PicoSleepNet: An Ultra Lightweight Sleep Stage Classification by Spike Neural Network using Single-channel EEG Signal

Shengnan Liu, Haoming Chu, Yukun Feng, Yulong Yan, Ning Ma, and Yuxiang Huan

Abstract—**This study introduces PicoSleepNet, an ultra-lightweight sleep stage classification method that utilizes a spiking neural network (SNN) with single-channel electroencephalogram (EEG) signals. Traditional methods use multi-bit Nyquist sampling and dense computing, which result in high complexity and power consumption, hindering their deployment on wearable devices. To address these limitations, we propose an innovative pipeline combining single-bit sub-Nyquist level-crossing sampling (LCS) and sparse computing based on SNN. First, LCS adaptively encodes EEG signals into event-driven spike sequences, reducing data volume by 6.98× while preserving essential signal characteristics compared to Nyquist sampling. Second, a sparse recurrent spiking neural network (RSNN) architecture, optimized by the masked backpropagation and sparse regularization (Masked-BPSR) technique, improves performance and reduces computational costs. Third, quantization-aware training (QAT) ensures that the model maintains high accuracy with low-bitwidth quantization, significantly reducing computational power consumption and enabling hardware-friendly deployment. Compared with current state-of-the-art sleep staging approaches, PicoSleepNet achieves competitive performance on three public datasets (Sleep-EDF-20, Sleep-EDF-78, and ISRUC-Sleep) with accuracies of 83.5%, 77.9%, 79.4% and macro-F1 scores of 75.2%, 68.1%, 77.2%, respectively. Meanwhile, by leveraging the computational sparsity design of RSNN and the joint optimization of Masked-BPSR and QAT, PicoSleepNet achieves an ultra-lightweight model with only 14.0-25.8K parameters (reduced by nearly 2×) and 681.4-842.0K operations (reduced by 27×), reducing computational power consumption by 1480×. This approach demonstrates the feasibility of deploying ultra-lightweight sleep staging systems in wearable devices and neuromorphic hardware, paving the way for broader applications in real-time health monitoring.**

Index Terms—**Ultra Lightweight, Sleep Stage Classification, Single-Channel EEG, Level-Crossing Sampling, Spiking Neural Network, Quantization-Aware Training, Low Power Computing**

This work was supported in part by the Key-Area Research and Development Program of Guangdong Province, China, under Grants 2023B0303030003, 2023B0303040002, and 2021B0909060002. (Shengnan Liu, Haoming Chu, and Yukun Feng contributed equally to this work.) (Corresponding author: Ning Ma, Yuxiang Huan.)

Shengnan Liu is with the School of Biological Science and Medical Engineering, Beihang University, Beijing, China (e-mail: liushengnan@buaa.edu.cn).

Haoming Chu, Ning Ma, and Yuxiang Huan are with the Guangdong Institute of Intelligence Science and Technology, Hengqin, Zhuhai 519031, Guangdong, China (e-mail: chuhaoming@gdiist.cn; maning@gdiist.cn; yxhuan@gdiist.cn).

Yukun Feng is with the State Key Laboratory of Internet of Things for Smart City and the Department of Computer and Information Science, University of Macau, Macao 999078, China, and also with the Guangdong Institute of Intelligence Science and Technology, Hengqin, Zhuhai 519031, Guangdong, China (e-mail: fengyukun@gdiist.cn).

Yulong Yan is with the School of Information Science and Technology, Fudan University, Shanghai 200433, China (e-mail: ylyan17@fudan.edu.cn).

I. Introduction

SLEEP is a vital process in human life and significantly impacts both physical and mental well-being [1]. Sleep staging serves as a primary method to assess sleep quality and diagnose sleep disorders [2]. Traditional sleep monitoring requires participants to wear polysomnography (PSG) equipment overnight. The PSG equipment incorporates multiple electrodes and sensors to capture various physiological signals, including electroencephalogram (EEG), electrooculogram (EOG), electrocardiogram (ECG), electromyogram (EMG), and respiration [3]. According to the American Academy of Sleep Medicine (AASM) manual, sleep physicians divide PSG into multiple 30-second epochs and assign a label to each epoch, such as wakefulness (Wake), non-rapid eye movement stages (N1, N2, N3), or rapid eye movement (REM) [4]. Since this manual sleep staging process is tedious, time-consuming, and highly dependent on the individual expertise of physicians, numerous automatic sleep staging approaches based on machine learning and deep learning have been developed, aiming to reduce the burden on physicians and provide effective sleep assessments.

Traditional PSG devices are expensive and inconvenient, making long-term continuous monitoring impractical. These limitations have driven wearable devices to emerge as an important direction. However, wearable sleep monitoring faces several technical constraints, including limited physiological signal acquisition channels and stringent requirements for both computational complexity and power efficiency. Most sleep staging approaches primarily employ convolutional neural networks (CNNs) or recurrent neural networks (RNNs) in their network architectures, where CNNs are capable of extracting time-frequency information from signals, and RNNs are adept at processing temporal dependencies. Although these deep learning approaches [5]–[11] have achieved remarkable classification accuracy, their complex network architectures and huge computational overhead pose significant challenges for practical implementation in wearable devices. For example, DeepSleepNet achieved 82% classification accuracy but required approximately 21M parameters [12]. To address these challenges, several studies have focused on developing lightweight sleep staging approaches, mainly using single-channel EEG signals and optimizing the neural network structure. For instance, U-Time adopted a fully convolutional

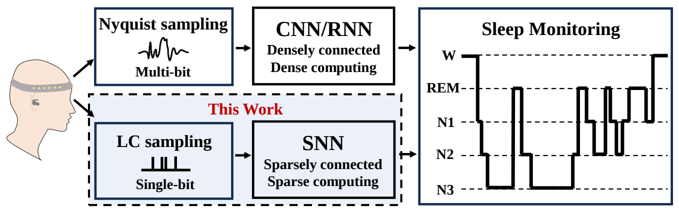

Fig. 1. The proposed method utilizes single-bit level crossing (LC) sampling and sparsely connected SNNs to achieve efficient and sparse computing for sleep stage monitoring.

encoder-decoder network, reducing parameters by an order of magnitude compared to DeepSleepNet [13]. TinySleepNet combined CNN and unidirectional long short-term memory (LSTM), reducing parameters to 1.3M while improving accuracy to 85.4% [14]. MicroSleepNet achieved real-time processing on mobile devices with 82% accuracy through lightweight convolution and attention mechanisms [15]. LightSleepNet further compressed parameters to 43.08K and reduced computational complexity to 45.76 MFLOP through an innovative 1D-CNN design, while maintaining 83.8% accuracy [16]. Although these traditional artificial neural network (ANN)-based sleep staging methods have made significant progress in reducing model size and complexity by adopting single-channel EEG and simplified network structures, they fundamentally remain within the dense computing paradigm. These models typically execute fully activated matrix multiplications across all layers, even for inactive neurons. Moreover, these models usually rely on multi-bit Nyquist-sampled EEG input, leading to huge data volume and high acquisition power consumption. Therefore, this study aims to design an end-to-end low-power pipeline from signal acquisition to model inference, capable of performing complete on-device computation under strict power constraints (As shown in Fig.1).

Spiking neural network (SNN) is a biologically inspired computational model that reduces power consumption by performing sparse computing on binary spike sequences. SNN can effectively exploit the temporal dynamics of physiological signals and capture complex patterns of physiology. In recent years, SNNs have been increasingly applied to biomedical classification tasks due to their event-driven computing and energy-efficient processing. For example, SNNs have been successfully used in image-based disease detection, ECG signal classification, neurological disorder diagnosis, cognitive state detection, and EEG sleep staging [17]–[22]. Currently, existing EEG sleep staging methods involving SNNs are typically used as a component to improve the performance of ANNs, but do not consider computational complexity and power consumption [23], [24]. Jia et al. proposed a sleep staging model based on a hybrid spiking neural network (HSNN) with 6-channel EEG signals as input, demonstrating the great potential of SNN in sleep staging [25]. However, it does not incorporate sparsity mechanisms, resulting in a relatively large number of model parameters. Additionally, they employed Ben's spiking algorithm (BSA) to encode EEG signals as input to the HSNN. BSA is fundamentally an offline post-processing algorithm applied to Nyquist-sampled signals, and thus cannot reduce data volume at the system level.

In this study, we propose PicoSleepNet, a model that employs level-crossing sampling (LCS) to encode EEG signals as input to a recurrent spiking neural network (RSNN). LCS is a data-driven sub-Nyquist sampling strategy that enables direct acquisition of single-bit spike sequences compatible with SNNs through level-crossing analog-to-digital converters (LC-ADCs), eliminating the need for additional spike encoding [26]. Compared to conventional multi-bit Nyquist sampling, LCS significantly reduces data volume and improves energy efficiency. To further reduce RSNN complexity, we incorporate a masked backpropagation with sparsity regularization (Masked-BPSR) module to promote synaptic and neuronal sparsity during training. This leads to a highly sparse network structure with significantly fewer operations and parameters, supporting efficient inference with minimal computational overhead. Finally, we apply quantization-aware training (QAT) to compress model weights into low-bit representations. QAT simulates quantization effects during training to retain accuracy while substantially reducing relative power and memory requirements. LCS, Masked-BPSR, and QAT together form a tightly integrated pipeline from signal acquisition to model inference, enabling end-to-end low-power optimization. This makes PicoSleepNet particularly well suited for deployment on energy-constrained wearable devices and neuromorphic systems. The main contributions of this paper are as follows:

1) *Sub-Nyquist EEG encoding via LCS:* We introduce LCS to convert continuous raw EEG signals into sparse spike sequences, achieving a 6.98× reduction in data volume while preserving key temporal characteristics.

2) *Masked-BPSR for sparse optimization of RSNN:* We propose a masked sparsity regularization method that jointly constrains synaptic connections and neuronal firing, reducing operations by 1.98× and parameters by 1.73× without degrading classification accuracy.

