Neural Architecture Search of Time-to-First-Spike-Coded Spiking Neural Networks for Efficient Eye-based Emotion Recognition

Qianhui Liu
School of Artificial Intelligence, Shandong University
Jinan, Shandong, China
qhliu@sdu.edu.cn

Jing Yang
School of Software, Shandong University
Shandong, Jinan, China

Miao Yu*
School of Computing, National University of Singapore
Singapore

Trevor E. Carlson
School of Computing, National University of Singapore
Singapore

Gang Pan
College of Computer Science and Technology, Zhejiang University
Hangzhou, Zhejiang, China

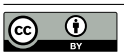
Haizhou Li
School of Artificial Intelligence, The Chinese University of Hong Kong (Shenzhen)
Shenzhen, Guangdong, China
haizhouli@cuhk.edu.cn

Zhumin Chen*
School of Artificial Intelligence, Shandong University
Jinan, Shandong, China

Abstract

Eye-based emotion recognition enables eyewear devices to perceive users' emotional states and support emotion-aware interaction. However, deploying such functionality on their resource-limited embedded hardware remains challenging. Time-to-first-spike (TTFS)-coded spiking neural networks (SNNs) offer a promising solution due to their extremely sparse and energy-efficient computation, where each neuron emits at most one binary spike. While prior works have primarily focused on improving TTFS SNN training algorithms, the role of network architecture has been largely overlooked. This is particularly critical, as spike timing in TTFS SNNs is tightly coupled with architectural design, and eye-based emotion recognition requires compact yet highly efficient networks. In this paper, we propose TNAS-ER, the first neural architecture search (NAS) framework tailored to TTFS SNNs for eye-based emotion recognition. TNAS-ER presents a novel ANN-assisted search strategy that leverages a ReLU-based ANN counterpart to guide architecture optimization and stabilize training of the TTFS SNN. TNAS-ER employs an evolutionary algorithm, with weighted and unweighted average recall jointly defined as fitness objectives for emotion recognition. Extensive experiments demonstrate that TNAS-ER achieves high recognition performance with significantly improved efficiency. Furthermore, we evaluate TNAS-ER on neuromorphic hardware, confirming its superior energy efficiency and strong potential for real-world applications.

*Corresponding Author.



This work is licensed under a Creative Commons Attribution 4.0 International License.
MM '26, Rio de Janeiro, Brazil
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2213-4/2026/11
https://doi.org/10.1145/3767308.3835971

CCS Concepts

• **Computing methodologies** → **Computer vision**; **Bio-inspired approaches**.

Keywords

Spiking Neural Networks, Neural Architecture Search, Event Camera, Eye-based Emotion Recognition

ACM Reference Format:
Qianhui Liu, Jing Yang, Miao Yu, Trevor E. Carlson, Gang Pan, Haizhou Li, and Zhumin Chen. 2026. Neural Architecture Search of Time-to-First-Spike-Coded Spiking Neural Networks for Efficient Eye-based Emotion Recognition. In *Proceedings of the 34th ACM International Conference on Multimedia (MM '26), November 10–14, 2026, Rio de Janeiro, Brazil.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3767308.3835971

1 Introduction

Eye-based emotion recognition has gained increasing attention for its non-invasive, privacy-preserving, and flexible nature compared to facial or EEG emotion recognition [29]. Recent approaches employ event cameras to avoid motion blur and support a wider dynamic range than conventional RGB cameras [8], enabling a precise capture of eye regions under challenging motion and lighting conditions. These advantages make eye-based emotion recognition particularly promising for wearable applications, such as immersive VR/AR interactions and emotion-aware mobile services [7, 11, 33].

Despite its potential, deploying eye-based emotion recognition models on embedded eyewear platforms remains challenging. As these devices are compact and battery-powered, their memory, computational and energy resources are often limited [33], necessitating recognition systems that are both accurate and highly efficient.

