CONVERSION OF SPARSE ARTIFICIAL NEURAL NETWORK TO SPARSE SPIKING NEURAL NETWORK CAN SAVE UP TO 99% OF ENERGY

Anonymous authors
Paper under double-blind review

ABSTRACT

Artificial Neural Networks (ANNs) are becoming increasingly important but face the challenge of the large scale and high energy consumption. Dynamic Sparse Training (DST) aims to reduce the memory and energy consumption of ANNs by learning sparse network topologies, which ultimately results in structural connection sparsity. Meanwhile, Spiking Neural Networks (SNNs) have attracted increasing attention due to their biological plausibility and event-driven nature, which ultimately results in temporal sparsity. To bypass the difficulty of directly training SNNs, converting pre-trained ANNs to SNNs (ANN2SNN) is becoming a popular approach to obtain high-performance SNNs. Here for the first time, we investigated the advantage of dynamically spare trained ANNs for conversion into sparse SNNs. By adopting Cannistraci-Hebb Training (CHT), a state-of-the-art brain-inspired DST family that resembles synaptic turnover during neuronal connectivity learning in brain circuits, we examined the extent to which connectivity sparsity impacts the accuracy and energy efficiency of SNNs across different conversion approaches. The results show that sparse SNNs can achieve accuracy comparable to or even surpassing that of dense SNNs. Moreover, sparse SNNs can reduce energy consumption by up to 99% compared with dense SNNs. Furthermore, driven by the interest in understanding the physical dynamic interactions between firing rate and accuracy in SNNs, we systematically analyzed the temporal relationship between the saturation of firing rate and accuracy in SNNs. Our results reveal a significant time lag where firing rate saturation precedes accuracy saturation. We also demonstrate that the magnitude of this time lag is significantly different between sparse and dense networks, where the average time lag of sparse SNNs is higher than that of dense SNNs. By combining the structural sparsity of DST and temporal sparsity of SNNs, we make a step forward to the brain-like computational network architecture with high performance and energy efficiency.

1 INTRODUCTION

Artificial neural networks (ANNs) have become central to various scientific, industrial, and economical applications. However, the high memory and energy costs of fully-connected ANNs create an urgent need for network architectures that consume much lower energy but still deliver comparable performance. One promising direction to reduce energy consumption is to sparsify a neural network(Blalock et al., 2020). Dynamic sparse training (DST), where both weights and topology evolve during training, has shown promises to discover sparse topology with close-to-dense performance and fewer links in ANNs(Mocanu et al., 2018; Jayakumar et al., 2020; Evci et al., 2020; Yuan et al., 2021; Zhang et al., 2024b). On hardwares that support sparse computation(Mishra et al., 2021; Dai et al., 2024), sparse networks require less computation, which translates the structural connection sparsity into energy savings(Tmamna et al., 2024).

Spiking Neural Networks (SNNs) are networks consisting of brain-inspired spiking neurons. Unlike neurons in ANNs, spiking neurons process information in time series with event-driven spikes(Eshraghian et al., 2023). Because of the event-driven nature of SNNs(Modaresi et al., 2023) and temporal sparsity of activations, SNNs thus hold promises to be far more energy-efficient

than conventional ANNs on neuromorphic hardware(Davies et al., 2018; Merolla et al., 2014; DeBole et al., 2019; Song et al., 2022; Huo et al., 2023; Zhang et al., 2024a). In practice, however, training high-performance SNNs directly remains challenging due to the non-differentiablity of spiking neurons(Li et al., 2024), which has motivated alternative strategies for obtaining performant SNNs. One popular option is ANN2SNN conversion: train a high-performance ANN with conventional methods and convert it to SNN. Conversion bypasses difficulties of direct training by utilizing well-established ANN training method. SNN conversion often yields near-ANN accuracy at low latency(Li et al., 2021; Bu et al., 2023; Huang et al., 2024; Wang et al., 2025), while being much more energy-efficient thanks to its event-driven nature.

However, to the best of our knowledge, prior ANN2SNN conversion works have focused most exclusively on dense networks(Li et al., 2021; Bu et al., 2023; Huang et al., 2024; Wang et al., 2025; Song et al., 2022; Han et al., 2020; Jiang et al., 2023), while conversion on dynamically sparse trained networks has never been studied. This gap is important because introducing structural connection sparsity into SNN conversion could combine benefits from both sides: the event-driven nature of SNNs reduces computation at temporal level, and introducing sparsity into structural connection can further reduce computation at structural level. Thus here, we extensively explored this intersection.

In this article we adopt established ANN2SNN algorithms(Yang et al., 2025; Wang et al., 2022; Chen et al., 2024) to investigate the extent to which introducing structural sparsity into these methods could influence the accuracy and energy consumption. Cannistraci-Hebb Training(CHT) as a brain-inspired, network-science-driven dynamic sparse training family, could learn not only weight but also topology of the networks, achieving close or even superior performance compared to the dense counterparts in ANNs(Zhang et al., 2024b; 2025; Hanming et al., 2025). Here we employ 2 sparse ANN structures, MLP(Rumelhart et al., 1986) and VGG-16(Yu et al., 2021), trained with CHT for ANN2SNN conversion through three representative conversion approaches that operate on different principles: CS-QCFS(Yang et al., 2025), SNM(Wang et al., 2022), and AEC(Chen et al., 2024). Our results show that, across all 3 conversion methods and 2 ANN structures, sparse SNNs can achieve close or even superior accuracy compared with their dense counterparts. More importantly, regardless of the SNN conversion method, sparse SNNs can achieve a remarkable energy reduction(as large as 99%).

