2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Gaining the Sparse Rewards by Exploring Lottery Tickets in Spiking Neural Networks

Hao Cheng[1*], Jiahang Cao[1*], Erjia Xiao[1], Mengshu Sun[2], Renjing Xu[1†]

***Abstract*— Deploying energy-efficient deep learning algorithms on computational-limited devices, such as robots, is still a pressing issue for real-world applications. Spiking Neural Networks (SNNs), a novel brain-inspired algorithm, offer a promising solution due to their low-latency and low-energy properties over traditional Artificial Neural Networks (ANNs). Despite their advantages, the dense structure of deep SNNs can still result in extra energy consumption. The Lottery Ticket Hypothesis (LTH) posits that within dense neural networks, there exist winning Lottery Tickets (LTs), namely sub-networks, that can be obtained without compromising performance. Inspired by this, this paper delves into the spiking-based LTs (SLTs), examining their unique properties and potential for extreme efficiency. Then, two significant sparse *Rewards* are gained through comprehensive explorations and meticulous experiments on SLTs across various dense structures. Moreover, a sparse algorithm tailored for spiking transformer structure, which incorporates convolution operations into the Patch Embedding Projection (ConvPEP) module, has been proposed to achieve Multi-level Sparsity (MultiSp). MultiSp refers to (1) Patch number sparsity; (2) ConvPEP weights sparsity and binarization; and (3) ConvPEP activation layer binarization. Extensive experiments demonstrate that our method achieves extreme sparsity with only a slight performance decrease, paving the way for deploying energy-efficient neural networks in robotics and beyond.**

I. Introduction

The current research in Artificial Intelligence (AI) [1], [2], [3], [4] has entered a new stage, where the computational costs of algorithms are becoming more diverse. How to efficiently deploy corresponding algorithms in various resource-constrained scenarios, such as mobile edge devices and robotics, has become an urgent problem to solve. To tackle this issue, our approach analyses the problem from a dual-level perspective: (1) Redesigning existing models at the *neuron level* to obtain an efficient new network structure; (2) Developing numerous sparse algorithms to reduce the parameter size of original dense models at the *structure level*.

About redesigning new neurons, SNN is acclaimed as the third generation of neural networks and has increasingly gained great interest from researchers in recent years due to its distinctive properties: high biological plausibility, temporal information processing capability, and low power consumption. Distinct from ANNs that process data in a continuous manner, SNNs operate on binary time-series data, utilizing low-power Accumulation (AC) operations over the more energy-intensive Multiply-Accumulate (MAC) operations found in ANNs. This fundamental difference not only promises significant energy savings but also aligns closely with biological neural processing. Additionally, SNNs follow their biological counterparts and inherit complex temporal dynamics from them, endowing SNNs with powerful abilities to extract image features in a variety of tasks, including recognition [5], [6], tracking [7], and images generation [8]. Hence, this paper leverages SNNs to address the energy-efficiency problem fundamentally.

However, utilizing dense SNNs could still lead to extra energy consumption. For exploring more sparse models at the structure level, recent studies have focused on applying pruning techniques in SNNs, exploring methods such as model pruning [9], [10], [11], [12], model quantization [13], [14], [15], and knowledge distillation [16], [17]. However, these methods face their own set of limitations. Many are constrained to simpler models [9], [10], or they lead to significant performance degradation [17], [15], especially in more complex spiking model architectures. This gap underscores the need for more effective pruning strategies that can maintain or enhance performance while accommodating the intricate dynamics of advanced SNNs.

To further mitigate these issues, the Lottery Ticket Hypothesis (LTH) offers a promising solution. LTH suggests that within a randomly initialized dense neural network, there exist efficient sub-networks, which can achieve the comparable accuracy of the full network within the same or fewer iterations. Building upon this concept, the Multi-Prize Lottery Tickets (MPLTs [18]) hypothesis further refines this approach by focusing on efficient connection selection without the necessity of weight training, enhancing both weight sparseness and binarization for improved performance.

In this paper, we delve into the Lottery Tickets (LTs) in the SNN scenarios and provide comprehensive guidance for efficient sparse spiking-based methods. Although some existing work indicates the presence of LTs in SNNs [19], [20], they still need weight training which costs additional resources. More importantly, they have not conducted a detailed analysis of the outcomes from Spiking Lottery Tickets (SLTs) and have not further explored the properties of SLTs under Multi-Prize SLTs (MPSLTs) and Multi-level Sparse (MultiSp) conditions. Based on this, this paper proposes an SLTs exploring algorithm for both standard CNN-

*Equal contribution; †Corresponding author.
[1]Hao Cheng*, Jiahang Cao* and Erjia Xiao are with MICS Thrust, The Hong Kong University of Science and Technology (Guangzhou),
Email: hcheng046@connect.hkust-gz.edu.cn, jcao248@connect.hkust-gz.edu.cn, exiao469@connect.hkust-gz.edu.cn
[2] Mengshu Sun is with Beijing University of Technology,
Email: sunms@bjut.edu.cn
[1]Renjing Xu† is with MICS Thrust, the Hong Kong University of Science and Technology (Guangzhou),
Email: renjingxu@hkust-gz.edu.cn

