LLM Calculator

Gleb Skiba

Task definition

The task of this work could be described with next subtasks:

1. Choose a LLM model with few than 4B parameters.
2. Fine-tune chosen model to calculate next math expression: A+B. A, B are positive integers, as big as possible.
3. Calculate the model’s quality on a dataset consisting of the sum of random integers with Accuracy metric.

Related Work

1. These papers [2,3,4] consider the pre-trained LLMs to solve text generation tasks. They achieve good results on different benchmarks in different evalution types. In this work, we will use them to perform good results after supervise fine-tuning on specific dataset.
2. Instruction tuning [6] is a technique used to align pretrained language models with human instructions. It enables targeted customization of LLMs to specific tasks, enhancing their ability to generate more accurate and contextually relevant responses and improving the zero-shot performance. The dataset used for instruction tuning can be human-written machine-generated or collected from web. Creating high-quality instruction tuning datasets can be expensive and time-consuming. In this study, we utilize a simple Python program to generate input-output pairs for arithmetic tasks.
3. In this paper [1] proposes a fine-tuned LLaMA-based model Goat to outperform GPT-4 on arithmetic tasks, due to consistent tokenization of numbers by LLaMA. The authors decompose challenging tasks like multi-digit multiplication and division into learnable tasks and leverage basic arithmetic principles.

Approach description

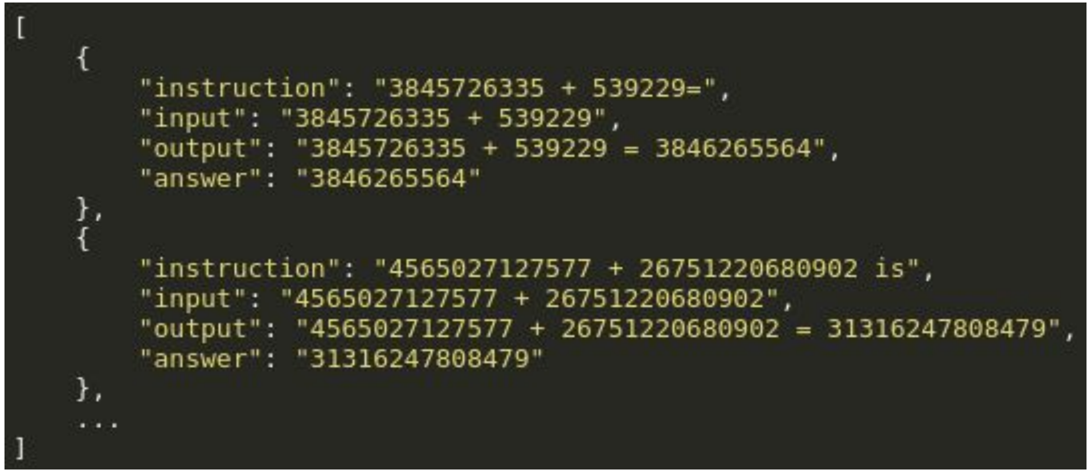
The approach of this work copies the approach of the Goat [1] model and can be described with the next steps:

1. Choose a model that fits the restrictions of this task.
2. Create a synthetic dataset of model instructions that describe the mathematical expression of addition and its result.
3. Fine-tune model in supervised mode on this dataset.

Dataset format

1. Dataset consists of 846600 instructions.
2. Each instruction created with the next algorithm (for more details you can see create_dataset.py):
   a. Created pairs of two random integer numbers, A and B respectively, 1 <= A, B <= 10 ^ 16.
   b. From each pair of numbers created strings with next format: “A + B" is an instruction for the model and “A+B=C” is an output of the model (C is a sum of A and B).
   c. With probability 0.1 symbol ‘+’ replaced with ‘plus’ in each instruction.
   d. With probability 0.05 symbol ‘+’ replaced with ‘and’ and string ‘Sum of ‘ added to each instruction.
   e. With probability 0.2 instruction added with model output into the final dataset, otherwise to the begin of instruction added special prompt (‘solve ’, ‘Q: ‘, ‘Answer of’ and etc, form more detail you can see template.json file) and only then added with model output into final dataset.