Driven by the interest in investigating the underlying mechanism of how SNNs achieve high-performance while consuming less energy compared with ANNs, we quantitatively studied the temporal relationship in the aspect of saturation between SNNs' accuracy and firing rate where firing rate directly influences the energy consumption. The saturation of dense SNNs's accuracy is commonly observed(Song et al., 2022; Han et al., 2020; Jiang et al., 2023; Bu et al., 2022; You et al., 2024). However, the quantitative relationship between saturation time of accuracy and firing rate has never been studied in current literature. Our results show that there exists a significant phenomenon that in SNNs firing rate saturate before accuracy. What is more interesting is that there also exists a significant difference in positive time lag between sparse and dense networks, which could be a potential cause of the energy and performance advantage of sparse SNNs over their dense counterparts.

By combining the sparsity in temporal activation domain and structural connection domain, this work makes a step forward to brain-like, low-power consumption neural network.

2 METHODS

In this section we first introduce methods related to sparse ANN training and sparse SNN conversion. Please see Appendix A for details of 3 conversion methods. Secondly, the energy measurement for SNNs and the principle of energy-saving by sparsity is explained. Thirdly, we demonstrate how to judge the saturation point. Finally, we present our experiment setup.

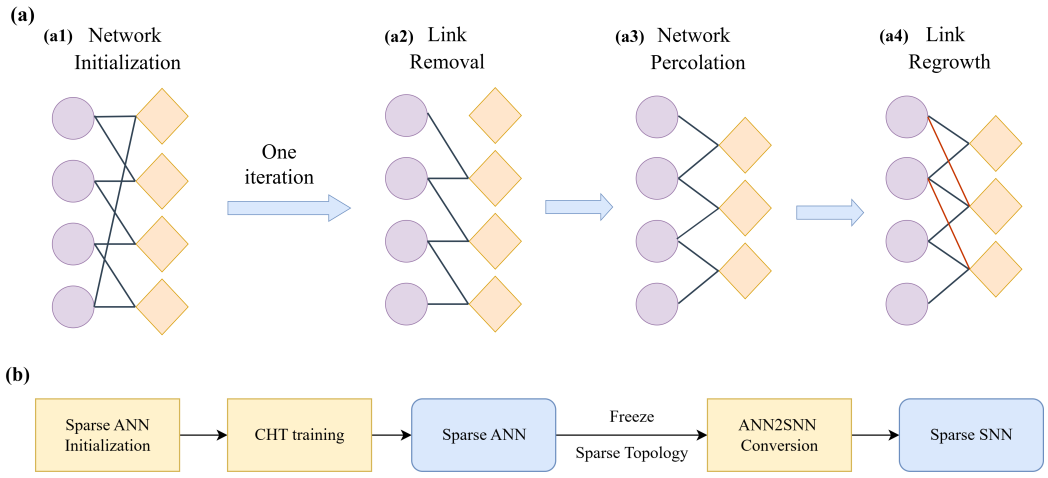


Figure 1: Overview of employed methods. (a) Illustration of CHT. (b) experiment pipeline for sparse models

2.1 Sparse ANN training & sparse SNN conversion

2.1.1 Cannistraci-Hebb Training

A sparse neural network is one neural network in which the number of connections between two layers is less than in a fully connected network of the same architecture. Cannistraci-Hebb Training (CHT) (Zhang et al., 2024b; Hanming et al., 2025; Zhang et al., 2025) is a network-driven and brain-inspired dynamic sparse training (DST) family that aims to learn the connectivity in networks. Here, sparsity level denotes the fraction of removed links compared with fully connected layer. During topology evolution a fraction of individual connections are canceled and new connections(generally, but not necessary, in the same amount of canceled) are grown.

Figure 1(a) presents the mechanism of CHT. First, a sparse network topology and weights are initialized. For each topology evolution iteration during training, the procedure is:

(i) remove links according to their absolute weights.

(ii) remove inactive neuron and incomplete path (network percolation).

(iii) predict and grow new links using CHT network link prediction.

In this study, CHT soft rule (CHTs)(Zhang et al., 2025) is used on MLP with sparsity level of 99 %, and CHT-Conv(Hanming et al., 2025) is used on VGG-16 with sparsity level of 50%.

2.1.2 Sparse SNN conversion methods adaption

To adapt existing SNN conversion methods which are originally designed for dense SNN into sparse SNN, we do following adaption (see Figure 1(b)): First a sparse ANN is trained using CHT, then we record its topology and then convert the pre-trained sparse ANN to SNN. Topology of sparse SNN is frozen during the conversion.

2.2 Energy measurement and reduction

2.2.1 Energy measurement for SNN

On event-driven neuromorphic hardware, one spiking layer's forward computation is triggered only when spikes are emitted, which is the source of energy consumption(Eshraghian et al., 2023; Modaresi et al., 2023). Thus, the energy consumption of SNN is highly related with the number of spikes.

