CSB-RNN: A Super Real-time RNN Framework with Compressed Structured Block

Runbin Shi[1], Peiyan Dong[2], Tong Geng[3], Martin Herbordt[3], Hayden So[1], Yanzhi Wang[2]
[1]The University of Hong Kong, [2]Northeastern University, [3]Boston University
{rbshi,hso}@eee.hku.hk,dong.pe@husky.neu.edu,{tgeng,herbordt}@bu.edu,yanz.wang@northeastern.edu

ABSTRACT

This paper presents CSB-RNN, an optimized full-stack RNN framework with the novel compressed structured block (CSB) technique. The CSB-pruned RNN model comes with both fine-granularity that benefits the pruning rate and regular structure that facilitates the hardware-parallelism. Further, we propose a novel hardware architecture for inferencing the CSB-pruned model, which solves the block-workload imbalance issue and achieves an over 95% hardware utilization. CSB-RNN achieves 1.7×-3.6× improvement on the pruning rate comparing to the prior art. With the addition of novel architecture, the compressed-RNN inference reaches a super real-time latency of 23μs-67μs on FPGA implementation.

1 INTRODUCTION

RNN is widely adopted for its high-accuracy on temporal-sequence analysis, such as machine translation [2], speech recognition [3], or even stock-price prediction [5]. However, the increasingly large model size and tremendous computational workload of RNN hampers its deployment on embedded (edge) device, which strictly demands on real-timing processing with limited hardware resources. To address this issue, pruning techniques on RNN models are proposed, which enables the models to be smaller, require less storage demand and provide higher potential performance by eliminating the redundant (near-zero) weights and their related arithmetic-operations in the model. The pruning schemes in most existing works [4, 6] are in a *non-structured* fashion and thus bring *extreme irregularities* in the processing, which is unfriendly to either the modern parallel device or the hardware-architecture design. So that the performance degradation from the hardware inefficiency encroaches upon the gains from model compression. Therefore, researches start to explore other pruning approaches, i.e. *Structured* pruning [7]. During structured pruning, regular computational patterns are required and maintained. Although these structured-pruned models facilitate the hardware parallelism and are relatively hardware friendly, the coarse pruning-granularity (structure) leads to either a significant degradation on model-accuracy or limited pruning rates. To keep the accuracy loss acceptable, the attainable pruning rates delivered in the existing structured pruning schemes are *far less than* that the ones with the non-structured scheme, and thus the model redundancy is under-utilized.

In this paper, we propose an *adaptive fine-grained* structured pruning technique that provides the same or even higher compression rate as non-structured pruning while guaranteeing potential hardware efficiencies. During our training phase, the weight matrix is divided into *fine-grained blocks* and a structured pruning is conducted on each block independently. The pruned blocks are

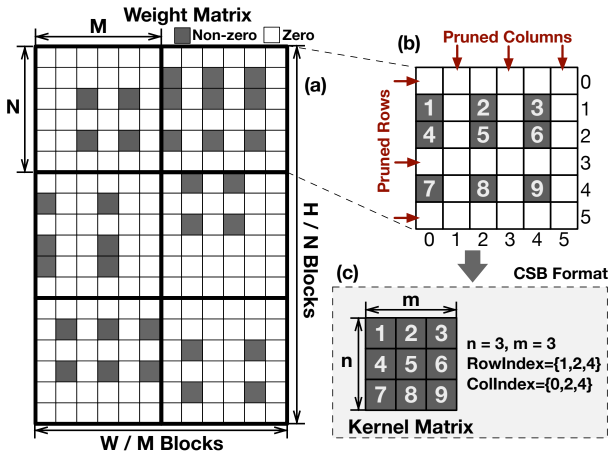

Figure 1: Fine-grained CSB-pruning: Partition the weight matrix to blocks, on which the row-column structured pruning is conducted independently. The pruned blocks are compressed and stored in CSB format.

encoded to the novel *Compressed Structured Block (CSB)* format for inference, which significantly reduces the weight-storage demand while retaining the fine-grained content in the original model. To realize a real-time performance, there are still many challenges to design a parallel architecture which can exploit the benefits of the proposed pruning technique perfectly. For instance, the architecture should be adaptive to various RNN types (i.e., LSTM, GRU, etc.); Programmability should be provided that applies to multiple RNN-layers with different hyper-parameters and further different pruning-granularities (block size) in CSB-pruning technique; Particularly, the parallel architecture should handle massive pruning-blocks with imbalanced workloads but maintain perfect hardware utilization. To address the issues above, we propose an architecture-compilation co-optimization to realize the best flexibility and performance. A novel programmable dataflow architecture is designed, which supports the *workload-sharing* to balance the workload among parallel processing units. Further, the architecture is dynamically adaptive to different block sizes and thus can seamlessly process RNN-layers with different pruning-granularities. Meanwhile, the compilation scheme is proposed that analyzes the compressed weight matrix once and generates control-instruction to conduct the continuous inference in a workload-balanced manner.

