Sparsity Through Spiking Convolutional Neural Network for Audio Classification at the Edge

Cong Sheng Leow*, Wang Ling Goh†, Yuan Gao*
Email:leowcs@ime.a-star.edu.sg, ewlgoh@ntu.edu.sg, gaoy@ime.a-star.edu.sg
*Institute of Microelectronics (IME), Agency for Science, Technology and Research (A*STAR), Republic of Singapore
†School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Republic of Singapore

***Abstract*—Convolutional neural networks (CNNs) have shown to be effective for audio classification. However, deep CNNs can be computationally heavy and unsuitable for edge intelligence as embedded devices are generally constrained by memory and energy requirements. Spiking neural networks (SNNs) offer potential as energy-efficient networks but typically underperform typical deep neural networks in accuracy. This paper proposes a spiking convolutional neural network (SCNN) that exhibits excellent accuracy of above 98 % on a multi-class audio classification task. Accuracy remains high with weight quantization to INT8-precision. Additionally, this paper examines the role of neuron parameters in co-optimizing activation sparsity and accuracy.**

***Index Terms*—spiking neural networks, voice detection, convolutional neural networks, neuromorphic**

I. Introduction

Artificial neural networks (ANNs) and convolutional neural networks (CNNs) can excel in audio classification using both temporal and spectral features at the expense of computational resources [1], [2]. The sparse information encoding in bio-inspired spiking neural networks (SNNs) can potentially help with energy efficiency for edge implementations [3]. In a recent study, [4] have shown that optimization on both weight sparsity and activation sparsity can yield greater energy efficiency than just exploiting weight sparsity.

However, SNNs generally underperform the ANNs and CNNS [3], [5]. In the quest for energy-efficient audio classifiers, different SNN architectures have been explored in recent years in various forms as seen in [6]–[8]. Thus, it is of interest to explore the combination of SNN and CNN to achieve a balance of performance and energy efficiency.

- Propose an SCNN architecture for multiclass classification for voice detection at the edge using backpropagation techniques.
- Examine the impact of first spiking layer encoding on the optimization for sparsity and accuracy.
- Examine effectiveness of Random and Bayesian optimization for SCNN's neuron parameters.

Section II details the implementation the experiment using Python. Section III provides and discusses the results obtained before concluding in section IV.

II. Architecture Implementation

A. Experiment

Going beyond binary classification problems, the free-spoken digit dataset (FSDD) was used where an audio recording of spoken digits '0' to '9' was sampled at 8 kHz [9]. For each audio recording, the period(s) where there is silence was trimmed based on a cut-off threshold of 10 dB. 16 MFCC filters were used to extract MFCCs from each audio at a hop length of 512. To feed MFCCs into the networks in Fig. 1, zero-padding was used to pad in both the horizontal and vertical dimensions as padding has shown to have minimal to no impact on the performance of CNNs [10].

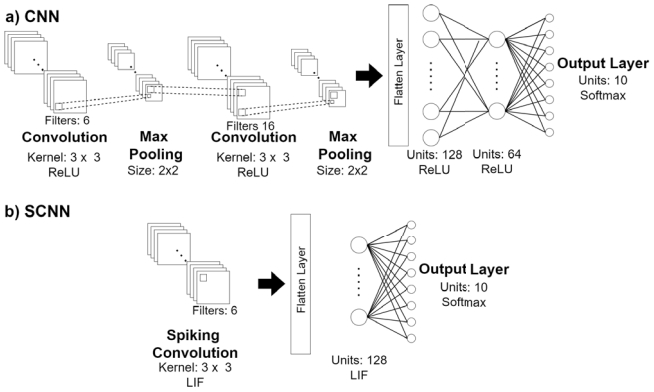
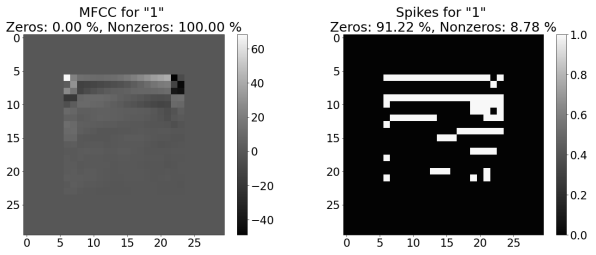


Fig. 1. Architecture of a (a) convolutional neural network and the (b) spiking convolutional neural network proposed for voice keyword classification.