Model Average Spike Firing Rate(MASFR) is the firing rate averaged over all neurons in the model:

$$\mathrm{MASFR}(T) = \frac{\sum_{neuron \in model} \sum_{t=1}^{T} \boldsymbol{spike}^{l}_{neuron}(t)}{T \cdot N_{model}} \tag{1}$$

where N_{model} is the number of neurons in the model.

Correspondingly, Layer Average Spike Firing Rate (LASFR)(Yao et al., 2023) measures the average spike frequency of all neurons in the layer during inference time T:

$$LASFR^{l}(T) = \frac{\sum_{neuron \in l} \sum_{t=1}^{T} \boldsymbol{spike}^{l}_{neuron}(t)}{T \cdot N_{l}} \tag{2}$$

where N_l is the number of neurons in layer l.

Because of the event-driven property, the required number of FLOPs to forward layer l's spikes in time T is(Yao et al., 2023) $FL^{l}_{SNN}(T) = T \cdot LASFR^{l}(T) \cdot FL^{l}_{ANN}$, where FL^{l}_{ANN} denotes the FLOPs of ANN layer l which is determined by the structure of this layer (e.g. linear or convolution).

Additionally, spikes in SNNs are binary. Thus the computation needed to forward spikes to next layer is only AC(Accumulation), which is much cheaper than MAC(Multiply-and-Accumulate) used in traditional ANNs(Yao et al., 2023). Note that, for the input layer we follow prior works and use Direct Input Encoding (DIE)(Rathi & Roy, 2021): the static image is fed to the SNN for T times, hence the first layer's operation still can be deemed as MAC.

So, the total energy consumption of the whole model is(Yao et al., 2023):

$$E(T) = E_{MAC} \cdot FL^{1}_{SNNConv}(T) + E_{AC} \cdot \left(\sum_{n=2}^{N} FL^{n}_{SNNConv}(T) + \sum_{m=1}^{M} FL^{m}_{SNNFC}(T) \right) \tag{3}$$

where N is number of total convolution layers, M is number of total fully-connected layer. FL^n is the required number of FLOPs for n^{th} layer. We follow the reference and set $E_{\mathrm{MAC}} = 4.6\,\mathrm{pJ}$, $E_{\mathrm{AC}} = 0.9\,\mathrm{pJ}$ for the energy consumption of single MAC and AC operation(Yao et al., 2023).

2.2.2 Energy saving from structural connection sparsity

In sparse layers, the number of links is reduced to $(1 - sparsity)$ compared with that of fully-connected. On hardware which supports sparse computation(Mishra et al., 2021; Dai et al., 2024), ideally the number of FLOPs needed for inference will be reduced to $(1 - sparsity)$ of that needed by dense layer.

$$FL_{ANN}(sparse) = (1 - sparsity) \cdot FL_{ANN}(dense) \tag{4}$$

For instance, a fully-connected linear layer requires $n_{\mathrm{in}} \cdot n_{\mathrm{out}}$ FLOPs to forward once where n_{in}, n_{out} is the input and output dimension of the linear layer. However, a sparse linear layer only requires $(1 - sparsity) \cdot n_{\mathrm{in}} \cdot n_{\mathrm{out}}$ FLOPs on hardware supporting sparse computation. For an MLP with sparsity level of 0.99, this corresponds to a large reduction in the number of FLOPs (99%).

2.3 Saturation time analysis

With inference time steps T increasing, SNN properties such as accuracy(Song et al., 2022; Han et al., 2020; Jiang et al., 2023; Bu et al., 2022; You et al., 2024) and MASFR tend to saturate. Saturation means when T is larger than a certain point, increasing T doesn't bring significant improvement but instead converges to a stable value with some noise.

In this study, we used following method, which has shared concept with early stop, to determine the saturation time: If the relative improvement between time steps is continuously no greater than 1% over 10 time steps, then the current time is determined as saturation point.

2.4 Experiment setup

In this experiment, we adopt MLP and VGG-16 (keep only one fully-connected layer to adapt for datasets not as challenging as ImageNet, see Yu et al. (2021)) network structures to train on popular image classification datasets CIFAR-10 and CIFAR-100. During sparse ANN training and

ANN2SNN conversion, the best hyper-parameters are determined by grid search. The grid search space is shown in Table1. Every pair of sparse/dense models shown in Results 3 use the same set of hyper-parameters.

Table 1: Hyper-parameters and grid search space of three methods.

(a) Method 1

lr	(0.05, 0.01, 0.005, 0.001, 0.0005)
bs	(32, 64, 128)
L	(2, 4, 8, 16, 32)

(b) Method 2

bs	(32, 64, 128)

(c) Method 3

lr	(0.001, 0.0001, 0.00001)
bs	(32, 64, 128)

3 RESULTS

3.1 SNN ACCURACY ANALYSIS

SNN accuracy is related to its inference time steps T where as T increases, more spikes are emitted which stabilize inference and make inference accuracy gradually converge . Figure 2 plots how SNN accuracy changes with T. Note that for method 1&2(Yang et al., 2025; Wang et al., 2022), spiking neurons operate integration and firing at every time step, therefore we have inference time steps in $ts = [1, 2, 3, ..., 64]$ for Method 1&2. However, for method 3(Chen et al., 2024) the integration and firing operation is done in two separate time windows instead of at every time step. So on method 3 we treat T as time window size which is a hyper-parameter. We display results with different $T \in ts = [2, 4, 8, 16, 32, 64]$.

