, stressful, and difficult for many, as many of these systems still require users to have some baseline knowledge of taxes to know where to find required information. Tax payers without such knowledge benefit greatly through the use of tax experts to assist in the filing of their taxes. However, using professional tax assistance can be expensive and unfeasible for many. Automated expert systems can help address this and make assistance in areas where an extremely high amount of domain knowledge is needed, such as tax payment, more readily accessible. The recent emergence of Large-Language Models (LLMs) has shown promise as a potential solution to the development of such expert systems (Floridi and Chriatti 2020). In particular, tax payers may benefit greatly from a chat-agent that can answer questions specific to their financial situation. While LLMs are well suited to be an underlying technology behind chat agents, it is often necessary to fine-tune them for specific use-cases (Beltagy et al. 2019).

Preprint. Under review.

Language models are probabilistic models of natural language that aim to model how people speak, such that they can understand and produce natural language sequences (Bengio et al. 2000). In the last few years, language models have begun to perform well enough to be embedded in commercial products and applications and have begun to provide significant economic value globally (Ray 2023). This recent progress of LLMs has been fueled by both massive amounts of training data, as well as advancements in deep learning, particularly with the introduction of the Transformer architecture (Vaswani et al. 2017).

Extant LLMs primarily leverage the Transformer architecture. However, there remain some limitations when attempting to train Transformer-based LLMs for highly specific tasks. One limitation is the need for a large amount of training data. BERT (Devlin et al. 2018) demonstrated that models can learn from unlabeled training data in an unsupervised fashion; however, there still remains a need for enough context specific data. This can be particularly difficult in contexts such as tax assistance. Additionally, with many LLMs having upwards of billions of parameters trained on billions to trillions of data points, it can be difficult to effectively update enough parameters during fine-tuning to store additional information. Therefore, we seek to compare two promising methods that aim to address these limitations: In-context Learning (ICL) and Retrieval Augmented Generation (RAG). Specifically, we evaluate their applicability to the context of tax assistance.

In the remainder of this paper, we (1) review related work, (2) propose our research design, (3) present our results and discussion, and (4) conclude our work.

2 Related Work

2.1 The Transformer Model

The Transformer model (Vaswani et al. 2017) marked a significant shift in the architecture of NLP models from complex recurrent neural networks (RNNs) and convolutional neural networks (CNNs) to a simpler model based entirely on attention mechanisms. The Transformer model leverages attention mechanisms to weigh different words in a sequence for improved contextual understanding. The model comprises encoder and decoder blocks, allowing it to perform tasks like machine translation and text generation. The model replaces recurrent and convolutional layers previously used in sequential learning tasks with self-attention layers to helps capture relationships between words in a sequence. Additionally, the model incorporates positional encoding which assists in maintaining the order of words. We review the details of the Transformer model further in the following sections.

2.1.1 Attention Mechanism

The attention mechanism is at the core of the Transformer. It allows the model to weigh different parts of the input sequence when processing each word, giving more emphasis to relevant words for understanding context. Instead of processing words in a sequence, as seen in traditional sequential models like RNNs, the Transformer considers all words simultaneously through attention scores. These scores determine the importance of each word concerning others in the sequence, enabling the model to capture long-range dependencies effectively.

2.1.2 Encoder-Decoder Architecture

The Transformer model consists of two main components: the encoder and the decoder. The encoder processes the input sequence, encoding information into a fixed-dimensional space through multiple layers of self-attention and feed-forward neural networks. This encoded representation is then passed to the decoder, which generates the output sequence by attending to the encoded information and producing one word at a time. This architecture is especially useful for tasks like machine translation, where the encoder understands the input language, and the decoder generates the translated output.

2.1.3 Self-Attention

Self-attention is a fundamental concept in the Transformer model that enables it to capture relationships between different words in a sequence. In self-attention, each word in the input sequence attends to all other words to determine its importance concerning them. This mechanism allows the model to weigh words based on their contextual relevance, considering the entire context of the sequence simultaneously. Self-attention is computed by creating queries, keys, and values for each word and calculating attention scores, which are used to produce weighted representations.

2.1.4 Positional Encoding

Unlike RNNs or CNNs, the Transformer lacks inherent sequential information. To address this, positional encoding is introduced. Since the model processes words in parallel, it needs a way to understand the order of words in a sequence. Positional encoding provides the Transformer with information about the position of words by adding fixed-length vectors to word embeddings. These vectors convey positional information and help the model differentiate between words based on their positions in the sequence.