Time-to-first-spike-coded (TTFS) spiking neural networks (SNNs) offer a promising direction for efficient computing. SNNs are brain-inspired models that process information through discrete spike

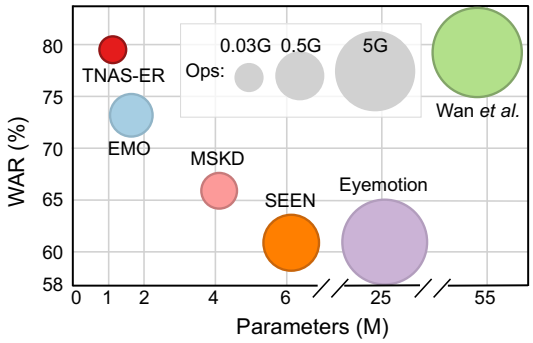


Figure 1: Comparison of TNAS-ER with state-of-the-art eye-based emotion recognition models on the SEE dataset, in terms of Weighted Average Recall (WAR), the number of parameters and operations. Circle size reflects the FLOPs (or SynOps for SNNs).

events, enabling event-driven and energy-efficient processing [19, 24]. In TTFS SNNs, each neuron emits at most one spike, resulting in extremely sparse activity and significantly reduced computation [14, 35]. This property makes TTFS SNNs well-suited for efficient inference on resource-constrained devices.

Existing research has primarily focused on training algorithms to improve the performance of TTFS SNNs [26, 31, 35, 40], while the role of network architecture in determining both accuracy and computational efficiency has been largely overlooked. However, architectural design is particularly critical. First, in TTFS SNNs, spike timing is tightly coupled with network architecture, making feature representations and overall performance sensitive to architecture choice. Second, practical wearable systems impose strict constraints on model size and energy consumption, requiring networks that are not only accurate but also highly compact and efficient. These considerations highlight the importance of exploring architecture design strategies tailored to TTFS SNNs for eye-based emotion recognition. Nevertheless, TTFS SNNs remain difficult to design, as their extremely sparse and timing-dependent spike activity leads to unstable gradients and challenging optimization.

This paper proposes TNAS-ER, a novel neural architecture search (NAS) framework tailored to TTFS SNNs for eye-based emotion recognition. TNAS-ER leverages a ReLU-based ANN (hereafter ReLU ANN) counterpart to stabilize training, guide architecture optimization, and enhance performance, effectively mitigating the difficulty of directly applying TTFS SNNs in NAS. By analyzing how ReLU ANN and TTFS SNN collaborate at each stage of the search, an ANN-assisted search strategy that bridges the two domains is introduced. TNAS-ER employs an evolutionary algorithm, with weighted and unweighted average recall jointly defined as fitness objectives for emotion recognition. Extensive experiments on two publicly available event-based single-eye emotion recognition datasets demonstrate that TNAS-ER achieves high recognition performance with significantly improved efficiency (see Figure 1). Furthermore, we deploy TNAS-ER on the neuromorphic hardware YOSO [6, 36], and the overall system achieves a low latency of 48 ms and energy consumption of 9.0 mJ, confirming its superior energy efficiency and strong potential for real-world applications.

The main contributions of this work are summarized as follows:

- We present the first exploration of TTFS SNNs for eye-based emotion recognition, achieving state-of-the-art accuracy with over 1.6×, 5.6× and 5.8 × reductions in parameters, operations and latency, respectively, compared to existing ANN and SNN baselines.
- We develop TNAS-ER, the first NAS framework specifically designed for TTFS SNNs, which automatically discovers efficient and high-performing architectures and outperforms manually designed and randomly searched architectures.
- We introduce an ANN-assisted search strategy that leverages the mapping between TTFS SNNs and ReLU ANNs to effectively stabilize training and guide architecture optimization.

2 Related Work

2.1 Eye-based Emotion Recognition

Eye-based emotion recognition has emerged as a promising alternative to facial or EEG-based approaches [11]. Compared with EEG-based methods, it offers a more convenient and comfortable user experience, and unlike facial analysis, it remains effective when the face is partially occluded or when privacy concerns restrict full-face data usage. Recent studies have shifted toward single-eye observations to overcome camera synchronization and bandwidth constraints in binocular setups [33]. To ensure reliable perception under challenging motion and lighting conditions, Zhang *et al.* [37] employed event cameras to capture the eye region and proposed an ANN-SNN hybrid network for joint frame–event processing. Han *et al.* [9] designed a hierarchical event–RGB fusion framework for multi-scale semantics integration. Although these multimodal methods achieve high accuracy, they require handling both frame and event streams, resulting in large and computationally intensive networks. To improve efficiency, Wan *et al.* [29] proposed a sparse transformer for event-based single-eye emotion recognition; however, the model still relies on a relatively large architecture. Wang *et al.* [30] distilled a large ANN–SNN teacher network into a unimodal SNN, reducing computation cost but depending on manually designed architectures that limit adaptability and efficiency.