For each experiment dense and sparse models share the same hyper-parameter configuration, which yields the best sparse models in grid search space.

The first and second row in Figure 2 present results of CIFAR10 and CIFAR100 trained on MLP. As can be seen here, on both datasets, sparse ANNs can achieve much higher accuracy than dense ANNs, showing the superiority of CHT training on ANNs. Also, comparing the accuracy between sparse and dense SNNs across 3 conversion methods, it could be observed that this accuracy advantage can be well preserved in sparse SNNs, which means on SNNs sparse networks can consistently achieve higher accuracy than the dense ones.

Third and Fourth row in Figure 2 present experiments using VGG-16 on CIFAR10 and CIFAR100. For VGG-16, on both datasets, sparse ANNs trained by CHT have similar accuracies with dense ANNs. After conversion, sparse SNNs can achieve close or even superior performance compared to its dense counterparts.

Moreover, it could be observed that in method 1&2, there is no clear difference between the saturation time of sparse and dense networks. This means although with less links in the network, sparse SNNs can have similar information-processing efficiency compared with dense SNNs.

Considering the impact of connectivity, the above results show that: sparse ANNs trained by CHT can match or even exceed dense ANNs. And after ANN2SNN conversion, sparse SNNs well preserve this advantage with similar inference latency compared with dense ANNs.

3.2 SNN ENERGY CONSUMPTION ANALYSIS

For method 1&2, we assign time T as the saturation time of SNN accuracy (see Section 2.3), while for method 3 T denotes the time steps where SNN reaches maximum accuracy. Reason (for method 3) is that its neurons' integration and firing operations are done in two separate time windows instead of every time step . Using T as reference time point, we calculate its energy according to Equation 3. The reason for assigning saturation time (for method 1&2) is, if we assign T smaller than saturation time, SNN's performance is not fully exploited. However, if we assign T larger than saturation time, it will cause waste of energy because further energy consumption does not bring significant improvement in accuracy.

