The Hardware Impact of Quantization and Pruning for Weights in Spiking Neural Networks

Clemens JS Schaefer, Pooria Taheri, Mark Horeni, and Siddharth Joshi

***Abstract*—Energy efficient implementations and deployments of Spiking neural networks (SNNs) have been of great interest due to the possibility of developing artificial systems that can achieve the computational powers and energy efficiency of the biological brain. Efficient implementations of SNNs on modern digital hardware are also inspired by advances in machine learning and deep neural networks (DNNs). Two techniques widely employed in the efficient deployment of DNNs – the quantization and pruning of parameters, can both compress the model size, reduce memory footprints, and facilitate low-latency execution. The interaction between quantization and pruning and how they might impact model performance on SNN accelerators is currently unknown. We study various combinations of pruning and quantization in isolation, cumulatively, and simultaneously (jointly) to a state-of-the-art SNN targeting gesture recognition for dynamic vision sensor cameras (DVS). We show that this state-of-the-art model is amenable to aggressive parameter quantization, not suffering from any loss in accuracy down to ternary weights. However, pruning only maintains iso-accuracy up to 80% sparsity, which results in 45% more energy than the best quantization on our architectural model. Applying both pruning and quantization can result in an accuracy loss to offer a favourable trade-off on the energy-accuracy Pareto-frontier for the given hardware configuration.**

***Index Terms*—Spiking neural networks (SNNs), quantization, pruning, sparsity, model compression, efficient inference.**

I. Introduction

SPIKING neural networks (SNNs) offer a promising way of delivering low-latency, energy-efficient, yet highly accurate machine intelligence in resource-constraint environments. SNNs promise to reduce communication and compute overheads drastically compared to their rate-coded artificial neural network (ANN) counterparts through judicious use of spike communication between neurons. Additionally, spiking temporal dynamics are well suited to event-stream tasks such as event-driven audio and vision classification problems [1], [2]. SNN accelerators incur lower costs for spike storage and communication than the data movement costs in their ANN counterparts. Practical deployments of SNN models could deliver unprecedented performance for edge-intelligence if SNN models can deliver accuracy with low-precision computations and SNN accelerators can exploit sparsity.

Parameter quantization and parameter pruning are widely used to reduce model footprint [3], [4]. Quantization, especially fixed-point integer quantization, can enable the use of cheaper lower precision arithmetic hardware and simultaneously reduce the required memory bandwidth for accessing all the parameters in a layer. On the other hand, pruning compresses a model by identifying unimportant weights and zeroing them out. Judicious use of both is critical in practical deployments of machine intelligence when the system might be constrained by size, weight, area and power (SWAP). Developing a better understanding of the interaction between quantization and pruning is critical to developing the next generation of SNN accelerators.

Efforts to quantize SNN parameters have recently gained traction. Authors in [5] developed a second-order method to allocate bit-widths per-layer an reduce the model size by 58% while only incurring an accuracy loss of 0.3% on neuromorphic MNIST. Other work [6] demonstrated up to a 2× model size reduction using quantization with minimal (2%) loss in accuracy on the DVS gesture data set. Separately, the authors in [7] demonstrated quantization in both inference and training, allowing for a memory footprint reduction of 73.78% with approx. 1% loss in accuracy on the DVS gesture data set. Similarly, authors in [8] showed that SNNs with binarized weights can still achieve high accuracy at tasks where temporal features are not task-critical (e.g., DVS Gesture 0.69% accuracy loss). They note that while on tasks highly dependent on temporal features such binarization incurs a greater degradation in accuracies hits (e.g., Heidelberg digits 16.16% accuracy loss). Authors in [9] directly optimize the energy efficiency of SNNs obtained by conversion from ANNs by adding an activation penalty and quantization-aware training to the ANN training. They demonstrated a ten-fold drop in synaptic operations while only facing a minimal (1-2%) loss in accuracy on the DVS-MNIST task.