2.2 Spiking Neural Networks with TTFS Coding

SNNs are brain-inspired models that transmit information through discrete spikes, mimicking biological communication [24]. Unlike ANNs, they operate in an event-driven manner, activating computations only when spikes occur. TTFS SNNs encode information in spike timing, allowing each neuron to fire at most once and thereby achieving extremely sparse and energy-efficient computation. Early works such as Rueckauer *et al.* [25] converted ANNs to TTFS SNNs but were limited to shallow networks. Zhang *et al.* [38] extended the conversion to deeper networks with reverse coding, and Park *et al.* [23] further improved it using dynamic thresholds and dendritic mechanisms. Yang *et al.* [35] later proposed a near-lossless conversion strategy addressing temporal distortions. Zhao *et al.* [40] proposed a conversion method for transformer architecture with TTFS-coding. In addition to conversion-based methods, Wei *et al.* [31] enabled direct TTFS training from pretrained ANNs. Stanojevic *et al.* [26] established an identity mapping between ReLU networks

and TTFS SNNs, enabling both conversion-based and direct training of TTFS models. Despite these advances, their architectural design remains unexplored. Given the crucial role of architecture in determining accuracy and efficiency [20, 34] and the intrinsic coupling between spike timing and architectural design in TTFS SNNs, an exploration of TTFS network architecture design is essential.

2.3 Neural Architecture Search

Neural Architecture Search (NAS) aims to automatically discover high-performing neural network architectures. Existing NAS methods generally fall into three categories: reinforcement learning (RL)-based [2, 13, 43], gradient-based [3, 15, 32], and evolutionary computation (EC)-based approaches [17, 21, 42]. Since RL-based NAS typically incurs high computational costs and large action spaces [17], most SNN-based NAS frameworks adopt gradient-based or EC-based approaches. Che *et al.* [4] adapted DARTS to develop the first differentiable hierarchical search for SNNs, and Liu *et al.* [16] extended it by incorporating spatial and temporal compression. Yan *et al.* [34] proposed a branchless supernet for gradient-based multi-objective optimization. However, gradient-based NAS relies on stable gradients and strong learning capabilities, which are difficult to satisfy in TTFS SNNs due to their sparse and timing-dependent nature. Na *et al.* [20] proposed an EC-based NAS framework with an effective SNN search space and backbone design. Che *et al.* [5] introduced evolutionary SNN neurons for transformer architecture search. Pan *et al.* [21] presented a multi-scale evolutionary search inspired by brain topologies. Building on these advances, we adopt EC-based NAS for TTFS SNNs, which avoids bi-level optimization instability and enables a more reliable search process.

3 TNAS-ER Framework

3.1 Preliminary

TTFS Spiking Neurons: In TTFS SNNs, information is encoded by spike timing. Given the spike timing t_j^{n-1} of presynaptic neurons j, the membrane potential V_i^n of neuron i in layer n updates as [26]:

$$\tau_c \frac{dV_i^{(n)}}{dt} = \begin{cases} \sum_j W_{ij}^n H\left(t - t_j^{n-1}\right), t < t_{\min}^n \\ 1, \qquad\qquad t_{\min}^n \le t \le t_{\max}^n \end{cases} \tag{1}$$

where $W_{ij}^{(n)}$ denotes the synapse weight, τ_c is the time constant, and H denotes the Heaviside function. $t_{\min}^{(n)}$ and $t_{\max}^{(n)}$ define the temporal bounds of the two phases, with $t_{min}^{(n)} = t_{max}^{(n-1)}$. As illustrated in Figure 2, the membrane potential integrates incoming spikes before $t_{\min}^{(n)}$, then increases linearly with a slope of 1, defining the