3) *QAT for low-bit quantization:* We apply QAT to compress model parameters into low-bit representations, achieving 62.5× reduction in computational power consumption while maintaining model performance.

II. Methodology

Fig.2 illustrates the framework of PicoSleepNet. The LCS encodes EEG signals into spike sequences, effectively reducing data volume while preserving signal features. These spike sequences are fed into the RSNN model through a sliding window method. The Masked-BPSR technology dynamically prunes redundant synapses and inhibits neural firing during model training, effectively reducing model parameters and the number of operations, and forming a sparse RSNN with high computational efficiency. Furthermore, the QAT method quantizes the sparse RSNN, reducing computational power consumption while minimizing the accuracy loss resulting from quantization. Overall, PicoSleepNet effectively performs sleep staging while achieving ultra-low power consumption.

A. LCS Encoding

SNNs transmit information via binary spike events, requiring real-valued EEG signals encoded into spike sequences as

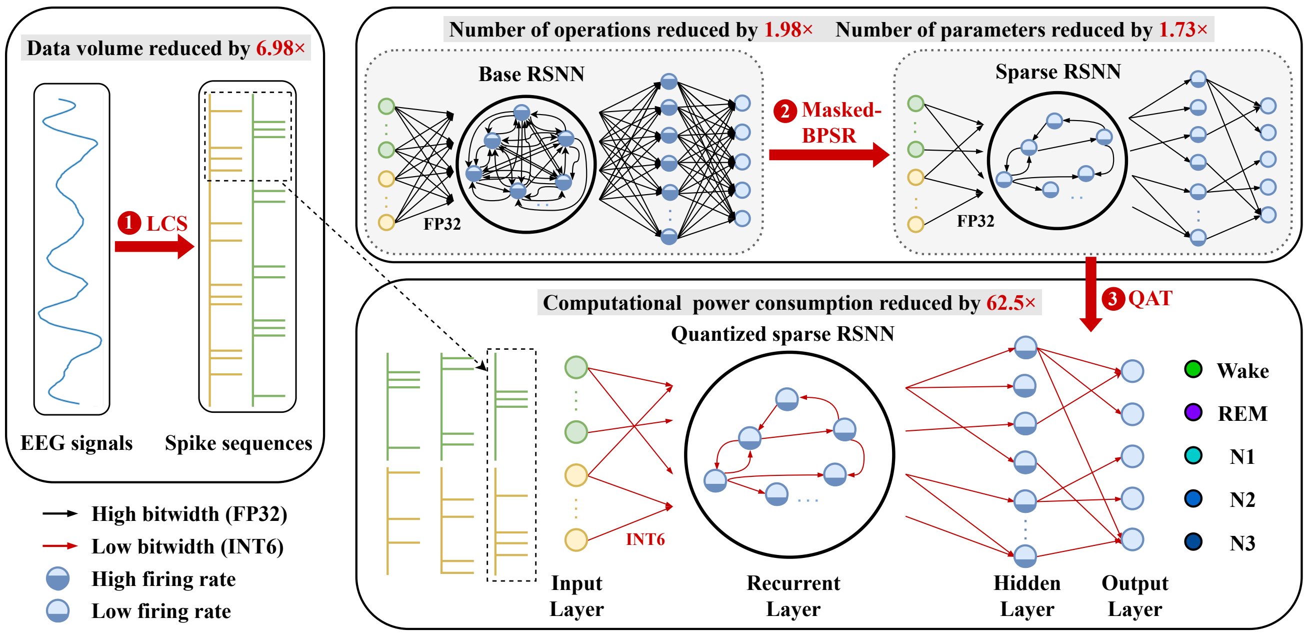

Fig. 2. The framework of PicoSleepNet. (1) The LCS method is used to encode the EEG signals into spike sequences, resulting in a 6.98× reduction in data volume compared to Nyquist sampling. The spike sequences are fed into the RSNN model through a sliding window method. (2) During training, the Masked-BPSR method produces a sparse RSNN, which reduces the number of operations by 1.98× and the number of parameters by 1.73× compared to the base RSNN (without Masked-BPSR). (3) Further, the QAT method generates a quantized sparse RSNN, with a computational power consumption reduction of 62.5× compared to the unquantized model.

input. The LCS is a non-uniform sampling technique inspired by neuromorphic sensors [27]. LCS converts analog signals directly into discrete neural spikes, making it ideal for SNN processing. Previous studies have demonstrated that the LCS provides an effective method for encoding digital EEG signals [28]. The sampling points of the LCS are determined by signal level changes instead of the fixed clock frequencies in traditional Nyquist sampling. This important property causes the sampling density to increase when the signal changes rapidly, and decrease when the signal changes slowly. This adaptive sampling effectively reduces data volume while preserving key signal information. Taking the EEG signals in the Sleep-EDF-20 dataset (100 Hz Nyquist sampling, 14-bit resolution) as an example, LCS generates 2 channel spike sequences with only 1 bit representing the spike, and the average data volume decreases by 6.98×. LC-ADCs convert analog EEG signals directly into spike sequences with no need for additional spike-encoding circuits, which is less power-consuming than Moreover, LC-ADCs convert analog EEG signals directly into spike sequences with no need for additional spike-encoding circuits, which is less power-consuming than traditional Nyquist sampling sensors. Therefore, this study employs the LCS method for EEG encoding.

The LCS predefines a threshold, and whenever the amplitude change of the EEG signal exceeds the threshold, a positive or negative spike potential is generated. Specifically, assuming that the threshold is set to δ, a positive spike will be generated when the signal rises by more than δ, and a negative spike will be generated when the signal falls by more than δ. Since the EEG signals used in this study are digital rather than analog, the LCS encoding process is slightly different from the original LCS due to the discreteness of the digital signals. The LCS encoding process is implemented by Algorithm 1.

Algorithm 1 The LCS Encoding Process

```
Input: $X[1:N]$ representing EEG signal with $N$ sampling
    points, $\delta$ representing LCS threshold
Output: $S^+$ for positive spikes, $S^-$ for negative spikes
1:  $last\_point \leftarrow X[0]$
2:  for $n = 1$ to $N$ do
3:      $s \leftarrow \lfloor (X[n] - last\_point)/\delta \rfloor$
4:      if $s \neq 0$ then
5:          $last\_point \leftarrow X[n]$
6:      end if
7:      if $s > 0$ then
8:          $S^+[n] \leftarrow s$
9:      else if $s < 0$ then
10:         $S^-[n] \leftarrow s$
11:     end if
12: end for
```

Fig.3 illustrates an example of the LCS encoding process. The LCS effectively encodes the EEG signal into positive and negative spike sequences, which reflect the temporal-frequency variations of the EEG. In different sleep stages, the generated spikes have different characteristics. For instance, the spike sequence presents uniform and sparse spikes in the Wake clip, while the spike sequence exhibits more widely spaced and dense spikes in the N3 clip.

B. RSNN Model

The RSNN model in this study is constructed based on leaky integrate-and-fire (LIF) neurons, which have been widely

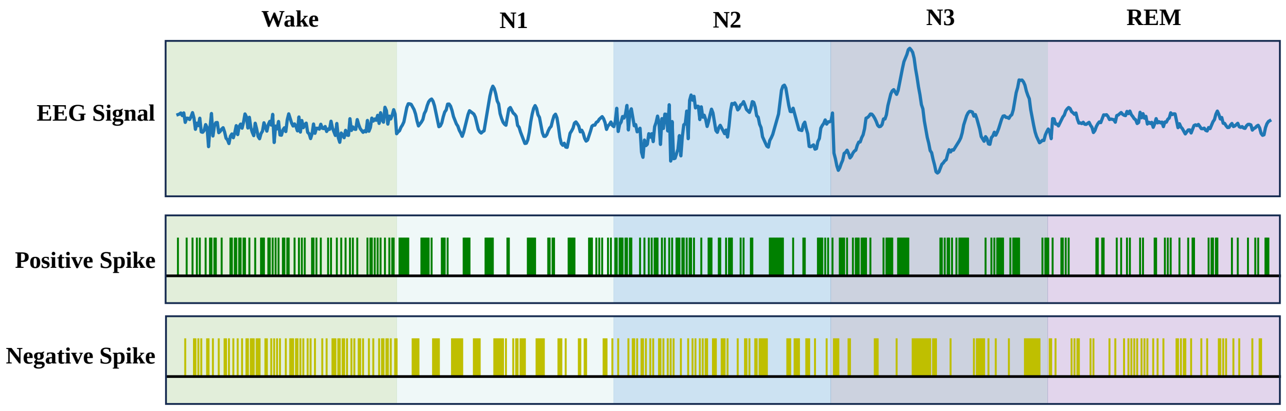

Fig. 3. An example of EEG encoding using the LCS. The figure captures five clips of a subject during one night from the Sleep-EDF-20 dataset, corresponding to five sleep stages. Each clip is 1.5 seconds long and sampled at 100 Hz.

used in SNN studies [29]. The LIF neuron model simulates the dynamics of biological neuron spiking behavior, including current integration, firing, and leakage processes. It possesses both temporal and spatial characteristics. Although LIF neurons increase the complexity of individual neurons compared to traditional ANNs, they facilitate the construction of sparse, event-driven RSNN. Moreover, the LIF neuron strikes a good balance between computational complexity and biological interpretability while retaining the fundamental characteristics of biological neurons.