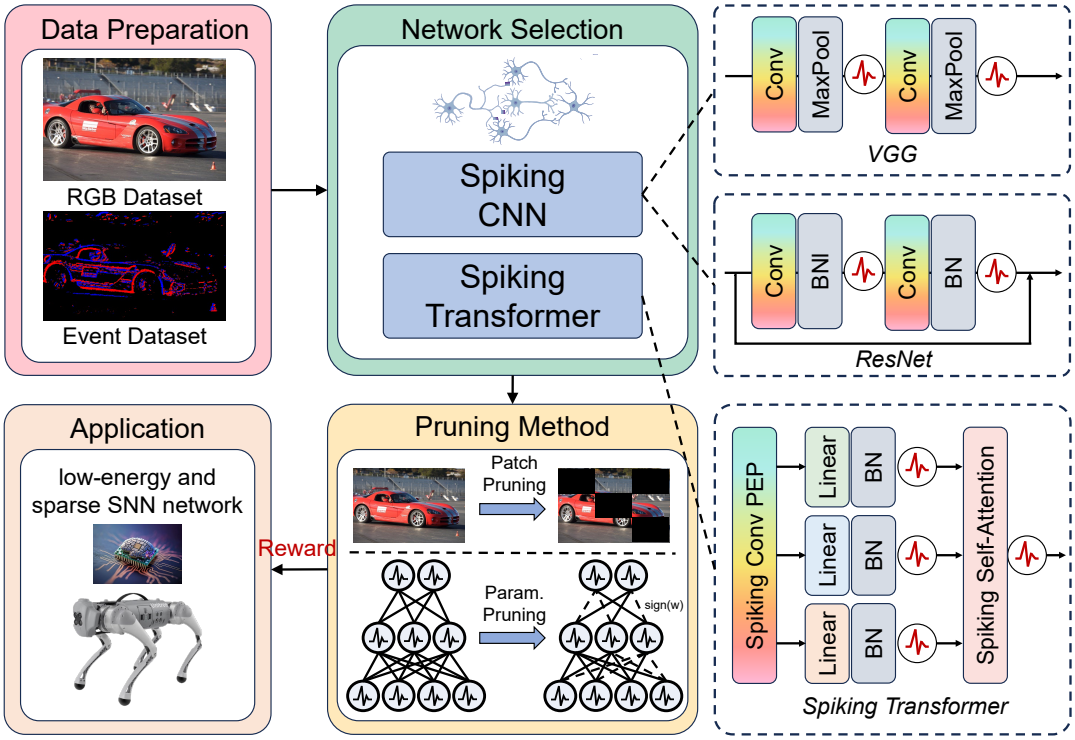

Fig. 1. The pipeline of our Spiking Lottery Tickets (SLT). The procedure begins with data preparation, utilizing either RGB or event datasets. It then progresses to the selection of network architectures, with options including CNN-based and transformer-based spiking models, where the rainbow-colored module can be sparsified by our SLT approach. The core of the process involves applying the patch pruning and parameter pruning methods, which yields rewards and returns a refined SNN network that is both energy-efficient and sparsely connected, making it ideal for implementation in resource-limited devices, e.g., robots.

based structures and transformer-based structures. The whole pipeline of our SLT method is illustrated in Figure 1. By comparing our SLTs with the original LTs, we gain the following two rewards:

> ***Reward 1:*** *Under the CNN-based models, the inner SLTs achieve higher sparsity and fewer performance losses compared to LTs.*
> ***Reward 2:*** *Under the transformer-based models, SLTs incur less accuracy loss compared to LTs counterparts at the same level of MultiSp.*

Our contributions can be summarized as follows:

- We obtain two ***Rewards*** that are applicable to SLTs for both CNN-based and transformer-based structures and provide a comprehensive analysis of the extreme sparsity outcomes.
- We propose a Multi-level Sparsity Exploring Algorithm for spiking-based transformers. This algorithm could effectively achieve multi-level sparsity results.
- We conduct extensive experiments on both RGB datasets and event datasets. Results demonstrate that our SLTs outperform the standard LTs by up to 4.58% while achieving extreme energy savings (>80.0%).

II. Related Works and Background

Spiking Neuron Networks. Spiking neural network is a bio-inspired algorithm that simulates the real process of signaling that occurs in brains. Compared to the artificial neural network (ANN), it transmits sparse spikes instead of continuous representations, which brings advantages such as low energy consumption and robustness. In this paper, we adopt the widely used Leaky Integrate-and-Fire (LIF) model [21], which is suitable to characterize the dynamic process of spike generation and can be defined as:

$$V[n] = \beta V[n-1] + \gamma I[n], \tag{1}$$
$$S[n] = \Theta(V[n] - \vartheta_{\text{th}}), \tag{2}$$

where n is the time step and β is the leaky factor that controls the information reserved from the previous time step; $V[n]$ is the membrane potential; $S[n]$ denotes the output spike which equals 1 when there is a spike and 0 otherwise; $\Theta(x)$ is the Heaviside function. When the membrane potential exceeds the threshold ϑ_{th}, the neuron will trigger a spike and resets its membrane potential to $V_{\text{reset}} < \vartheta_{\text{th}}$. The LIF neuron achieves a balance between computing cost and biological plausibility.

Pruning methods in Spiking Neural Networks. To further improve the energy efficiency of SNN, a number of works on SNN pruning have been proposed and well-validated on neuromorphic hardware. Shi [22] propose a pruning scheme that exploits the output spike firing of the SNN to reduce the number of weight updates during network training. Guo [23] dynamically removes non-critical weights in training by using the adaptive online pruning algorithm. Apart from seeking the help of pruning, Rathi [10] and Takuya [24] pursue sparse SNN by using Knowledge distillation and

quantization. Several works try to combine LTH with SNNs: Kim et al. [19] first investigate how to scale up pruning techniques towards deep SNNs and reveal that winning tickets consistently exist in deep SNNs across various datasets and architectures. They also propose a kind of Early-Time ticket that could alleviate the heavy search cost. Yao et al. [20] contribute a novel approach by introducing a probabilistic modeling method for SNNs. This method allows for the theoretical prediction of the probability of identical behavior between two SNNs, accounting for the complex spatio-temporal dynamics inherent to SNNs.

III. Method

Method Overview. The core focus of this paper lies in exploring the intrinsic properties of SLTs across different structures, including CNN-based models and transformer-based models, while comparing their performance with the original LTs across RGB and event-based datasets. For CNN-based models, we adopt the multi-prize lottery tickets hypothesis for obtaining optimal sub-networks and examine the binarized condition in detail. For transformer-based models, we investigate how extreme sparsity affects performance by integrating the parameter-level and patch-level sparsity methods and subsequently provide a thorough discussion about the balances between different sparsities. After obtaining the resulting sparse and energy-efficient networks, we could obtain two distinct rewards (mentioned in Section I) that benefit the implementation in resource-limited applications. The overview of our method is depicted in Figure 1.

