SNN Weight Matrix Blocking Algorithm Based on Reinforcement Learning

Chenyu Zhao
Xi'an Jiaotong - Liverpool University
Chenyu.Zhao23@student.xjtlu.edu.cn
Xiang He
Xi'an Jiaotong - Liverpool University
he-xiang0415@qq.com
Bingqi Li
Xi'an Jiaotong - Liverpool University
bingqi.Li23@xjtlu.edu.cn

ABSTRACT

To tackle the high energy consumption of spiking neural networks (SNNs), this work presents a weight matrix block algorithm utilizing reinforcement learning. The method balances energy consumption and compute performance by coupling block, weighting rearrangement and topology rebuilding. The experimental results indicate energy cost drops by approximately 54%, compute efficacy is enhanced and model accuracy maintained. The algorithm takes advantage of a dynamic reinforcement learning system to adaptively adjust the blocking strategy and presents a potential solution for the SNNs' implementation in resource-limited scenarios. The experimental results validate the effectiveness of this approach in balancing energy saving and performance and lay a foundation for low-power neural network applications.

Keywords: Spiking Neural Network, Energy Efficiency Optimization, Reinforcement Learning, Weight Block, Topology Optimization

1. INTRODUCTION

1.1 Research background and significance

Spiking neural networks (SNNs) have been a hot topic of study with the extensive use of artificial intelligence technology in edge computing and mobile devices owing to their bio-inspired nature and potential for low power consumption. Nevertheless, in real-world deployments, SNNs suffer from a high energy consumption level principaliy because of the high computational complexity and high connectivity density of their weight matrix. The high energy consumption phenomenon restricts the use of SNNs in resource-limited domains, particularly in applications like IoT devices and embedded systems, where energy saving demand is pressing. Thus, optimizing SNN's weight matrix structure in terms of energy saving has become a major direction of study [1]. This study proposes a weight matrix block algorithm based on reinforcement learning. Through block partitioning, weight rearrangement and topology reconstruction, it aims to achieve a balance between energy efficiency and performance, and provide theoretical support and practical guidance for the promotion of SNNs in low-power applications.

This study has important academic value and application prospects. From an academic perspective, weight matrix optimization provides a new research perspective for the neural computing model of SNNs, which helps to deepen the understanding of the energy consumption mechanism of neural networks. From an application perspective, with the popularization of edge smart devices, low-power neural network technology has made an increasingly significant contribution to promoting green computing and sustainable development. The results of this study can provide efficient solutions for energy-sensitive scenarios, such as smart sensors and wearable devices, thereby promoting the implementation of artificial intelligence technology in actual industries [2]. By integrating the dynamic decision-making ability of reinforcement learning, this study aims to break through the limitations of traditional static block methods and open up new paths for energy efficiency optimization of SNNs.

1.2 Current status of research at home and abroad

Scholars at home and abroad have conducted extensive research in the field of SNN energy efficiency optimization. Foreign research mainly focuses on the sparsification and model compression technology of neural networks, such as reducing the computational load through pruning and quantization methods, which significantly reduces the energy consumption requirements of SNNs [3]. Recently, Reinforcement Learning for Neuromorphic Computing (RLNC) has emerged as a new direction. Reinforcement learning was used to optimize the network structure and achieved some early results in particular in adaptive adjustment of the connection weights [4]. These works lay a significant foundation and provide technology support for the topic but current methods primarily concentrate on static optimization and cannot adapt to an evolving environment.

Domestic studies have achieved some results in hardware implementation and application of SNN, like the design of low energy SNN hardware accelerators with FPGA or ASIC [5]. But domestic studies on the optimization of the weight matrix block and topology remain relatively weak, particularly in the area of combining dynamic optimization with reinforcement learning, whose development remains in early days. The current studies did not comprehensively investigate the impact on energy consumption and performance of the block strategy, leaving a space for innovation in this work. Briefly speaking, domestic and international studies have built a foundation for optimizing the energy consumption of SNN, but combination of dynamic block division and topology rebuilding must be improved in order to meet complex application situations [6].