The membrane potential of the LIF neuron integrates all input spikes and it will leak if the firing threshold is not exceeded. When the membrane potential exceeds the firing threshold, the neuron generates an output spike and its membrane potential is reset to the resting potential. The dynamics of a LIF neuron can be described by the following equation:

$$v^t = \tau \cdot v^{t-1} \cdot \neg s^{t-1} + \sum w \cdot m^t + bias, \tag{1}$$

$$s^t = \mathcal{H}(v^t - V_{th}) = \begin{cases} 1, \text{if } v^t \geq V_{th} \\ \\ 0, \text{otherwise} \end{cases}, \tag{2}$$

where v^t represents the membrane potential of the neuron at time t, and τ denotes the potential leakage rate. m^t and s^t represent the input spike (pre-synaptic spike) and output spike (post-synaptic spike) of the neuron, respectively, where $m^t, s^t \in \{0, 1\}$. $\mathcal{H}$ is the Heaviside step function, controlling spike generation by comparing membrane potential v^t with threshold potential V_{th}. $\neg$ represents logical NOT operation, and the leakage and reset of membrane potential are controlled by $\neg s^t$. When $s^t = 0$, indicating that the neuron has not exceeded the firing threshold, the membrane potential decays according to τ. When $s^t = 1$, indicating that the output spike is generated, the membrane potential returns to resting potential. In the summation term, w represents the synaptic weights of the input spikes, and $bias$ represents the neuron bias.

Theoretically, both s^t and m^t are non-differentiable step functions as equation 2, which affect the learning of SNN. Since s^t only controls whether the membrane potential leaks or resets and does not provide information for the membrane potential update, its derivative does not need to be considered during the backpropagation process. Wu *et al.* proposed to use the derivative of the Gaussian cumulative distribution function to approximate the derivative of spike activity, making the spike function m^t of the LIF neuron differentiable [30]. Consequently, the approximate derivative of m^t is given by equation 3, and its shape can be controlled by coefficient α.

$$f'(v^t) \approx \frac{1}{\sqrt{2\pi\alpha}} e^{-\frac{(v^t - V_{th})^2}{2\alpha}}, v^t \in \mathbb{R}. \tag{3}$$

The RSNN model consists of four layers:

1) Input Layer: The input layer consists of two parts, each with 40 neurons, for a total of 80 neurons. These two parts are used for positive and negative spike sequence input, respectively. For each sleep epoch, the EEG signal is encoded by the LCS to form a positive spike sequence and a negative spike sequence with the same length. Each spike sequence is fed into the model using a sliding window of length 40 points (Fig.2), and there is no overlap between sliding windows. For example, a 30-second EEG signal sampled at 100 Hz can be encoded into positive and negative spike sequences with a length of 3000 points each. Consequently, 75 iterations are required to complete the input of one epoch using the sliding window.

2) Recurrent Layer: The recurrent layer is a reservoir [31] containing 150 neurons. The output spikes from this layer are fed back into the same layer through recurrent connections with synaptic weights w_{li}. Therefore, neurons in the recurrent layer receive spikes from both the input layer and the recurrent layer. The dynamics of the neurons in the recurrent layer are described in Equation 4.

$$v_i^t = \tau_i \cdot v_i^{t-1} \cdot \neg s_i^{t-1} + \sum_{j \in \mathbb{L}_1} w_{ji} \cdot m_j^t + \sum_{l \in \mathbb{L}_2} w_{li} \cdot s_l^{t-1} + bias_i, i \in \mathbb{L}_2, \tag{4}$$

where $\mathbb{L}_1$ represents the input layer and $\mathbb{L}_2$ represents the recurrent layer. Initially, neurons in the recurrent layer are fully connected.

3) Hidden Layer: The number of neurons in the hidden layer is 50.

4) Output Layer: The output layer contains 5 neurons corresponding to the five sleep stages, namely Wake, N1, N2, N3, and REM.

All connections in the RSNN, including inter-layer connections and recurrent connections, are sparse via Masked-BPSR to minimize model complexity. In general, the RSNN model effectively maps sleep EEG spike sequences to high-dimensional states via a recurrent structure and extracts hidden features for sleep stage classification.

C. Masked-BPSR Training

Neuron firing rate and synaptic density can significantly affect the computational complexity during the inference of the SNN model. To achieve a lightweight model, it is necessary to minimize the neuron firing rates and synaptic density while maintaining accuracy. The BPSR algorithm has been developed for SNN learning [32]. In this paper, we further improve the BPSR method and develop Masked-BPSR to achieve better learning performance and sparsity optimization.

BPSR combines backpropagation and spike sparsity regularization, using the l^2 norm to constrain neuron spike firing rates. Since $s^t \in \{0,1\}$, using l^2 norm only results in a two-fold gradient difference compared to the l^1 norm. Therefore, Masked-BPSR adopts the l^1 norm for the spike sparse penalty, which is expressed by equation 5:

$$L_{spike} = \lambda_s \sum_{i\in\mathbb{L}_1}^{\mathbb{L}_{n-1}} \sum_t \left\| s_i^t \right\|_1 \quad s_i^t \in \{0,1\}, \tag{5}$$

where λ_s is the spike sparsity coefficient. The output layer $\mathbb{L}_n$ is excluded because its spikes do not trigger synaptic operations, and penalizing them would affect model accuracy. Adding spike sparsity regularization in the loss function can limit the neuron firing rates and reduce spike redundancy, thereby reducing communication and computation overhead.

To achieve synaptic sparsity, BPSR proposes a reconnection mechanism based on synaptic weights and gradients, optimizing network structure through synaptic pruning and growth. In BPSR, the pruning method operates independently and cannot be combined with synaptic sparsity to form an effective systematic optimization of SNN. In addition, BPSR uses the l^1 norm on synaptic weights to achieve synaptic sparsity, which inhibits the weight growth of synapses that do not need to be pruned, thus affecting accuracy. Synaptic sparsity can be achieved using the l^0 norm on weight parameters. However, l^0 regularization is not differentiable, making it impossible to learn synaptic sparsity during backpropagation. In Masked-BPSR, we introduce synaptic-mask sparsity, which uses learnable mask parameters overlaid on synaptic weights to indicate whether the corresponding synaptic connections are pruned or grown. Taking the recurrent layer as an example, the membrane potential calculation of the LIF neuron becomes:

$$v_i^t = \tau_i \cdot v_i^{t-1} \cdot \neg s_i^{t-1} + \sum_{j\in\mathbb{L}_1} \widetilde{w}_{ji} \cdot m_j^t + \sum_{l\in\mathbb{L}_2} \widetilde{w}_{li} \cdot s_l^{t-1} + bias_i, i \in \mathbb{L}_2, \tag{6}$$

$$\widetilde{w}_{ji} = w_{ji} \cdot \mathcal{H}(k_{ji}), \ \widetilde{w}_{li} = w_{li} \cdot \mathcal{H}(k_{li}), \tag{7}$$

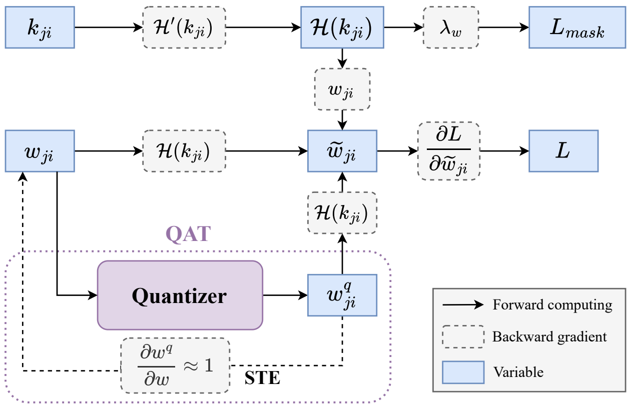

Fig. 4. The forward computation of synaptic-mask sparsity. The solid arrows represent forward computation, and the dashed arrows represent backpropagation. The gray dashed variable represents the backward gradient of the forward computation. The purple dashed box shows the computation process during QAT fine-tuning.

where k_{ji}, and k_{li} both represent synaptic masks, indicating the importance of synapses at corresponding positions. They determine whether synapses are pruned or grown and are initialized to 1. The synaptic mask $\mathcal{H}(k_{ji})$ is binarized by the Heaviside function, thus the synaptic weight w_{ji} is updated to $\widetilde{w}_{ji}$. Specifically, when $k_{ji} \geq 0$, the corresponding synaptic weight is preserved, otherwise it is pruned. The forward and backward propagation computation process of the synaptic mask is shown in Fig.4. Consequently, the l^0 norm of synaptic weights can be represented by the l^1 norm of synaptic mask parameters, given by:

$$\left\|\widetilde{w}_{ji}\right\|_0 = \left\|\mathcal{H}(k_{ji})\right\|_1. \tag{8}$$

This leads to the synaptic mask sparse penalty equation:

$$L_{mask} = \lambda_w \sum_{j\in\mathbb{L}} \left\|\mathcal{H}(k_{ji})\right\|_1, \tag{9}$$

where λ_w is used to adjust the intensity of the synaptic-mask sparsity. In addition, the calculation of $\mathcal{H}'(k_{ji})$ can refer to equation 3.

This study uses cross-entropy loss to measure the difference between model predictions and labels. The loss function incorporates two regularization constraint terms for spike and synaptic-mask sparsity. The loss function is given by:

$$L = -\sum_c y_c \log(p_c) + \lambda_s \sum_{i\in\mathbb{L}_1}^{\mathbb{L}_{n-1}} \sum_t \left\| s_i^t \right\|_1 + \lambda_w \sum_{j\in\mathbb{L}} \left\|\mathcal{H}(k_{ij})\right\|_1. \tag{10}$$

The details of parameter gradient derivation are thoroughly described in [32] and thus omitted here.

