Fine-Tuning Large Language Models (DESIGN.MD)

UNH - COMP741/841 Chris Puzzo, Christian Jackson 4/15/2024

I. System Overview (cpuzzo/cjackson)

This project is about using a pre-trained transformer to create the proof of concept for a custom chat bot. We plan to use a Large Language Model (LLM) which is a type of deep learning model. These models are useful for natural language processing (NLP) and question and answer text generation. They are trained with a large amount of general purpose data from Internet sources. These specific LLMs are known as transformers.

In many cases, these models are not trained well enough to answer questions or draw on knowledge from specific domains. To address this gap, the process of Fine Tuning is used to specialize the transformer to be knowledgeable on a specific domain. An example of this would be GitHub Co-Pilot, which is a transformer trained on code-bases.

It's useful to fine-tune an existing LLM versus simply retraining a model from the ground up because training an LLM is costly and time consuming. In addition, it would be extremely difficult to meet the performance of the current transformers on the market. We plan to use Meta Platforms' open source Llama 2 as our general LLM to fine-tune. They offer three model sizes from 7 to 70 billion parameters. We will start by using the 7B parameter and will change to a larger model if it proves insufficient for our use case. In addition to Llama 2 we will be using the python Perimeter Efficient Fine Tuning (PEFT) library as our fine-tuning engine and Quantized Low Ranking Adaptation (QLoRA). We picked these tools due to their ability to (theoretically) be used to deploy a fine-tuned model on consumer-grade hardware. We plan on using JSON/XML data extracted from text message back ups. Since we are providing our own data, we can assure its validity and lack of PII or other concerns related to questionably sourced data.

II. Content Knowledge Background (cpuzzo)

There are a couple of Content Knowledge areas that a good understanding of is essential to understanding how our project will function. The first AI paradigm that needs to be understood is Neural AI, particularly deep learning. The transformers we are using are built on Deep Learning Neural networks. In addition, an understanding of the inner workings of Neural Networks is needed to understand how fine-tuning works.

Another important thing to understand is fine-tuning and how PEFT differs from "standard" fine tuning . PEFT differs in that instead of tuning all the parameters of a model, it only tunes a small number of (extra) model parameters instead of all the model's parameters' " (PEFT, n.d.). Basically it can tune a model using a much smaller set of data to get results that, while not perfect, are good enough for the majority of use cases out there. The ability to compare results of PEFT with that of standard fine-tuning will be important to the creation of our chatbot because it will show us if PEFT really is good enough or if full fine-tuning is required. Also an understanding of what Transformers and Natural Language Processing (NLP) is important to our project as the model we are using Llama is a Transformer that is used for NLP.

III. AI System Development and Evaluation (cjackson)

The general steps required for LLM fine-tuning are:

1. Select a base pre-trained general purpose LLM.
2. Select a dataset for a particular problem that is labeled so it can be used for training the model.
3. Preprocess the dataset (data cleaning, split into training, validation, and test sets).
4. Fine-tuning to adapt the general LLM to our task using our training dataset.

This last fine-tuning step is typically done by updating the model's parameters based on the new training dataset. Instead we will investigate alternative methods: PEFT, and LoRA to avoid loading the entire LLM into memory and modifying its weights. Instead, we'll create additional smaller neural matrices called adapters that are used in combination with the original LLM for generating inferences.

Specifically, we'll use QLoRA, which is a memory efficient way of creating an adapter for an existing LLM but claims to provide performance similar to the LoRA method. The original model's weights are left unchanged, and we have an adaptor that modifies the inferences of the LLM at runtime. An interesting additional benefit of this approach is that the LLM can be tuned for multiple purposes, and the resulting adapters can be swapped in and out or used currently against a single in-memory LLM. The following steps have been adapted from an online tutorial (Das, 2024):

1. *Using Colab*
2. *Install required libraries*
3. *Load dataset*
4. *Create Bitsandbytes configuration*
5. *Load the pre-trained model (Llama 2)*
6. *Tokenization*
7. *Test the model with zero shot inferencing*
8. *Pre-process our dataset (TBD)*
9. *Prepare the model for QLoRA*
10. *Setup PEFT for fine-tuning*
11. *Train the PEFT adapter*
12. *Evaluate the model qualitatively (human)*
13. *Evaluate the model quantitatively (investigate ROUGE metric)*

The LLM we're using is available from Meta Platforms, Inc. in various parameter sizes at: https://llama.meta.com/llama2/

We expect that we may need to use the following libraries:

- **Bitsandbytes** to load the model efficiently
- **peft:** for efficient fine-tuning.
- **datasets:** a Hugging Face library for accessing datasets obtained from that site
- **einops:** for tensor operations.

IV. Risks Exploration (cjackson/cpuzzo)

Algorithmic discrimination The base LLM that we've selected (Llama2) most likely contains biases regardless of how careful the creators, Meta Corporation, have been to try to mitigate them. Also, we're introducing another dataset when fine-tuning and there is the potential for introducing new biases or amplifying existing biases in the model. These biases could be further propagated or even magnified in the fine-tuned model, leading to bad output (discriminatory, or inappropriate).