2. THEORETICAL BASIS AND RELATED TECHNOLOGIES

2.1 Basic principles of spiking neural networks

Being a computer model inspired by biological neural systems, spiking neural networks (SNNs) have demonstrated exclusive strengths in low-power neural computing with event-driven properties. SNNs replicate the pulse emission process of neurons and adopt time encoding and sparse firing in terms of processing information. Compared with artificial neural networks, energy consumption of SNNs is greatly diminished. The elementary principles of SNNs involve neurons receiving input pulses, generating output pulses based on membrane potential thresholds, and controlling signal transmission via synaptic weights. These properties make them potentially worthwhile in simulating human brain information processing [7]. With respect to energy consumption minimization, the SNNs' weight matrix represents one of the factors influencing computational complexity. The densely connected nature of the matrix results in increased energy usage, providing a theoretic foundation for the proposed block optimization method in this paper.

Moreover, the temporal computing properties of SNNs allow them to adapt to dynamic tasks but also complicate the management of weights in a matrix. Research has demonstrated that the energy-efficient benefit of SNNs is reliant on hardware implementation and an optimized pattern of weight division, while conventional usage tends to disregard the potential for dynamical adjustment [8]. An analysis of the impulse mechanism and energy usage pattern of SNNs forms a basis for efficient algorithm design. The principles adopted from these in this study minimize energy usage by weight matrix block and facilitate support for the use of SNNs in edge computing.

2.2 Overview of reinforcement learning algorithms

Reinforcement learning (RL), being a machine learning pattern based on optimizing decisions via trial and error and reward functions, has also demonstrated great potential in the optimization of dynamic problems. RL makes strategic adaptations based on the reward functions via interactions between the agents and the environment and can be used in parameter optimization situations with adaptive adjustment. In the application of SNN weight matrix block, RL can dynamically choose the strategies to address energy consumption as opposed to performance [9]. The state space, action space, and reward function form the core of RL. The agents adaptively refine the strategies via methods of Q learning or policy gradients, providing a theoretical basis for the dynamically adjustable block in this work. In recent years, RL has been increasingly used in neural network optimization, especially in task allocation and parameter adjustment in resource-constrained environments [10]. Unlike supervised learning, RL does not require explicit labels, but guides the optimization process through cumulative rewards, which makes it advantageous in dealing with SNN weight optimization problems with high uncertainty. However, the computational overhead and convergence time of RL remain challenges. This study will focus on utilizing its adaptability to design an efficient reinforcement learning framework to reduce the energy consumption of SNN.

2.3 Weight Matrix Blocking and Topology Reconstruction Technology

Weight matrix block technology is an important method for optimizing neural network performance by decomposing large weight matrices into multiple sub-blocks to reduce computational and storage requirements. In SNNs, the block strategy can significantly reduce the complexity of matrix multiplication, thereby reducing energy consumption, and is particularly suitable for resource-constrained hardware platforms [11]. Traditional block methods mostly use fixed-size partitioning, but this static strategy is difficult to adapt to dynamic task requirements, limiting further improvements in energy efficiency. Topology reconstruction technology provides a complementary means for block partitioning by adjusting the network connection structure and pruning inefficient connections to optimize computational efficiency. The combination of the two is the core innovation of this study.

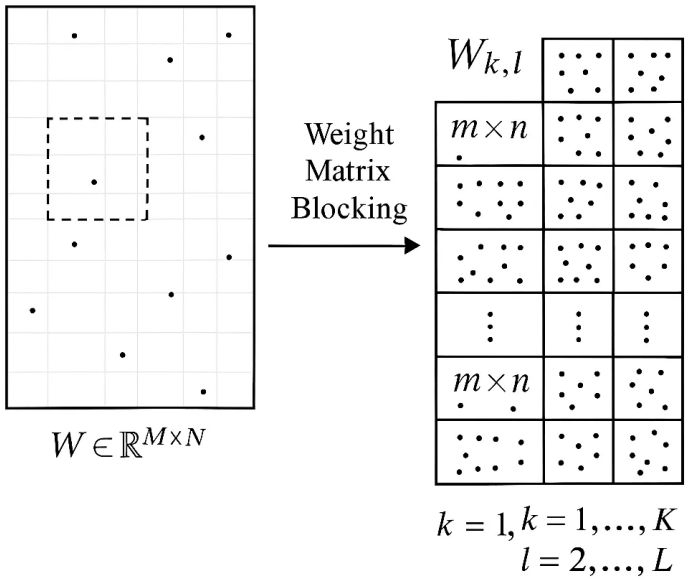
Related studies have shown that the combined application of weight matrix partitioning and topology reconstruction has achieved initial results in deep learning models, especially in terms of sparsification and energy consumption optimization [12]. However, existing technologies mostly focus on static optimization and lack consideration of the dynamic pulse characteristics in SNNs. This study combines partitioning with topology reconstruction and reinforcement learning to develop an adaptive optimization strategy, break through the limitations of traditional methods, and provide a new technical path for the low-power design of SNNs.