D. Low-bitwidth Quantization by QAT

Neuroscience research has found that a single synapse stores only a few bits of information [33]. SNN models can also be represented using low-bitwidth data, further reducing overhead. Quantization enables fixed-point computations instead of floating-point operations, which can significantly

TABLE I
DETAILS OF THREE DATASETS USED IN OUR EXPERIMENTS (EACH SAMPLE IS 30 SECONDS OF EPOCH)

Dataset	Subjects	Sampling Rate	EEG Channel	Wake	N1	N2	N3	REM	Total Samples
Sleep-EDF-20	20	100Hz	Fpz-Cz	8285	2804	17799	5703	7717	42308
				19.6%	6.6%	42.1%	13.5%	18.2%	
Sleep-EDF-78	78	100Hz	Fpz-Cz	65951	21522	69132	13039	25853	195479
				33.7%	11.0%	35.4%	6.7%	13.2%	
ISRUC-Sleep	10	200Hz	C3-A2	1674	1217	2616	2016	1066	8589
				19.5%	14.2%	30.4%	23.5%	12.4%	

reduce resource overhead and computational power consumption. However, quantization inevitably introduces quantization errors and leads to accuracy loss, especially in low-bitwidth quantization. This study implements QAT based on the model, which enables the model to adapt to the effects of quantization and compensate for accuracy losses by retraining and fine-tuning parameters upon quantization. Compared to post-training quantization methods, QAT generally achieves smaller accuracy degradation.

Specifically, this study applies quantizers to parameters in the forward computation, including synaptic weights, biases, and membrane potentials. In the backpropagation process, the straight-through estimator (STE) method [34] is introduced to address the challenges of quantized gradient computation. The STE disregards quantization errors by treating the quantizer as an approximately equivalent mapping (as illustrated in Fig.4). Taking synaptic weights as an example:

$$\frac{\partial w^q}{\partial w} \approx 1. \tag{11}$$

Therefore, the parameter gradient can be expressed as:

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial w^q} \cdot \frac{\partial w^q}{\partial w} \approx \frac{\partial L}{\partial w^q}. \tag{12}$$

Based on the STE method, the gradient descent method can be used to update the parameters, enabling the perception of parameter changes during the QAT process.

III. DATASETS AND EVALUATION METRICS

A. Datasets

We evaluate our method using three public datasets, namely Sleep-EDF-20, Sleep-EDF-78, and ISRUC-Sleep. Table I summarizes these datasets. Our main results are obtained using an epoch-wise data split strategy, where 10% of all epochs are randomly selected as the test set. For fair comparison with prior works that employ subject-wise protocols, we additionally conduct subject-wise k-fold cross-validation, where k is set to 20, 10 and 10 for Sleep-EDF-20, Sleep-EDF-78 and ISRUC-Sleep respectively.

1) Sleep-EDF-20: The Sleep-EDF-20 dataset is from the Sleep-EDF Expanded database and contains 20 subjects aged 25-34 [35], [36]. Each subject underwent PSG recording for two consecutive nights, but due to equipment failures, only 39 sleep recordings are available. The PSG recordings in this dataset were sampled at 100 Hz, including EEG (Fpz-Cz, Pz-Oz), EOG, and EMG. Sleep experts manually labeled each 30-second epoch into 8 categories according to the R&K standard (W, N1, N2, N3, N4, REM, MOVEMENT, and UNKNOWN) [2]. Consistent with previous studies [15], [16], [37], [38], we follow the AASM standard by merging the stages N3 and N4 into N3, considering five stages (W, N1, N2, N3, and REM) [4]. In our experiments, we use the raw data from the Fpz-Cz EEG channel.

2) Sleep-EDF-78: The Sleep-EDF-78 dataset is an expanded version of the Sleep-EDF-20 dataset and contains 78 healthy Caucasian subjects aged 25-101 [35], [36]. Similar to Sleep-EDF-20, two consecutive nights of PSG recordings were collected for each subject. Due to equipment failures, only 153 sleep recordings are available. Sleep-EDF-78 has the same PSG sampling rate, leads, and labeling method as Sleep-EDF-20. Similarly, we consider five sleep stages and use the raw data from the Fpz-Cz EEG channel.

3) ISRUC-Sleep: The ISRUC-Sleep dataset was recorded by the Sleep Medicine Center of Coimbra University between 2009 and 2013 [39]. PSG recordings were collected from subjects for 8 hours throughout the night, including EEG (F3-A2, C3-A2, O1-A2, F4-A1, C4-A1, O2-A1), EOG, ECG, and EMG, with a 200Hz sampling rate. Two experienced sleep experts independently scored each 30-second PSG epoch according to the AASM standard and divided it into five sleep stages (W, N1, N2, N3, and REM). The ISRUC-Sleep-III dataset has been used by an SNN model for sleep staging, which contains 10 PSG records from 10 healthy subjects [25]. Therefore, it is also selected in this study, using the C3-A2 EEG channel.

B. Evaluation Metrics

Two key metrics are employed to evaluate the performance of our model, namely accuracy (ACC) and macro-F1 score (MF1). The overall ACC indicates the proportion of correctly predicted samples to the total number of samples, which can be expressed by:

$$\mathrm{ACC} = \frac{\sum_{i=1}^{K} \mathrm{TP}_i}{M}, \tag{13}$$

where M is the total number of samples and K is the number of classes. TP_i is the number of true positives for the $i\text{-}th$ class.

MF1 is a common metric to evaluate the performance of the model in multi-classification problems on unbalanced datasets, which is given by:

$$\mathrm{MF1} = \frac{1}{K} \sum_{i=1}^{K} \frac{2 \cdot \mathrm{PR}_i \cdot \mathrm{RE}_i}{\mathrm{PR}_i + \mathrm{RE}_i}, \tag{14}$$

TABLE II
PERFORMANCE OF THE PROPOSED MODEL IN SLEEP STAGING WITHOUT QUANTIZATION

Dataset	Per-Class F1					Overall Metrics	
	Wake	N1	N2	N3	REM	ACC	MF1
Sleep-EDF-20	89.2	34.0	88.4	87.1	75.9	83.3	74.9
Sleep-EDF-78	90.7	30.8	82.1	74.0	66.5	78.1	68.8
ISRUC-Sleep	89.5	55.5	77.7	88.1	72.9	79.0	76.7

where $\mathrm{PR}_i = \frac{\mathrm{TP}_i}{\mathrm{TP}_i+\mathrm{FP}_i}$, represents the precision, which is the proportion of true positive samples among all samples predicted to be positive. $\mathrm{RE}_i = \frac{\mathrm{TP}_i}{\mathrm{TP}_i+\mathrm{FN}_i}$, represents the recall, which measures the proportion of correct predictions for each class. TP_i, TN_i, FP_i, and FN_i represent the number of true positive, true negative, false positive, and false negative samples for the i-th class, respectively.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

The models in this study are built with PyTorch 2.3 and trained on an NVIDIA RTX-3080 GPU. We evaluate the performance of the proposed method from the following aspects: (1) Sleep staging performance and generalization capability are evaluated on three representative datasets. (2) The performance and computational power consumption of quantized models using QAT under different quantization precisions are compared. (3) The proposed method is compared with the current state-of-the-art approaches to validate its relative advantages. (4) Ablation experiments are carried out to assess the individual and combined contributions of each component in the proposed framework. (5) A detailed hyperparameter selection process is presented to show how key parameters are tuned to optimize complexity and performance trade-offs.

A. Sleep Staging Performance

Table II presents the sleep staging performance of unquantized PicoSleepNet on the Fpz-Cz channel from the Sleep-EDF-20 and Sleep-EDF-78 datasets, and the C3-A2 channel from the ISRUC-Sleep dataset. The evaluation metrics include per-class F1 scores, overall ACC, and MF1.

Notably, results show the highest performance in the Wake stage with F1 scores from 89.2% to 90.7% across all datasets and relatively high performance in the N2 and N3 stages (74.0-88.4%). The REM stage achieves F1 scores ranging from 66.5% to 75.9%. However, the F1 scores for the N1 stage are relatively low (30.8-55.5%). This performance distribution pattern aligns with commonly observed results in existing studies, where the N1 stage is typically the most challenging to classify due to its transitional characteristics and limited samples. In terms of overall metrics, the overall ACCs for the Sleep-EDF-20, Sleep-EDF-78, and ISRUC-Sleep datasets reach 83.3%, 78.1%, and 79.0%, respectively, with the corresponding MF1 scores of 74.9%, 68.8%, and 76.7%.

B. Quantization Performance And Power Savings

To evaluate computational power consumption across different quantization precisions, we implement fixed-point arithmetic units with various bitwidths and single-precision floating-point arithmetic units based on the IEEE 754 standard for adders and multipliers using Verilog HDL. The logic synthesis is performed with 100 MHz clock frequency in UMC 22nm Low Power technology using Synopsys Design Compiler, under the corner of 1.0V supply voltage and 25°C operating temperature. Table III presents the computational power consumption for arithmetic units with different bitwidths, where FP32 represents 32-bit floating-point operations and INTn represents n-bit fixed-point operations. The results show that the power consumption of fixed-point operations is significantly lower than that of floating-point operations. Moreover, the power consumption of both fixed-point adders and multipliers decreases significantly with decreasing bitwidths.

TABLE III
SIMULATION TEST RESULTS OF POWER CONSUMPTION OF COMPUTING UNITS WITH DIFFERENT BITWIDTHS

Bitwidth	Adder Power (μW)	Multiplier Power (μW)
FP32	29.5	126
INT32	2.79	126
INT16	1.38	31.8
INT12	1.03	16.7
INT8	0.665	6.76
INT6	0.483	3.51
INT4	0.311	1.04
INT3	0.226	0.435

[1] The logic synthesis is performed with 100 MHz clock frequency in UMC 22nm Low Power technology using Synopsys Design Compiler, under the corner of 1.0V supply voltage and 25°C operating temperature.