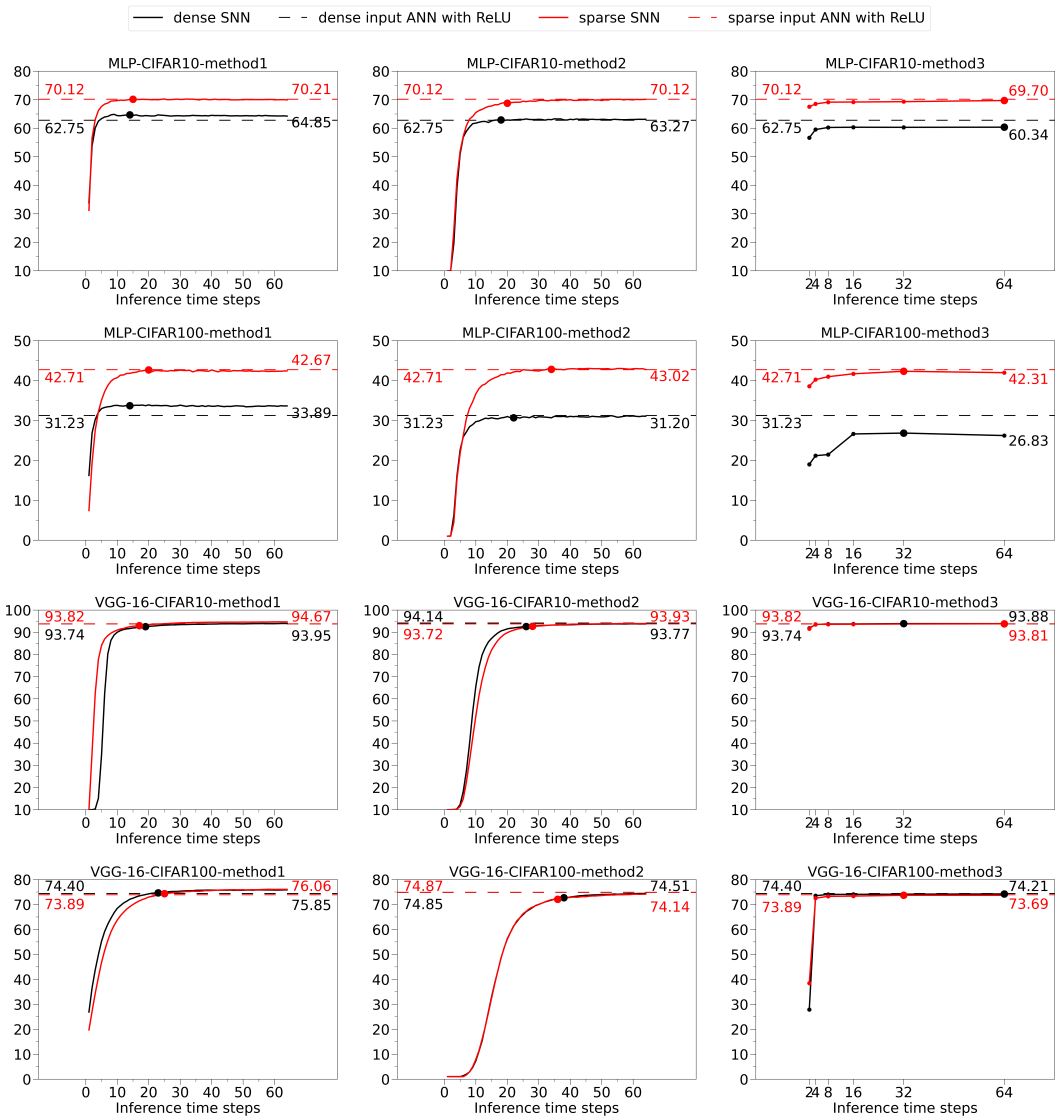


Figure 2: SNN accuracy(%) vs inference time steps. The accuracy curves in black are for dense models and red curves are for sparse models. While the solid lines represent how SNN model(after conversion) accuracy change with time, dashed horizontal lines represent ANN(before conversion) accuracy. The point marked on curve indicates the time: when SNN accuracy saturates(method 1&2) / when SNN accuracy reaches maximum(method 3). Each row correspond to a combination of network architectures (MLP and VGG-16) and datasets, while the columns correspond to different ANN2SNN conversion methods. The black/red numbers near the left y-axis report the dense/sparse ANN accuracy; the black/red numbers at the right report the maximum dense/sparse SNN accuracy shown in the figure.

Results are summarized in Table 2. For connectivity, 'fc' denotes fully-connected and represent dense model. 's(sparsity)' is sparse model with sparsity level in the bracket. Energy (E) refers to SNN theoretical energy consumption to classify a single image. The extent of energy reduction brought by sparsity is calculated with reduction $= \frac{E_{\text{dense}} - E_{\text{sparse}}}{E_{\text{sparse}}} \times 100\%$.

Results show regardless of architectures, datasets or conversion methods, sparse SNNs are always more energy-efficient than dense SNNs theoretically. That is because sparse SNNs benefit from structure connection sparsity that reduces active links compared with dense ANNs.

Table 2: Energy consumption comparison.

dataset	model	method	connectivity	T	energy(μJ)	reduction(%)
CIFAR10	MLP	QCFS	fc	14	622.13	98.94
			s(0.99)	15	6.59	
		SNM	fc	18	810.99	98.76
			s(0.99)	20	10.02	
		AEC	fc	64	2797.42	99.00
			s(0.99)	64	28.00	
	VGG-16	QCFS	fc	19	394.02	48.25
			s(0.5)	17	203.91	
		SNM	fc	26	912.14	42.65
			s(0.5)	28	523.11	
		AEC	fc	32	434.22	19.00
			s(0.5)	64	351.73	
CIFAR100	MLP	QCFS	fc	14	625.76	98.58
			s(0.99)	20	8.86	
		SNM	fc	22	986.16	98.04
			s(0.99)	34	19.34	
		AEC	fc	32	1409.70	99.00
			s(0.99)	32	14.09	
	VGG-16	QCFS	fc	23	629.51	41.80
			s(0.5)	25	366.37	
		SNM	fc	38	1104.36	49.16
			s(0.5)	36	561.45	
		AEC	fc	32	715.55	49.30
			s(0.5)	64	362.79	

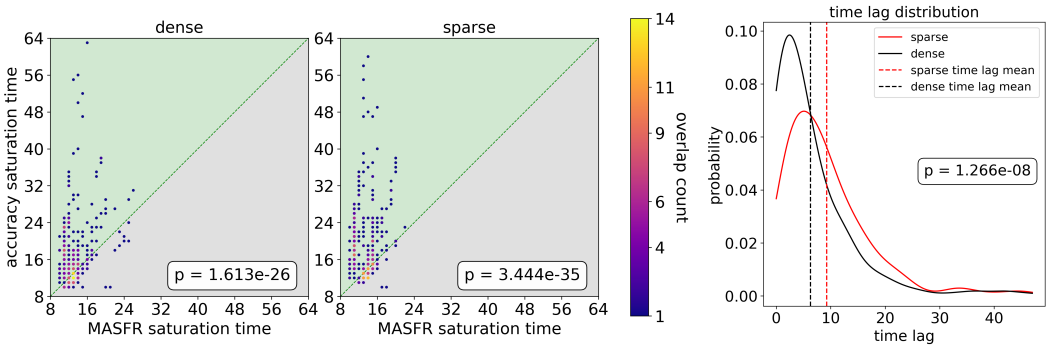
Therefore, regardless of conversion method, for MLP with sparsity level of 99%, sparse SNNs save up to 99% of energy; the smallest observed reduction 98.04% is still incredible. For CNN with sparsity level of 50%, sparse SNNs achieve energy reduction usually higher than 40%. One exception is method 3 on MLP for VGG-16, where reduction is 19.00% because the time with highest accuracy for sparse SNN is twice as that of dense SNN.

3.3 Time lag between saturation of Firing rate and Accuracy

For a deeper investigation of the underlying mechanism of how spikes firing rate affects model performance in SNNs, we here systematically studied the relationship between saturation time of accuracy and Model Average Spike Firing Rate (MASFR) in SNNs, where saturation time is calculated using the same algorithm described in Section 2.3. Our analysis utilizes data from all grid search experiments involving method 1&2 across four architecture-dataset combinations, as in method 1&2 the integration and firing operations and accuracy updates happen at every time step. By incorporating a wide range of methods, architectures, datasets, and hyper-parameter configurations, we aim to understand the general dynamics of SNNs obtained through ANN2SNN conversion.