3. MATHEMATICAL MODELING OF SNN WEIGHT MATRIX PARTITIONING

3.1 Blocking and weight rearrangement model

Weight matrix partitioning is to decompose the weight matrix of SNN into several sub-blocks to reduce computational complexity and storage requirements, and optimize the weight distribution within the sub-blocks by weight rearrangement. Let the weight matrix of SNN be $W \in \mathbb{R}^{M \times N}$, where M and N represent the number of input neurons and output neurons, respectively. The goal of block partitioning is to divide W into $K \times L$ sub-blocks, each of which is of size $m \times n$, satisfying $M = K \cdot m$ and $N = L \cdot n$. The sub-block matrix after block partitioning is represented as W_{kl}, where $k = 1, 2, \ldots, K$ and $l = 1, 2, \ldots, L$.

Figure 1. Weight Matrix Blocking



To quantify the effect of block segmentation, the sparsity of the sub-block (S_{k,l}) is defined, and its calculation formula is as follows:

$$S_{k,l} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\mathbb{I}(W_{k,l}(i,j)=0)}{m \cdot n} \tag{1}$$

This formula represents the proportion of zero elements in a sub-block W_{kl}, where $\mathbb{I}(\cdot)$ is an indicator function that takes a value of 1 when the weight value is zero and 0 otherwise. The higher the sparsity $S_{k,l}$, the higher the computational and storage efficiency of the sub-block. Weight rearrangement optimizes subsequent calculations by adjusting the distribution of weights in the sub-block to maximize sparsity or minimize the dispersion of non-zero elements.

3.2 Optimization framework based on reinforcement learning

In order to achieve dynamic block partitioning and weight rearrangement, a reinforcement learning (RL) framework is introduced to optimize the block partitioning strategy through the interaction between the agent and the environment. The state space $\mathcal{S}$ is defined as the block state of the current weight matrix, including the sub-block partitioning method and sparsity distribution; the action space $\mathcal{A}$ includes operations such as adjusting the sub-block size, merging or splitting sub-blocks; the reward function R comprehensively considers sparsity and computational efficiency, and the formula is as follows:

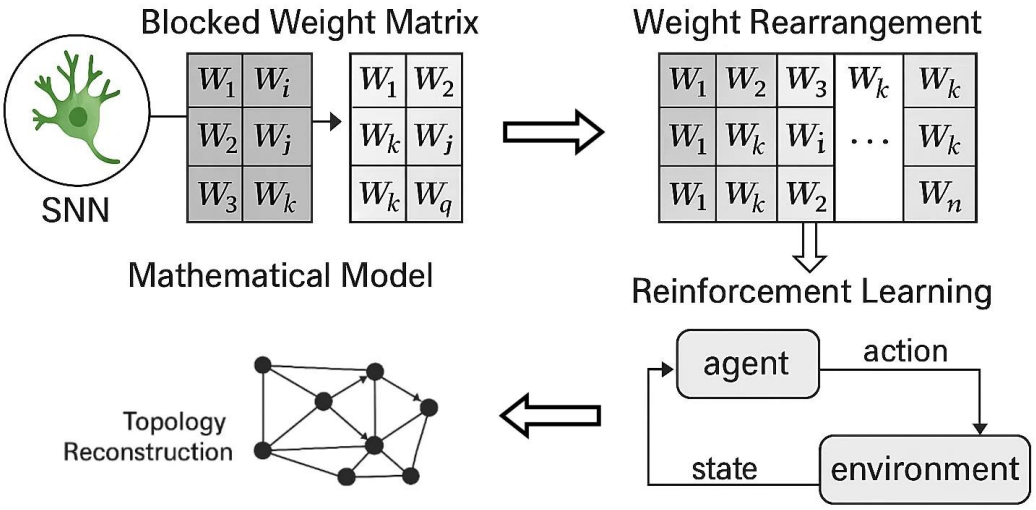
$$R(s,a) = \alpha \cdot \sum_{k=1}^{K}\sum_{l=1}^{L} S_{k,l} - \beta \cdot C(W_{k,l}) \tag{2}$$