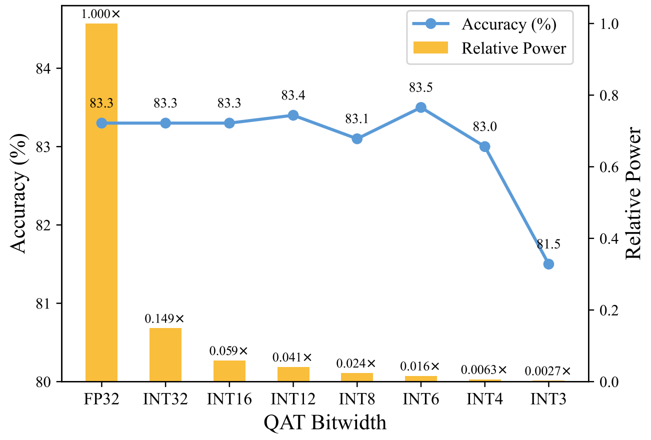

Fig. 5. Model performance and power consumption using QAT at different quantization precisions on the Sleep-EDF-20 dataset. Accuracy is shown as a blue line (left axis), and power consumption is shown as yellow bars on a logarithmic scale (right axis). Note: The relative power indicates the proportion of computational power consumption for each model relative to the FP32 model.

To determine the optimal quantization precision, we have analyzed model performance and computational power consumption across different bitwidths on three datasets. Fig.5 shows the results for the Sleep-EDF-20 dataset using QAT. It should be noted that the relative power consumption performance is calculated based on the simulation results in Table III combined with the number of addition and multiplication operations of the model. Relative power represents the relative proportion of the computational power consumption under the quantization condition to the FP32 model. The results show that the model quantized by the QAT method can maintain almost the same accuracy at low-bitwidth (e.g., INT6 and INT4) quantization precision, while its relative power consumption can be reduced by more than 62.5× compared to the FP32 model. In addition, during the quantization process, especially for INT4 and INT3, the number of addition operations decreased and the number of multiplication operations increased. This is because some existing synaptic weights are quantized to 0, which reduces the number of neural firings and increases the number of membrane potential decays. Considering both ACC loss and power optimization, INT6 is selected as the quantization precision of the final model on the Sleep-EDF-20 dataset. Consequently, INT6 QAT quantization achieves a 62.5× reduction in power consumption while increasing accuracy by 0.2%. The increase in accuracy is attributed to parameter fine-tuning during QAT, enabling the model to better adapt to quantization errors and support lower bitwidths. The experiments are repeated on two other datasets. For the Sleep-EDF-78 dataset, the INT6 model is selected as the final model. The accuracy of this model is 77.9%, which is 0.2% lower than the pre-quantization model, and its relative power reduces by 62.5×. On the ISRUC-Sleep dataset, the INT4 model is selected, which achieves an accuracy of 79.4%. Its accuracy is improved by 0.4% and its relative power reduces by 185× of the FP32 model. Furthermore, quantization causes some synaptic weights to be quantized to 0, reducing the number of synapses. The quantized model requires fewer bits to represent each parameter, so the size of the INT6 and INT4 models can be reduced by over 5.3× and 8×, respectively.

C. Comparison With State-of-the-Art Approaches

We compare PicoSleepNet with several recent lightweight staging approaches, particularly those optimized for wearable and mobile health applications. Brief descriptions for each compared approach are as follows:

• **DeepSleepNet** [12] exploits a custom CNN architecture to extract single-channel EEG time-frequency features and uses BiLSTM for sequence modeling. The model has a large number of parameters, about 25M.

• **U-time** [13] is a fully convolutional encoder-decoder network based on the U-Net architecture. The model contains about 1.2M trainable parameters.

• **MicroSleepNet** [15] uses one-dimensional group convolution, an efficient channel and spatial attention module, and dilated convolution to achieve effective feature extraction and fusion, significantly reducing the number of model parameters to 48.2K and computational operations to 49M.

• **LightSleepNet** [16] adopts a lightweight 1D CNN structure with approximately 43.1K parameters and 45.8M operations.

• **CNN Transformer** [40] reduces data dimensions and extracts features through four convolutional filter layers, and combines transformer to learn temporal change features for low-power sleep staging. The model has approximately 300K parameters.

• **SleepNet-Lite** [41] utilizes a multi-scale convolutional block and an inverted residual block to achieve a lightweight model (41.7K parameters) for sleep staging.

• **EfficientSleepNet** [42] is a lightweight sleep stage classification model that integrates depthwise separable convolutions, grouped convolutions, channel reordering, and a novel channel attention mechanism. The model has 83.8K parameters.

• **SleepSatelightFTC** [43] is a lightweight sleep stage classification model that uses self-attention mechanisms on both time-domain and frequency-domain inputs and has 470K parameters.

• **ESSN** [44] is an ingeniously structured sleep sequence network with 270K parameters and 350M floating-point operations.

• **LEEGNet** [45] introduces knowledge distillation to achieve a lightweight model with 970K parameters for sleep staging.

• **SleepyCo** [46] is a novel automatic sleep scoring framework that combines a feature pyramid and supervised contrastive learning. The model has 2.4M parameters.

• **HSNN** [25] employs a structural design that is more in line with the characteristics of biological neural networks and uses 6-channel EEG signals for sleep staging.

Table IV shows the performance and complexity of PicoSleepNet and traditional ANN models on the Sleep-EDF-20 and Sleep-EDF-78 datasets. PicoSleepNet achieves 83.5% accuracy and 75.2% MF1 on the Sleep-EDF-20 dataset, and 77.9% accuracy and 68.1% MF1 on the Sleep-EDF-78 dataset. In terms of model complexity, the number of parameters of PicoSleepNet is only about half of the most lightweight model and reduces computational operations by 27×. These results reveal that PicoSleepNet has achieved a significant breakthrough in model complexity while maintaining comparable performance to traditional methods. Specifically, in the proposed model, each synapse needs to store a weight parameter, each neuron needs to store a bias parameter, while the membrane potential decay coefficient and threshold are shared by all neurons. Therefore, the total number of parameters in PicoSleepNet is only 22.2-25.8K. In contrast, existing most lightweight models use the CNN architecture with shared kernel parameters, resulting in 41.7-83.8K parameters. Although sharing variables significantly reduces the number of parameters, the computational complexity of these models is still very high. In these two datasets, PicoSleepNet requires only about 681.4-730.2K operations, while the others require at least 18.4M operations. We also calculate the power consumption of these models according to Table III. The relative power indicates the proportion of the computational power consumption of this method relative to our FP32 model. Since other models did not report the number of additions and multiplications, all of the operations are considered additions to estimate the minimum computational power. Table IV shows that unquantized PicoSleepNet has more than 23× of the power consumption advantage over these models. Considering that PicoSleepNet can further reduce power consumption by 62.5× via QAT, the final INT6 PicoSleepNet model can achieve a 1480× computational power reduction.

TABLE IV
COMPARISON AMONG PROPOSED METHOD AND STATE-OF-THE-ART MODELS ON SLEEP-EDF-20 AND SLEEP-EDF-78 DATABASES

Database	Method	Architecture	Channels	ACC	MF1	Parameters	Operations	Relative Power
Sleep-EDF-20	DeepSleepNet† [12]	CNN+LSTM	Fpz-Cz	81.9	76.6	25.0M	-	-
	MicroSleepNet† [15]	CNN	Fpz-Cz	82.8	75.3	48.2K	49.0M	>63.424
	U-Time† [13]	U-Net	Fpz-Cz	-	79.0	1.2M	-	-
	LightSleepNet† [16]	CNN	Fpz-Cz	83.8	75.3	43.1K	45.8M	>59.282
	SleepNet-Lite† [41]	CNN	Fpz-Cz	85.2	79.5	41.7K	18.4M	>23.816
	EfficientSleepNet [42]	CNN	Fpz-Cz	83.3	77.5	83.8K	-	-
	SleepSatelightFTC† [43]	CNN	Fpz-Cz	85.7	77.7	470K	-	-
	ESSN† [44]	CNN+LSTM	Fpz-Cz	87.1	81.4	270K	350M	>287.577
	RSNN+BPSR	RSNN	Fpz-Cz	82.0	73.8	28.8K	1035.6K	1.383
	PicoSleepNet (FP32)†	RSNN	Fpz-Cz	**81.4**	**72.6**	**26.6K**	**795.2K**	**1.075**
	PicoSleepNet (FP32)	RSNN	Fpz-Cz	**83.3**	**74.9**	**24.6K**	**737.0K**	**1.000**
	PicoSleepNet (INT6)	RSNN	Fpz-Cz	**83.5**	**75.2**	**22.2K (INT6)**	**681.4K (INT6)**	**0.016**
Sleep-EDF-78	DeepSleepNet† [12]	CNN+LSTM	Fpz-Cz	77.8	71.8	25.0M	-	-
	MicroSleepNet† [15]	CNN	Fpz-Cz	79.5	71.8	48.2K	49.0M	>60.495
	U-Time† [13]	U-Net	Fpz-Cz	-	76.0	1.2M	-	-
	CNN Transformer [40]	CNN	Fpz-Cz	77.5	-	300.0K	-	-
	LEEGNet [45]	CNN	Fpz-Cz	82.4	-	970K	-	-
	EfficientSleepNet [42]	CNN	Fpz-Cz	80.5	75.0	83.8K	-	-
	SleepSatelightFTC† [43]	CNN	Fpz-Cz	84.8	77.8	460K	-	-
	SleepyCo† [46]	CNN	Fpz-Cz	84.6	79.0	2.4M	-	-
	RSNN+BPSR	RSNN	Fpz-Cz	77.1	66.6	29.8K	1212.1K	1.531
	PicoSleepNet (FP32)†	RSNN	Fpz-Cz	**76.2**	**66.3**	**27.3K**	**770.2K**	**0.997**
	PicoSleepNet (FP32)	RSNN	Fpz-Cz	**78.1**	**68.8**	**27.9K**	**772.3K**	**1.000**
	PicoSleepNet (INT6)	RSNN	Fpz-Cz	**77.9**	**68.1**	**25.8K (INT6)**	**730.2K (INT6)**	**0.016**

[1] The relative power indicates the proportion of computational power consumption for each method relative to our FP32 (No-quant) model. A lower value indicates higher power efficiency. † indicates results obtained using subject-wise k-fold cross-validation.