Characterizing the interaction between hardware designs, SNN model quantization, and SNN model sparsity is critical to the design of next-generation SNN hardware platforms. Prior work has shown 5-bit parameter quantization and $10-14\times$ model compression through pruning [10]. However, existing work has primarily focused on static data sets like (neuromorphic) MNIST/CIFAR where there is minimal temporal information when compared to other event-based datasets like DVS gestures. We address this critical gap in literature by

Manuscript received 13 February 2023; revised 13 March 2023; accepted 20 March 2023. Date of publication 22 March 2023; date of current version 12 May 2023. This work was supported in part by ASCENT, one of six centers in JUMP, an SRC Program sponsored by DARPA. This brief was recommended by Associate Editor D. Liu. *(Corresponding author: Clemens JS Schaefer.)*

The authors are with the Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556 USA (e-mail: cschaef6@nd.edu; ptaheri@nd.edu; mhoreni@nd.edu; sjoshi2@nd.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSII.2023.3260701.

Digital Object Identifier 10.1109/TCSII.2023.3260701

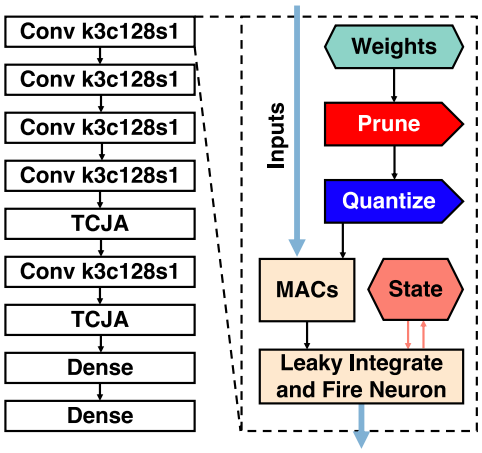


Fig. 1. On the left the schematic SNN architecture proposed by [11] using stacked spiking convolution blocks with 3×3 weight kernels, 128 output channels and striding of 1. Followed by temporal-channel joint attention (TCJA) blocks and spiking dense blocks for classification. On the right is a simplified flow within a spiking layer: weights are first pruned and then quantized (preprocessing), inputs are multiplied and accumulated with preprocessed weights before being fed into the logic of the leaky integrate and fire neurons, which additionally relies on a stored state (membrane potential).

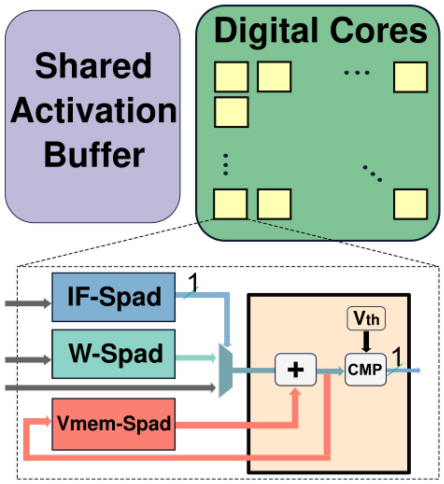


Fig. 2. Eyeriss-like spiking architecture consisting of a shared activation buffer connected to an array of digital cores. A single digital core has dedicated memory for inputs (IF-Spad), weights (W-Spad), state (Vmem-Spad) and spiking threshold (V_{th}). The digital cores possess arithmetic components, e.g., an accumulator or a comparator (CMP) for spike generation.

comprehensively analyzing the energy efficiency derived from quantizing and pruning a state-of-the-art SNN [11] trained for the DVS gestures dataset[1]. We: (i) propose two hybrid approaches which apply quantization and pruning together to the weights of SNNs, (ii) analyze the overall energy impact of quantization and pruning together as well as separately, and (iii) disentangle the effect of quantization-induced sparsity on energy consumption.

II. Methods

A. Spiking Neural Networks

Spiking neural networks are motivated by biological neural networks, where both communication and computation use action potentials. Similar to the work in [11], we use a standard leaky-integrate and fire neuron, written as:

$$u(t)_i^l = \tau u(t-1)_i^l + \sum_j s(t)_{ij}^{l-1} w_{ij}^l,$$
$$s(t)_i^l = H(u(t)_i^l - V_{th}) \quad (1)$$

where $u(t)_i^l$ is the membrane potential of the i^{th} neuron in layer l at time step t and decays by the factor τ while accumulating spiking inputs $s(t)_{ij}^{l-1}$ from the previous layer weighted by w_{ij}. We do not constrain the connectivity of these layers. When the membrane potential exceeds the spike threshold V_{th}, the neuron issues a spike (see eq. (1)) through the Heaviside step function H and resets its membrane potential to a resting potential (set to 0 in this work).

