Homework 5
Build Your Own LLM

11-685: Introduction to Deep Learning (Fall 2023)

OUT: **October 27, 2023**
FINAL DUE: **December 8, 2023 11:59 PM**

Start Here

• Collaboration policy:

– You are expected to comply with the University Policy on Academic Integrity and Plagiarism.

– You are allowed to talk with / work with other students on homework assignments

– You can share ideas but not code, you must submit your own code.

- Your code will be compared against all other semester codes and current one using MOSS.

1. Introduction

In this homework, you will be learning how to build your own Large Language Model, starting from building your own architecture, pretraining it, and then finetuning it for two different tasks. Since this is a project style homework, we will not be providing starter code. You will be evaluated similar to how a project is, in terms of the experiments you tried out and how you approach the problem. More details on this to be released later, but your final report should contain a description of the problem, the task that you were asked to solve, approach and methods, results and discussion, followed finally by conclusions on the entire process of building an LLM.

1 Background

First, let's start off with defining what language models are, a language model is a model that gives a probability distribution over the next token in a sequence given the previous tokens in the sequence, that is: $\mathrm{P}(Y1 \mid Y1 : t-1)$. The basic language model consists of building blocks such as an encoder and a decoder, and more generally, a transformer architecture. When this basic language model is pretrained and finetuned for tasks using millions of parameters, it becomes a large language model.

1.1 Readings

In order to complete this homework, a few readings are essential, make sure to go through the following:

- To familiarize yourself with the transformer architecture (Which you will be asked to implement) Please read this: https://arxiv.org/abs/1706.03762
- To understand how GPT-3 uses this transformer and attention mechanism, please go through this: https://arxiv.org/abs/2005.14165

1.2 NanoGPT

For this homework, you are needed to build your own GPT, and an excellent resource online is this: https://www.youtube.com/watch?v=kCc8FmEb1nY. This explains how an LLM can be built clearly enough that we felt we need not make our own video, and it should help you too! Bear in mind, this video links to a repository of nanoGPT which you can take inspiration from, but should not copy in writing your transformer model for the LLM. It is recommended to watch this a few times end to end.

2 Your Language Model

The language model that you need to develop will involve either only a decoder only architecture, or an encoder decoder architecture. Be sure not to take an already implemented architecture online, as we will perform MOSS checking on popular codebases online. But you can talk inspiration from them. Either of these architectures work well for the tasks described below.

3 Phase 1: Pretraining

You will be pretraining on a subset of OpenWebText data. HuggingFace is useful for loading OpenWebText data. (https://openwebtext2.readthedocs.io/en/latest/). OpenWebText in the link given here has about 65 GB of data. Now we understand the compute difficulties, so it is okay to train the model on a subset of data, 13-15 GB. Feel free to use more if compute allows. The goal of this assignment is to teach how to build an LLM from scratch, and as such, for the finetuning tasks, keeping in mind the compute difficulties, we will be lenient in the evaluation cutoffs.

- First, write your model architecture, and ensure that it can train on a subset of the data.
- This would involve writing the dataloader, making use of it to train the model, and perform predictions.
- Once you have validated on a subset of the data, extend it and train for longer until your chosen loss goes down, using your compute.

4 Phase 2: Finetuning

Constraints:

- You will be using the same LLM for both the tasks below. Do not write two separate LLMs finetuned for each task separately.
- We will release cutoffs for both the tasks at a later time, and more details on how to submit for evaluation.
- Remember that you have limited compute. If you can come up with a way to pretrain/fine-tune in a very efficient way, that can lead to a research paper as well, similar to this competition: https://llm-efficiency-challenge.github.io/challenge. But this is not essential to complete this homework.

Fine-tuning steps:

- In a sense, finetuning is the same as pretraining, just with a different corpus. You will take the model checkpoint resulting from pretraining, and further improve its performance on specific downstream tasks.
- You will reuse the pretraining stage training structure, which includes the dataloader and the LM training loss. However, you will need to think about ways to embed task-specific instructions in the training text.
- Specifically, in order to avoid forgetting, you will randomly mix and match finetuning samples from all tasks, so that the model simultaneously improves on all tasks.

4.1 CNN Daily Mail for Summarization Task

The CNN DailyMail Dataset is an English-language dataset containing just over 300k unique news articles as written by journalists at CNN and the Daily Mail. The current version supports both extractive and abstractive summarization, though the original version was created for machine reading and comprehension and abstractive question answering. Reference: https://huggingface.co/datasets/cnn_dailymail

In this task, you will be made to use the CNN Daily Mail dataset for the task of summarization, and your model will be evaluated on BLEU. A reference of the current benchmark for this task is here:
https://paperswithcode.com/sota/document-summarization-on-cnn-daily-mail

4.2 SQuAD for QA task

Stanford Question Answering Dataset (SQuAD) is a reading comprehension dataset, consisting of questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage, or the question might be unanswerable.
Reference Link: https://huggingface.co/datasets/squad

For this task, you will be finetuning your language model in order to perform question answering. You will be evaluated on ROGUE, and a leaderboard of current state of the art can be found here:
https://rajpurkar.github.io/SQuAD-explorer/

4.3 Bonus task (Above 100 points)

In addition to the two finetuning tasks above, if you perform another finetuning task, you can receive bonus points on top of the 100. One place where you can find such tasks is GLUE, but there are many more, feel free to use whatever you like, even an ongoing competition.

5 Contributions

We will ask each one of the team members to write about their contributions. Making an efficient LLM training and finetuning procedure can lead to something novel.

6 Timeline

- Writeup out: October 26, 2023
- Quiz on Readings and Video: November 11, 2023
- Phase 1 Preliminary Report November 18, 2023
- Phase 2 and Final Report including Finetuning and end to end: December 8, 2023

7 Deliverables

- Project-style report, sections of which will be posted on Piazza.
- The report can include your model architecture, pretraining procedure, evaluation mechanism.
- Trained model.pt
- Scores on finetuning tasks

8 Conclusion

In this assignment, you will be guided on how to construct your own Large Language Model (LLM), from architecture design to pretraining and finetuning for two different tasks. The LLM should be pretrained on a subset of OpenWebText data, and then finetuned on the CNN Daily Mail dataset for a summarization task and the Stanford Question Answering Dataset (SQuAD) for a question answering task. The aim of this assignment is to provide you with a comprehensive understanding of how to build and fine-tune a LLM from scratch.

9 References

[1] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. Advances in neural information processing systems. 2017;30. Link to paper: `https://arxiv.org/abs/1706.03762`

[2] Brown T, Mann B, Ryder N, Subbiah M, Kaplan JD, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, Agarwal S. Language models are few-shot learners. Advances in neural information processing systems. 2020;33:1877-901. Link to paper: `https://arxiv.org/abs/2005.14165`

[3] NanoGPT Link: `https://www.youtube.com/watch?v=kCc8FmEb1nY`

[4] OpenWebText Link: `https://openwebtext2.readthedocs.io/en/latest/`

[5] LLM Efficiency Challenge NeurIPS: `https://llm-efficiency-challenge.github.io/challenge`

[6] CNN DailyMail Link: `https://huggingface.co/datasets/cnn_dailymail`

[7] CNN DailyMail Benchmark Link: `https://paperswithcode.com/sota/document-summarization-on-cnn-daily-mail`

[8] SquadQA Link: `https://huggingface.co/datasets/squad`

[9] SquadQA Leaderboard: `https://rajpurkar.github.io/SQuAD-explorer/`