A train-test split of 80-20 for 20 epochs was used to train the networks (sufficient for unity training accuracy), with the average accuracy obtained for more than 50 iterations. The Adam optimizer with cross entropy loss was used to optimize the network and evaluate the loss. To approximate the computational resources in terms of energy and memory, floating point operations per second (FLOPS) and number of network parameters were used respectively, with FLOPS calculated through *fvcore*, a package developed by the Meta Artificial Intelligence Research Team [11].

B. CNN

The CNN proposed by Adhish Thite (one of the contributors to the FSDD [9]) was used as a reference due to its effectiveness in the dataset and shallower architecture as compared to the ones in [1]. The network was optimized to obtain the architecture shown in Fig. 1.



(a) CNN Output (b) SCNN Output

Fig. 2. The output of one filter has an increase in the number of zeros or sparsity from 0.0 % in the first layer after convolution (left) to 91.2 % after the spiking activation (right).

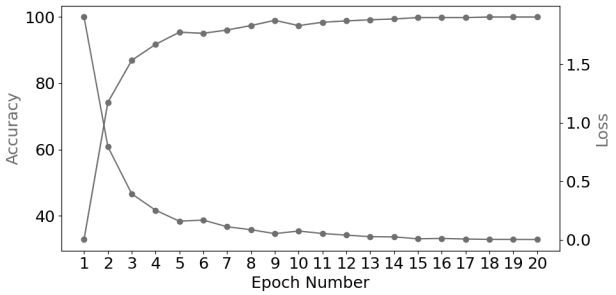


Fig. 3. Accuracy (left axis, red) and loss (right axis, blue) against epoch number for SCNN training.

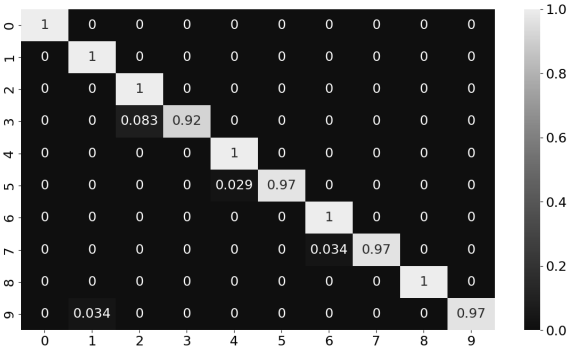


Fig. 4. Confusion matrix obtained for FSDD classification using SCNN with architecture in Model B (true labels along the rows and predicted labels along the columns).

C. SCNN

Using the open-source package *snnTorch* [12] with *PyTorch*, a CNN can be converted to SCNN such as the one proposed in Fig. 1 with autograd available [13]. The leaky integrate-and-fire (LIF) neuron was chosen for its effectiveness and simplicity [5].

Traditionally, training SNNs has been challenging since conventional gradient descent cannot be applied to the non-differentiable spikes [3]. While alternatives such as spike-time-dependent plasticity (STDP) has been proposed, the implementation is challenging and the performance of the network is typically poor as compared to their DNN-counterparts. *SnnTorch* makes backpropagation possible for training the SNNs through surrogate gradient [14]. The default neuron using the sigmoid function as the surrogate gradient function was implemented with a slope of 25 for a total of 10 time steps at a threshold of 1.0.

To simulate hardware performance of the SCNN proposed, the models were quantized to INT8 (8-bit integer) precision and re-evaluated through quantization-aware training (QAT). The quantization was implemented using the cosine annealing method proposed by [15] through *Brevitas* [16] and *snnTorch*. Since QAT uses floating-point operations, FLOPS remains constant.

D. Hyperparameter Tuning

Both manual search and automated search were deployed to study the optimal LIF neuron parameters in the SCNN. The three main parameters varied were the threshold voltage, the voltage decay rate - beta, and the number of time steps for the spiking layers. The activation sparsity of the first layer output was examined further for its role in propagating sparsity throughout the network. For the automated search, *Tune* [17] was used to implement Random and Bayesian optimization. As the direct impact of sparsity on the hardware energy efficiency is unclear, a multi-objective optimization is not possible.

III. Experimental Results

A. Comparison between CNN and SCNN

Table I shows the accuracy of SCNN models with standard deviation, parameter numbers and FLOPS. At 98.7 %, the initial SCNN model A performed better than the CNN at 97.3 % with the training plot in Fig. 3. However, the CNN used 83.702K parameters and 3.326M FLOPS while the SCNN used 1.616M parameters and 0.281G FLOPS, showing that hardware exploitation is required for exploiting the sparsity as seen in Fig. 2. With a reduction of the time the computational expenditure reduced significantly at the expense of the accuracy as seen in Table I.