We illustrate our exemplar SNN model in 1, with max pooling and batch norm layers (in the convolutional layers) omitted for clarity. The model is trained using backpropagation-through-time with arc-tangent surrogate gradients [12], [13].

[1]Code https://github.com/Intelligent-Microsystems-Lab/SNNQuantPrune

B. Quantization

In this work, we only quantify the effects of quantizing synaptic weights, employing standard quantization techniques like clipping and rounding as described in [14]:

$$Q_u(x) = \text{clip}\left(\text{round}\left(\frac{x}{s_{\text{in}}}\right), -2^{b-1}, 2^{b-1}-1\right) \cdot s_{\text{out}}.$$

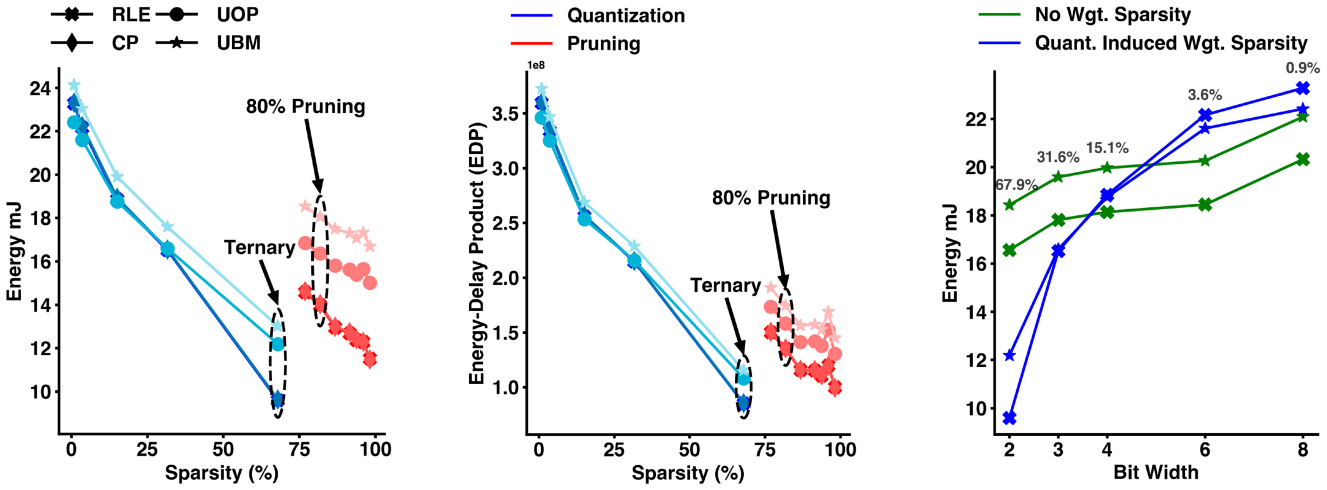
Here, Q_u is the quantizer function, b is the number of bits allocated to the quantizer, and x is the value to be quantized. Additionally, s_{in} and s_{out} are learnable scale parameters. The scale parameters are initially set to 3× the standard deviation of the underlying weight distribution added to its mean. In this work, we implement quantization-aware training through gradient scaling [15] to minimize accuracy degradation.

C. Pruning

We implement global unstructured pruning by enforcing a model-wide sparsity target (ω) across all the SNN model parameters (weights). During pruning, we rank the weights by magnitude and prune the ω% lowest.

D. Energy Evaluation

We use an Eyeriss-Like architecture [16] illustrated in Figure 2. This accelerator implements a grid of digital processing elements (PEs) with input spikes and membrane partial sums supplied from a shared activation buffer. Each digital core has scratchpad memories for input spikes, weights, neuron state, and spiking threshold. The PEs can leverage multiple sparse representation schemes, incorporating features like clock gating and input read skipping. These sparsity-aware techniques are applied for input spikes, weights, and output spikes. Across the PE grid, inputs can be shared diagonally, weights can be multicasted horizontally, and partial sums can be accumulated vertically. We removed multipliers from the PE and modified the input scratch pads to be only 1-bit wide. This architecture is evaluated for our SNN workload of