Figure 3(a) presents a scatter plot of MASFR saturation time versus accuracy saturation time for dense (left panel) and sparse (right panel) SNNs. Time lag is defined as

$$\text{time lag} = \text{accuracy saturation time} - \text{MASFR saturation time} \tag{5}$$



(a) Saturation time comparison

(b)time lag distribution

Figure 3: Time lag analysis. (a)Points located in the green-shaded region above the y=x line indicate that firing rate saturate earlier than accuracy. The colors of the points corresponds to the number of overlapping data points. (b)The distribution of positive sparse time lag and positive dense time lag. The vertical lines represent the mean value. Sparse curves are in red and dense curves are in black.

To statistically evaluate the significance of this phenomenon, a one-sided Wilcoxon signed-rank test was conducted between the saturation time of accuracy and MASFR. The results are: for dense SNNs $p\_value = 1.613 \times 10^{-26}$, sparse SNNs $p\_value = 3.444 \times 10^{-35}$, and all SNNs $p\_value = 6.119 \times 10^{-62}$. These results provide strong statistical evidence that MASFR saturation precedes accuracy saturation. Given the diversity of the experimental settings, this conclusion suggests that the observed time lag is a general characteristic of SNNs.

This phenomenon can be qualitatively understood from the perspective of rate decoding(used in method 1&2). The sufficient condition for accuracy saturation is the stabilization of firing rates in the output layer neurons. Since MASFR is an average across all neurons in the network, it takes additional time for firing rate of neurons in the last layer to stabilize after MASFR's saturation. Thus there is a time lag between saturation of accuracy and MASFR.

After recognizing that positive time lag is significantly dominant, we further focused on studying the difference between positive time lag of sparse SNNs and positive time lag of dense SNNs. The distribution of time lag of two connectivity are plot using Kernel Density Estimation in Figure 3(b). A two-sided Mann-Whitny U test between time lag of sparse SNNs and time lag of dense SNNs shows $p\_value = 1.266 \times 10^{-8}$, which gives solid evidence to the difference in time lag between sparse SNNs and dense SNNs. Moreover, the mean value of time lag of sparse SNNs are higher than that of dense SNNs. This phenomenon is very interesting because it shows how structural connectivity impacts the SNN mechanism.

To summarize, this study uncovers a phenomenon of converted SNN using rate coding: the time lag between accuracy and MASFR. Moreover, the time lag of sparse SNNs are significantly different from the time lag of dense SNNs, while the average time lag of sparse SNNs are higher than dense SNNs. This may be a potential cause of the accuracy and energy advantage of sparse SNNs over dense SNNs.

4 Discussion

Here for the first time, we perform the investigation on the sparse SNNs converted with dynamically sparse trained ANNs. We show that sparse SNNs can achieve close-and in some cases superioraccuracy compared to their dense counterparts while being substantially more energy-efficient, making an energy reduction up to 99%. By transferring the sparse advantage of both structural connections and activations in brain networks into artificial neural networks, this work makes a step forward to

brain-like network architecture. Additionally, this study quantitatively shows the significant time lag in saturation between accuracy and firing rate, unveiling the underlying physical mechanism of how spiking neural networks process spikes and turn spiking information into model performance. Also, the results show that the positive(which is dominant) time lag is significantly different between sparse and dense networks, which might be a potential cause for the performance and energy gap between networks with different sparsity level.

Despite its significance, this work has certain limitations. Limited by available hardware, we analyze theoretical energy consumption rather than measuring real energy consumption. Our energy calculation is based on future hardware with the support of both sparse and event-driven computation.

Looking ahead, there are several promising directions that could extend the results of this work. The first direction is to study how to use CHT to directly train SNNs so as to further evaluate the effectiveness of DST methods(such as CHT) on SNNs. What's more, future study can extend time lag phenomenon study to various architectures, datasets and methods to fully understand the SNN model and time dynamics. Last but not least, we suggest the development of sparse neuromorphic hardware, as it is promising to implement energy-efficient architectures in reality.

REPRODUCIBILITY

Code of this research is sumbmitted as supplementary material. Please read README.md for more information.

REFERENCES

Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. What is the state of neural network pruning? *Proceedings of machine learning and systems*, 2:129–146, 2020.

Tong Bu, Jianhao Ding, Zhaofei Yu, and Tiejun Huang. Optimized potential initialization for low-latency spiking neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 11–20, 2022.

Tong Bu, Wei Fang, Jianhao Ding, PengLin Dai, Zhaofei Yu, and Tiejun Huang. Optimal ann-snn conversion for high-accuracy and ultra-low-latency spiking neural networks. *arXiv preprint arXiv:2303.04347*, 2023.

Yunhua Chen, Ren Feng, Zhimin Xiong, Jinsheng Xiao, and Jian K Liu. High-performance deep spiking neural networks via at-most-two-spike exponential coding. *Neural Networks*, 176: 106346, 2024.