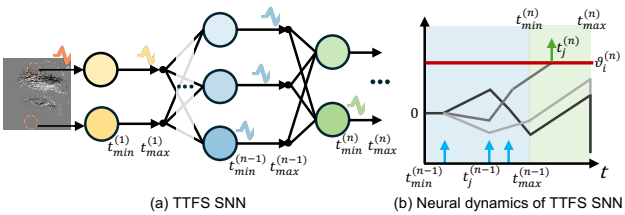


Figure 2: TTFS SNN and its neural dynamics.

$B1$ model [26]. A neuron fires when $V_i^{(n)} \ge \vartheta_i^{(n)}$ and then enters a refractory period, ensuring that each neuron fires at most once.

Mapping Between TTFS SNN and ReLU ANN: Under the above model, a bidirectional mapping can be established between TTFS SNNs and their counterpart ReLU ANNs with the same architecture [26]. The firing time is linearly related to the ReLU activation $x_i^{(n)}$:

$$t_i^{(n)} = t_{max}^{(n)} - x_i^{(n)} \tag{2}$$

The temporal window is defined as:

$$t_{max}^{(n)} = t_{min}^{(n)} + (1+\zeta) \cdot X_{max}^{(n)} \tag{3}$$

where ζ is a scaling factor, ensuring that the time window is large enough to encode the maximum possible activation value from ReLU ANNs. Given the synaptic weights W_{ij}^n and firing thresholds ϑ_i^n of the SNN, the corresponding weight w and bias b of the counterpart ReLU network are defined as [26]:

$$w_{ij}^{(n)} = \mathcal{M}(W_{ij}^{(n)}) = W_{ij}^{(n)}, \quad b_i^{(n)} = -\vartheta_i^{(n)} + \frac{t_{\max}^{(n)} - t_{\min}^{(n)}}{\tau_c} \tag{4}$$

where $\mathcal{M}$ is a function that maps the TTFS SNN weights $W_{ij}^{(n)}$ to the ReLU ANN weights $w_{ij}^{(n)}$. Under this mapping, two networks can produce identical logits and losses for the same input. Tanojevic *et al.* further demonstrated that a TTFS SNN and its ReLU ANN counterpart exhibit identical weight updates and training trajectories [26]:

$$\begin{aligned} \delta w_{ij}^{(n)} &= \mathcal{M}(W_{ij}^{(n)} - \eta \frac{\mathrm{d}\mathcal{L}}{\mathrm{d}W_{ij}^{(n)}}) - \mathcal{M}(W_{ij}^{(n)}) \\ &= -\eta \frac{\mathrm{d}\mathcal{L}}{\mathrm{d}W_{ij}^{(n)}} = \Delta W_{ij}^{(n)} \end{aligned} \tag{5}$$

This mapping between ReLU ANN and TTFS SNN allows TTFS SNN to inherit the learning capabilities of ANNs to enhance the performance, forming the foundation for our architectural design.

3.2 ANN-Assisted TTFS SNN Search

Neural architecture design is crucial for enhancing the efficiency and capability of SNNs [16, 20]. However, TTFS SNNs present unique challenges due to their single-spike temporal coding and layer-wise time scheduling, which result in unstable gradients and high sensitivity to architectural configurations [26]. To address this challenge, TNAS-ER proposes an ANN-assisted search strategy (see Figure 3). Instead of relying solely on TTFS SNN learning, we leverage a ReLU ANN counterpart that shares a bidirectional mapping with the TTFS SNN to assist the search process. The framework consists of three stages: supernet training, evolutionary search, and subnet retraining with fine-tuning. We describe how TTFS SNN and the ANN counterpart collaborate at each stage below.

