Project Proposal: Weight Pruning as a Service

Jieren Deng
Department of Computer Science and Engineering
University of Connecticut
Storrs, USA
jieren.deng@uconn.edu

Hongwu Peng
Department of Computer Science and Engineering
University of Connecticut
Storrs, USA
hongwu.peng@uconn.edu

***Abstract*—Deep neural network (DNN) weight pruning has attracted noticeable attention as it enables the deployment of large DNN models on resource-constrained platforms. However, network pruning requires investing significant development effort and computational costs. In this project, we propose a framework (a user-server-based setting) to search differentially private tickets and Differentially Private Pattern Ticket (), which enables efficient and privacy preserving lottery ticket search to users. Specifically, we propose (i) a pruning-as-a-service framework to eliminate the significant costs of human labor and computation required for DNN weight pruning. We develop (ii) an efficient and privacy-preserving method, , through pattern ticket and the proposed noisy top-k ticket algorithm, an approach of differential privacy (DP), to search the differentially private lottery ticket. Also, we prove (iii) the convergence for the proposed searching differentially private tickets method.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

With the ongoing democratization of machine learning [8], there are increasing needs to execute giant DNN models on resource-constrained platforms such as mobile and edge devices with limited computing and storage capacities [14], [22]. Neural network (weight) pruning has been proposed to reduce model parameters and computational cost, which can produce a smaller network that is computational cost-efficient while maintaining similar performance compared to the original network [7], [10], [11], [13], [15], [19], [23]. Machine-Learning-as-a-Service (MLaaS) [18] providers such as Google [20], Intel [25], and ARM [1] have shown great interest in network pruning. However, recent research shows that network pruning requires significant human effort and tremendous computational resources with high costs. Additionally, there lacks a general and unified method in developing pruning techniques [2]. The dataset/network pair and evaluation metrics vary significantly across different works. The hyperparameters also differ from each other or are left unspecified [2]. Hence, it becomes extremely difficult for non-experts to reproduce the results for network pruning, sometimes even from well-crafted open source projects.

In addition, recent studies show that publicly shared gradients in the training process can reveal sensitive properties of the training data to a third-party, as shown in Figure **??**. This means that sharing gradient is a risk for data privacy. Therefore, it is important to avoid any gradient sharing during the pruning process, in order to protect users' data privacy and provide a privacy-preserving network pruning.

II. Background

A. Differential Privacy

Differential Privacy (DP) is a privacy model that is known to render maximum privacy by minimizing the chance of individual record identicfation [3]. In principle, DP defines the bounds of how much information can be reverted to a third party/adversary about someone's data being present in a particular database. Usually, ϵ, is used to describe the level of privacy by a randomized privacy-preserving algorithm over a particular database. Let $\mathcal{D}$ be a collection of datasets. Two datasets D and D' are adjacent if they differ by the addition or removal of at most one user data. A mechanism $\mathcal{M} : \mathcal{D} \rightarrow \mathcal{R}$ with domain $\mathcal{D}$ and range $\mathcal{R}$ is ϵ-differentially private if for any two adjacent datasets $D, D' \in \mathcal{D}$ and for any subest of outputs $S \subseteq \mathcal{R}$,

$$Pr[\mathcal{M}(D) \in S] \leq e^{\epsilon} Pr[\mathcal{M}(D') \in S] \tag{1}$$

We state our results with ϵ-DP formulation since the Laplace mechanism we use is pure differential privacy.

Laplace Mechanism. One of the most common building blocks of differentially private algorithms is the *Laplace mechanism* [6], which is used to answer numerical queries. A numerical query is a function $q : \mathcal{D} \rightarrow \mathbb{R}^k$ (i.e. it outputs a $k-$dimensional vector of numbers). The Laplace mechanism is based on a concept called *sensitivity*, which measures the worst-case effect one record can have on a numerical query:

$$\Delta_q = \max_{D,D' \in \mathcal{D}} ||\varphi(D) - \varphi(D')|| \tag{2}$$

The Laplace mechanism works by adding Laplace noise (having density $f(x|b) = \frac{1}{2b} \exp(-|x|/b)$ and variance $2b^2$) to query answers. The chosen variance depends on ϵ and the sensitivity. We use the notation $\texttt{Lap}(b)$ to refer to the Laplace noise. For any numerical query $q : \mathcal{D} \rightarrow \mathbb{R}^k$, the Laplace mechanism outputs:

$$\mathcal{M}(\mathcal{D}, q, \epsilon) = q(\mathcal{D}) + (\mu_1, ..., \mu_k) \tag{3}$$

where μ_i is independent random variables sampled from $\texttt{Lap}(\frac{\Delta_q}{\epsilon})$. Thus, the Laplace Mechanism is *ϵ-differentially*

private. The choice of DP mechanism is flexible, our framework is not subject to any particular mechanism. For prototype design, we select Laplace mechanism as it achieves pure DP and easy to compose. However, one can also replace it with other options such as Gaussian mechanism.