Parameters Sparse for CNN-based structures. Our investigation begins with the conduction of standard Lottery Tickets (LTs [25]), which uncover efficient sub-networks within randomly initialized neural networks without the need for conventional weight training. Building upon this foundation, we extend our exploration to the domain of Multi-prize Lottery Tickets (MLTs [18]), which introduce the added benefit of binary weight and activation representations, further enhancing the network's energy efficiency and operational effectiveness. As the LT process unfolds, the network's connections become increasingly sparse, driven by an evolving threshold that refines weight selection. Sub-networks emerge by retaining only those weights that surpass this threshold. Employing the strategies of LTs and MLTs allows our spiking-based CNN models to progressively achieve the sparse rewards of energy conservation and minimal performance degradation through the strategic pruning and binarization of weights.

Multi-level Sparsity for transformer-based structures. Transformer-based models have more complex structures than CNN's, therefore, here we emphasize the sparsity of spiking transformer from two perspectives: *parameter-level* and *patch-level*. Regarding the sparsification of the parameters, we investigate the CNN-based patch embedding projection (ConvPEP) module where the existence of its redundancy has been proved [26]. MLTs methods are also utilized to achieve sparse and binarized PEP. Moreover, in the SNN-based ViT, patch-level redundancy exists, which motivates us to apply patch-pruning techniques for getting sparse rewards. We consider the multi-level sparsity in the spiking transformer as MultiSp, which includes 1) ConvPEP weights sparsity and binarization; 2) ConvPEP activation binarization; and 3) ConvPEP input patch number sparsity.

Algorithm 1 Multi-level Sparsity Exploration of Spiking Transformers

Input: Spiking transformer $F(\cdot)$; ConvPEP $F_c(\cdot)$; Weights of ConvPEP module W_c; Pruning scores of ConvPEP module S_c; Loss function $\mathcal{L}$; Training Dataset $(P, label)$; Parameter pruning rate and epoch $\{P_a, N_a\}$; Patch pruning rate and epoch $\{P_b, N_b\}$; Patch number n_p; Patch embedding and Position embedding $PatE, PosE$.
Output: Return optimal binarized parameter-sparsed and patch-sparsed subnetwork $G_s(\cdot)$.
Randomly Initialize the weights of ConvPEP module W_c and its pruning scores S_c.
Initialize the pruning masks of ConvPEP module by $\forall m_c \in M_c, m_c = 1$, its binary subnet weights $B_c \leftarrow sign(W_c)$ and gain term $\alpha \leftarrow ||M_c \odot W_c||_1 / ||M_c||_1$.
for $k = 1$ to $N_a + N_b$ **do**
  if $k \leq N_a$ **then** ▷ ***Stage 1: param. pruning***
    $S_c \leftarrow S_c - \eta \nabla_{S_c} \mathcal{L}(\alpha M_c \odot W_c)$
    Generating the sorting indices r_c and updating the pruning mask M_c where its score exceeds P_a.
    Update the gain term $\alpha \leftarrow ||M_c \odot W_c||_1 / ||M_c||_1$.
    Return binarized sparse ConvPEP module $F_s(\cdot)$.
  else if $k > N_a$ **then** ▷ ***Stage 2: patch pruning***
    Initialize Patch-level LTs index id_p.
    $PatE \leftarrow F_s(P)$
    $id_p \leftarrow$ Sorting $PatE$ based on $P_b n_p \odot PosE$
  end if
end for
$P_{plt} \leftarrow$ Pick the Patch-level LTs according to the $id_p \odot P$
Return $G_s(\cdot) := F(P_{plt}; M_c) \leftarrow F(F_s(P_{plt}; \alpha(m_c \odot w_c)))$

The Algorithm 1 presents further details of discovering spiking multi-level sparse lottery tickets. The algorithm consists of two stages, (1) *Stage 1: parameter-level pruning* and (2) *Stage 2: patch-level pruning*. In the first stage, we focus on refining the weight connections and binarizing both weights and activation output. The gain term α needs to be continuously updated until the correct winning tickets are found. W_c is the ConvPEP weights and M_c is the mask that is repeatedly updating until obtaining the final Mask M_c and its corresponding binarized sparse ConvPEP module $F_s(\cdot)$. After finding the parameter-level tickets, the updated mask M_c and weights W_c would be kept and start the second stage for exploring patch-level sparsity. The original patches P with different index id_p from Position Embedding $PosE$ are fed into the sparse ConvPEP $F_s(\cdot)$ and obtain corresponding Patch Embedding $PatE$. By comparing the magnitude of output $PatE$, we can identify the top-k patches P_{plt} that have the most impact on the overall performance. P_{plt} corresponds to the patch-level SLTs of our spiking-based transformer. Eventually. we could obtain the resulting binarized parameter-sparsed and patch-sparsed subnetwork

TABLE I

EVALUATING THE PERFORMANCE OF DENSE ANN/SNN, ORIGINAL LTS/SLTS AND MPLTS/MPSLTS FOR VGG-9/ RESNET-19 UNDER CIFAR10, CIFAR100, DVS128GESTURE AND CIFAR10-DVS.

Architecture	Method	CIFAR10 Acc(%)	CIFAR100 Acc(%)	DVS128Gesture Acc(%)	CIFAR10-DVS Acc(%)
VGG-9 / ResNet-19 (2.26M) / (12.63M)	ANN	88.10/92.05	68.64/76.74	92.92/94.83	78.65/80.14
	SNN	87.71/91.78	67.94/75.66	92.78/93.27	78.41/80.06
	LTs	87.73/91.87	67.56/75.62	92.43/93.83	77.92/80.02
	SLTs	87.67/91.66	65.81/74.76	91.73/91.77	77.82/79.41
	MPLTs	87.52/90.26	66.74/72.01	87.12/89.98	77.01/78.66
	MPSLTs	**87.76/91.82**	**67.22/72.43**	**91.70/93.22**	**77.94/79.71**