Supernet training. We first construct an SNN supernet composed of searchable blocks, each containing multiple candidate operations (see Section 3.3). During training, we adopt a uniform sampling strategy to activate different candidate paths, enabling efficient exploration of the search space. However, directly training a TTFS SNN supernet is highly unstable, as different candidate operations may yield inconsistent $t_{\max}^{(n)}$ values, causing temporal misalignment within one layer. Moreover, NAS supernets typically include skip

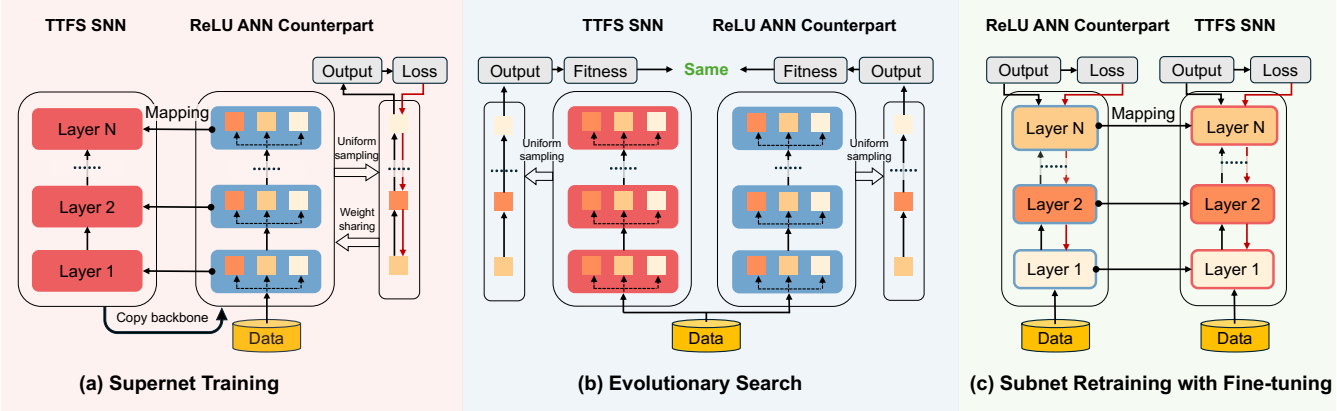


Figure 3: ANN-assisted TTFS SNN search strategy in TNAS-ER: (a) Supernet training with a ReLU ANN counterpart sharing the same backbone as the TTFS SNN and can be mapped back to the spiking domain; (b) Evolutionary search can be performed on either the TTFS SNN or the ANN counterpart, yielding identical high-performing architectures; (c) Subnet retraining in the ANN domain, followed by transfer to the TTFS SNN and SNN fine-tuning.

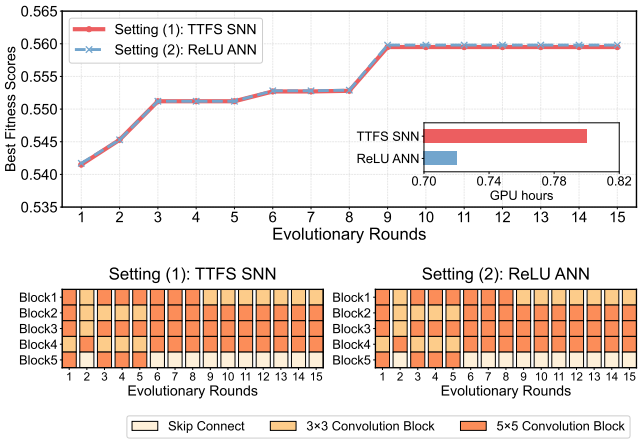


Figure 4: Evolutionary search quality comparison between TTFS SNN and its counterpart ANN: Top - fitness curves and search time, Bottom - best architecture evolution.

operations to explore variable network depths, yet in TTFS SNNs, such connections disrupt strict layer-wise temporal scheduling.

To address these issues, we introduce the ReLU ANN counterpart that shares the same backbone architecture as the TTFS SNN and conduct the supernet training in the ANN domain. Unlike TTFS SNN, ReLU ANN operates on continuous activations, avoiding temporal dependencies and enabling stable gradient-based optimization. This makes it more suitable for training large and flexible supernets with diverse candidate operations. After training, the learned parameters can be transferred to the TTFS SNN via the mapping. In this way, the ANN serves as an optimization surrogate that provides stable training signals while preserving compatibility with the TTFS SNN. Overall, this ANN-assisted training scheme eliminates temporal inconsistencies and provides better-initialized weights for candidate blocks, thereby improving search reliability and facilitating the discovery of high-performing architectures.