To further reduce compute, the SCNN was reduced to just 3 layers in Model B - 1 convolutional layer and 2 fully-connected layers. The SCNN performed well and remained resilient to the architectural changes as seen in Table I. Model B, did not suffer accuracy degradation ($\leq$ 1.0 %) from Model A and the confusion matrix is shown in Fig. 4. Additionally, the leaner network has significant improvement in the number of parameters and FLOPS over Model A.

While pooling can further reduce compute, the spiking layer does not work well with max-pooling applied on the output spike map (see Fig. 2. Using a 6C3-P2-128FC-10FC (P2 referring to max-pooling with size 2) SCNN architecture at 10 time steps, a 93.9 % test accuracy was obtained at 0.183M parameters and 26.722M FLOPS. The accuracy dropped more than 4.0 % when compared to both Model A and B as any error or information loss due to pooling was propagated throughout the network.

TABLE I
PERFORMANCE OF DIFFERENT ARCHITECTURES USING FSDD.

Model	SCNN Architecture*	Test Accuracy	Quantized Test Accuracy(INT8)	No. of parameters	FLOPS
A	6C3-16C3-128FC-64FC-10FC	98.7±0.57 %	74.1±9.22 %	1.616 M	281.000 M
	6C3-16C3-128FC-64FC-10FC (only 1 timestep)	95.4±3.24 %	61.6±12.30 %		28.0850 M
B	6C3-128FC-10FC	98.4±0.35 %	97.0±0.77 %	0.702 M	88.930 M
	6C3-128FC-10FC (only 1 timestep)	93.1±1.77 %	91.3±2.40 %		8.8930 M

* nCx refers to an n-channel convolutional neural network with kernel size x, and yFC refers to fully connected layers with y number of neurons.

B. Quantization results

Table I shows the quantization test accuracy without quantized states. Model B remains resilient to quantization with a slight 1.0 % drop in accuracy. Similar to the unquantized models, time steps play a significant role in providing high performance as seen in Table I. However, Model A suffered severe degradation after quantization, which can be attributed to the increase in quantization error with increased layers as compared to Model B. Furthermore, the increased standard deviation after quantization is worth further study. Nonetheless, the resiliency after quantization of Model B instilled confidence in efficient hardware-level implementation.

C. Comparison with related works

Table II compares this work with related solutions for audio classification. As compared to ANN in [2], this work was able to achieve just around 1.0 % accuracy drop with a shallower network. Compared to CNN using IA-optimization in [18], Model B was able to perform better at the same network depth.

This work was able to achieve higher performance than most other SNN solutions and achieve competitive performance when quantized. Most existing SNNs for audio classification were trained at FP32 or FP64 precision, which is not representative to hardware implementation. Similar to [14], this work confirms the ability for SNN-based solution to achieve competitive accuracy through surrogate-gradient backpropagation. Moreover, increased activation sparsity as seen in Fig. 2 exhibits potential to be exploited for increased energy-saving such as in [4].

In the more recent works proposed such as in [6] and [19], the networks requires spiking dataset which is non-trivial without specialized hardware. Although the proposed network cannot achieve direct spike-encoding such as in [8], the use of well-tested MFCCs enables acceleration opportunities to be done for potential energy-saving [20].

D. Hyperparameter, Sparsity, and Accuracy

In the manual search for both Model A and B, the accuracy decreased and sparsity increased with the increase in threshold voltage as seen in Fig. 5. However, the number of time steps and beta did not exhibit any clear relationship. Based on the manual search, the search space of the automated search was constrained.