2.2 Transformer-based Language Models

Since the introduction of Transformers, a number of Transformer-based LLMs have emerged including BERT (Devlin et al. 2018), GPT-3, and LLaMA (Touvron et al. 2023). BERT was designed to pre-train deep bidirectional representations from unlabeled text. BERT is able to operate on unlabeled text by jointly conditioning on both left and right contexts in all layers. Devlin et al. published BERT-base with 110 million parameters and BERT-large with 340 million parameters. Additionally, BERT is able to be fine-tuned with the addition of a single additional output layer. GPT-3 is a generative pre-trained Transformer released by OpenAI in 2020. GPT-3 differs from the traditional Transformer architecture in that it is a decoder-only model. GPT-3 was trained with 175 billion parameters, however the model is closed-source. In contrast, LLaMA was released in 2023 by Meta as an open-sourced LLM. The models released by Meta range from 7 to 65 billion parameters, with LLaMA-13B outperforming GPT-3 on most benchmarks. LLaMA outperforms previous LLMs by training fewer parameters on more tokens than what is typically used. For the remainder of this paper we will focus on LLaMA due to its open-source nature, use of publicly available training data, and state-of-the-art performance.

LLaMA was trained on 4.5TB of data from 7 different data sources including CommonCrawl, C4, GitHub, Wikipedia, Gutenberg and Books3, ArXiv, and Stack Exchange. There is no clear component of LLaMA's training that would make it tailored to highly specific contexts such as tax assistance. LLaMA differs from the traditional Transformer architecture by leveraging various improvements from previously proposed LLMs such as pre-normalization (Zhang and Sennrich 2019), the SwiGLU activation function (Shazeer 2020), and rotary positional encodings (Su et al. 2021).

2.3 In-Context Learning

ICL is an emerging paradigm being used for NLP tasks in which LLMs make predictions only based on contexts augmented with a few examples (Dong et al. 2023). To perform ICL, an LLM is provided with a few examples to form a demonstration context. During ICL, the demonstration context and query are provided to the LLM as a single prompt to receive a contextualized prediction. ICL has demonstrated its ability to improve LLM communication and incorporate specific human knowledge, making it well suited for teaching LLMs to perform complex tasks such as tax assistance. One of the most significant advantages of ICL is that it eliminates the need for additional training data, or updating the LLMs parameters at all. One limitation of ICL; however, is that it is highly sensitive to the prompting template used, and can only ingest a certain amount of new context, limited by the maximum prompt size of a given model.

2.4 Retrieval Augmented Generation

To overcome the limited context that can be provided to an LLM via ICL, RAG has emerged as an additional technique for updating how an LLM operates. RAG approaches augment the parametric memory of LLMs with a non-parametric store of additional information that can be accessed by the model (Lewis et al. 2020), fusing emerging deep learning technology with traditional retrieval technology (Li et al. 2022). A major advantage of RAG is that you can easily construct and scale document sets for specific contexts that you would like a pre-trained LLM to retrieve from, making it extremely well suited to contexts such as tax assistance where there are hundreds of tax forms and instructions that are tedious for people to understand themselves.

3 Research Design

3.1 Research Testbed

Our research test bed is comprised of two components: the SARA data set (Holzenberger et al. 2020) and a collection of IRS tax forms, instructions, and documents. The SARA data set consists of rules extracted from the statutes of the US Internal Revenue Code (IRC) along with a set of natural language questions. These questions can only be answered accurately by referring to the provided rules. The IRC contains rules and definitions related to the imposition and calculation of taxes, organized into sections that define terms and include general rules along with exceptions.

The data set consists of 256 training examples and 120 test examples. It has two types of examples: statutory reasoning and numerical reasoning.

Statutory reasoning: These examples consists of a context describing a scenario, followed by a statement which is either entailed or contradicted based on the premise and applicable tax laws. An example is demonstrated below.

Context: Alice's income in 2015 is $100000. She gets one exemption of $2000 for the year 2015 under section 151(c). Alice is not married.

Statement: Alice's total exemption for 2015 under section 151(a) is equal to $6000.

Answer: Contradiction

Numerical reasoning: These examples also consist of a context followed by a question. In these examples, the question asks about applicable taxes in the given context. Each example has singular numerical answer.

Context: In 2017 Alice was paid $75845. Alice has a son Bob. From September 1st 2015 to November 3rd 2019 Alice and Bob lived in the same home which Alice maintained. In 2017 Alice takes the standard deduction.

Statement: How much tax does Alice have to pay in 2017?