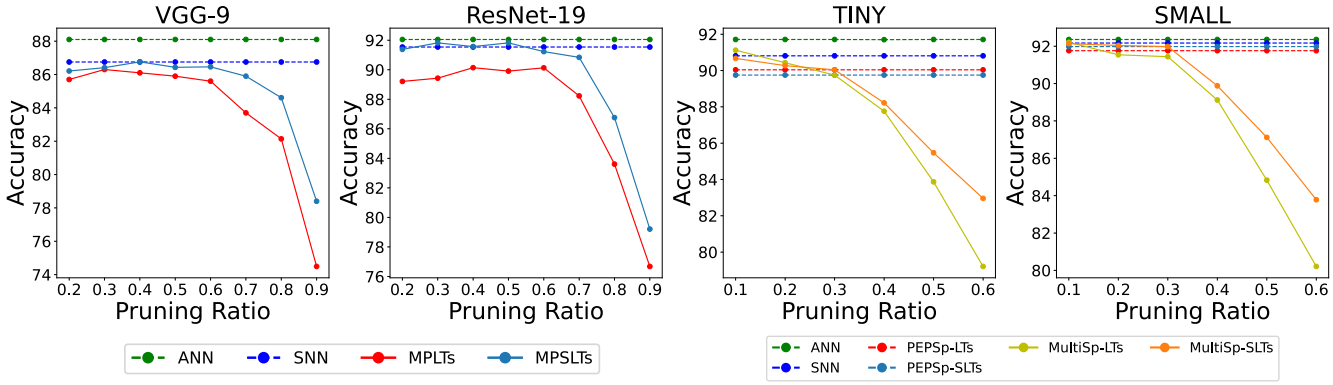

Fig. 2. The performance changes under different LTs and SLTs due to varying parameter-level and patch-level pruning ratios on CIFAR10.

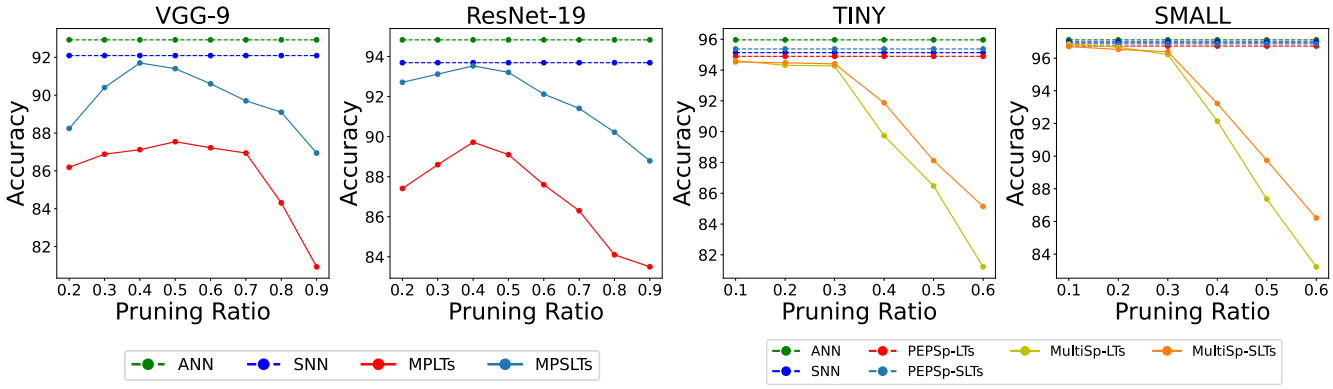

Fig. 3. The performance changes under different LTs and SLTs due to varying parameter-level and patch-level pruning ratios on DVS128Gesture.

$G_s(\cdot)$ with valuable multi-level winning tickets.

IV. EXPERIMENTS AND ANALYSIS

A. Experimental Setting

In this paper, to verify our finding rewards and proposed Algorithm 1, we use two CNN-based structures VGG-9 and ResNet-19, and two transformer-based structures TINY and SMALL referring to the model scale of DeiT-Tiny and Deit-Small [27]. The spiking version of the above structures are from [15], [5]. We adopt two RGB-based datasets CIFAR10 and CIFAR100, two event-based datasets DVS128Gesture [28] and CIFAR10-DVS [29]. When exploring parameter sparse LTs and SLTs in CNN modules, we use pruning ratio $P_a \in \{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8\}$, $P_b \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$ is the patch-level pruning ratio for the input patch number of ConvPEP. For the parameters of SNN, we adopt the most commonly used LIF neuron with timestep $T = 4$ and decay rate $\lambda = 0.99$ in our main experiments. Additionally, in order to explore the specific relationship between various SNN component parameters and spiking lottery tickets, we further use time step $T \in \{1, 2, 4, 6, 8\}$ and decay rate $\lambda \in \{0.99, 0.8, 0.6, 0.4, 0.2\}$. We use the Adam optimizer with a base learning rate of 0.1 to facilitate the learning process. Surrogate gradient training methods [30] for SNN are adopted. The models for conducting experiments are implemented based on Pytorch and SpikingJelly [31]. Since our main focus is finding the best spiking lottery tickets while maintaining accuracy, we only use the most primitive direct training without any training tricks in all our experiments. Furthermore, the Appendix VI includes more detailed interpretations.

TABLE II

THE PERFORMANCE OF VIT/SPIKING-VIT, PEPSP-LTS/SLTS, MULTISP-LTS/SLTS WITH DIFFERENT PATCH AND PARAM. PRUNING RATIO FOR TINY/SMALL UNDER CIFAR10, CIFAR100, DVS128GESTURE AND CIFAR10-DVS.

Architecture	Datasets	ViT	Spiking-ViT	PEPSp-LTs	PEPSp-SLTs	MultiSp-LTs	MultiSp-SLTs
TINY / SMALL (5M)/(22M)	*Patch Pruning Ratio*	0/0	0/0	0/0	0/0	0.3/0.3	**0.3/0.3**
	Param Pruning Ratio	0/0	0/0	0.4/0.4	0.4/0.4	0.4/0.4	**0.4/0.4**
	CIFAR10 (%)	91.71/92.36	90.81/92.17	90.04/91.76	89.75/91.98	89.87/91.44	**90.21/91.98**
	CIFAR100 (%)	74.37/75.12	73.84/75.43	74.31/75.22	73.68/74.86	73.21/74.09	**73.82/74.89**
	DVS128Gestrure (%)	95.96/97.12	95.13/96.99	94.89/96.74	95.37/96.88	94.27/96.22	**94.40/96.39**
	CIFAR10-DVS (%)	78.43/80.01	77.86/79.77	76.61/78.42	77.14/78.91	76.69/78.02	**77.03/78.43**

