Project Work I. - 2023/24/2

Parameter Efficient Fine-tuning of Largue Language Models

Author: Sándor Zsombor
Supervisor: Csanády Bálint

Introduction

Large language models have been in public's center of attention since the reveal of OpenAI's ChatGPT. Since then, multiple open source models have been released, showing greater and greater natural language understanding capability, while simultaneously decreasing model sizes.

However fine-tuning a large language model with traditional methods, like prompt engineering or full fine-tuning, is still an expensive task, the former requiring a large amount of manual attention, and the latter intensive in resources. The main goal of this semester's project was to experiment with different parameter efficient fine tuning methods, and gain an insight on the quality of the results.

Parameter-efficient fine-tuning

Intuitively, if we only wish to modify the output for a certain downstream task, it is expected, that the full modification of the model is not required. This is the essence of all PEFT methods: we prepare a pretrained model for tuning, with the "new model" having a small fraction of the original's parameter count. Then we train the model as if we were to full fine-tune. We mainly experimented with 2 PEFT methods.

Prompt tuning [1]

Doing prompt engineering, we try to customize the prompt to nudge the model into a direction with the desired output. Prompt tuning stems from the same idea, but applied to the embeddings. We prefix the embedding matrix with virtual prompt embeddings; we keep these as the only tunable parameters of the resulting model.

Low-rank adaptation [2]

Despite the layers of LLM's containing a large number of parameters, the intrinsic rank of the matrices which arise after tuning is often low. We may try to leverage this by tuning layers in a lower dimensional space. This idea is the foundation of a technique called low-rank adaptation (LoRA).

With LoRA, we extend selected parameter-matrices with a lower dimensional transformation. Let M be an $n \times k$ parameter-matrix. We create two additional matrices: $A \in \mathbb{R}^{n \times r}$ and $B \in \mathbb{R}^{r \times k}$, where r is a hyperparameter of the techinque (optimally close to the rank of M). We initialize A sampled from a standard normal distribution, and B as an all-zero matrix.

During the computation of the layer output, we now consider $M + A \cdot B$, instead of simply multiplying by M. We may freeze M while finetuning, effectively reducing it's size from nk to $r(n + k)$ parameters, from which the parameter efficiency follows. In practice, it is possible to control the weight of AB respective to M using the α hyperparameter: M is replaced by $M + \frac{\alpha}{r} AB$

Problem setting

• Dataset

We performed measurements using the IMDB movie review dataset, containing 50000 reviews, each labelled with `positive` or `negative` based on the sentiment of the review. The dataset is split into a `train` and `test` set in a balanced fashion, each containing 25000 samples. We used the former for training, the latter for evaluation, often with a sample size of 2000.

• Model and infrastructure

We used the LLaMA2 family of models by Meta [3], which are shown to be able to achieve state-of-the-art results on many natural language processing tasks, including semantic understanding. For instruction-tuning the model, we used the Stanford Alpaca prompt format, which proved to be able to provide higher accuracy, than the LLaMA2 format, after some initial test.

We mainly experimented with the smaller, 7B and 13B models. The computations for the 7B and 13B models were run on a single A100 80GB GPU, while the 70B model needed 4 GPUs. The code for the experiments was written in Python 3.9, utilizing the `transformers` [4] and `peft` [5] libraries by Hugging Face.

Model size	7B	13B	70B
Accuracy	51.78%	82.564%	94.24%

Table 1: Accuracy without training, 25000 test samples

N \ V	8	25	80
50	64.9%	61.2%	57.3%
150	86.25%	76%	83.55%
500	88.4%	89.75%	94.1%
2500	95.3%	95.55%	96.15%

Table 2: Prompt-tuning results, 7B, 2000 test samples

N \ d	0.0	0.1	0.2
50	91.05% (2)	90.4% (2)	92.15% (2)
150	95.4% (2)	94.1% (1)	95.3% (2)
500	95.3% (1)	96.05% (1)	96.35% (2)
2500	96.1% (1)	96.85% (2)	96.25% (2)

Table 3: LoRA results, 7B, 2000 test samples

Results

• Base measurements

For later comparison, Table 1 contains the models' accuracy, without any training. Since the LLaMA2 chat models were not trained on the Alpaca prompt format, the lower results are expected.

• Prompt-tuning

Measurements provided for the 7B. Trained over N samples, with V virtual tokens, for 2 epochs. A learning rate of 0.03 was used. Table 2 contains accuracy measured over 2000 samples.

As expected, with a low number of training samples, the results are unstable, however with a larger sample size, the number of tokens increase performance.

• LoRA

Trained over N samples, with $r = 8$, $\alpha = 16$, d dropout, and `q_proj` and `v_proj` as LoRA application targets. Table 3 contains best accuracy measured over 2000 samples, with the number of training epochs in parenthesis.

Analyzing these results, we tried to achieve the best possible results using the full train sample with all model sizes. The 7B and 13B models were able to achieve state-of-the-art accuracy, with both models beating the previous known best scores (at time of writing) [6]. We weren't able to beat the 13B results with the 70B model. The reason for this is not suspected to be the model's competence, rather the lengthy inference and training times, which make hyperparameter-optimization challenging.

Table 4 contains the accuracy for the experiments. These were trained and evaluated on all training and test samples, respectively. Training used $r = 8$, $\alpha = 16$, 0.1 dropout, $5 \cdot 10^{-4}$ learning rate and a constant learning rate scheduler.

Model size	7B	13B	70B
Accuracy	96.98%	97.32%	96.812%

Table 4: LoRA best results, 25000 test samples

Future work

In this semester, we experimented with two PEFT methods on a binary semantic categorization problem using the LLaMA2 model. Our future plans include extending these experimentations to the newly released LLaMA3 family, and measuring performance with other techniques, such as the recent LoReFT [7]. We also plan to continue testing accuracy with models trained on a limited sample size, which — merged with the idea of LlamBERT [8] —could provide useful results in the realm of real world applications of LLMs. Finally, we may attempt to achieve similar results on NLP tasks other than semantic categorization.

References

[1] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning, 2021.

[2] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.

[3] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[4] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[5] Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. Peft: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`, 2022.

[6] Papers with code - IMDb benchmark (sentiment analysis). `https://paperswithcode.com/sota/sentiment-analysis-on-imdb`. Accessed: 2024-05-15.

[7] Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. Reft: Representation finetuning for language models, 2024.

[8] Bálint Csanády, Lajos Muzsai, Péter Vedres, Zoltán Nádasdy, and András Lukács. Llambert: Large-scale low-cost data annotation in nlp, 2024.