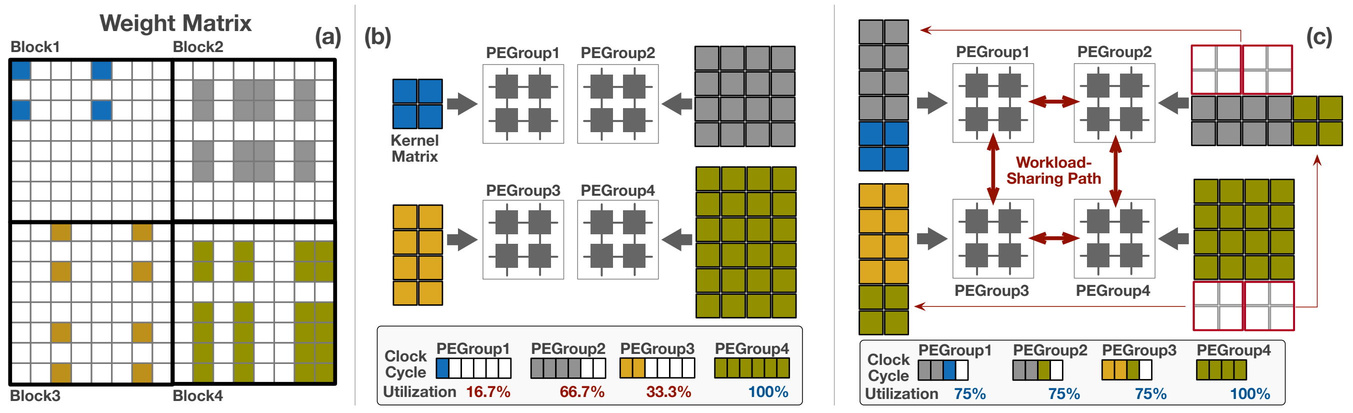

Figure 2: (a) demonstrates the inter-block workload-imbalance with CSB-pruning; (b) the imbalance leads to hardware un-utilization on vanilla architecture; (c) concept of workload-sharing in the novel architecture for inference with CSB-pruned model.

2 NOVEL TECHNIQUES

2.1 Pruning with Compressed Structured Block

We propose the compressed structured block (CSB) pruning technique that benefits both the pruning flexibility and the hardware-parallelism. As illustrated in Fig. 1(a), in the pruning flow, the weight matrix (with size of $W \times H$) is uniformly sliced to *blocks* (size $M \times N$), which is the basic unit for the subsequent structured pruning. Then the row- and column-wise (structured) pruning is performed on each block (Fig. 1(b)). Thus, only the weights locate at the cross-points of the un-pruned rows and columns are remained. The CSB-based pruning technique integrates two-fold advantages of both the unstructured pruning and coarse-grained structured pruning. On the one hand, CSB provides adequate flexibility in model pruning. Because each block is processed independently, the pruning rate varies among blocks that help to remain the essential part of the model. On the other hand, as Fig. 1(c), the un-pruned weight values in each block compose a dense *kernel matrix* (with a size of $m \times n$) that makes the inference computation be friendly to parallel hardware.

2.2 CSB-Inference with Imbalanced Workload

The CSB-pruning scheme provides the fine-grained structured pruning block, and the pruning ratio varies among blocks. As a result, the *imbalanced block-wise workload* brings challenges to exploit the inter-block parallelism on hardware. Fig. 2 demonstrates a workload-imbalance case that results in a low hardware utilization with the basic design. The weight matrix (Fig. 2(a)) is partitioned to 4×4 blocks, and the un-pruned weight counts of each block are different. During the inference, the block-workloads are allocated to 2×2 PEGrounps that each contains multiple processing elements (PEs), as Fig. 2(b). As a result, the execution time is bounded by the PEGroup4, and the other ones are severely under-utilized. We handle this issue with the *workload-sharing* technique to process the CSBs with the imbalanced workload pattern, as demonstrated in Fig. 2(c). In the architecture part, workload-sharing paths are set for workload migration between the neighboring PEGroups. Meanwhile, the compilation scheme is proposed that analyzes the CSB-pruned model and schedules the sharing to realize a balanced execution and optimal hardware utilization.

Table 1: CSB-Pruning Evaluation Results.

RNN Application	Model Size #Weight+#Bias	Pruning Technique	Hardware Friendly	Pruning Rate	Model Size After Pruning
Speech Recognition	13.2M+8K	non-structured	NO	15.7X	840K+8K
		row-column [7]	YES	8X	1.65M+8K
		row-balanced [4]	YES	8.9X	1.48M+8K
		CSB (this paper)	YES	13.5X	978K+8K
Machine Translation	36M+12K	non-structured	NO	16.3X	2.2M+12K
		bank-blanced [1]	YES	3.3X	10.9M+12K
		CSB (this paper)	YES	12X	3M+12K

3 PRELIMINARY RESULTS

The CSB-pruning flow is realized with PyTorch, and the inference architecture is implemented on FPGA with the configuration of 256 PEs. Two benchmark RNN-models in speech recognition and machine translation domain are used to evaluate the pruning rate and the hardware performance. As Table 1 presents, for both models, our CSB method reaches the pruning rate that is close to the non-structured method. Compared to the existing hardware-friendly pruning methods, CSB-pruning achieves a significant improvement on compression rate, from 1.7× to 3.6×. With the addition of the architecture optimization, the RNN-inference latency reaches a super real-time performance of 23μs and 67μs on two benchmark models respectively.

REFERENCES

[1] Shijie Cao et al. 2019. Efficient and effective sparse LSTM on FPGA with Bank-Balanced Sparsity. In *FPGA'19*. ACM, 63–72.
[2] Kyunghyun Cho et al. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).
[3] Alex Graves et al. 2013. Speech recognition with deep recurrent neural networks. In *ICASSP'13*. IEEE, 6645–6649.
[4] Song Han et al. 2017. Ese: Efficient speech recognition engine with sparse lstm on fpga. In *FPGA'17*. ACM, 75–84.
[5] Sreelekshmy Selvin et al. 2017. Stock price prediction using LSTM, RNN and CNN-sliding window model. In *ICACCI'17*. IEEE, 1643–1647.
[6] Runbin Shi et al. 2019. E-LSTM: Efficient inference of sparse LSTM on embedded heterogeneous system. In *DAC'19*. IEEE, 1–6.
[7] Wei Wen et al. 2018. Learning intrinsic sparse structures within long short-term memory. In *ICLR'18*.