Poster: Packing-aware Pruning for Efficient Private Inference based on Homomorphic Encryption

Parsa Ghazvinian[1] Robert Podschwadt[2] Prajwal Panzade[3] Mohammad H. Rafiei[4] Daniel Takabi[5]

Department of Computer Science, *Georgia State University*, *Atlanta*, GA
[1]pghazvinian1@gsu.edu, [2]rpodschwadt1@gsu.edu, [3]ppanzade1@student.gsu.edu, [4]mrafiei@gsu.edu, [5]takabi@gsu.edu

***Abstract*—Due to the extensive application of machine learning in a wide range of fields and the necessity of privacy, privacy-preserving machine learning (PPML) solutions have recently gained significant traction. One group of approaches relies on Homomorphic Encryption (HE), which enables us to perform computation over encrypted data. However, even with state-of-the-art schemes, HE operations are still significantly slow compared to their plaintext counterparts and require considerable memory. Therefore, here we propose a framework that automatically makes an already trained model HE-friendly using a learning-based method. Then, it performs a packing-aware pruning method that prunes the model's parameters in arbitrary block shape in correspondence with the ciphertext packing method while maintaining the model's performance. This allows for dropping a significant number of HE operations and reducing latency and memory consumption of the private inference (PI). We evaluate our method on the LeNet model trained on MNIST; our results demonstrate that even at a 98% pruning ratio, the model performs almost the same as the baseline model. At the same time, the number of required HE operations is reduced by a factor of 15 which implies 9.63 and 4.04 factors of reduction in the latency and the required memory of PI.**

***Index Terms*—Packing-Aware Pruning, Private Inference, Privacy-Preserving Machine Learning, Homomorphic Encryption**

I. Introduction

Machine learning as a service (MLaaS) has become the predominant solution for complex computations of machine learning models. However, privacy concerns hinder the prevailing adoption of MLaaS. Homomorphic encryption allows performing computation on encrypted data without decryption, so it is adopted as one of the primary solutions for PPML. Nevertheless, high memory requirements and latency limit applying HE solutions to arbitrary models. Pruning the parameters of a neural network (NN) effectively addresses these limitations in the plaintext domain. However, conventional plaintext pruning methods offer minor benefits in the HE domain, even at high ratios, as they do not consider the ciphertext packing method[1]. The underlying reason is that most HE schemes perform operations in single instruction multiple data (SIMD) format and pack multiple values inside one ciphertext on which the operation is performed. An HE operation can only be skipped if its operand ciphertext is all-zero, but regular pruning methods introduce zeros in random locations, so typically, some non-zero values reside in ciphertexts, preventing it from dropping corresponding operation; this could be translated to a minor improvement in latency or memory consumption. [2] presents a pruning technique that utilizes HE-friendly structures to identify and prune specific portions of model parameters. However, their method requires the client to evaluate the activation functions. [1] utilizes permutation and expansion of the packed model weights to prune ciphertext packs. However, their proposed method requires two transformation operations on input and output data, and they just evaluated their proposed method on simple fully-connected networks. Here, we propose a framework that automatically transforms an already trained model to its HE-friendly version and performs packing-aware pruning, which significantly reduces the number of HE operations, latency, and memory consumption, and performs the PI non-interactively.

II. The Proposed Methodology

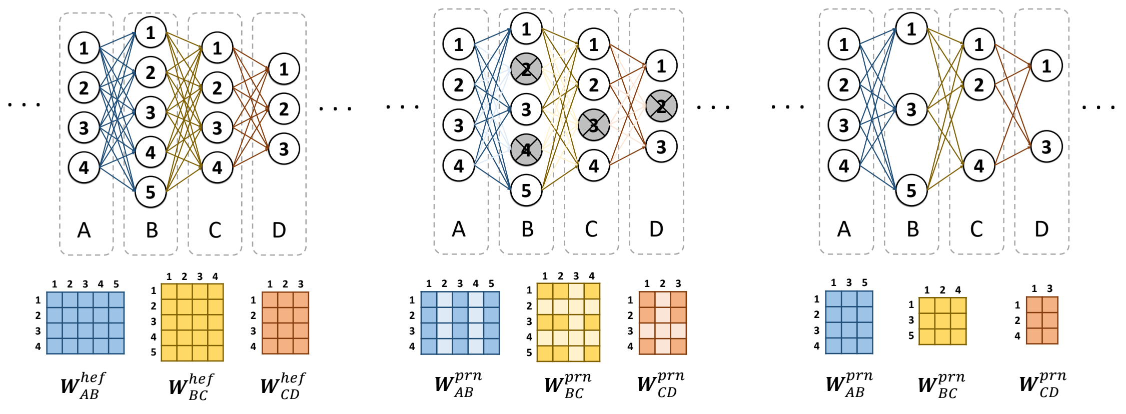
Let the "original model" be the model trained on plaintext data; our method has two steps to identify the HE-friendly packing-aware pruned version of it with comparable accuracy and then perform PI on it. We describe these steps in the followings:

A. Step 1. Making the Original Model HE-Friendly

In this step, all max-pooling layers are converted to average pooling with the same window size and strides, one at a time, starting with the latest one to the first. After the conversion of each pooling layer, we retrain the afterward layers a few epochs, then fine-tune the whole model with a relatively small learning rate to recover the accuracy loss imposed by replacements. We start from the latest max-pooling layer since it would produce a lower propagated error than if we started from the first one. Similarly, the activation layers are updated starting from the latest layer; we replace each with a trainable polynomial with a pre-defined degree or a square function. In the case of polynomials, to obtain the polynomial parameters (i.e., polynomial multipliers) automatically, we only train them as trainable model parameters for a few epochs, with relatively small magnitude initial weights and a small learning rate. Next, we fine-tune the whole model with a small learning rate (in the case of the square function, only the fine-tuning step is required). The outcome of step 1 is a HE-friendly version of the "original model"(Fig1a).

B. Step 2. Packing-Aware Pruning

Zhu et al. [4] propose an over-training pruning method using a binary gradient mask to enforce zeroing low-magnitude



(a) Original/He-Friendly (b) Pruned model with(left)/ without zeroed weights (right)

Fig. 1: Example fully-connected layer's pruning process with pruning blocks to be weight columns

weights gradually until it achieves a predefined target sparsity. By defining a block constraint to this method which is implemented by the TensorFlow Model Optimization toolkit[1], we are able to prune the weight matrix in any arbitrary block shape (block pruning), then by matching it with the ciphertext packing, we perform an effective packing-aware pruning. Based on our packing method (described in II-C), we set the block shape equal to the weight matrix columns in a fully-connected layer, which implies pruning the neurons. This consequently prunes the rows of the next layer's weight matrix (i.e., outgoing connections of the pruned neuron) (Fig1b). We convert convolution layers into the equivalent fully-connected layers to leverage the same pruning method for convolution layers. In this case, the transformed weight matrix's columns are equivalent to the convolution operation's filters, which means filter pruning. Finally, we drop the pruned (zeroed) columns (filters) from the model and fine-tune it again to recover the resulting accuracy loss, so we end up with a compressed version of the pruned model (Fig1b) ready for PI.

C. Step 3. Private Inference

We assume the model is plaintext and the input data is encrypted for PI. We adopt the batch packing technique proposed by [3] in which we group the data into batches of multiple instances and pack the same feature of every instance into a ciphertext. This means the number of ciphertexts is equal to the number of features. We should also encode plain values into plaintext since we can only perform operations between ciphertexts and ciphertexts or ciphertexts and plaintexts. We take the HE-friendly packing-aware pruned model of step 2 and extract the weights and layer configurations, to perform PI. We implement the algorithms for PI using the HE primitives.

III. Experiments

We evaluate our method on the LeNet trained on the MNIST dataset, which originally contains three convolution layers and one fully-connected layer. As demonstrated in Table I, the model has almost the same performance as the baseline model even at 98% sparsity; while the number of HE operations is reduced by a factor of 15, it consequently reduces the latency and the required memory of PI by factors of 9.63 and 4.04, respectively. We use CKKS implementation of SEAL[2] with 128-bit security supporting real numbers.

TABLE I: Results of performing PI of LeNet trained on MNIST in different sparsities. Unpruned HE-friendly model is the baseline, testing accuracy, pruning time and PI latency in seconds, the total number of HE operations, and the memory required to perform PI in GB.

Sparsity	baseline	0.73	0.89	0.98
Accuracy	0.99	0.99	0.98	0.97
Pruning Time	-	111	210	393
Latency	1272	502	425	132
HE Operation	8.8e5	2.9e5	2.2e5	5.8e4
Memory	303	166	166	75

References

[1] Ehud Aharoni et al. "HE-PEx: Efficient Machine Learning under Homomorphic Encryption using Pruning, Permutation and Expansion". In: *arXiv preprint arXiv:2207.03384* (2022).

[2] Yifei Cai et al. "Hunter: He-friendly structured pruning for efficient privacy-preserving deep learning". In: *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*. 2022, pp. 931–945.

[3] Nicholas Dowlin et al. "CryptoNets: Applying Neural Networks to Encrypted Data with High Throughput and Accuracy". In: *International Conference on Machine Learning*. PMLR. 2016, pp. 201–210.

[4] Michael Zhu and Suyog Gupta. "To prune, or not to prune: exploring the efficacy of pruning for model compression". In: *arXiv preprint arXiv:1710.01878* (2017).

[1]https://www.tensorflow.org/model_optimization/api_docs/python/tfmot/sparsity/keras/prune_low_magnitude

[2]https://github.com/Microsoft/SEAL