Xilai Dai, Yuzong Chen, and Mohamed S Abdelfattah. Kratos: An fpga benchmark for unrolled dnns with fine-grained sparsity and mixed precision. In *2024 34th International Conference on Field-Programmable Logic and Applications (FPL)*, pp. 156–163. IEEE, 2024.

Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *Ieee Micro*, 38(1):82–99, 2018.

Michael V DeBole, Brian Taba, Arnon Amir, Filipp Akopyan, Alexander Andreopoulos, William P Risk, Jeff Kusnitz, Carlos Ortega Otero, Tapan K Nayak, Rathinakumar Appuswamy, et al. Truenorth: Accelerating from zero to 64 million neurons in 10 years. *Computer*, 52(5):20–29, 2019.

Jason K Eshraghian, Max Ward, Emre O Neftci, Xinxin Wang, Gregor Lenz, Girish Dwivedi, Mohammed Bennamoun, Doo Seok Jeong, and Wei D Lu. Training spiking neural networks using lessons from deep learning. *Proceedings of the IEEE*, 111(9):1016–1054, 2023.

Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the lottery: Making all tickets winners. In *International conference on machine learning*, pp. 2943–2952. PMLR, 2020.

Bing Han, Gopalakrishnan Srinivasan, and Kaushik Roy. Rmp-snn: Residual membrane potential neuron for enabling deeper high-accuracy and low-latency spiking neural network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13558–13567, 2020.

Li Hanming, Yusong Wang, Yingtao Zhang, and Carlo Vittorio Cannistraci. Cannistraci-hebb training of convolutional neural networks. *Preprints*, September 2025. doi: 10.20944/preprints202509.1904.v1. URL https://doi.org/10.20944/preprints202509.1904.v1.

Zihan Huang, Xinyu Shi, Zecheng Hao, Tong Bu, Jianhao Ding, Zhaofei Yu, and Tiejun Huang. Towards high-performance spiking transformers from ann to snn conversion. In *Proceedings of the 32nd ACM international conference on multimedia*, pp. 10688–10697, 2024.

Dexuan Huo, Jilin Zhang, Xinyu Dai, Jian Zhang, Chunqi Qian, Kea-Tiong Tang, and Hong Chen. Anp-g: A 28nm 1.04 pj/sop sub-mm2 spiking and back-propagation hybrid neural network asynchronous olfactory processor enabling few-shot class-incremental on-chip learning. In *2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, pp. 1–2. IEEE, 2023.

Siddhant Jayakumar, Razvan Pascanu, Jack Rae, Simon Osindero, and Erich Elsen. Top-kast: Top-k always sparse training. *Advances in Neural Information Processing Systems*, 33:20744–20754, 2020.

Haiyan Jiang, Srinivas Anumasa, Giulia De Masi, Huan Xiong, and Bin Gu. A unified optimization framework of ann-snn conversion: towards optimal mapping from activation values to firing rates. In *International Conference on Machine Learning*, pp. 14945–14974. PMLR, 2023.

Yuhang Li, Shikuang Deng, Xin Dong, Ruihao Gong, and Shi Gu. A free lunch from ann: Towards efficient, accurate spiking neural networks calibration. In *International conference on machine learning*, pp. 6316–6325. PMLR, 2021.

Yuhang Li, Shikuang Deng, Xin Dong, and Shi Gu. Error-aware conversion from ann to snn via post-training parameter calibration. *International Journal of Computer Vision*, 132(9):3586–3609, 2024.

Paul A Merolla, John V Arthur, Rodrigo Alvarez-Icaza, Andrew S Cassidy, Jun Sawada, Filipp Akopyan, Bryan L Jackson, Nabil Imam, Chen Guo, Yutaka Nakamura, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345 (6197):668–673, 2014.

Asit Mishra, Jorge Albericio Latorre, Jeff Pool, Darko Stosic, Dusan Stosic, Ganesh Venkatesh, Chong Yu, and Paulius Micikevicius. Accelerating sparse deep neural networks. *arXiv preprint arXiv:2104.08378*, 2021.

Decebal Constantin Mocanu, Elena Mocanu, Peter Stone, Phuong H Nguyen, Madeleine Gibescu, and Antonio Liotta. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature communications*, 9(1):2383, 2018.

Farhad Modaresi, Matthew Guthaus, and Jason K Eshraghian. Openspike: an openram snn accelerator. In *2023 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5. IEEE, 2023.

Nitin Rathi and Kaushik Roy. Diet-snn: A low-latency spiking neural network with direct input encoding and leakage and threshold optimization. *IEEE Transactions on Neural Networks and Learning Systems*, 34(6):3174–3182, 2021.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

Joonghyun Song, Jiwon Shirn, Hanseok Kim, and Woo-Seok Choi. Energy-efficient high-accuracy spiking neural network inference using time-domain neurons. In *2022 IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, pp. 5–8. IEEE, 2022.

Jihene Tmamna, Emna Ben Ayed, Rahma Fourati, Mandar Gogate, Tughrul Arslan, Amir Hussain, and Mounir Ben Ayed. Pruning deep neural networks for green energy-efficient models: A survey. *Cognitive Computation*, 16(6):2931–2952, 2024.

Tianqi Wang, Qianzi Shen, Xuhang Li, Yanting Zhang, Zijian Wang, and Cairong Yan. Precise spiking neurons for fitting any activation function in ann-to-snn conversion. *Applied Intelligence*, 55(6):463, 2025.