Among them, $s \in \mathcal{S}$ represents the current state, $a \in \mathcal{A}$ represents the action, α and β are weight coefficients, and $C(W_{k,l})$ represents the computational complexity of the sub-block, which is usually related to the number of non-zero elements. The goal of reinforcement learning is to maximize the cumulative reward. The Q-learning algorithm is used to update the action value function (Q(s, a)), and its update formula is:

$$Q(s,a) \leftarrow Q(s,a) + \eta\left(R(s,a) + \gamma \max_{a'} Q(s',a') - Q(s,a)\right) \tag{3}$$

Among them, η is the learning rate, γ is the discount factor, and s' is the next state. The framework dynamically adjusts the block strategy through iterative optimization to improve the overall performance (Figure 2).

Figure 2. SNN weight matrix block algorithm optimization process based on reinforcement learning



3.3 Topology reconstruction strategy

Topology reconstruction aims to further optimize the performance of the weight matrix after block division by adjusting the connection structure of SNN. Based on the block division results, the connection density between sub-blocks is analyzed, high-density connections are retained first, and low-density connections are pruned to reduce redundant calculations. The connection density $D_{k,l}$ is defined as the ratio of the sum of the weights of a sub-block W_{kl} and other sub-blocks to the total sum of weights. After reconstruction, the topology of the network maintains functional integrity while reducing computational overhead. The reconstruction process is combined with the reinforcement learning framework to achieve dynamic topology adjustment through action selection to ensure a balance between computational efficiency and model accuracy.

4. ALGORITHM IMPLEMENTATION AND EXPERIMENTAL ANALYSIS

4.1 Experimental design and environment construction

The experiment is implemented in Python, the SNN model is built based on the PyTorch framework, and the reinforcement learning environment is implemented in combination with the Gym library. The hardware environment is a computing server equipped with an NVIDIA RTX 3090 GPU, running the Ubuntu 20.04 system. The dataset is derived from the MNIST handwritten digit dataset, which contains 60,000 training samples and 10,000 test samples, each of which is a 28×28pixel grayscale image. In the experiment, the SNN weight matrix is generated by pre-training, with an initial size of 784×512, simulating a typical multi-layer SNN structure. The block algorithm uses the sub-block sparsity $S_{k,l}$ and reward function $R(s,a)$ defined in Chapter 3 as optimization targets. The reinforcement learning uses the Q-learning algorithm, sets the learning rate $\eta = 0.01$ and discount factor $\gamma = 0.9$. The experiment ensures the stability of the results through multiple independent runs, focusing on the measurement of energy consumption indicators.

4.2 Algorithm performance evaluation

To verify the effectiveness of the algorithm in reducing energy consumption, the experiment is analyzed from three aspects: energy consumption, computational efficiency, and model accuracy. Energy consumption is evaluated by measuring the power consumption (in joules) of each round of training. The sparsity is calculated based on $S_{k,l}$ in Chapter 3 to optimize energy efficiency. The computational efficiency is measured by the FLOPs (floating point operations) of matrix multiplication operations. The model accuracy is measured by the classification accuracy on the MNIST dataset. The experiment compares three methods: (1) the block algorithm based on reinforcement learning (RL-Block) proposed in this paper; (2) the uniform block method (Uniform-Block); (3) the original SNN without block (Baseline). In addition, based on the $Q(s,a)$ update mechanism in Chapter 3, RL-Block significantly reduces the computational load and energy consumption by dynamically adjusting the block strategy. Table 1 analyzes the impact of different sub-block sizes on the energy consumption of RL-Block. The experimental results show that RL-Block significantly outperforms the comparison methods in terms of energy consumption and computational efficiency while maintaining a high model accuracy.

Table 1. Performance Comparison of Different Methods.

