2025 Design, Automation & Test in Europe Conference (DATE 2025)

A DRAM-Based Processing-in-Memory Accelerator for Privacy-Protecting Machine Learning

Bokyung Kim
Dept. of Electrical and Computer Engineering
Rutgers University, USA
bk.kim@rutgers.edu

The sensational outcomes of machine learning (ML) are witnessed. As the big success relies on continual training with massive data encompassing sensitive information, deep neural network (DNN) models easily leak private information. For instance, large pre-trained language models contain a substantial volume of private information, which can be acquired by querying with appropriate prompts. This raises concerns regarding privacy in ML, and DNN models are evolving for privacy-preserving ML (PPML).

Differential privacy (DP) is considered the gold-standard solution and accordingly is applied to the representative training methodology, stochastic gradient descent (SGD), for PPML. Instead of per-batch gradients of SGD (Fig. 1(a)), DP-SGD treats every sample in a batch as an individual sample (i.e., it keeps not only each sample's weight gradients (per-sample gradients) to derive norms but also the norms to clip the gradients. Finally, all clipped gradients are aggregated and added with noise (Fig. 1(b)).

As DP-SGD is memory-bound, reweighted DP-SGD [4] (R-DP-SGD) is developed to handle the high memory demand by additional backpropagation. While DP-SGD applies the gradient norm to each per-sample weight gradient in clipping, R-DP-SGD removes the necessity to store all the per-sample gradients by reweighting the loss and re-executing the backpropagation, as described in Fig. 1(c). R-DP-SGD comprises four distinct computing phases from one forward- and two back-propagations: ***forward, activation gradient, per-sample gradient norm, and per-batch gradient***. Sacrificing computation resources, R-DP-SGD recovers memory requirements by almost SGD with the same privacy level as DP-SGD. However, R-DP-SGD further aggravates the computation burden compared to SGD, which already shows high overhead. The need to efficiently accelerate DP training has critically emerged.

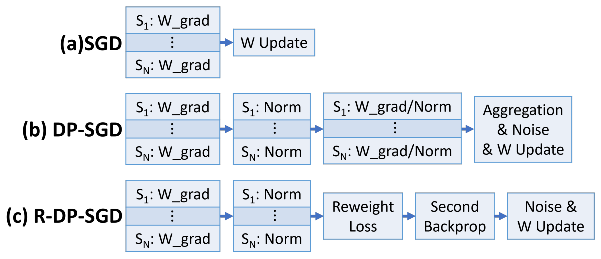

Fig. 1. (a) Non-private SGD with the benefit of batch training, (b) DP-SGD with high memory overhead, and (c) compute-intensive R-DP-SGD.

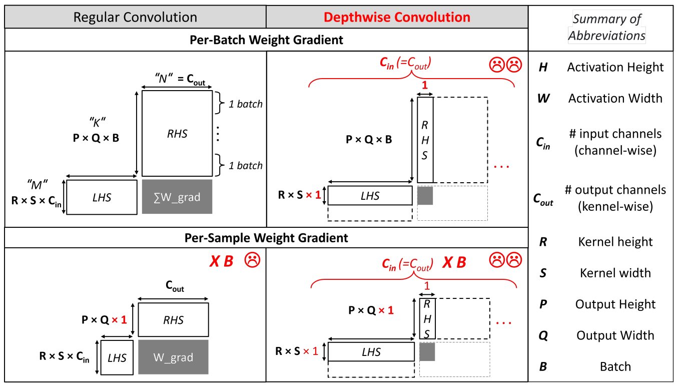

Fig. 2. GEMMs of per-batch and per-sample weight gradients in regular and depthwise convolution.

I. Motivation of Our Accelerator Design

Previous accelerators have been tailored for non-private DNNs, focusing on higher-dimensional tensor processing. The systolic array, where processing engines are placed in an array form, is one of the most successful designs for high performance. An operator's left- and right-hand sides are unrolled and computed in the systolic array, performing general matrix multiplication (GEMM). Correspondingly, a representative off-the-shelf accelerator, tensor processing unit (TPU), leverages the systolic array for non-private DNNs.

However, DP-applied SGD is largely computed on low-dimensional tensors, and therefore, the previous accelerators do not match the requirement of private SGD. While domain-specific accelerators for DP training are rarely designed, DiVa [5] proposes a systolic-array-based accelerator for R-DP-SGD. As the weight-stationary (WS) dataflow of TPU is inappropriate for R-DP-SGD, DiVa presents an output-stationary (OS)-based design. It also introduces a post-processing unit (PPU) for norm computation in R-DP-SGD.

Our study uncovers that the systolic array design limits the performance and efficiency of the privacy-secured training. Following our observation, **the per-sample gradient norm has a distinctive computing nature oriented toward low-dimensional tensor (e.g., vector) processing**. cAs a result of the mismatch with matrix-tailored systolic arrays, when accommodating DP-applied SGDs into systolic arrays, they can hardly avoid **repetition of under-utilization scenarios** owing to vector-shaped operands, particularly in depthwise convolution. As shown in Fig. 2, C_{in} and $Cout$ are one in depthwise