Yuchen Wang, Malu Zhang, Yi Chen, and Hong Qu. Signed neuron with memory: Towards simple, accurate and high-efficient ann-snn conversion. In *IJCAI*, pp. 2501–2508, 2022.

Hongchao Yang, Suorong Yang, Lingming Zhang, Hui Dou, Furao Shen, and Jian Zhao. Cs-qcfs: Bridging the performance gap in ultra-low latency spiking neural networks. *Neural Networks*, 184:107076, 2025.

Man Yao, Jiakui Hu, Guangshe Zhao, Yaoyuan Wang, Ziyang Zhang, Bo Xu, and Guoqi Li. Inherent redundancy in spiking neural networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 16924–16934, 2023.

Kang You, Zekai Xu, Chen Nie, Zhijie Deng, Qinghai Guo, Xiang Wang, and Zhezhi He. Spikezip-tf: Conversion is all you need for transformer-based snn. *arXiv preprint arXiv:2406.03470*, 2024.

Xiaohong Yu, Wei Long, Yanyan Li, Xiaoqiu Shi, and Lin Gao. Improving the performance of convolutional neural networks by fusing low-level features with different scales in the preceding stage. *IEEE Access*, 9:70273–70285, 2021.

Geng Yuan, Xiaolong Ma, Wei Niu, Zhengang Li, Zhenglun Kong, Ning Liu, Yifan Gong, Zheng Zhan, Chaoyang He, Qing Jin, et al. Mest: Accurate and fast memory-economic sparse training framework on the edge. *Advances in Neural Information Processing Systems*, 34:20838–20850, 2021.

Jilin Zhang, Dexuan Huo, Jian Zhang, Chunqi Qian, Qi Liu, Liyang Pan, Zhihua Wang, Ning Qiao, Kea-Tiong Tang, and Hong Chen. Anp-i: a 28-nm 1.5-pj/sop asynchronous spiking neural network processor enabling sub-0.1-μ j/sample on-chip learning for edge-ai applications. *IEEE Journal of Solid-State Circuits*, 59(8):2717–2729, 2024a.

Yingtao Zhang, Jialin Zhao, Wenjing Wu, and Alessandro Muscoloni. Epitopological learning and cannistraci-hebb network shape intelligence brain-inspired theory for ultra-sparse advantage in deep learning. In *The Twelfth International Conference on Learning Representations*, 2024b.

Yingtao Zhang, Diego Cerretti, Jialin Zhao, Wenjing Wu, Ziheng Liao, Umberto Michieli, and Carlo Vittorio Cannistraci. Brain network science modelling of sparse neural networks enables transformers and llms to perform as fully connected. *arXiv preprint arXiv:2501.19107*, 2025.

A APPENDIX: ANN2SNN CONVERSION METHODS

To thoroughly examine how sparsity affects SNN, we choose three conversion methods with distinct principles: CS-QCFS, SNM and AEC.

A.1 METHOD1: CS-QCFS

CS-QCFS converts ANNs to SNNs by first replacing ReLU with a Softplus Quantization Clip-Floor-Shift (S-QCFS) activation function on pre-trained ANNs for further training to learn learn layer-wise thresholds. Then, S-QCFS function is replaced with channel-wise S-QCFS(CS-QCFS) function to finetune channel-wise thresholds in convolution layers, which aims to capture heterogeneous activation distribution across channels. Finally, the ANN weights are directly transferred to SNN. Channel-wise thresholds of CS-QCFS are used as SNN spiking threshold, after applying a soft-plus function to ensure positivity of SNN's threshold.

A.2 METHOD2: SNM

Signed Neuron with Memory (SNM) allows both positive and negative spikes. If membrane potential exceeds positive threshold, neuron will emit a positive spike ($+1$). Despite this classical Integrate-Fire(IF) neuron behavior, for SNM neuron if membrane potential is lower than negative threshold, neuron will emit a spike (-1). SNM also includes a memory mechanism to make sure the number of positive spike is no less than number of negative spike, which ensures non-negative firing rate. Neuron-wise normalization is applied to determine the spiking threshold, in which for each neuron in a pre-trained ANN the maximum activation is set as the neuron's spiking threshold.

A.3 METHOD3: AEC

AEC (At-Most-Two-Spike Exponential Coding) splits each layer's inference window into a decoding phase and an encoding phase. During decoding phase neurons only accumulate inputs from previous layer and stay silent. During encoding phase each neuron can emit up to two spikes: primary and compensate; A primary spike occurs once membrane potential (accumulated in decoding phase) exceeds primary threshold. A reset-by-subtraction is done after primary spike. Then a compensate spike occurs if the membrane potential after reset exceeds compensate threshold. Both primary and compensate threshold decays exponentially with time. The output of the layer is the product of thresholds(at spike time) and spikes. After replacing ANN neurons by AEC neurons with fixed initialization, a further training on SNN is applied to only finetune the threshold parameters and BatchNorm parameters. Finally in inference BatchNorm layers are folded into preceding convolution layer to yield output SNN.

B APPENDIX: LLMS USAGE

We used LLMs to polish the language of the article while the principle, main logic, and the innovations of the article are by the authors. We also used LLMs to search literature, but they are all verified and read by authors.