TABLE V
COMPARISON AMONG PROPOSED METHOD AND STATE-OF-THE-ART MODELS ON ISRUC-SLEEP DATABASE

Database	Method	Architecture	Encoding	Channels	ACC	MF1	Neurons	Synapses	Parameters	Operations	Relative Power
ISRUC-Sleep	HSNN [25]	HSNN	BSA	6-leads EEG	76.0	75.0	4242	>574250	>578.0K	-	-
	RSNN+BPSR	RSNN	LCS	C3-A2	76.7	72.9	285	29992	30.3K	2614.5K	1.536
	PicoSleepNet (FP32)†	RSNN	LCS	**C3-A2**	**77.6**	**75.2**	**285**	**29896**	**30.2K**	**1736.5K**	**1.043**
	PicoSleepNet (FP32)	RSNN	LCS	C3-A2	**79.0**	**76.7**	**285**	**28146**	**28.4K**	**1662.7K**	**1.000**
	PicoSleepNet (INT4)	RSNN	LCS	C3-A2	**79.4**	**77.2**	**285**	**13701**	**14.0K (INT4)**	**842.0K (INT4)**	**0.0054**

[1] The relative power indicates the proportion of computational power consumption for each method relative to our FP32 (No-quant) model. A lower value indicates higher power efficiency. † indicates results obtained using subject-wise k-fold cross-validation.

Table V shows the comparison of PicoSleepNet with the previous SNN method. PicoSleepNet achieves 79.4% accuracy and 77.2% MF1 score on the ISRUC-Sleep dataset. This performance is 3.4% higher in accuracy and 2.2% higher in macro-F1 than HSNN, while we only need a single-channel EEG instead of 6 channels in HSNN. PicoSleepNet uses Masked-BPSR to minimize the complexity of the model, requiring only 285 neurons and 13,701 synapses, which is an order of magnitude less than HSNN. Notably, the connections between different layers in HSNN are fully connected, and the connectivity within the reservoir layer is not reported, so the number of synapses and parameters in Table V only count the synapses between different layers. Although HSNN did not report the number of operations, it requires 2000 steps per epoch, while ours only takes 75 steps and has far fewer synapses. Therefore, we can infer that the computational complexity of our model is significantly lower. In addition, PicoSleepNet adopts LCS in EEG encoding. The algorithm complexity of the LCS encoding method is only O(N), which greatly reduces the preprocessing overhead compared to the O(NM) of BSA encoding. LCS signals can also be directly collected by neuromorphic sensors with LC-ADCs, which can greatly reduce the power consumption of signal acquisition and avoid the data encoding process. Futhermore, low-bitwidth PicoSleepNet is more suitable for deployment on resource-constrained wearable devices. These advantages make PicoSleepNet easier to integrate with low-power sensors and neuromorphic processors, potentially enabling ultra-low-power end-to-end sleep staging systems.

We also replace our training method with BPSR for comparison. The results show that Masked-BPSR can achieve higher accuracy and lower model complexity in all three datasets, indicating that synaptic-mask sparsity can achieve better pruning optimization performance. In addition, under subject-wise k-fold cross-validation, the overall accuracy and MF1 scores are computed by aggregating predictions across all folds. Due to the dynamic spiking activity and the progressive synaptic pruning during training, the number of model parameters,

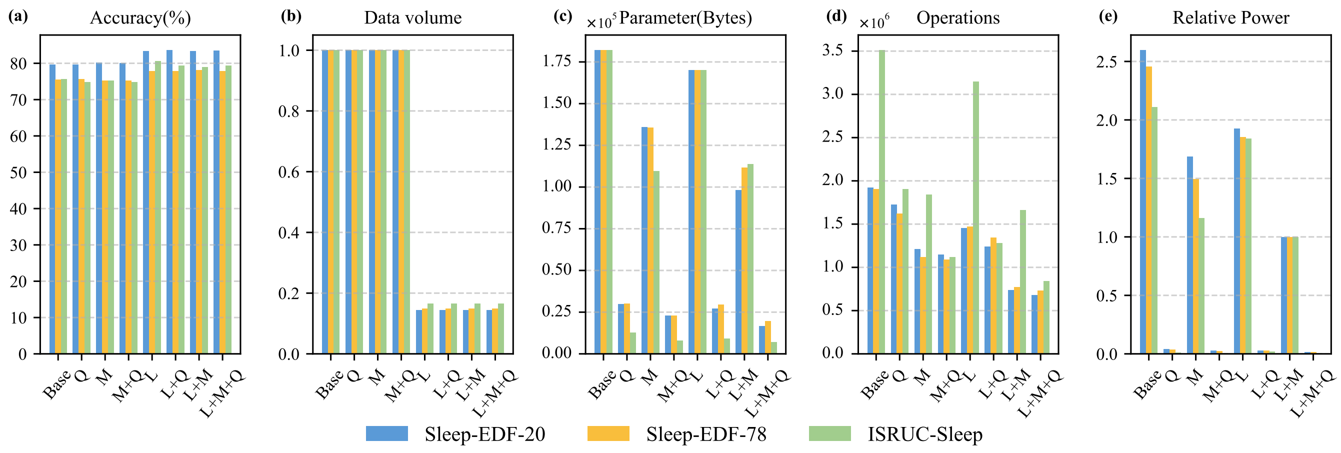

Fig. 6. A full-factorial ablation study evaluating the independent and combined effects of three lightweight techniques, LCS encoding (L), Masked-BPSR (M), and QAT (Q), on three benchmark datasets (Sleep-EDF-20, Sleep-EDF-78, and ISRUC-Sleep). The eight configurations include all possible combinations of these techniques, with "Base" denoting the baseline model without any lightweight technique. Performance metrics include classification accuracy, data volume, size of parameters, number of operations, and relative power consumption.

computational operations, and relative power consumption vary across folds. Therefore, we report their average values in table IV and table V.

D. Ablation Study of Model Components

To systematically evaluate the independent contributions and interactive effects of the three lightweight techniques, LCS encoding, Masked-BPSR, and QAT, we conducted a full-factorial ablation study on three datasets (Sleep-EDF-20, Sleep-EDF-78, and ISRUC-Sleep). As shown in the Fig.6, the study includes eight combinations of lightweight strategies. The figure summarizes the performance of each configuration in terms of accuracy, parameters, operations, relative power consumption, and data volume.

We use BSA to serve as a comparison baseline for evaluating model performance in the absence of LCS. BSA is a commonly used sparse encoding approach in SNNs, which has been widely applied in EEG-based spike encoding [47]–[49]. However, as a post-processing method on Nyquist-sampled signals, BSA requires complete signal acquisition and storage. In contrast, LCS is an event-driven, sub-Nyquist sampling method that generates single-bit spikes directly during the signal acquisition process, unifying sampling and encoding. LCS naturally produces outputs in a format inherently compatible with SNNs, eliminating the need for additional spike encoding modules. By triggering non-uniform sampling in the time domain based on signal dynamics, LCS aligns well with the event-driven computation paradigm of SNNs, effectively reducing redundant input and unnecessary computational overhead. Notably, we use the optimal model architecture for each encoding method. For BSA encoding, the best performance is achieved when using 20, 180, 50, and 5 neurons in the input layer, recurrent layer, hidden layer, and output layer, respectively. Subsequently, Masked-BPSR and QAT are performed on these corresponding baseline architectures to ensure fair and reproducible comparisons.

Fig.6(b) demonstrates that LCS achieves up to 6.98× data volume reduction compared to BSA, along with lower spike encoding power consumption, making it especially suitable for deployment on resource-constrained, low-power edge devices. Fig.6(c) and Fig.6(d) illustrate the effectiveness of Masked-BPSR in reducing model complexity. As shown in Fig.6(c), the parameter count is reduced by 42.2%, 34.3%, and 33.2% on the Sleep-EDF-20, Sleep-EDF-78, and ISRUC-Sleep datasets, respectively. In addition, Masked-BPSR also shows a strong compression effect under BSA coding, demonstrating its generality and effectiveness. Under joint compression of QAT and Masked-BPSR, the model sizes are reduced by 89.1%, 87.6%, and 91.7%. Fig.6(d) shows that Masked-BPSR significantly reduces the number of operations while maintaining model performance. For the SNN model, the number of operations depends mainly on the accumulation of membrane potential caused by the propagation of spikes in synapses and the spontaneous decay of the membrane potential. For each synaptic operation, only one addition is performed between the membrane potential and the synaptic weight, since the input spike $m^t \in \{0, 1\}$ (Eq. 1). The spontaneous decay of the membrane potential requires one multiplication operation. Regardless of the encoding method, Masked-BPSR significantly reduces the number of parameters and operations. Specifically, under LCS encoding, Masked-BPSR reduces the number of operations by 49.4%, 47.4%, and 47.1% on the Sleep-EDF-20, Sleep-EDF-78, and ISRUC-Sleep datasets, respectively. Notably, the sampling rate of the ISRUC-Sleep dataset is 200 Hz, which is twice that of the two EDF datasets. As a result, the input spikes of the ISRUC-Sleep data after LCS or BSA cause a doubling of time steps in inference, thus showing almost double the number of operations. These results demonstrate that the Masked-BPSR can significantly reduce model complexity and computational costs while preserving performance, which is crucial for improving model efficiency and reducing energy costs. Furthermore, Fig.6(e) illustrates that QAT compresses the floating-point model into a low-bit representation, enabling

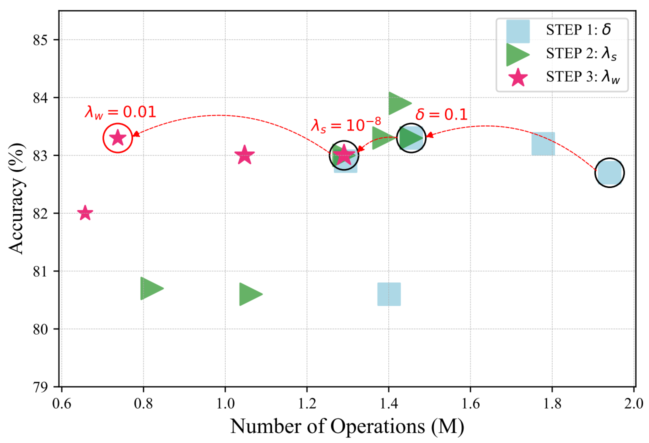

Fig. 7. Three-step hyperparameter tuning for sparsity optimization on the Sleep-EDF-20 dataset. Each point shows a tested configuration with its accuracy and normalized operations. Blue squares, green triangles, and magenta stars indicate tuning of δ, λ_s, and λ_w, respectively. The marker size indicates the amount of the parameter. Circled markers denote the selected optimal values at each step.

at least 62.5× reduction in relative power consumption without compromising performance.