[Sequential Composition] Let $M_1 : D \rightarrow$ ***Distr(R)*** *be an* (ϵ_0)*-private computation, and let* $M_2 : D \rightarrow$ ***Distr(R')*** *be an* (ϵ_0')*-private computation in the first argument for any fixed value of the second argument. Then, the function*

$$M \mapsto M_1(d)M_2(d) \tag{4}$$

is $(\epsilon_0 + \epsilon_0')$*-private.*

[Post-Processing] *Let* $\mathcal{M}$ *be an* ϵ*-differentially private mechanism and* g *be an arbitrary mapping from the set of possible outputs to an arbitrary set. Then* $g \circ \mathcal{M}$ *is* ϵ*-differentially private.*

B. Neural Network Pruning

Weight Pruning. There are two types of weight pruning methods: non-structured pruning and structured pruning. The non-structured pruning was first introduced by Han et. al. [10], which is an iterative and heuristic method to prune the connections and neurons in original networks. The non-structured pruning is not hardware friendly which limits the capacity for accelerating inferences [4], [5], [9]. Structured pruning is designed to fit the characteristic of hardware which can significantly reduce the computation [12], [17], [21]. The accuracy of structured pruning methods drops noticeably when the pruning rate increases compared to non-structured pruning method. Recently, pattern pruning [16] was designed to incorporate unique pattern masks for the convolutional kernel. However, these weight pruning techniques do not have a theoretical guarantee in convergence. Otherwise, server side has to provide another pruned candidate with the hope of convergence or help debug for each unconverged case, which puts a huge burden on the server side. Compared to the aforementioned weight pruning approaches, our proposed method provides a theoretical proof for convergence guarantee.

Lottery Ticket Hypothesis (LTH). The Lottery Ticket Hypothesis was proposed by Frankle et al. [7], showing that a subnetwork of a randomly-initialized network can replace the original network with the same performance. The "winning-ticket" by [24] shown that a "good initialized" subnetwork has a better performance than the randomly initialized subnetwork. However, LTH pruning method has no guarantee in training data privacy-preserving during identifying winning tickets which could be the severely sensitive information leakage in finding the lottery ticket. To address this issue, our proposed method provides training data privacy-preserving during identifying winning tickets.

1

REFERENCES

[1] C. Banbury, C. Zhou, I. Fedorov, R. M. Navarro, U. Thakkar, D. Gope, V. J. Reddi, M. Mattina, and P. N. Whatmough, "Micronets: Neural network architectures for deploying tinyml applications on commodity microcontrollers," *arXiv preprint arXiv:2010.11267*, 2020.

[2] D. Blalock, J. J. G. Ortiz, J. Frankle, and J. Guttag, "What is the state of neural network pruning?" *arXiv preprint arXiv:2003.03033*, 2020.

[3] M. A. P. Chamikara, P. Bertók, D. Liu, S. Camtepe, and I. Khalil, "An efficient and scalable privacy preserving algorithm for big data and data streams," *Computers & Security*, vol. 87, p. 101570, 2019.

[4] X. Dai, H. Yin, and N. K. Jha, "Nest: a neural network synthesis tool based on a grow-and-prune paradigm," *arXiv preprint arXiv:1711.02017*, 2017.

[5] X. Dong, S. Chen, and S. Pan, "Learning to prune deep neural networks via layer-wise optimal brain surgeon," in *Advances in Neural Information Processing Systems*, 2017, pp. 4860–4874.

[6] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*. Springer, 2006, pp. 265–284.

[7] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," *arXiv preprint arXiv:1803.03635*, 2018.

[8] C. Garvey, "A framework for evaluating barriers to the democratization of artificial intelligence," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[9] Y. Guo, A. Yao, and Y. Chen, "Dynamic network surgery for efficient dnns," in *Advances In Neural Information Processing Systems*, 2016, pp. 1379–1387.

[10] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[11] Y. He, G. Kang, X. Dong, Y. Fu, and Y. Yang, "Soft filter pruning for accelerating deep convolutional neural networks," *arXiv preprint arXiv:1808.06866*, 2018.

[12] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1389–1397.

[13] N. Lee, T. Ajanthan, and P. Torr, "Snip: Single-shot network pruning based on connection sensitivity," in *International Conference on Learning Representations*, 2018.

[14] E. Li, L. Zeng, Z. Zhou, and X. Chen, "Edge ai: On-demand accelerating deep neural network inference via edge computing," *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 447–457, 2019.

[15] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, "Rethinking the value of network pruning," *arXiv preprint arXiv:1810.05270*, 2018.

[16] X. Ma, F.-M. Guo, W. Niu, X. Lin, J. Tang, K. Ma, B. Ren, and Y. Wang, "Pconv: The missing but desirable sparsity in dnn weight pruning for real-time execution on mobile devices." in *AAAI*, 2020, pp. 5117–5124.

[17] X. Ma, S. Lin, S. Ye, Z. He, L. Zhang, G. Yuan, S. Huat Tan, Z. Li, D. Fan, X. Qian *et al.*, "Non-structured dnn weight pruning–is it beneficial in any platform?" *arXiv*, pp. arXiv–1907, 2019.

[18] M. Ribeiro, K. Grolinger, and M. A. Capretz, "Mlaas: Machine learning as a service," in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2015, pp. 896–902.

[19] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6105–6114.

[20] T. M. O. Toolkit, https://www.tensorflow.org/model$_optimization$, 2021.

[21] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," *Advances in neural information processing systems*, vol. 29, pp. 2074–2082, 2016.

[22] X. Xu, Y. Ding, S. X. Hu, M. Niemier, J. Cong, Y. Hu, and Y. Shi, "Scaling for edge inference of deep neural networks," *Nature Electronics*, vol. 1, no. 4, pp. 216–222, 2018.

[23] T. Zhang, S. Ye, K. Zhang, J. Tang, W. Wen, M. Fardad, and Y. Wang, "A systematic dnn weight pruning framework using alternating direction method of multipliers," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 184–199.

[24] H. Zhou, J. Lan, R. Liu, and J. Yosinski, "Deconstructing lottery tickets: Zeros, signs, and the supermask," in *Advances in Neural Information Processing Systems*, 2019, pp. 3597–3607.

[25] N. Zmora, G. Jacob, L. Zlotnik, B. Elharar, and G. Novik, "Neural network distiller: A python package for dnn compression research," *arXiv preprint arXiv:1910.12232*, 2019.