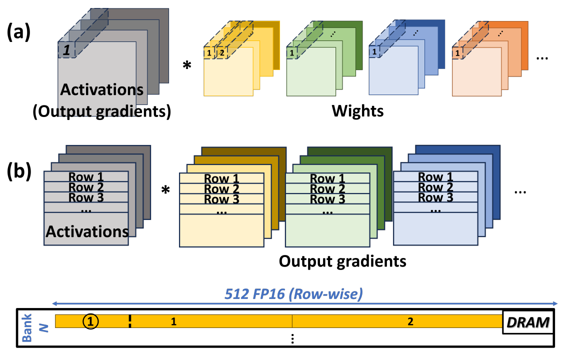

Fig. 3. Computing schemes, (a) *FAG_COMP* and (b) *WG_COMP*.

convolution, narrowing "*N*" and "*M*" and correspondingly causing under-utilization. It becomes severe in per-sample weight gradient computing, where computing results are not accumulated across different samples. Also, as systolic arrays pursue high productivity in matrices, **data transfer issues** occur in linear models as a limitation behind the production. Numerous partial results are generated in linear models, but the bandwidth is limited to transfer them; therefore, innumerable movements are triggered in R-DP-SGD.

II. Method

Against the conventional hardware design, where the data transfer issues have been indicated as a bottleneck, processing-in-memory (PIM) has manifested its effectiveness in data movement by integrating memory storage and computing [2, 1, 3]. DRAM-PIM leveraging high bandwidth memory (HBM) is in the limelight because of its memory capacity, high bandwidth, reliability, and low-cost commodity. These features make DRAM-PIM advantageous for data- and computation-intensive applications like PPML.

Leveraging the benefits of DRAM-PIM, we propose a differentially private PIM accelerator, introducing 1) a novel micro-architecture and 2) a new dataflow scheme to support various operations and peculiar requirements from diverse computing phases of R-DP-SGD. As a micro-architecture design, we design a low-cost computing unit at the bank level, **full operation adder tree (*FOAT*)**. Two functions are implemented in FOAT: self-multiplication and hold. Unlike prior adder designs, FOAT can compute not merely fundamental operations but also norms by the two functions without other peripheries.

At the architecture level, as we noticed that a consistent computing scheme could harm DP training computing efficiency, we first characterize different computing phases to find their requirement and develop a **custom-built computing scheme**. The proposed accelerator executes R-DP-SGD on two dataflows; *FAG_COMP* is for forward and activation gradient and *WG_COMP* for per-batch and -sample gradients, as described in Fig. 3(a) and (b), respectively. *FAG_COMP* priorizes channel-wise deta for computing, while *WG_COMP* conducts intra row-wise computing first. The prioritized chunks are mapped and computed in the DRAM-PIM design.

III. Evaluation

We compare the proposed accelerator to DiVa. Various networks are used to attest our design's improvement: VGGs (large models), ResNets (shallow and deep layers), MobileNetV2, MNasNet (light models), and BERT models (NLP). The proposed accelerator achieves 13.8$\times$ and 123.8$\times$ improvement on average in performance and energy efficiency than the baseline. Models with depthwise convolution (light models) achieved the most notable improvement, 77.6$\times$ and 16.8$\times$ in MobileNetV2 and MNasNet, respectively. In the case of the integrated version, the average improvement increased to 13.85$\times$. The PE array consumes higher power (21.2W) than the memory array (1.24W). Also, whereas a PPU of DiVa needs 8 adder trees of 7 stages, ours adds a 4-staged FOAT per bank. The proposed accelerator presents 9.1$\times$ energy improvement on average in the benchmarks except for MobileNetV2. MobileNetV2 shows dramatically high energy efficiency over 800$\times$ due to the synergy effect of the performance and power efficiency of DPIMA. The proposed accelerator is implemented in DRAM, and therefore, its area increment is from FOAT. Thanks to the small DRAM fabrication technology, the estimated area in 5 nm is 0.002 mm^2 per adder tree and 0.512 mm^2 in total increase. On the contrary, DiVa reports 82 mm^2 with the 65 nm node due to the PE array. Ours saves the huge area dedicated to the systolic array.

IV. Conclusion

It has become urgent to accelerate R-DP-SGD in PPML efficiently. However, prior systolic-array-based designs display limitations, i.e., under-utilization increases massive data transfers, by vectorized tensors. Therefore, we propose a DRAM-PIM accelerator with a novel processing unit, *FOAT*, enabling various operations of R-DP-SGD and a customized dataflow scheme by catching different requirements. Experiments prove 13.8$\times$ and 123.8$\times$ performance and energy efficiency improvement than the systolic baseline.

Acknowledgments

This work is supported by Rutgers University Startup Fund 304291 for Prof. Kim.

References

[1] B. Kim, E. Hanson, and H. Li. "An efficient 3d reram convolution processor design for binarized weight networks". In: *IEEE Transactions on Circuits and Systems II: Express Briefs* 68.5 (2021), pp. 1600–1604.
[2] B. Kim and H. Li. "Leveraging 3D Vertical RRAM to Developing Neuromorphic Architecture for Pattern Classification". In: *2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. IEEE. 2020, pp. 258–263.
[3] B. Kim, S. Li, and H. Li. "INCA: Input-stationary Dataflow at Outside-the-box Thinking about Deep Learning Accelerators". In: *2023 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE. 2023, pp. 29–41.
[4] J. Lee and D. Kifer. "Scaling up differentially private deep learning with fast per-example gradient clipping". In: *Proceedings on Privacy Enhancing Technologies* 2021.1 (2021).
[5] B. Park, R. Hwang, D. Yoon, Y. Choi, and M. Rhu. "DiVa: An Accelerator for Differentially Private Machine Learning". In: *2022 55th IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE. 2022, pp. 1200–1217.