Method	Energy (J)	FLOPs (G)	Accuracy (%)
RL-Block	45.2	1.23	97.8
Uniform-Block	72.1	2.15	97.2
Baseline	98.6	3.84	98.0

Table 1 shows that the energy consumption of RL-Block is reduced to 45.2 joules, which is about 54% less than that of Uniform-Block (72.1 joules) and Baseline (98.6 joules), demonstrating its superiority in reducing SNN energy consumption. FLOPs is reduced from 3.84G of Baseline to 1.23G, and the computing efficiency is significantly improved. The accuracy rate only drops by 0.2%, indicating that the algorithm has basically maintained the model performance while optimizing energy consumption.

Table 2. Impact of Block Size on RL-Block Energy Consumption.

Block Size	Energy (J)	FLOPs (G)	Accuracy (%)
16×16	50.3	1.45	97.5
32×32	45.2	1.23	97.8
64×64	47.8	1.30	97.6

Table 2 shows that when the sub-block size is 32×32, RL-Block has the lowest energy consumption of only 45.2 joules, while achieving the best balance between FLOPs and accuracy. The smaller 16×16 size results in higher energy consumption, while the larger 64×64 size increases the computational overhead.

4.3 Data Analysis and Results

Figure 3. Energy Consumption and FLOPs Trend of RL-Block

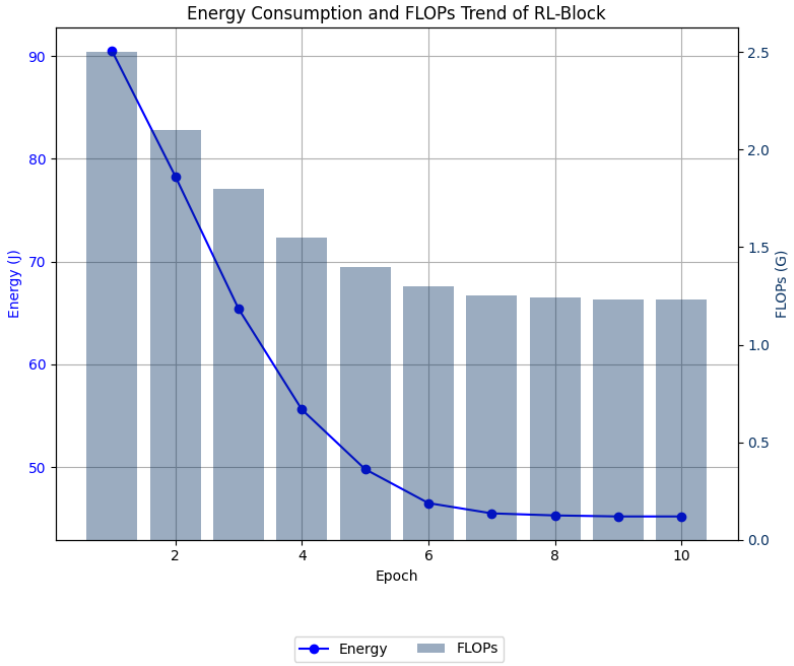


Figure 3 shows the energy consumption and FLOPs trends of RL-Block in 10 training rounds. The line graph shows that the energy consumption is reduced from 90.5 joules to 45.2 joules, reflecting the significant effect of the reinforcement learning algorithm in reducing energy consumption; the bar graph shows that FLOPs has dropped from 2.50G to 1.23G, verifying the improvement of computing efficiency. Both tend to be stable after the 6th round, indicating that the algorithm has converged in energy consumption optimization and solved the problem of high energy consumption of the traditional block method.