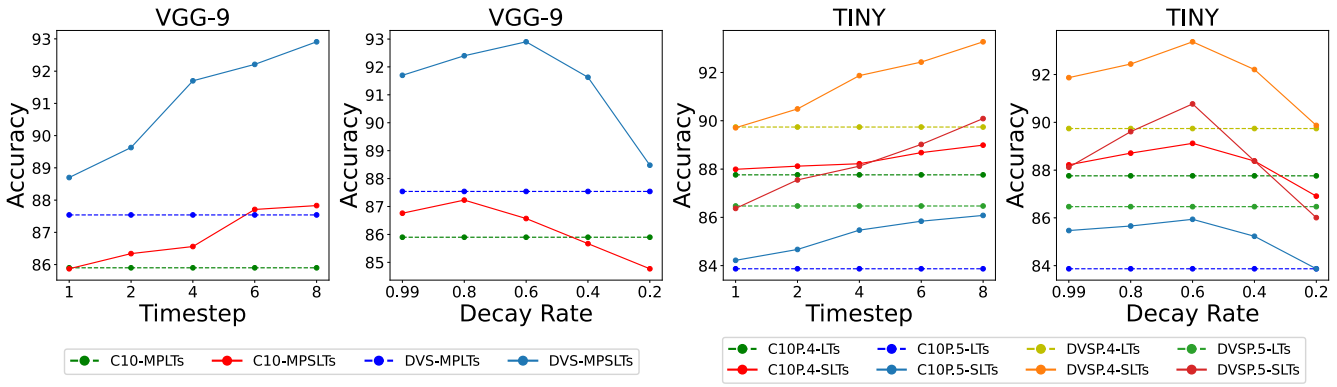

Fig. 4. The performance effect under different Timestep and Decay Rate to VGG-9 and TINY in CIFAR10 (C10) and DVS128Gesture (DVS). The P.4 and P.5 indicate the parameter Pruning Ration is 0.4 and 0.5.

B. General Performance Analysis

To comprehensively explore the inherent characteristics of spiking LTs and compare them with ANN LTs, Figure 2, 3, 7, and 8 respectively evaluate our used four structures on four different datasets under different pruning ratios P_a and P_b.

CNN-based Sparsity. In the study focusing on structures with convolution operations, based on the left two sub-figures of the aforementioned four figures, we first investigate the performance of Multi-prize LTs (MPLTs) and SLTs (MPSLTs) under CNN-based structures across different values of P_a. Additionally, we compare their performance with the corresponding original dense ANNs and SNNs. Through the observation, we can conclude that MPSLTs outperform MPLTs across different structures and types of datasets. This observation supports the ***Reward 1*** mentioned in the Section I. In addition, the optimal winning tickets in both MPLTs and MPLTs occur at the pruning ratio of $P_a = 0.4$. Therefore, in the subsequent exploration regarding patch-level sparsity, we will fix the $P_a = 0.4$ for the ConvPEP.

Transformer-based Sparsity. We attempt to explore the performance of Multi-level LTs (MultiSp-LTs) and SLTs (MultiSp-SLTs) in spiking transformers under different P_b. We also compare the ConvPEP sparse LTs (PEPSp-LTs) and SLTs (PEPSp-SLTs) which only perform the parameter-level sparsity in the ConvPEP modules serving as baselines. As illustrated in the right two sub-figures, both MultiSp-LTs and MultiSp-SLTs can maintain similar performance as PEPSp-LTs and PEPSp-SLTs at $P_b = 0.3$. However, during the descent process, MultiSp-SLTs exhibit smaller performance degradation compared to MultiSp-SLTs, indicating that MultiSp-SLTs possess better sparse robustness. This aligns with the ***Reward 2*** proposed above.

Overall Results. To summarize the best-performing MPLTs and MPSLTs under different CNN-based structures, we present Table I. When $P_a = 0.4$, the MPSLTs with the best performance under VGG-9/ResNet-19 on CIFAR10, CIFAR100, DVS128Gesture and CIFAR10-DVS yield accuracy improvement with $0.24/1.56$, $0.48/0.42$, $4.58/3.24$ and $0.93/1.05$ (%) compared to MPLTs. It is important to note that MPSLTs gain more performance increases under the event-based dataset compared to the RGB dataset. Finally, regarding the results of standard SNN/ANN transformers, PEPSp-LTs/PEPSp-SLTs and MultiSp-LTs/MultiSp-SLTs, we select the best-performing results with the most suitable sparse parameters, namely $\{P_a; P_b\} = \{0.4, 0.3\}$, and summarize them in Table II. Our Multip-SLTs also achieve better outcomes than the ANN's version, demonstrating the effectiveness of our spiking-based sparsity method.

C. Analysis on SNN-related Parameters

In this section, we clarify the impact of the unique timestep T and decay rate λ in SNNs on the performance under different types of SLTs In Figure 4 and Figure 9, we select $T \in \{1, 2, 4, 6, 8\}$ and $\lambda \in \{0.99, 0.8, 0.6, 0.4, 0.2\}$ as our parameter variables. We conduct experiments with VGG-9 and TINY on four datasets. Specifically for VGG-9, we demonstrate the performance variation of MPSLTs under the modification of T and λ. Under the TINY, we select patch-level pruning ratio $P_b = 0.4/0.5$ (P.4/P.5), which perfor-