(a) (b) (c)

Fig. 3. The Effect of sparsity on energy (Fig. 3(a)) and energy-delay product (EDP) (Fig. 3(b)). Weight sparsity across different target levels (75%, 80%, 85%, 90%, 92.5%, 95%, and 97.5%) are achieved during guided pruning and compared against weight quantization across different levels of precision (8b, 6b, 4b, 3b, and terenary). Quantization-induced weight sparsity for these models is logged as the number of values set to 0 after quantization and finetuning. Results are presented across sparse representation schemes like uncompressed bitmask (UBM), uncompressed offset pair (UOP), coordinate payload (CP), and run length encoding (RLE). For models with limited sparsity (e.g., less than 25%) the UOP scheme performs best, however RLE typically outperforms other schemes at higher sparsity levels. Note although RLE and CP nearly overlap each other, RLE is strictly better, although by a small margin. The highlighted models are ternarized models and pruned models with 80% sparsity target, all delivering iso-accuracy, demonstrating a clear advantage for ternary quantization. Fig. 3(c) Disentangles the energy gains obtained from sparsity from the benefits obtained due to quantization. Quantization-induced sparsity reduces as model precision increases, leading to applying sparse representations to weights being unsuitable at higher precisions. We still employ sparse representation schemes for input and output spikes to better leverage activity sparsity in SNNs. The quantization-induced weight sparsity levels are annotated above the data points.

choice using high-level component-based energy estimation tools [17], [18] calibrated to a commercially available 28 nm CMOS node. We determine spike activity statistics through the training-time sparsity. Tensor-sparsity statistics and calibrated energy costs are used for loop analysis, where loop interchange analysis is employed to search for efficient dataflows [18]. All results enforce causality, preventing temporal loop reordering.

III. Results

We evaluate the pruning and quantization performance of the SNN model [11] on the DVS gesture data set [1] using spike-count for classification. Our baseline model has weights quantized to 8 bits while delivering 97.57% accuracy on DVS gestures. Figure 1 shows the SNN architecture and the location in the computational graph where pruning and quantization operations were applied to the weights. Finetuning experiments started with pre-trained floating point weights and lasted 50 epochs with a linear warm-up and cosine decay learning rate schedule with a peak learning rate of 0.001.

Figure 3 illustrates how different rates of sparsity achieved through pruning and quantization interact with the different compression schemes studied. These include uncompressed bitmask (UBM), uncompressed offset pair (UOP), coordinate payload (CP), and run length encoding (RLE). We see that for highly quantized SNNs, such as those resulting from ternarization the accelerator incurs lower energy and latency in computing the SNN operations when compared to pruned SNNs that deliver the same accuracy. In part, when terenerizing the weights, there are only three representation levels available, and most weights are quantized to 0. This allows ternarization to simultaneously benefit from both low-precision computing and high weight sparsity and model compression. At higher levels of precision, there is insufficient quantization-induced sparsity for the network to benefit from sparse representation (see upper left of Figs 3). At higher sparsity rates, RLE outperforms other representation formats. RLE and CP formats incur similar overheads and deliver similar performance in energy and energy-delay-product (EDP), with RLE being, on average,.3% better on both metrics. The 8b and 6b models for quantization incur similar energy/energy-delay costs, as seen by their clustering in the upper right corner of both plots in Fig. 3. In contrast, 4b, 3b, and ternary models benefit from both sparsity and reduced numerical precision, with ternarization gaining the most.

We better disentangle the benefits of quantization and quantization-induced sparsity, by examining its variation across multiple quantization levels in Fig. 3. For 8 and 6 bit weights the model incurs substantial overheads due to the additional sparse-storage format related metadata. We still employ sparse-storage formats for spike activity, with RLE storage performing better due to the extreme sparsity in spike activity. At the higher precision levels, with a crossover point for 4-b weights, employing RLE and ignoring any sparsity in weights delivers higher performance. However, for 3 bit and weight ternarization, there is a significant improvement to be derived from employing sparse storage formats in weights too. We additionally show the energy breakdown, normalized to computing energy for data movement across the accelerator in Fig. 4. The larger energy cost of transferring metadata from DRAM for 8 bit sparse weights results in greater energy incurred for the entire model. Remarkably, the ternarization includes significantly more utilization of the intermediate memories which in turn leads to improved energy-efficiency.