Dataset example



```
[
    {
        "instruction": "3845726335 + 539229=",
        "input": "3845726335 + 539229",
        "output": "3845726335 + 539229 = 3846265564",
        "answer": "3846265564"
    },
    {
        "instruction": "4565027127577 + 26751220680902 is",
        "input": "4565027127577 + 26751220680902",
        "output": "4565027127577 + 26751220680902 = 31316247808479",
        "answer": "31316247808479"
    },
    ...
]
```

Model selections

1. Due to the limitation of 4B parameters, the following models were considered:
   a. Falcon-RW-1B [3] – 1B parameters causal decoder-only model trained on 350B tokens of a high-quality web dataset built by leveraging stringent filtering and large-scale deduplication. It is intended for use as a research artifact, to study the influence of training on web data alone.
   b. BLOOM-3b / BLOOM-560M [2] – it's architecture is essentially similar to GPT3 (auto-regressive model for next token prediction), but has been trained on 46 different languages and 13 programming languages.
   c. Pythia-1b / Pythia-1.4b / Pythia-2.8b [4] – decoder-only autoregressive language models designed specifically to facilitate understanding how models behave along two axes: training and scaling.
2. As a base model, **BLOOM-3b** was chosen because it has the largest allowable number of parameters. This property potentially contributes to model quality, but I didn't have enough resources to do any comparative experiments.

Training setup

1. Model was trained in 8 bit mode.
2. Model was trained with AdamW optimizer with 3e-4 learning rate and 100 warm-up steps.
3. Model was trained with 8 batch size for 1 epoch.
4. Model was trained with LoRA [5] technique with next hyperparameters:
    a. Model was trained with 8 LoRA's matrix rank.
    b. Model was trained with LoRA scaling factor.
    c. LoRA matrices applied only to attention blocks.
    d. Model was trained without LoRA bias.
    e. Model was trained with 0.05 LoRA dropout.

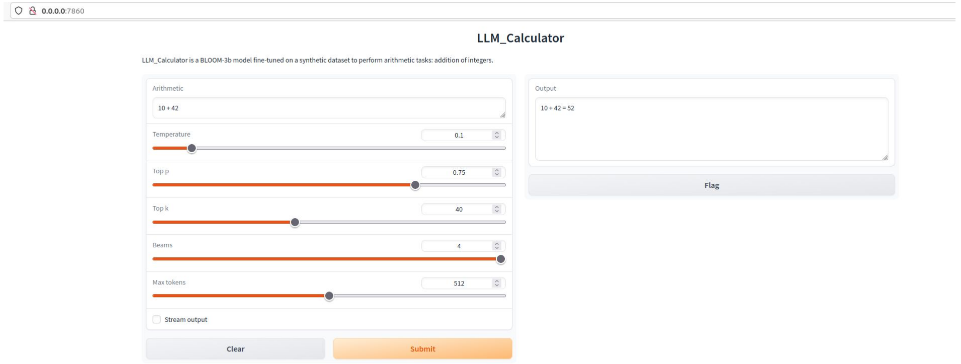
Model quality

1. To evaluate model quality Accuracy metric was used.
2. Model was evaluated on 100 random integer numbers in the range [1; 10^16].
3. For more details look at benchmark.sh script.

Model name	Accuracy
BLOOM-3b fine-tuned	23.13 %

Model inference

1. Model inference implemented with gradio.
2. To run model inference you should:
   a. Setup environment.
   b. Start app.
   c. Open this link “http://0.0.0.0:7860” in the browser.
   d. Enter numbers :)


0.0.0.0:7860

LLM_Calculator

LLM_Calculator is a BLOOM-3b model fine-tuned on a synthetic dataset to perform arithmetic tasks: addition of integers.

Arithmetic: 10 + 42

Temperature: 0.1

Top p: 0.75

Top k: 40

Beams: 4

Max tokens: 512

Stream output

Output: 10 + 42 = 52

Conclusions

1. Model achieves 26% Accuracy.
2. To improve model we can implement next modifications:
   a. Try to use LLM with more parameters, for example, LLaMA or LLaMA-2.
   b. Try to fine-tuned model with more epochs on the bigger dataset.

References

1. Tiedong Liu, Bryan Kian Hsiang Low. 2023. Goat: Fine-tuned LLaMA Outperforms GPT-4 on Arithmetic Tasks. arXiv preprint arXiv:2305.14201.
2. Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ili´c, Daniel Hesslow, Roman Castagné, et al. 2023. BLOOM: A 176B-Parameter Open-Access Multilingual. arXiv preprint arXiv:2211.05100.
3. Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, Julien Launay. 2023. The RefinedWeb Dataset for Falcon LLM: Outperforming Curated Corpora with Web Data, and Web Data Only. arXiv preprint arXiv:2306.01116.
4. Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien. 2023. Pythia: A Suite for Analyzing Large Language Models Across Training and Scaling. arXiv preprint arXiv:2304.01373.
5. Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen. 2021. LoRA: Low-Rank Adaptation of Large Language Models. arXiv preprint arXiv:2106.09685.
6. Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. arXiv preprint arXiv:2210.11416.