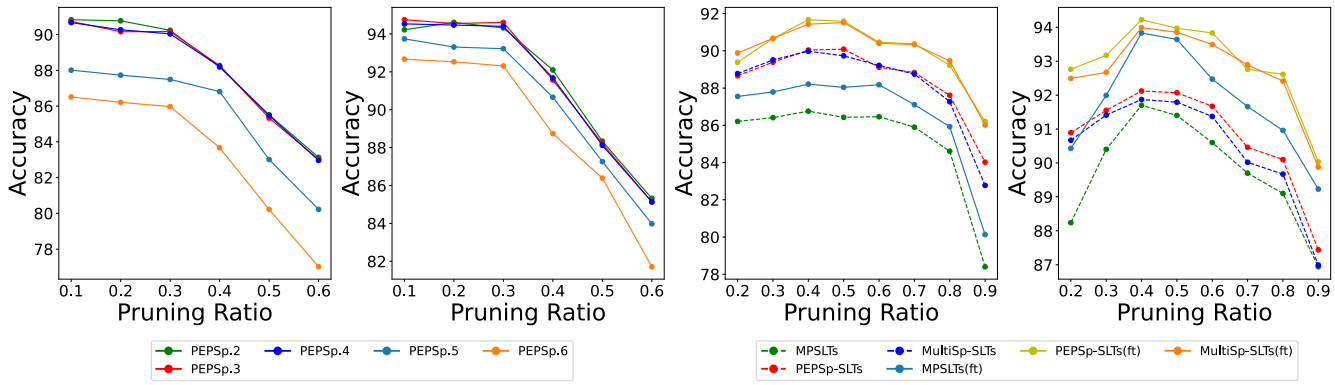

Fig. 5. The left two figures: The impact of different param. pruning ratios $(0.2, 0.3, 0.4, 0.5, 0.6)$ on overall performance of patch-lever sparsity. The right two figures: The Fing-Tuning (FT) effect to MPTSLTs, and PEPSp-SLTs and MultiSp-SLTs.

mance just begins to decline in terms of patch-level sparsity. We then illustrate the performance variation of MultiSp-SLTs under different T and λ compared with MultiSp-LTs (abbreviated as SLTs/LTs in Figure 4 and Figure 9).

Time Step. The MPSLTs of VGG-9 and the MultSp-SLTs on transformer show performance improvement with the increase of timestep T, also maintaining the properties of the two rewards discovered above.

Decay Rate. The performances of MPSLTs of VGG-9 and the MultSp-SLTs on transformer do not simply exhibit a monotonic increase or decrease with the value change of decay rate. Instead, it shows an increase followed by a decrease as λ decreases. This implies that in our pursuit of spiking-based MultiSp-SLTs in the future, we need to first make a more detailed selection of decay rate λ values.

D. Comparisons of Power Consumption

To better understand the effects of our spiking LTs on energy using, we estimate the theoretical power consumption on neuromorphic chips (detailed in Appendix VI-A). We compute the energy results of VGG-9 in CIFAR10, where the MPSLTs only cost 0.079 mJ per image, exhibiting merely 11.2% (0.079/0.708) energy consumption compared to its ANN counterpart. In addition, our MPSLT further reduces energy consumption by 15.1% on top of the dense SNN, i.e., 0.079 mJ *vs.* 0.093 mJ. This **extreme energy-savings** further validates the possibility of deploying our algorithms on resource-constrained machines in the future.

E. Ablation Study

Evaluation of the Balance Between P_a and P_b. Since the MultiSp-SLTs in transformer-based models represent a multi-level combination of ConvPEP parameter-level sparsity (PEPSp) and patch-level sparsity, the modification in different pruning ratios P_a and P_b are bound to influence each other. In this section, we further validate the performance variation of PEPSp with additional $P_a \in \{0.2, 0.3, 0.5, 0.6\}$ (abbreviated as Sp.x) under different values of P_b. The two left sub-figures of Figure 5 illustrate that the performance trends of overall patch-level sparsity are similar when executing P_a modification. However, for overall performance under different P_b, there exists a specific threshold between

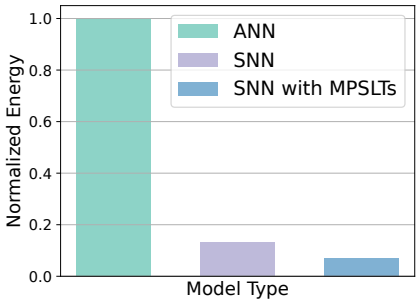

Fig. 6. Comparison of energy consumption using (a) ANN, (b) SNN without any pruning, and (c) SNN with MPSLTs. Our method achieves extreme energy-saving with only 0.079 mJ/image.

$P_a = 0.4$ and $P_a = 0.5$. A noticeable decline in overall performance would occur once P_a exceeds this threshold.

Effect of Fine-tuning Strategy. Since the LTs pruning methods do not train the weights directly, therefore, we hope to explore how fine-tuning can further affect the final performances. We conduct fine-tuning (FT) separately for MPSLTs, PEPSp-SLTs, and MultiSp-SLTs. As shown in the right sub-figures in Figure 5, the results demonstrate that SLTs would generate performance improvements through additional fine-tuning strategy.

V. Conclusion

In summary, this paper tackles the pressing need for energy-efficient algorithms by exploring the extreme sparsity in low-energy spiking neural networks. In detail, we focus on considering the existence of lottery tickets in SNNs and their corresponding unique properties compared with ANN lottery tickets. Two valuable sparse rewards are investigated under the spiking CNN-based and transformer-based models. In addition, we propose a sparse algorithm tailored to the spiking transformer, which incorporates convolution operations into the Patch Embedding Projection (PEP) module, achieving multi-level sparsity. Comprehensive experiments show that our approach results in significant sparsity while minimally impacting performance, thereby facilitating the implementation of energy-efficient neural networks across robotics and additional fields.

REFERENCES

[1] L. Yang, Y. Han, X. Chen, S. Song, J. Dai, and G. Huang, "Resolution adaptive networks for efficient inference," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2369–2378.

[2] L. Yang, H. Jiang, R. Cai, Y. Wang, S. Song, G. Huang, and Q. Tian, "Condensenet v2: Sparse feature reactivation for deep networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3569–3578.

[3] Z. Zheng, L. Yang, Y. Wang, M. Zhang, L. He, G. Huang, and F. Li, "Dynamic spatial focus for efficient compressed video action recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.

[4] L. Yang, Z. Zheng, J. Wang, S. Song, G. Huang, and F. Li, ": An adaptive object detection system based on early-exit neural networks," *IEEE Transactions on Cognitive and Developmental Systems*, 2023.

[5] Z. Zhou, Y. Zhu, C. He, Y. Wang, S. Yan, Y. Tian, and L. Yuan, "Spikformer: When spiking neural network meets transformer," *International Conference on Learning Representations (ICLR)*, 2023.

[6] S. Deng, Y. Li, S. Zhang, and S. Gu, "Temporal efficient training of spiking neural network via gradient re-weighting," *arXiv preprint arXiv:2202.11946*, 2022.