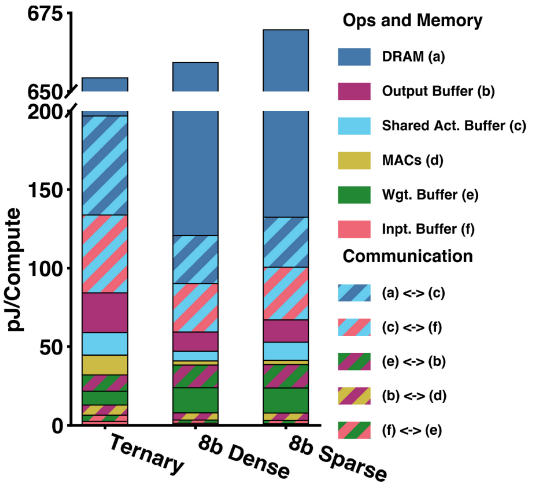


Fig. 4. A breakdown of normalized energy for different data-movement and operations for three different configurations. First, we see that ternarization incurs minimal DRAM access which is where significant efficiency is gained. Next, for the configuration with 8 bit sparse models, storing and accessing the additional metadata incurs significant costs. This metadata overhead does not exist for the dense 8 bit model. In each case, we employ RLE sparse-storage. Data movement to and from the shared activation buffer incurs the second highest energy costs in our analysis.

Although both pruning and quantization try to leverage model robustness to improve energy, they operate along different principles. Quantization leverages model overparameterization to facilitate computations at a lower precision while pruning reduces the redundancy in the model parameters to compress it. Consequently, pruning and quantization can often be at odds, requiring a careful study of their interaction. We study two strategies for pruning and quantizing SNNs: i) *cumulatively*, where the model is first finetuned with pruning, and after half the finetuning epochs, we commence quantization aware training, and ii) *jointly*, where the model is simultaneously pruned and quantized (as shown in Fig. 1).

Figure 5 demonstrates how the different strategies for quantizing and pruning impact the model accuracy and energy-efficiency (top) / energy-delay product (bottom) for our architecture of choice. Although 80% pruning does not yet suffer from accuracy loss, it is 45% costlier than ternary quantization, which can maintain accuracy for this model. If constrained to iso-accuracy, a pure quantization approach outperforms all other alternatives, with the 3-bit quantization with 80% pruning occupying a larger footprint than the ternary network. However, more fine-grained trade-offs between the model accuracy and energy can be achieved across various combined strategies, as shown in the inset of Fig. 5. The mixed pruning and quantization schemes enable operation along the Pareto curve when some loss in accuracy can be tolerated. Cumulative quantization and pruning maintains accuracy in low-energy regimes, however, this doesn't translate to EDP improvements due to encoding overhead.

We examine how the layer sparsity statistics change across the best iso-accuracy model resulting from quantization, pruning, cumulative quantization & pruning and joint quantization & pruning in Figure 6. We observe that the pruning and mixed schemes all have a relatively similar sparsity distribution among layers, with the only difference being that the