Answer: $15037

In addition to the SARA data set, we also collected a set of 2,412 IRS forms, instructions and news releases. To collect this data, we visited irs.gov, which provides a directory of every PDF that is hosted on the site. We collected forms, which are specific documents that must be submitted by various classifications of tax payers, instructions, which help tax payers understand how to use forms, and news releases, which contain up to date information about any extenuating circumstances that may impact tax payers.

3.2 LLaMA Implementation

We implemented our baseline chat bot using LLaMA-2 model with 7 billion parameters. It has been tested on a Tesla P100-PCIE-16GB gpu. Inference for batchsize of 8 with max sequence length of 4096 tokens requires approximately 15 Giga Bytes of memory. We then evaluate LLaMA-2's ability to perform ICL.

Our chat bot system's interface is terminal based. Figure-1 shows a screen shot of the interface. The first step to interacting with the chat bot is to specify the type of desired output a system tag. The output can either be a reasoning based response or factual retrieval. We illustrate this process in the following set of prompt examples with one example for each output type.

system: Please answer by performing statutory reasoning step by step to determine if last statement in query is entailed or contradicted according to the statutes of tax law.

system: Please answer by performing numerical analysis to calculate tax applicable to the given context. Provide a single numerical answer.

system: Give factual information about tax laws.

Confidence Score: To evaluate the model's confidence in its response, we implement two techniques that do not require model logits. The first strategy is to elicit the model to give a confidence estimate itself in its responses. The second strategy is a consistency based approach.

```
========================================


system: Please answer by performing statutory reasoning step by step to determine if last statement in query is
entailed or contradicted according to the statutes of tax law.
Understood!

========================================


Alice’s income in 2015 is $100000. She gets one exemption of $2000 for the year 2015 under section 151(c). Alice
 is not married. Alice’s total exemption for 2015 under section 151(a) is equal to $6000.
> Assistant:  Sure, I'd be happy to help you with that! Here's the step-by-step statutory reasoning to determine
 if the last statement in the query is entailed or contradicted according to the statutes of tax law:
Step 1: Identify the statutes relevant to the query
Relevant statutes:
* Internal Revenue Code (IRC) section 151(a)
Step 2: Determine the facts stated in the query
Facts:
* Alice's income in 2015 is $100,000.
* Alice gets one exemption of $2,000 for the year 2015 under section 151(c).
* Alice is not married.
Step 3: Apply the statutes to the facts
IRC section 151(a) provides that the total exemption for a taxpayer who is not married is $6,000.
Facts: Alice's income in 2015 is $100,000.
Conclusion:
Based on the relevant statutes and the facts provided, we can conclude that Alice's total exemption for 2015 is
$6,000, which is equal to the amount provided under IRC section 151(a).
Therefore, the last statement in the query is ENTAILED.

========================================
```

Figure 1: Chat bot Interface

The consistency based approach works by varying the temperature of the prompt to generate multiple responses for the same query. Then the score can be calculated based on most similar responses.

3.3 Retrieval Augmented Generation Implementation with LLaMA and LangChain

In addition to evaluating LLaMA-2 7B's performance in ICL, we also evaluated its ability to be integrated into a RAG system. In this implementation, we leverage LangChain (Topsakal et al. 2023), a framework that assists in combining multiple components into a single NLP system, or chat bot. Additionally, LangChain makes it feasible to index a corpus such that an LLM can perform document retrieval to augment its responses. In our implementation we also used Facebook AI Similarity Search as the retrieval mechanism, giving our chat bot a means to perform embedding similarity analysis between the prompt and corpus.

4 Results and Discussion

We evaluate our chat bot's ICL performance with test examples from the SARA data set. In the first task of statutory reasoning, the model performs with an accuracy of 50%. We find that this performance is due to the model predicting that a prompt is entailed with tax instructions 100% of the time with an evenly split data set. In the second task of numerical reasoning, the model performs with an accuracy of 40%, however this is a more difficult task as the model needs to get its predicted tax amount due exactly right. We attempted to augment our model evaluation with the output of a confidence score; however, we found that confidence score provided by the model is always 100%.