[7] J. Zhang, B. Dong, H. Zhang, J. Ding, F. Heide, B. Yin, and X. Yang, "Spiking transformers for event-based single object tracking," in *CVPR*, 2022, pp. 8801–8810.

[8] J. Cao, Z. Wang, H. Guo, H. Cheng, Q. Zhang, and R. Xu, "Spiking denoising diffusion probabilistic models," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 4912–4921.

[9] E. O. Neftci, B. U. Pedroni, S. Joshi, M. Al-Shedivat, and G. Cauwenberghs, "Stochastic synapses enable efficient brain-inspired learning machines," *Frontiers in neuroscience*, vol. 10, p. 241, 2016.

[10] N. Rathi, P. Panda, and K. Roy, "Stdp-based pruning of connections and weight quantization in spiking neural networks for energy-efficient recognition," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 4, pp. 668–677, 2018.

[11] Y. Chen, Z. Yu, W. Fang, T. Huang, and Y. Tian, "Pruning of deep spiking neural networks through gradient rewiring," *arXiv preprint arXiv:2105.04916*, 2021.

[12] Y. Liu, S. Xiao, B. Li, and Z. Yu, "Sparsespikformer: A co-design framework for token and weight pruning in spiking transformer," *arXiv preprint arXiv:2311.08806*, 2023.

[13] S. Xu, H. Li, B. Zhuang, J. Liu, J. Cao, C. Liang, and M. Tan, "Generative low-bitwidth data free quantization," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16*. Springer, 2020, pp. 1–17.

[14] Y. Zhou, S.-M. Moosavi-Dezfooli, N.-M. Cheung, and P. Frossard, "Adaptive quantization for deep neural network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[15] W. Fang, Z. Yu, Y. Chen, T. Huang, T. Masquelier, and Y. Tian, "Deep residual learning in spiking neural networks," *Advances in Neural Information Processing Systems*, vol. 34, pp. 21 056–21 069, 2021.

[16] A. Polino, R. Pascanu, and D. Alistarh, "Model compression via distillation and quantization," *arXiv preprint arXiv:1802.05668*, 2018.

[17] Y. Chen, Z. Yu, W. Fang, Z. Ma, T. Huang, and Y. Tian, "State transition of dendritic spines improves learning of sparse spiking neural networks," in *ICML*. PMLR, 2022, pp. 3701–3715.

[18] J. Diffenderfer and B. Kailkhura, "Multi-prize lottery ticket hypothesis: Finding accurate binary neural networks by pruning a randomly weighted network," in *ICLR*, 2021. [Online]. Available: https://openreview.net/forum?id=U_mat0b9iv

[19] Y. Kim, Y. Li, H. Park, Y. Venkatesha, R. Yin, and P. Panda, "Exploring lottery ticket hypothesis in spiking neural networks," in *European Conference on Computer Vision*. Springer, 2022, pp. 102–120.

[20] M. Yao, Y. Chou, G. Zhao, X. Zheng, Y. Tian, B. Xu, and G. Li, "Probabilistic modeling: Proving the lottery ticket hypothesis in spiking neural network," *arXiv preprint arXiv:2305.12148*, 2023.

[21] K. Roy, A. Jaiswal, and P. Panda, "Towards spike-based machine intelligence with neuromorphic computing," *Nature*, vol. 575, no. 7784, pp. 607–617, 2019.

[22] 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," *Frontiers in neuroscience*, vol. 13, p. 405, 2019.

[23] W. Guo, M. E. Fouda, H. E. Yantir, A. M. Eltawil, and K. N. Salama, "Unsupervised adaptive weight pruning for energy-efficient neuromorphic systems," *Frontiers in Neuroscience*, vol. 14, p. 598876, 2020.

[24] S. Takuya, R. Zhang, and Y. Nakashima, "Training low-latency spiking neural network through knowledge distillation," in *COOL CHIPS*. IEEE, 2021, pp. 1–3.

[25] V. Ramanujan, M. Wortsman, A. Kembhavi, A. Farhadi, and M. Rastegari, "What's hidden in a randomly weighted neural network?" in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 893–11 902.

[26] X. Shen, Z. Kong, M. Qin, P. Dong, G. Yuan, X. Meng, H. Tang, X. Ma, and Y. Wang, "Data level lottery ticket hypothesis for vision transformers," *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19)*, 2023.

[27] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *International conference on machine learning*. PMLR, 2021, pp. 10 347–10 357.

[28] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza *et al.*, "A low power, fully event-based gesture recognition system," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7243–7252.

[29] W. Cheng, H. Luo, W. Yang, L. Yu, and W. Li, "Structure-aware network for lane marker extraction with dynamic vision sensor," *arXiv preprint arXiv:2008.06204*, 2020.

[30] 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 Processing Magazine*, vol. 36, no. 6, pp. 51–63, 2019.

[31] W. Fang, Y. Chen, J. Ding, D. Chen, Z. Yu, H. Zhou, T. Masquelier, Y. Tian, and other contributors, "Spikingjelly," https://github.com/fangwei123456/spikingjelly, 2020.

[32] M. Yao, G. Zhao, H. Zhang, Y. Hu, L. Deng, Y. Tian, B. Xu, and G. Li, "Attention spiking neural networks," *IEEE transactions on pattern analysis and machine intelligence*, 2023.

VI. APPENDIX

A. Theoretical Power Consumption

To calculate the theoretical energy consumption, we begin by determining the synaptic operations (SOPs). The SOPs for each block can be calculated using [5]:

$$\mathrm{SOPs}(l) = fr \times T \times \mathrm{FLOPs}(l) \tag{3}$$

where l denotes the block number in the model, fr is the firing rate of the input spike train of the block and T is the time step of the spike neuron. $\mathrm{FLOPs}(l)$ refers to floating point operations of l block.

