ComGrad: Community Service of Pre-computed Gradients

Botos Csaba, 4 Apr 2024

1 Introduction

There is a common trend in usage across public-domain LLM projects developed in low computational resource environments: 1) collect task specific data, 2) take a pre-trained model (such as Gemma [1]) and 3) fine-tune it with an efficient optimization method, such as QLoRA [2]. While there is a growing literature of best practices to speed up the convergence of the fine-tuning process, due to the diversity of the user specific tasks, the most effective methods, to this date, are mainly task-agnostic methods. Coincidentally, thanks to the development of model sharing platforms (such as the the HuggingFace Hub [3]), the users are incentivized to share their own fine-tuned parameters of the original models, with rich task descriptions. For example, GPT-2 [4] has over 11,000 publicly accessible fine-tuned checkpoints. Since these fine-tuned versions use the same pre-trained initialization, the more tasks become available the more semantic information we can reconstruct about the structure of the parameter space around the common initialization point.

Harvesting such information from the freely available source has significant potential for at least two applications: **1) bootstrapping** - given a task description by the user, we can speed up the optimization process by better pre-initialization and restricting the trainable parameter set to those relevant to the task. **2) explainability** - the ability to describe a fine-tuned model, without having access to the training data, to the model-card, or even without running the model itself, allows scalable community trend analysis and security measures, such as detecting *backdoor attacks* [5].

We introduce a simple and effective method to approximate a bijective mapping between task-descriptions and fine-tuned model parameters, by leveraging the task-arithmetic literature on model editing [6, 7]. Our method improves over time as more community contributions become accessible.

2 Method

Here we describe how to map the task description $d_t \in \mathcal{D}$ of task $t \in \mathcal{T}$ to the parameters θ_t of a parametrized model $f_\theta(.)$ which was fine-tuned to perform task t. More specifically, building on the assumption that every θ_t is a result of a training initialized with the same pre-trained parameters θ_0, we are only interested in the task vector $\tau_t = \theta_t - \theta_0$.

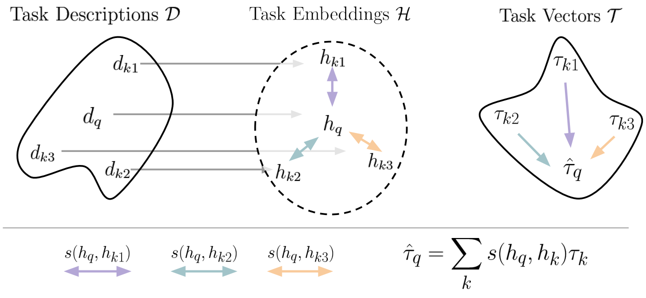

Figure 1: **Overview:** the objective is to predict the task vector of an unseen *query* task $\hat{\tau}_q$ via learning a metric space over the embeddings h of the task descriptions d. The similarity metric we use here is the inner-product $s = proj$.

Our approach is to learn an embedding of the task descriptions $h_\phi : \mathcal{D} \to \mathcal{H}$, parametrized by ϕ, such that for a given *query* task we can use the pairwise similarities between the embeddings of the query and every *key* task $\text{proj}(h_q, h_k)$ as a scaling factor for the following approximation, as shown in Figure 1:

$$\hat{\tau}_q = \sum_k \text{proj}(h_q, h_k)\tau_k. \tag{1}$$

In algebraic terms, this step could be considered as expressing our unseen query task in the basis of previously solved optimization problems, where the similarities $\text{proj}(h_q, h_k)$ are the *coordinates* w.r.t. the h_k basis vectors. As for the next step, our goal is to find a better, task-specific initialization for solving the query task q. Formally, we say that such *"better"* initialization $\hat{\theta}_q = \theta_0 + \hat{\tau}_q$ must satisfy the following inequality:

$$\mathcal{L}_q(f, \theta_0) \geq \mathcal{L}_q(f, \hat{\theta}_q) \forall q, k \in \mathcal{T}. \tag{2}$$

Namely, for every query and key task, the task-specific initialization $\hat{\theta}_q$ should achieve a lower loss value of the query task $\mathcal{L}_q(f, \theta)$ than the task-agnostic, pre-trained weights θ_0.