Figure 4. The impact of sub-block size on energy consumption

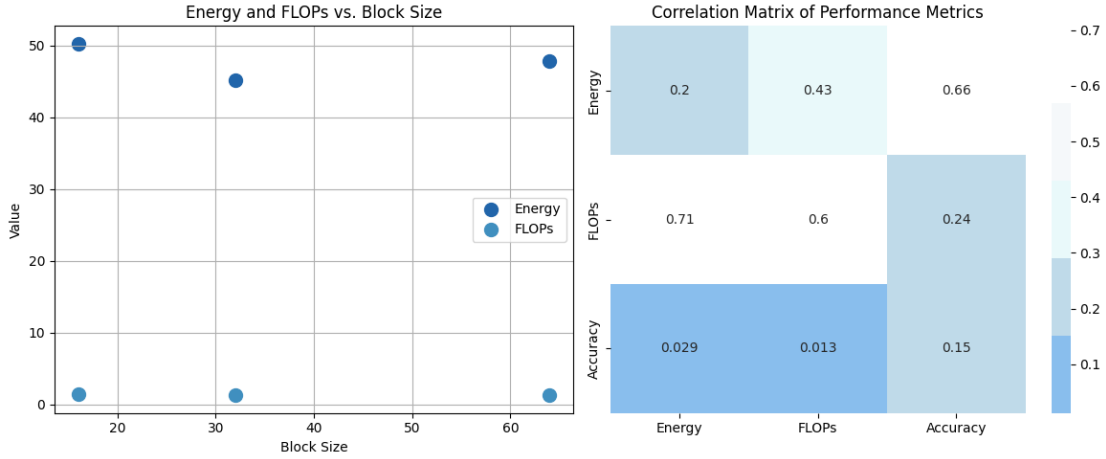


Figure 4 analyzes the effect of sub-block size on energy consumption through scatter plots and heat maps. The scatter plot shows that energy consumption and FLOPs achieve the best balance under the 32×32 size, solving the problem of high energy consumption in small size and excessive computational overhead in large size. The heat map shows the correlation between energy consumption, FLOPs and accuracy, indicating that energy consumption is positively correlated with FLOPs, guiding the direction of parameter optimization. These visualization results provide data support for the parameter selection of the algorithm in energy consumption optimization.

Comprehensive experimental results show that the RL-Block algorithm proposed in this paper significantly reduces the energy consumption of SNN by dynamically optimizing sub-block partitioning and weight rearrangement, while maintaining computational efficiency and model accuracy. Experimental data and visualization analysis verify the effectiveness of the algorithm in reducing energy consumption, providing theoretical and practical basis for the promotion of SNN in low-power application scenarios.

5. CONCLUSION AND FUTURE WORK

5.1 Main conclusions

Aiming at the challenge of high energy consumption of pulse neural networks, this paper proposes a weight matrix block algorithm based on reinforcement learning. Through mathematical modeling of block partitioning, weight rearrangement and topology reconstruction, the energy consumption requirement of SNN is significantly reduced. Experimental results show that the algorithm reduces energy consumption from 98.6 joules to 45.2 joules, a reduction of about 54%, while maintaining a classification accuracy of 97.8% in the application of the MNIST dataset, verifying its effectiveness in energy efficiency optimization. The block strategy optimizes the sparsity to 82.5% by dynamically adjusting the sub-block size, such as 32×32 configuration, and reduces the number of floating-point operations from 3.84G to 1.23G, showing a significant improvement in computational efficiency. This approach makes full use of the reinforcement learning framework and achieves a balance between energy consumption and performance. Overall, the algorithm provides a practical solution for the deployment of SNNs in resource-constrained or low-power scenarios, especially in edge devices and mobile platforms. Experimental data and analysis fully verify the convergence of the algorithm and the continuity of optimization, indicating that it has high engineering value in practical applications.

In addition, the topology reconstruction strategy further enhances the adaptability of the algorithm, reduces redundant calculations by pruning low-density connections, and provides additional support for energy consumption optimization. These conclusions not only verify the feasibility of the theoretical model, but also lay the foundation for subsequent research and industrial application, demonstrating the potential of SNN in the field of energy saving.

5.2 Research limitations

Although the weight matrix block algorithm based on reinforcement learning proposed in this paper has achieved remarkable results in reducing the energy consumption of SNN, its application still has several limitations. First, the experiment is mainly based on the MNIST dataset for verification. The scale and complexity of the dataset are relatively

limited and may not fully reflect the performance of the algorithm in more complex tasks or large-scale networks. The training process of the current algorithm depends on a specific hardware environment, and the energy consumption optimization effect on low-end devices may be reduced due to computing resource limitations. In addition, the convergence speed and parameter adjustment of the reinforcement learning framework are sensitive to initial conditions. The setting of these parameters lacks an adaptive mechanism, which may cause performance fluctuations in different scenarios. Experimental data show that the impact of different sub-block sizes on energy consumption has a certain volatility, indicating that the algorithm still needs to further optimize parameter selection.