To estimate the theoretical energy consumption of the spiking model, we assume that the operations are implemented on a $45nm$ hardware, with energy costs of $E_{MAC} = 4.6pJ$ and $E_{AC} = 0.9pJ$, respectively. According to [32], the calculation for the theoretical energy consumption of SNN is given by:

$$\begin{aligned} E_{SNN} = {} & E_{flop} \times \mathrm{FLOP}^{1}_{\mathrm{Conv}} \\ & + E_{sop} \times \left(\sum_{n=2}^{N} \mathrm{SOP}^{n}_{\mathrm{Conv}} + \sum_{m=1}^{M} \mathrm{SOP}^{m}_{\mathrm{FC}} \right) \end{aligned} \tag{4}$$

where N and M denote the number of spiking convolutional layers and spiking linear layers. We first sum up the SOPs of all $Conv$ layers (except the first layer), and FC layers and multiply by E_{flop}. For the first convolutional layer of SNN, we calculate the energy consumption utilizing FLOPs due to the spike encoding operation performed here.

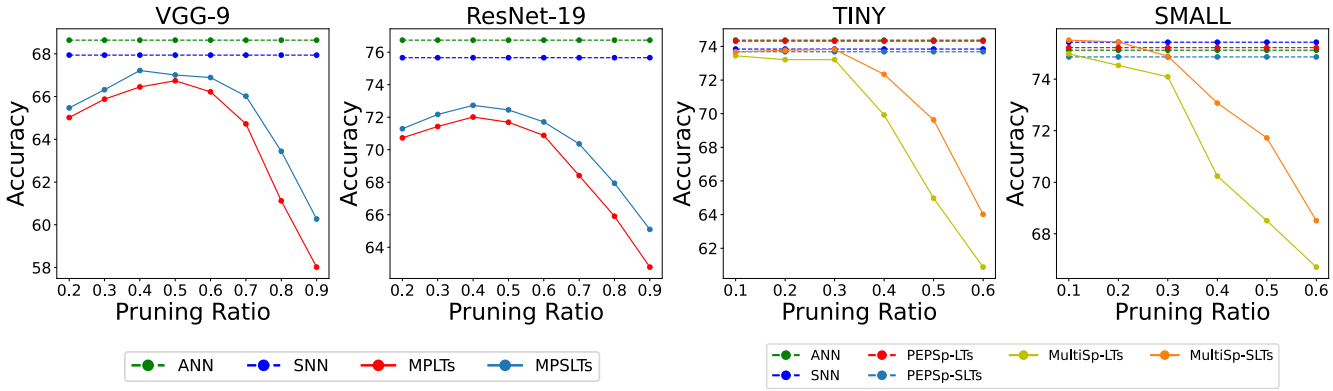

Fig. 7. The performance changes under different LTs and SLTs due to varying parameter-level and patch-level pruning ratios on CIFAR100.

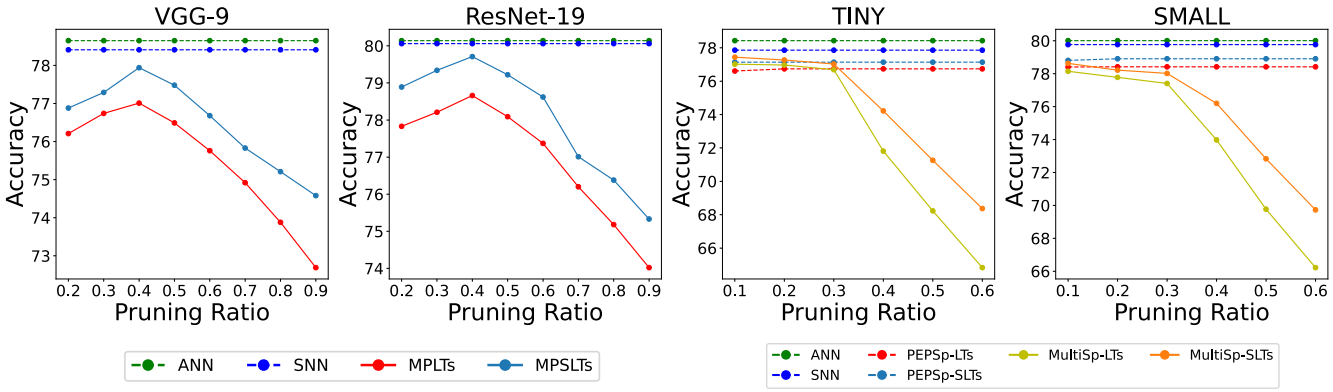

Fig. 8. The performance changes under different LTs and SLTs due to varying parameter-level and patch-level pruning ratios on CIFAR10DVS.

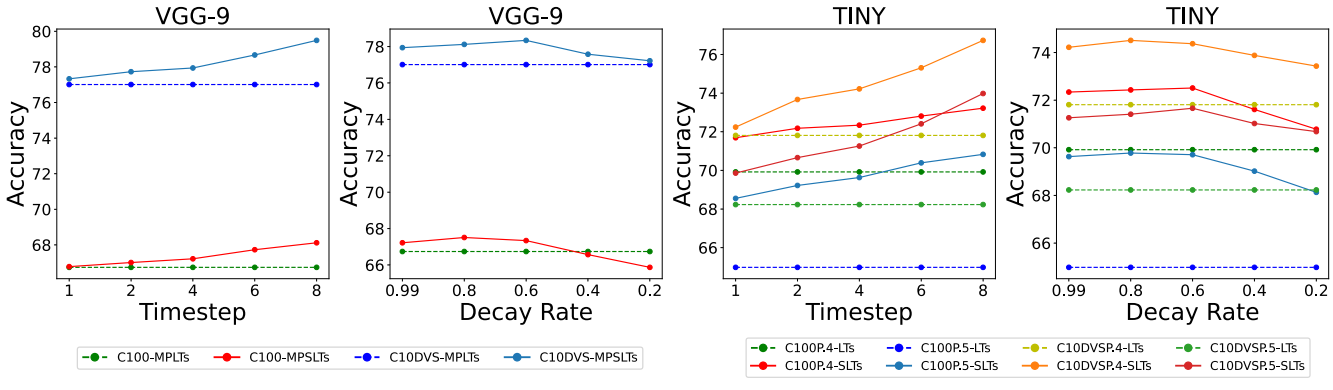

Fig. 9. The performance effect under different Timestep and Decay Rate to VGG-9 and TINY in CIFAR100 (C100) and CFIAR10-DVS (C10DVS). The P.4 and P.5 indicate the parameter Pruning Ration is 0.4 and 0.5.