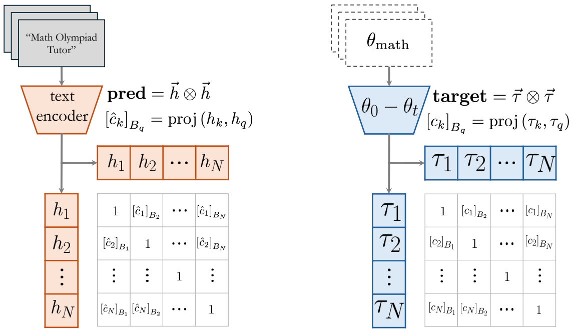
"Math Olympiad Tutor"
text encoder
pred $= \vec{h} \otimes \vec{h}$
$[\hat{c}_k]_{B_q} = \text{proj}(h_k, h_q)$
h_1 h_2 $\cdots$ h_N
h_1 h_2 $\vdots$ h_N
1 $[\hat{c}_1]_{B_2}$ $\cdots$ $[\hat{c}_1]_{B_N}$
$[\hat{c}_2]_{B_1}$ 1 $\cdots$ $[\hat{c}_2]_{B_N}$
$\vdots$ $\vdots$ 1 $\vdots$
$[\hat{c}_N]_{B_1}$ $[\hat{c}_N]_{B_2}$ $\cdots$ 1
θ_{math}
$\theta_0 - \theta_t$
target $= \vec{\tau} \otimes \vec{\tau}$
$[c_k]_{B_q} = \text{proj}(\tau_k, \tau_q)$
τ_1 τ_2 $\cdots$ τ_N
τ_1 τ_2 $\vdots$ τ_N
1 $[c_1]_{B_2}$ $\cdots$ $[c_1]_{B_N}$
$[c_2]_{B_1}$ 1 $\cdots$ $[c_2]_{B_N}$
$\vdots$ $\vdots$ 1 $\vdots$
$[c_N]_{B_1}$ $[c_N]_{B_2}$ $\cdots$ 1

Figure 2: Efficient training of the embedding function h_ϕ.

3 Proposed implementation

To achieve Eq. 2 for new, unseen tasks, we train the parametrized embedding h_ϕ such that for any pair of task descriptions in the training set their pairwise similarity approximates the similarity between their task vectors, illustrated in Figure 2.

Ground truth. We use the projection of the q-th task vector τ_q onto the k-th task vector τ_k. In other words, we compute the k-th coordinate of the q-th task in the basis of $B_q = \text{span}(\{\tau_t\}_{t \neq q})$:

$$[c_k]_{B_q} = \text{proj}(\tau_k, \tau_q) \tag{3}$$

Prediction. Consequently, we want to learn an embedding h_ϕ of the task descriptions d_t that we can use to express the corresponding task vector τ_t, as the linear combination of $\{\tau_t\}_{t \neq q}$, by approximating its coordinates:

$$[\hat{c}_k]_{B_q} = \text{proj}(h_\phi(d_k), h_\phi(d_q)) \tag{4}$$

Objective. The quality of the embedding is represented by $\mathcal{L}_{\text{emb}}$, we can iteratively minimize over the embedding parameters ϕ to approximate:

$$\phi^* = \arg\min_\phi \mathcal{L}_{\text{emb}}(\hat{c}, c) \tag{5}$$

4 Applications

Maintaining such a semantic mapping of fine-tuned parameters speeds up the adaptation process at scale, and encourages more public contributions as everyone benefits from sharing. Furthermore, projecting new parameters on the basis of explainable task-vectors allows one to infer the purpose of the model prior to running it to mitigate security risks [8].

References

[1] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

[2] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. arxiv 2023. *arXiv preprint arXiv:2305.14314*, 2023.

[3] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[4] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[5] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 707–723. IEEE, 2019.

[6] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.

[7] Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. *arXiv preprint arXiv:2305.12827*, 2023.

[8] David Cohen. Data scientists targeted by malicious hugging face ml models with silent backdoor. `https://shorturl.at/orK05`, 2024. Accessed: 2024.02.28.