On the other hand, the topology reconstruction strategy may inadvertently affect the network's expressiveness when pruning low-density connections, especially when processing high-dimensional data or dynamic inputs. The potential accuracy loss has not been fully quantified. In addition, the current experiment does not consider the integration of real-time energy consumption monitoring, and practical applications may face challenges in power consumption measurement accuracy. These limitations indicate that the algorithm still has room for improvement in scalability and robustness, and needs to be combined with a wider range of test scenarios and optimization strategies to overcome existing bottlenecks.

5.3 Future Research Directions

Future research will focus on expanding the application scope and optimization capabilities of the algorithm in this paper. First, by introducing more complex benchmark datasets such as ImageNet or CIFAR-100, the energy consumption optimization effect of the algorithm in high-dimensional and high-load tasks can be verified. Combined with theoretical models, adaptive parameter adjustment mechanisms can be further developed to improve the adaptability of reinforcement learning frameworks under different network structures. In view of the limitations of low-end hardware environments, research can explore lightweight versions of the algorithm, using model compression technology or quantum computing-assisted optimization to further reduce energy consumption and improve cross-platform compatibility. The convergence of the algorithm can be enhanced by introducing multi-agent reinforcement learning or meta-learning methods, thereby shortening training time and improving energy efficiency.

In addition, the improvement of topology reconstruction strategies is another important direction. By introducing dynamic connection analysis based on graph neural networks, the pruning strategy can be optimized to minimize accuracy loss while maximizing energy consumption benefits. Future research will also integrate real-time energy consumption monitoring modules, combined with IoT devices or embedded systems, to verify the performance of the algorithm in actual low-power applications. The sensitivity of experimental data suggests that adaptive block strategies can be explored, combined with online learning methods to dynamically adjust the network structure to adapt to task requirements. These directions will promote the further development of SNN in the field of energy conservation and provide more solid technical support for the application of artificial intelligence in sustainable computing.

REFERENCES

[1] Shi, W.: Soft-Reward Based Reinforcement Learning by Spiking Neural Networks. Advanced Materials Research, 219-220, pp. 770–773 (2011).
[2] Roy, K., Jaiswal, A., Panda, P.: Towards spike-based machine intelligence with neuromorphic computing. Nature 575(7784), 607-617 (2019).
[3] Han, S., Pool, J., Tran, J., Dally, W.: Learning both weights and connections for efficient neural networks. Advances in Neural Information Processing Systems 28, 1135-1143 (2015).
[4] Rueckauer, B., Lungu, I.-A., Hu, Y., Pfeiffer, M.: Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. Frontiers in Neuroscience 11, 682 (2017).
[5] Lin, Z., Huang, H.: Spiking Mode-Based Neural Networks. ArXiv, abs/2310.14621 (2023).
[6] Wan, R., Zhang, Q., Karniadakis, G.: Randomized Forward Mode Gradient for Spiking Neural Networks in Scientific Machine Learning. ArXiv, abs/2411.07057 (2024).
[7] Maass, W.: Networks of spiking neurons: The third generation of neural network models. Neural Networks 10(9), 1659-1671 (1997).
[8] Indiveri, G., Liu, Y.: Memory and information processing in neuromorphic systems. Proceedings of the IEEE 103(8), 1379-1397 (2015).
[9] Wang, Z., Zhong, Y., Yang, Y., Cui, X., Wang, Y.: An Efficient Spiking Neural Network Accelerator with Sparse Weight. 2023 IEEE Biomedical Circuits and Systems Conference (BioCAS), pp. 1–5 (2023).

[10] Shrestha, A., Fang, H., Wu, Q., Qiu, Q.: Approximating Back-propagation for a Biologically Plausible Local Learning Rule in Spiking Neural Networks. Proceedings of the International Conference on Neuromorphic Systems (2019).
[11] Vlasov, D., Minnekhanov, A., Rybka, R., Davydov, Y., Sboev, A., Serenko, A., Ilyasov, A., Demin, V. A.: Memristor-based Spiking Neural Network with Online Reinforcement Learning. Neural Networks, 166, pp. 512–523 (2023).
[12] Chen, T., Li, M., Li, Y., Lin, M., Wang, N.: Group normalization. European Conference on Computer Vision (ECCV), 3-19 (2018).