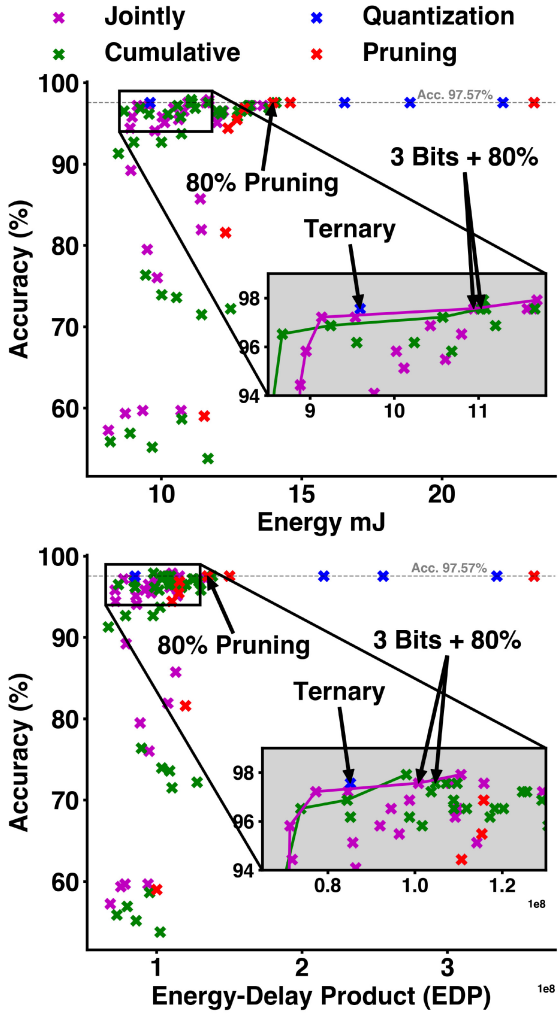


Fig. 5. Illustration of accuracy vs. energy trade-off (top) and accuracy vs. tradeoff between energy-delay product (bottom) for studied model compression techniques. We study the following interactions between pruning and quantization: quantization only, pruning only, cumulative quantized & pruned, and jointly quantized & pruned models. Inset of the figure highlights the points where the models start to degrade from iso-accuracy. Note that all quantized models down to ternary quantization maintain iso-accuracy meanwhile, pruned models degrade at higher energy cost or EDP. Mixed model using both quantization and pruning form the Pareto-curve once model accuracy degrades below iso-accuracy.

joint and cumulative approaches can deliver higher sparsity on some layers (e.g., the first TCJA-t). On the other hand, the ternary quantization scheme presents a significantly different sparsity distribution across layers. We propose that the significant energy advantages from terenarization can be attributed to the combination of ternary encoding and consistently higher sparsity levels achieved for the first layer, the last layer, and the two TCJA layers. The different sparsity levels across layers of the iso-accuracy models also hint at commonly observed non-uniform impact for different layer types on energy consumption, e.g., different tensor sizes affecting compiler mappings.

Previous work on quantized learning showed that SNN models could be compressed by 73.78% of the original footprint with a 1.04% degradation on test error on DVS gestures [7]. However, due to differences in the base model (baseline accuracy of 87.85% vs 97.57% in this work) and that work

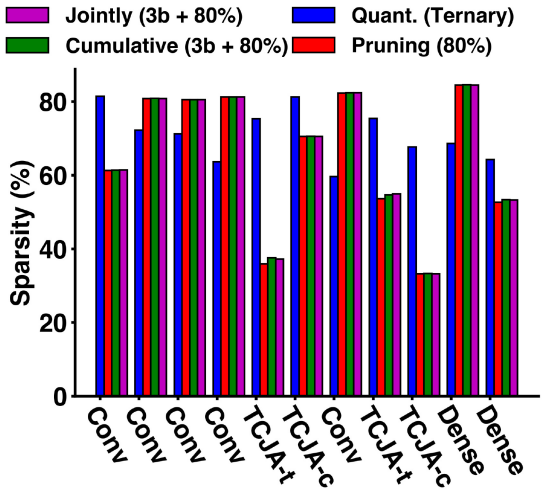


Fig. 6. Sparsity insight into the SNN model for the smallest iso-accurate models for quantization, pruning, cumulative quantization & pruning and joint quantization & pruning. The two mixed approaches replicate the internal sparsity distribution of the pruned model, meanwhile ternary quantization results in a significantly different sparsity distribution (note that the overall sparsity of the ternarized model is 67.9%).

having a different focus (quantizing learning vs. inference), a direct comparison with our work is not possible. Layer-wise pruning using second-order information has demonstrated benefits in both ANNs and SNNs compression [3], [5], where an SNN model trained for NMNIST could be shrunk by 58% with a negligible accuracy loss. On the MNIST data set, the authors in [19] show hardware-aware SNN compression delivering 90.1% accuracy with a corresponding estimated 3.1× and 4× improvement in energy and area, respectively. More closely related hardware-aware pruning and quantization efforts have trained VGG-based SNNs on CIFAR that were compressed with pruning by $10-14\times$ with an estimated $8-14\times$ reduction in energy while maintaining accuracy down to 5-bit quantization [10]. Other works targeting the CIFAR dataset use weight and connectivity training to achieve an $\approx 99\times$ compression with an accuracy loss of $\approx 3.5\%$ [20]. The authors in [21] demonstrate a 75% reduction in weights through pruning while achieving a $\approx 90\%$ accuracy on MNIST.