To evaluate our chat bot after RAG, we pass in the following prompt:

"How do I file my taxes?"

to both the baseline LLaMA-2 and to RAG LLaMA-2 that has access to our collection of IRS documents. The output of LLaMA-2 is shown in Figure-2 and the output of RAG LLaMA-2 is shown in Figure-3

As the responses from the two models indicate, our RAG LLaMA is able to give a much more specific response, referencing a tax form included in the corpus provided to it. We evaluated RAG-LLaMA with a small corpus of 9 documents, which is why the response references Form 8689. An additional benefit of RAG LLaMA is that it can also provide the document within the corpus that was referenced to generate the response to the user so they can read more directly from the augmenting text themselves.

```
 Unterscheidung between a "tax return" and an "income tax return". In the United States, the term "taxes" generally refers to the
 federal income tax, which is a tax on income earned by individuals and businesses. The process of filing taxes involves
 preparing and submitting a tax return to the relevant tax authority, such as the Internal Revenue Service (IRS) for federal
 taxes or a state or local tax authority for state or local taxes.

There are several steps involved in filing taxes:

1. Gather all necessary documents: This includes W-2 forms from your employer(s), 1099 forms for any self-employment income,
interest statements from banks and other financial institutions, and any other income or deduction documentation.
2. Choose a filing status: Your filing status will depend on your marital status and family situation. The most common filing
statuses are single, married filing jointly, married filing separately, head of household, and qualifying widow(er) with
dependent child.
3. Determine your income: Add up all of your income from various sources, including wages, salaries, tips, self-employment
income, interest, dividends, and capital gains.
4. Calculate your deductions: You may be eligible for deductions for things like charitable donations, medical expenses, mortgage
interest, property taxes, and student loan interest.
5. Figure out your tax liability: Use a tax table or tax software to calculate how much you owe in taxes based on your income,
deductions, and any credits you're eligible for.
6. Complete and submit your tax return: Once you have all of the necessary information, you can complete your tax return using
tax software or by hand. Be sure to sign and date your return, and mail it to the appropriate address by the deadline.
7. Check your refund: If you're due a refund, you can check the status of your refund online or through tax software.

It's important to note that there are different tax forms and instructions for different situations, so make sure you're using
the correct form for your situation. For example, if you have self-employment income, you'll need to use Form 1040-ES to make
```

Figure 2: LLaMA-2 Response

```
Form 8689 is used to determine how much US tax you owe to the USVI. You can either file it with your US tax return or separately
with the VIs Bureau of Internal Revenue.
```

Figure 3: LLaMA-2-RAG Response

5 Conclusion

Based on our evaluation of the smallest model among available LLaMAs (LLaMA-7B), they work well in a restricted knowledge domain using retrieval augmented generation for fact retrieval tasks. Further, they perform better at reasoning tasks as opposed to numerical analysis tasks as shown by our results on the SARA data set. Further, we see that while both RAG and ICL are promising techniques for fine-tuning the responses of an LLM, RAG is better at enhancing LLaMA-2's factual retrieval ability. Our findings demonstrate that these two techniques can help LLMs perform in extremely specific contexts that they were not pre-trained on, which can provide significant societal benefit to domains such as tax assistance, medical advice, etc.

References

[1] https://github.com/facebookresearch/llama

[2] Beltagy, I., Lo, K., & Cohan, A. (2019). SciBERT: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

[3] Bengio, Y., Ducharme, R., & Vincent, P. (2000). A neural probabilistic language model. *Advances in neural information processing systems*, 13.

[4] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[5] Dong, Q., Li, L., Dai, D., Zheng, C., Wu, Z., Chang, B., ... & Sui, Z. (2022). A survey for in-context learning. *arXiv preprint arXiv:2301.00234*.

[6] Floridi, L. & Chiriatti, M. (2020). GPT-3: Its nature, scope, limits, and consequences. *Minds and Machines*, 30, 681-694.

[7] Holzenberger, N., Blair-Stanek, A., & Van Durme, B. (2020). A dataset for statutory reasoning in tax law entailment and question answering. *arXiv preprint arXiv:2005.05257*.

[8] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33, 9459-9474.

[9] Li, H., Su, Y., Cai, D., Wang, Y., & Liu, L. (2022). A survey on retrieval-augmented text generation. *arXiv preprint arXiv:2202.01110*.

[10] Ray, P. P. (2023). ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope. *Internet of Things and Cyber-Physical Systems*.

[11] Shazeer, N. (2020). Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*.

[12] Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., & Liu, Y. (2023). Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 127063.

[13] Topsakal, O., & Akinci, T. C. (2023, July). Creating large language model applications utilizing langchain: A primer on developing llm apps fast. *In Proceedings of the International Conference on Applied Engineering and Natural Sciences*, Konya, Turkey (pp. 10-12).

[14] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M. A., Lacroix, T., ... & Lample, G. (2023). Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

[15] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

[16] Zhang, B., & Sennrich, R. (2019). Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32.