Based on the automated search results in Fig. 6, Bayesian search obtained the highest accuracy for both Model A and B, with Model B outperforming Model A. On the other hand, random search worked better for Model A than Model B. A higher number of time steps was observed for higher accuracy in Model B while beta and threshold value did not matter

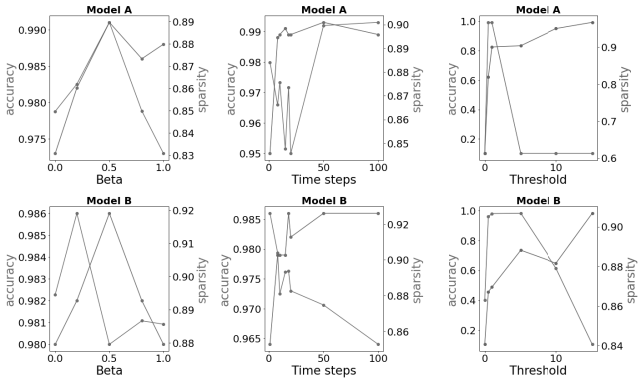


Fig. 5. The plots for accuracy (left axis) and sparsity (right axis) for the different SCNN architectures and the different hyperparameters. Rows 1, and 2 (top to bottom) show the results for Model A, and B respectively. Columns 1, 2, and 3 (left to right) show the results when beta, time steps, and threshold were varied respectively.

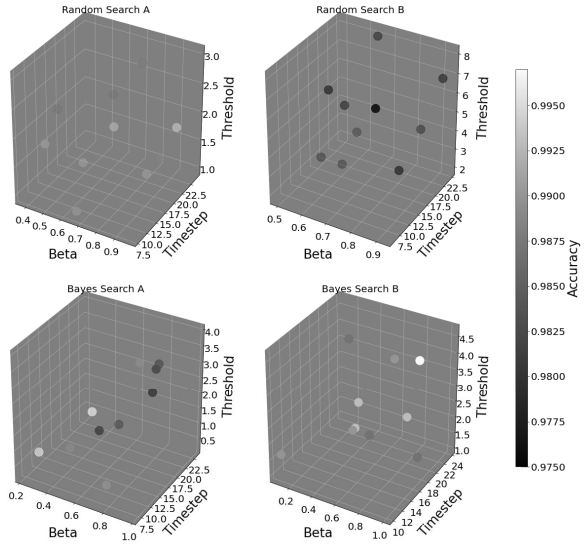


Fig. 6. The 3D scatter plots obtained from the top 10 searches for Model A (left) and B (right) using random search (top) and Bayesian search (bottom).

TABLE II
COMPARISON OF PROPOSED ARCHITECTURE WITH OTHER CNN AND SNN SOLUTIONS.

References	**[7]**	**[8]**	**[2]**	**[18]**	**[6]**	**[19]**	**This work**
Architecture Type	SOM-SNN	Spiking + CNN	ANN	CNN	SRNN	CSNN	SCNN (Model B)
Number of layers	3	3	5	3	3	3 Hidden layers	3
Precision	FP64	FP64	FP32	Not Stated	FP32	FP32	FP32 / INT8
Dataset	TIDIGITS ("0"-"9")	FSDD ("0"-"9")	FSDD ("0"-"9")	FSDD ("0"-"9")	Spiking Heidelberg Digits (SHD) / GSC / TIMIT	SHD	FSDD ("0"-"9")
Training Method	Maximum-Margin Tempotron	Backpropagation	Backpropagation	IA-Optimization	Backpropagation with multi-Gaussian surrogate gradient	Backpropagation with Surrogate Gradient and Kaiming initialization	Backpropagation with Surrogate Gradient
Test Accuracy	97.4 %	96.0 %	99.7 %	89.9 %	90.4 % / 92.1 % / 66.1 %	83.1 % (95.9 % Validation accuracy)	**98.4 %** (FP32) / **97.0 %** (INT8)

as much in the automated search. However, large time step is detrimental for energy efficiency. Alternatively, threshold voltage can be utilized for sparsity exploitation in hardware as observed in the manual search.

IV. CONCLUSION

This paper proposes a SCNN for voice keyword classification which achieved 98.4 % test accuracy (Model B) with minimal degradation after quantization. Based on automated optimization guided by initial manual exploration, the number of time steps and threshold voltage were key for accuracy and activation sparsity respectively. Further analysis of the relationship between the hyperparameters and actual hardware implementation will offer greater insights into actual performance, computational cost, and training with energy-saving methods through sparsity in SNN solutions.

ACKNOWLEDGMENT

This work was supported by the Agency for Science, Technology and Research (A*STAR), Singapore under the Cyber-Physiochemical Interface programme, Grant No. A18A1b0045.

REFERENCES

[1] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson, "Cnn architectures for large-scale audio classification," in *2017 IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, 2017, pp. 131–135, doi:10.1109/ICASSP.2017.7952132.

[2] J. Oruh, S. Viriri, and M. F. Ijaz, "Deep learning-based classification of spoken english digits," *Comput. Intell., Neurosci.*, vol. 2022, 2022, doi:10.1155/2022/3364141.