In summary, LCS is primarily used to compress input-side data volume and reduce spike encoding power consumption, Masked-BPSR is designed to reduce model complexity, and QAT reduces power consumption by converting the model into a low-bit representation. The synergistic integration of the three techniques leads to effective lightweight and low-power optimization, ensuring end-to-end efficiency from signal acquisition to model inference. The effectiveness of this integrated strategy is further validated through full-factorial ablation experiments conducted on three datasets, demonstrating that PicoSleepNet is particularly well-suited for deployment in resource-constrained environments such as wearable devices and neuromorphic systems.

E. Hyperparameter Selection Strategy

To optimize the trade-off between model accuracy and computational cost, we carefully tuned three key hyperparameters in our proposed framework: the threshold for LCS (δ), the spike sparsity coefficient λ_s, and the synaptic-mask sparsity coefficient λ_w. Each of these parameters plays a distinct role in promoting sparsity at steps of signal encoding and network computation. We adopted a three-step tuning procedure to progressively reduce model complexity while maintaining high classification performance. The entire tuning trajectory is visualized in Fig.7, where different marker shapes and colors correspond to the three hyperparameters being tuned, clearly illustrating the trade-offs and cumulative benefits of sparsity optimization.

1) LCS Threshold δ: The LCS module converts raw EEG signals into sparse spike sequences by emitting events whenever the signal crosses a predefined threshold δ. To evaluate the effect of δ on input sparsity and model performance, we varied its value from 0.06 to 0.14, while keeping λ_s and λ_w fixed at zero to isolate the effect of thresholding. As shown in Fig.7 (blue squares), $\delta = 0.1$ achieves the optimal balance: it yields 83.3% accuracy while reducing the input data volume by 6.98× and maintaining the number of operations at a manageable level (1.45M).

2) Spike Sparsity Coefficient λ_s: After fixing the LCS threshold, we introduce a spike sparsity constraint via λ_s to regulate the average firing rate of neurons across time. We conduct a grid search in the range from 0 to 10^{-6} and found that $\lambda_s = 10^{-8}$ (represented by green triangles in Fig.7) provides the best trade-off, reducing the number of operations by approximately 11% without causing accuracy degradation.

3) Synaptic-Mask Sparsity Coefficient λ_w: To further compress the network, we apply synaptic-mask sparsity controlled by λ_w. With δ and λ_s fixed, we vary λ_w from 0 to 0.05. As indicated by magenta stars in Fig.7, the optimal setting of $\lambda_w = 0.01$ results in a 42% reduction in parameter count and a 47% reduction in operations (down to 737K), while maintaining 83.3% accuracy.

This three-step tuning strategy enable us to derive a lightweight and energy-efficient model with only 24.6K parameters and 737K operations on the Sleep-EDF-20 dataset. The same three-step tuning strategy is applied to the Sleep-EDF-78 and ISRUC-Sleep datasets to identify dataset-specific hyperparameters. The selected values are $\delta = 0.13$, $\lambda_s = 10^{-8}$, $\lambda_w = 0.005$ for Sleep-EDF-78, and $\delta = 0.08$, $\lambda_s = 10^{-7}$, $\lambda_w = 0.01$ for ISRUC-Sleep. In addition, we conduct a sensitivity analysis to assess the robustness of model performance to small variations in each hyperparameter. The results show that model accuracy remains within approximately ±1% of the selected configuration when individual hyperparameters are slightly perturbed.

Finally, although this study focuses on algorithmic design and evaluation using publicly available datasets, real-world validation remains essential for assessing the practical feasibility of the proposed approach. In future work, we intend to integrate the proposed method with dedicated low-power neuromorphic hardware to develop a wearable sleep monitoring system. This will facilitate a comprehensive end-to-end evaluation of the performance and practical applicability of the proposed approach in real-world sleep monitoring scenarios. Such deployment will enable testing under realistic wearable conditions while accounting for motion artifacts, electrode variability, signal collection losses, and environmental noise. These factors are rarely captured in standard PSG benchmark datasets. Moreover, we will extend validation to diverse user groups to better evaluate the generalization ability and robustness of the model in practical settings.

V. Conclusion

This study presents PicoSleepNet, an ultra-lightweight SNN-based sleep stage classification method using single-channel EEG signals. By integrating low-power sampling, sparse SNN model, and low-bitwidth quantization, PicoSleepNet significantly reduces computational complexity and power consumption. In the signal encoding phase, the event-driven LCS realizes highly compressed EEG encoding over traditional Nyquist sampling. The model design employs a biologically inspired RSNN structure, reducing computational complexity via Masked-BPSR. In model deployment, the QAT method significantly reduces hardware overhead while maintaining accuracy. Experimental results demonstrate that PicoSleepNet achieves accuracies of 83.5%, 77.9%, 79.4% and macro-F1 scores of 75.2%, 68.1%, and 77.2% on the Sleep-EDF-20, Sleep-EDF-78, and ISRUC-Sleep datasets, respectively. Thanks to the sparse optimization of Masked-BPSR, the model requires only 14.0-25.8K parameters and 681.4-842.0K operations. In addition, QAT can further reduce the computational power consumption by 62.5× based on the sparse model with little loss in accuracy. Compared to previous methods, PicoSleepNet minimizes model complexity by reducing the number of parameters by nearly 2× and the number of operations by 27×, and achieves a significant reduction in computational power consumption by 1480× without sacrificing accuracy. PicoSleepNet provides a scalable, energy-efficient solution for EEG-based sleep monitoring, paving the way for continuous, real-time health monitoring applications. Future work will focus on integrating the framework with low-power sensors and neuromorphic processors to enhance end-to-end energy efficiency, with potential applications in broader health-related signal processing. Additionally, we plan to improve model accuracy while preserving energy efficiency by exploring more advanced architectural designs and incorporating strategies to address the N1 class imbalance, such as data augmentation, class re-weighting, and minority class oversampling [50]–[53].

References

[1] P. Maquet, "The role of sleep in learning and memory," *Science*, vol. 294, no. 5544, pp. 1048–1052, 2001.

[2] E. A. Wolpert, "A manual of standardized terminology, techniques and scoring system for sleep stages of human subjects." *Archives of General Psychiatry*, vol. 20, no. 2, pp. 246–247, 1969.

[3] S. A. Keenan, "An overview of polysomnography," *Handbook of clinical neurophysiology*, vol. 6, pp. 33–50, 2005.

[4] R. B. Berry, R. Brooks, C. Gamaldo, S. M. Harding, R. M. Lloyd, S. F. Quan, M. T. Troester, and B. V. Vaughn, "Aasm scoring manual updates for 2017 (version 2.4)," pp. 665–666, 2017.

[5] Y. Wei, Y. Zhu, Y. Zhou, X. Yu, and Y. Luo, "Automatic sleep staging based on contextual scalograms and attention convolution neural network using single-channel eeg," *IEEE Journal of Biomedical and Health Informatics*, 2023.

[6] K. Fei, J. Wang, L. Pan, X. Wang, and B. Chen, "A sleep staging model on wavelet-based adaptive spectrogram reconstruction and light weight cnn," *Computers in Biology and Medicine*, vol. 173, p. 108300, 2024.

[7] R. Li, B. Wang, T. Zhang, and T. Sugi, "A developed lstm-ladder-network-based model for sleep stage classification," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 31, pp. 1418–1428, 2023.

[8] H. Phan, O. Y. Chén, M. C. Tran, P. Koch, A. Mertins, and M. De Vos, "Xsleepnet: Multi-view sequential model for automatic sleep staging," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 5903–5915, 2021.

[9] Z. Jia, Y. Lin, J. Wang, X. Wang, P. Xie, and Y. Zhang, "Salientsleepnet: Multimodal salient wave detection network for sleep staging," *arXiv preprint arXiv:2105.13864*, 2021.

[10] A. H. Abdulnabi, G. Wang, J. Lu, and K. Jia, "Multi-task cnn model for attribute prediction," *IEEE Transactions on Multimedia*, vol. 17, no. 11, pp. 1949–1959, 2015.

[11] H. Phan, F. Andreotti, N. Cooray, O. Y. Chén, and M. De Vos, "Seqsleepnet: end-to-end hierarchical recurrent neural network for sequence-to-sequence automatic sleep staging," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 3, pp. 400–410, 2019.

[12] A. Supratak, H. Dong, C. Wu, and Y. Guo, "Deepsleepnet: A model for automatic sleep stage scoring based on raw single-channel eeg," *IEEE transactions on neural systems and rehabilitation engineering*, vol. 25, no. 11, pp. 1998–2008, 2017.

[13] M. Perslev, M. Jensen, S. Darkner, P. J. Jennum, and C. Igel, "U-time: A fully convolutional network for time series segmentation applied to sleep staging," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[14] A. Supratak and Y. Guo, "Tinysleepnet: An efficient deep learning model for sleep stage scoring based on raw single-channel eeg," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. IEEE, 2020, pp. 641–644.