IV. Conclusion

We analyze the energy-accuracy trade-off between quantization and pruning in state-of-the-art SNNs. We also provide a realistic analysis of how quantization and pruning might interact with a baseline digital SNN accelerator. Our results showed that exploiting quantization-induced sparsity, particularly for weight ternarization, can deliver tenfold improvement to EDP without loss of accuracy. Additional fine-grained control can be achieved by further trading-off energy and accuracy by employing hybrid pruning and quantization schemes to deliver multiple models that occupy the accuracy-efficiency frontier.

References

[1] A. Amir et al., "A low power, fully event-based gesture recognition system," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 7243–7252.
[2] B. Cramer, Y. Stradmann, J. Schemmel, and F. Zenke, "The heidelberg spiking data sets for the systematic evaluation of spiking neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 7, pp. 2744–2757, Jul. 2022.
[3] Z. Dong, Z. Yao, D. Arfeen, A. Gholami, M. W. Mahoney, and K. Keutzer, "HAWQ-V2: Hessian aware trace-weighted quantization of neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 18518–18529.
[4] Z. Zhuang et al., "Discrimination-aware channel pruning for deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 883–894.
[5] H. W. Lui and E. Neftci, "Hessian aware quantization of spiking neural networks," in *Proc. Int. Conf. Neuromorph. Syst.*, 2021, pp. 1–5.
[6] R. V. W. Putra and M. Shafique, "*Q*-SpiNN: A framework for quantizing spiking neural networks," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, 2021, pp. 1–8.
[7] C. J. Schaefer and S. Joshi, "Quantizing spiking neural networks with integers," in *Proc. Int. Conf. Neuromorph. Syst.*, 2020, pp. 1–8.
[8] J. K. Eshraghian and W. D. Lu, "The fine line between dead neurons and sparsity in binarized spiking neural networks," 2022, *arXiv:2201.11915*.
[9] M. Sorbaro, Q. Liu, M. Bortone, and S. Sheik, "Optimizing the energy consumption of spiking neural networks for neuromorphic applications," *Front. Neurosci.*, vol. 14, p. 662, Jun. 2020.
[10] S. S. Chowdhury, I. Garg, and K. Roy, "Spatio-temporal pruning and quantization for low-latency spiking neural networks," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, 2021, pp. 1–9.
[11] R.-J. Zhu et al., "TCJA-SNN: Temporal-channel joint attention for spiking neural networks," 2022, *arXiv:2206.10177*.
[12] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," *IEEE Signal Process. Mag.*, vol. 36, no. 6, pp. 51–63, Nov. 2019.
[13] J. H. Lee, T. Delbruck, and M. Pfeiffer, "Training deep spiking neural networks using backpropagation," *Front. Neurosci.*, vol. 10, p. 508, Nov. 2016.
[14] E. Park and S. Yoo, "Profit: A novel training method for sub-4-bit mobilenet models," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 430–446.
[15] J. Lee, D. Kim, and B. Ham, "Network quantization with element-wise gradient scaling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 6448–6457.
[16] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE J. Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, Jan. 2017.
[17] Y. N. Wu, J. S. Emer, and V. Sze, "Accelergy: An architecture-level energy estimation methodology for accelerator designs," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, 2019, pp. 1–8.
[18] A. Parashar et al., "Timeloop: A systematic approach to DNN accelerator evaluation," in *Proc. IEEE Int. Symp. Perform. Anal. Syst. Softw. (ISPASS)*, 2019, pp. 304–315.
[19] N. Rathi, P. Panda, and K. Roy, "STDP-based pruning of connections and weight quantization in spiking neural networks for energy-efficient recognition," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 4, pp. 668–677, Apr. 2019.
[20] Y. Chen, Z. Yu, W. Fang, T. Huang, and Y. Tian, "Pruning of deep spiking neural networks through gradient rewiring," 2021, *arXiv:2105.04916*.
[21] Y. Shi, L. Nguyen, S. Oh, X. Liu, and D. Kuzum, "A soft-pruning method applied during training of spiking neural networks for in-memory computing applications," *Front. Neurosci.*, vol. 13, p. 405, Apr. 2019.