In order to avoid this our best bet would be to source our data from multiple different sources and carefully look it over before using it. This way we can at least avoid adding any bias to the system.

Overfitting/Underfitting Fine-tuning our LLM might lead to overfitting, especially if the dataset used for fine-tuning is not diverse enough or is too small. We will need to test to ensure this doesn't happen. We also want to avoid underfitting, because we are not working with a massive dataset we might need to supplement the chat data with other sources that are similar but not exactly the same.

Protection of data privacy We aren't planning to use datasets that contain PII. However, if our training dataset does happen to contain PII, we risk training our model in a way that could perpetuate a data breach or result in an unintended disclosure of confidential information. In addition, where we source our data from can be a problem. The idea is to build a chatbot that is fine-tuned to talk like a specific person/group of people. However if we source our data from people who don't want their data shared that would be an invasion of their right of privacy.

Misalignment It's possible that our fine-tune model might not align perfectly with our expectations. If our approach of using QuLora does not adequately capture the nuances of the domain knowledge, we may not get the output that we're expecting. Compute resources Our approach is to avoid resource intensive processing (within reason). Fine-tuning LLMs can be very expensive. Our hope is that by using Parameter Efficient Fine-Tuning (PEFT) versus Full Fine Tuning (Instruction fine-tuning), we can address this concern. Dependence on Proprietary Technologies The LLM that we've selected to use (Llama 2) is open source and was pretrained on publicly sourced data. It's available in three sizes that range from 7B to 70B parameters that may help give us some interesting options for comparisons. This is a well known LLM provided by Meta with many open source libraries that will allow us to avoid a dependence on an expensive and rapidly changing technology like ChatGPT or Anthropic's Claude.

Transparency and explanation Our AI model is a deep learning neural model and as such it operates much like a "black box." It will not be possible for us to provide an explanation for how any specific output was generated beyond pointing at the original LLM and the training data we've used for fine-tuning it. This is a general problem with this type of architecture and there is a lot of research happening right now to provide more explanation about how answers are developed. This is not something that we can address in this project.

V. Project Milestones (cpuzzo/cjackson)

Week of 4/1:

- Research approaches and tools for fine tuning
- Write and submit project proposal

Week of 4/8

- Present proposal to class
- Begin design
- Find a dataset

Week of 4/15

- Submit project Design document
- Start coding the project

- Parsing, loading tuning dataset
- Deploy Llama 2 in a test environment
- Begin coding with the PEFT library
- Test the model with zero shot inferencing
- Pre-process our dataset
- Prepare the model for QLoRA

Week of 4/22

- Iterate on code development/test cycles
- Train the PEFT adapter
- Start fine-tuning with the dataset
- Evaluate performance
- Evaluate the model qualitatively (human)
- Evaluate the model quantitatively (investigate ROUGE metric)
- Present the project for the first time

Week of 4/29

- Attempt to fine-tune Llama 2 with a commercial PC
- Start writing final report

Week of 5/6

- Finish coding
- Submit final report

Week of 5/13

- Present the final version of the project

VI. References (cjackson/cpuzzo)

Amur, Dilli Prasad. 2023. “QLoRA: Fine-Tuning Large Language Models (LLM’s).” Medium (blog). November 28, 2023.https://medium.com/@dillipprasad60/qlora-explained-a-deep-dive-into-parametric-efficient-fine-tuning-in-large-language-models-llms-c1a4794b1766.

Das, Suman. 2024. “Fine Tune Large Language Model (LLM) on a Custom Dataset with QLoRA.” Medium (blog). January 25, 2024.https://dassum.medium.com/fine-tune-large-language-model-llm-on-a-custom-dataset-with-qlora-fb60abdeba07.

Dettmers, Tim, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. “QLoRA: Efficient Finetuning of Quantized LLMs.” arXiv.http://arxiv.org/abs/2305.14314. “Governments Setting Limits on AI – Communications of the ACM.” 2024. March 15, 2024.https://cacm.acm.org/news/governments-setting-limits-on-ai/.

“PEFT.” n.d. Accessed April 4, 2024.https://huggingface.co/docs/peft/en/index.

Tools and Libraries

Pagoni, Artidoro, Tim Dettmers, QLoRA July 2023. https://github.com/artidoro/qlora

Dettmers, Tim, Bitsandbytes version 0.43.0, March 2023 https://github.com/TimDettmers/bitsandbytes

HuggingFace.co, PEFT v0.10.0 https://huggingface.co/docs/peft/en/package_reference/config

HuggingFace.co, Datasets v2.18.0 https://huggingface.co/docs/datasets/en/index

Rogozhnikov, Alex, einops – v0.7.0 September 30, 2023 https://github.com/arogozhnikov/einops