[15] G. Liu, G. Wei, S. Sun, D. Mao, J. Zhang, D. Zhao, X. Tian, X. Wang, and N. Chen, "Micro sleepnet: efficient deep learning model for mobile terminal real-time sleep staging," *Frontiers in Neuroscience*, vol. 17, p. 1218072, 2023.

[16] Y. Liao, C. Zhang, M. Zhang, Z. Wang, and X. Xie, "Lightsleepnet: Design of a personalized portable sleep staging system based on single-channel eeg," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 1, pp. 224–228, 2021.

[17] G. R. Pedrollo, A. R. Franco, L. B. Bagesteiro, and A. Balbinot, "Spiking neural networks diagnosis of adhd subtypes through eeg signals evaluation," in *2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. IEEE, 2022, pp. 3166–3169.

[18] M. Sharifshazileh, K. Burelo, T. Fedele, J. Sarnthein, and G. Indiveri, "A neuromorphic device for detecting high-frequency oscillations in human ieeg," in *2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*. IEEE, 2019, pp. 69–72.

[19] L.-Y. Niu, Y. Wei, W.-B. Liu, J.-Y. Long, and T.-h. Xue, "Research progress of spiking neural network in image classification: a review," *Applied intelligence*, vol. 53, no. 16, pp. 19 466–19 490, 2023.

[20] Z. Yan, J. Zhou, and W.-F. Wong, "Energy efficient ecg classification with spiking neural network," *Biomedical Signal Processing and Control*, vol. 63, p. 102170, 2021.

[21] R. E. Turkson, H. Qu, C. B. Mawuli, and M. J. Eghan, "Classification of alzheimer's disease using deep convolutional spiking neural network," *Neural Processing Letters*, vol. 53, no. 4, pp. 2649–2663, 2021.

[22] S. Yang and B. Chen, "Effective surrogate gradient learning with high-order information bottleneck for spike-based machine intelligence," *IEEE transactions on neural networks and learning systems*, 2023.

[23] J. Li, B. Hu, Z.-H. Guan, and B. Zhang, "Sleep stage classification with spiking neural networks and transformers using multi-channel eeg data," in *2024 43rd Chinese Control Conference (CCC)*. IEEE, 2024, pp. 8345–8350.

[24] H. Jia, Z. Yang, P. Gao, M. Wu, C. Li, Y. Kan, and R. Zhang, "Automatic sleep staging via frequency-wise spiking neural networks," in *2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2022, pp. 1028–1033.

[25] Z. Jia, J. Ji, X. Zhou, and Y. Zhou, "Hybrid spiking neural network for sleep electroencephalogram signals," *Science China Information Sciences*, vol. 65, no. 4, p. 140403, 2022.

[26] J. Van Assche and G. Gielen, "Power efficiency comparison of event-driven and fixed-rate signal conversion and compression for biomedical applications," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 14, no. 4, pp. 746–756, 2020.

[27] Z. Wang, L. Ye, H. Zhang, J. Ru, H. Fan, Y. Wang, and R. Huang, "20.2 a 57nw software-defined always-on wake-up chip for iot devices with asynchronous pipelined event-driven architecture and time-shielding level-crossing adc," in *2020 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE, 2020, pp. 314–316.

[28] S. F. Hussain and S. M. Qaisar, "Epileptic seizure classification using level-crossing eeg sampling and ensemble of sub-problems classifier," *Expert Systems with Applications*, vol. 191, p. 116356, 2022.

[29] Y.-H. Liu and X.-J. Wang, "Spike-frequency adaptation of a generalized leaky integrate-and-fire model neuron," *Journal of computational neuroscience*, vol. 10, pp. 25–45, 2001.

[30] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, "Spatio-temporal backpropagation for training high-performance spiking neural networks," *Frontiers in neuroscience*, vol. 12, p. 331, 2018.

[31] S. Schliebs, H. N. A. Hamed, and N. Kasabov, "Reservoir-based evolving spiking neural network for spatio-temporal pattern recognition," in *Neural Information Processing: 18th International Conference, ICONIP 2011, Shanghai, China, November 13-17, 2011, Proceedings, Part II 18*. Springer, 2011, pp. 160–168.

[32] Y. Yan, H. Chu, Y. Jin, Y. Huan, Z. Zou, and L. Zheng, "Backpropagation with sparsity regularization for spiking neural network learning," *Frontiers in Neuroscience*, vol. 16, p. 760298, 2022.

[33] T. M. Bartol Jr, C. Bromer, J. Kinney, M. A. Chirillo, J. N. Bourne, K. M. Harris, and T. J. Sejnowski, "Nanoconnectomic upper bound on the variability of synaptic plasticity," *Elife*, vol. 4, p. e10778, 2015.

[34] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *arXiv preprint arXiv:1308.3432*, 2013.

[35] B. Kemp, A. H. Zwinderman, B. Tuk, H. A. Kamphuisen, and J. J. Oberye, "Analysis of a sleep-dependent neuronal feedback loop: the slow-wave microcontinuity of the eeg," *IEEE Transactions on Biomedical Engineering*, vol. 47, no. 9, pp. 1185–1194, 2000.

[36] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals," *circulation*, vol. 101, no. 23, pp. e215–e220, 2000.

[37] O. Tsinalis, P. M. Matthews, Y. Guo, and S. Zafeiriou, "Automatic sleep stage scoring with single-channel eeg using convolutional neural networks," *arXiv preprint arXiv:1610.01683*, 2016.

[38] H. Phan, F. Andreotti, N. Cooray, O. Y. Chén, and M. De Vos, "Dnn filter bank improves 1-max pooling cnn for single-channel eeg automatic sleep stage classification," in *2018 40th annual international conference of the IEEE engineering in medicine and biology society (EMBC)*. IEEE, 2018, pp. 453–456.

[39] S. Khalighi, T. Sousa, J. M. Santos, and U. Nunes, "Isruc-sleep: A comprehensive public dataset for sleep researchers," *Computer methods and programs in biomedicine*, vol. 124, pp. 180–192, 2016.

[40] Z. Yao and X. Liu, "A cnn-transformer deep learning model for real-time sleep stage classification in an energy-constrained wireless device," in *2023 11th International IEEE/EMBS Conference on Neural Engineering (NER)*. IEEE, 2023, pp. 1–4.

[41] H. Zhou, A. Liu, H. Cui, Y. Bie, and X. Chen, "Sleepnet-lite: a novel lightweight convolutional neural network for single-channel eeg-based sleep staging," *IEEE Sensors Letters*, vol. 7, no. 2, pp. 1–4, 2023.

[42] F. Wang, Z. Zheng, B. Hu, X. Yang, M. Tang, and H. Huang, "Efficientsleepnet: A novel lightweight end-to-end model for automated sleep staging on single-channel eeg," in *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2025, pp. 1–5.

[43] A. Ito and T. Tanaka, "Sleepsatelightftc: A lightweight and interpretable deep learning model for single-channel eeg-based sleep stage classification," *bioRxiv*, pp. 2024–08, 2024.

[44] Y. Chen, Y. Lv, X. Sun, M. Poluektov, Y. Zhang, and T. Penzel, "Essn: An efficient sleep sequence network for automatic sleep staging," *IEEE Journal of Biomedical and Health Informatics*, 2024.

[45] B. Wang, Y. Li, C. Zhang, A. Stefanidis, M. Zhou, and J. Su, "Leegnet: Lightweight eeg sleep stage classification network with knowledge distillation," in *2024 IEEE 5th International Conference on Pattern Recognition and Machine Learning (PRML)*. IEEE, 2024, pp. 277–283.

[46] S. Lee, Y. Yu, S. Back, H. Seo, and K. Lee, "Sleepyco: Automatic sleep scoring with feature pyramid and contrastive learning," *Expert Systems with Applications*, vol. 240, p. 122551, 2024.

[47] Y. Li, L. Fan, H. Shen, and D. Hu, "Hr-snn: An end-to-end spiking neural network for four-class classification motor imagery brain-computer interface," *IEEE Transactions on Cognitive and Developmental Systems*, 2024.

[48] C.-A. López-Herrera, H.-G. Acosta-Mesa, E. Mezura-Montes, and J.-A. Barradas-Palmeros, "Prediction of epileptic seizure using neuroevolved spiking neural network," in *Mexican International Conference on Artificial Intelligence*. Springer, 2024, pp. 135–146.

[49] N. Nuntalid, K. Dhoble, and N. Kasabov, "Eeg classification with bsa spike encoding algorithm and evolving probabilistic spiking neural network," in *Neural Information Processing: 18th International Conference, ICONIP 2011, Shanghai, China, November 13-17, 2011, Proceedings, Part I 18*. Springer, 2011, pp. 451–460.

[50] J. Fan, C. Sun, C. Chen, X. Jiang, X. Liu, X. Zhao, L. Meng, C. Dai, and W. Chen, "Eeg data augmentation: towards class imbalance problem in sleep staging tasks," *Journal of Neural Engineering*, vol. 17, no. 5, p. 056017, 2020.

[51] R. Jain and A. Ramakrishnan, "Modality-specific feature selection, data augmentation and temporal context for improved performance in sleep staging," *IEEE journal of biomedical and health informatics*, vol. 28, no. 2, pp. 1031–1042, 2023.

[52] H. Phan, F. Andreotti, N. Cooray, O. Y. Chén, and M. De Vos, "Joint classification and prediction cnn framework for automatic sleep stage classification," *IEEE Transactions on Biomedical Engineering*, vol. 66, no. 5, pp. 1285–1296, 2018.

[53] E. Eldele, Z. Chen, C. Liu, M. Wu, C.-K. Kwoh, X. Li, and C. Guan, "An attention-based deep learning approach for sleep stage classification with single-channel eeg," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 29, pp. 809–818, 2021.