[3] M. Pfeiffer and T. Pfeil, "Deep learning with spiking neurons: Opportunities and challenges," *Frontiers Neurosci.*, vol. 12, 2018, doi:10.3389/fnins.2018.00774.

[4] K. L. Hunter, L. Spracklen, and S. Ahmad, "Two sparsities are better than one: Unlocking the performance benefits of sparse-sparse networks," 2021, *arXiv:2112.13896* .

[5] J. D. Nunes, M. Carvalho, D. Carneiro, and J. S. Cardoso, "Spiking neural networks: A survey," *IEEE Access*, vol. 10, pp. 60 738–60 764, 2022, doi:10.1109/ACCESS.2022.3179968.

[6] B. Yin, F. Corradi, and S. M. Bohté, "Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks," *Nature Mach. Intell.*, vol. 3, no. 10, pp. 905–913, 2021, doi:10.1038/s42256-021-00397-w.

[7] J. Wu, Y. Chua, M. Zhang, H. Li, and K. C. Tan, "A spiking neural network framework for robust sound classification," *Frontiers Neurosci.*, vol. 12, 2018, doi:10.3389/fnins.2018.00836.

[8] J. Hu, W. L. Goh, Z. Zhang, and Y. Gao, "Voice keyword recognition based on spiking convolutional neural network for human-machine interface," in *2020 3rd Int. Conf. Intell. Auton Syst. (ICoIAS)*, 2020, pp. 77–82, doi:10.1109/ICoIAS49312.2020.9081859.

[9] Z. Jackson, C. Souza, J. Flaks, Y. Pan, H. Nicolas, and A. Thite, "Jakobovski/free-spoken-digit-dataset: v1.0.8," Aug. 2018, doi:10.5281/zenodo.1342401.

[10] M. Hashemi, "Enlarging smaller images before inputting into convolutional neural network: zero-padding vs. interpolation," *J. Big Data*, vol. 6, no. 1, p. 98, Nov. 2019, doi:10.1186/s40537-019-0263-7.

[11] M. Research, "fvcore (commit 279b7d0)," Sep. 2022. [Online]. Available: https://github.com/facebookresearch/fvcore

[12] J. K. Eshraghian, M. Ward, E. Neftci, X. Wang, G. Lenz, G. Dwivedi, M. Bennamoun, D. S. Jeong, and W. D. Lu, "Training spiking neural networks using lessons from deep learning," 2021, *arXiv:2109.12894*.

[13] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *Neural Inf. Process. Syst. (NIPS) Autodiff Workshop*, 2017.

[14] B. Cramer, S. Billaudelle, S. Kanya, A. Leibfried, A. Grübl, V. Karasenko, C. Pehle, K. Schreiber, Y. Stradmann, J. Weis, J. Schemmel, and F. Zenke, "Surrogate gradients for analog neuromorphic computing," *Proc. Nat. Acad. Sci.*, vol. 119, no. 4, p. e2109194119, 2022, doi:10.1073/pnas.2109194119.

[15] J. K. Eshraghian, C. Lammie, M. R. Azghadi, and W. D. Lu, "Navigating local minima in quantized spiking neural networks," 2022, *arXiv:2202.07221*.

[16] A. Pappalardo, "Xilinx/brevitas," (2021), doi:10.5281/zenodo.5779154.

[17] R. Liaw, E. Liang, R. Nishihara, P. Moritz, J. E. Gonzalez, and I. Stoica, "Tune: A research platform for distributed model selection and training," 2018, *arXiv:1807.05118*.

[18] Y. Zhang, W. Yu, L. He, L. Cui, and G. Cheng, "Signals classification based on ia-optimal cnn," *Neural Comput., Appl.*, vol. 33, no. 15, pp. 9703–9721, 2021.

[19] J. Rossbroich, J. Gygax, and F. Zenke, "Fluctuation-driven initialization for spiking neural network training," *Neuromorphic Comput. Eng.*, 2022, doi:10.1088/2634-4386/ac97bb.

[20] K. Yang, L. Zhu, and W. Shan, "Design of an ultra-low Power MFCC Feature Extraction Circuit with Embedded Speech Activity Detector," in *2021 IEEE Int. Conf. Integr. Circuits, Technol. Appl. (ICTA)*, Nov. 2021, pp. 82–83, doi:10.1109/ICTA53157.2021.9661980.