Evolutionary search. During the evolutionary search, candidate subnets are iteratively sampled, evaluated, and evolved to discover high-performing architectures. Each candidate subnet is treated as an individual, whose fitness is evaluated through forward inference using the inherited supernet weights. Higher-performing candidates generate new ones via mutation and crossover, progressively guiding the search toward optimal designs.

We consider two settings for architecture discovery: (1) transferring the trained ReLU ANN supernet to TTFS SNN and conducting the search based on SNN, and (2) performing the search directly on the trained ReLU ANN counterpart. In both settings, the network accuracy obtained from the forward pass is used as the fitness score. As shown in Figure 4, using a search space with five searchable blocks, the two settings yield nearly identical fitness curves and converge to the same architectures across all search rounds. This consistency arises from the mapping between TTFS SNNs and ReLU ANNs, which ensures consistent logits for the same inputs. Consequently, evolutionary search can be equivalently performed on either the ReLU ANN or the TTFS SNN. In practice, we perform the search in the ANN domain, as evaluating TTFS SNNs requires additional configuration of $t_{\max}$, resulting in longer search time.

Subnet retraining with fine-tuning. For the selected architecture, we explore three retraining settings: (1) retraining in the ANN domain and transferring to the TTFS SNN via the mapping, (2) performing (1) followed by SNN fine-tuning, and (3) directly retraining the SNN from scratch. Using a backbone where all searchable blocks adopt 3×3 convolution kernels, we observe in Figure 5 that ANN-assisted settings (1) and (2) outperform direct SNN retraining, highlighting the superior learning capability provided by the ANN counterpart. Although (1) and (3) share theoretically identical training trajectories, their accuracies diverge because this equivalence holds only in the absence of Batch Normalization (BN). In practice, (1) trains the ReLU ANN with BN, fuses BN into the weights, and transfers them to the TTFS SNN, yielding higher accuracy than (3),

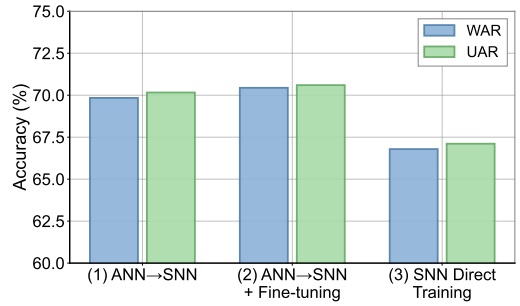


Figure 5: Subnet retrain quality comparison across different strategies.

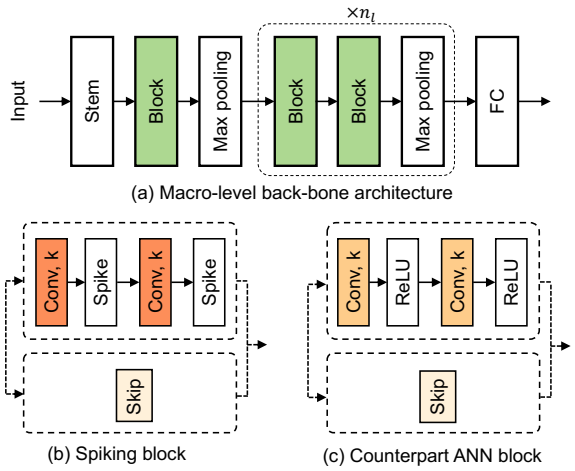


Figure 6: Hierarchical search space. Upper: Macro-level back-bone architecture, composed of searchable blocks. Bottom: Within one searchable block, the micro-level spiking block and its ANN counterpart.

which trains without BN. Furthermore, setting (2) fine-tunes the trained model to better adapt to spiking dynamics while supporting adaptation to deployment requirements for target hardware.

It is worth noting that although TNAS-ER leverages the ANN counterpart to assist the search, the search objective, search space, and performance evaluation are all defined within the TTFS SNN domain. The ReLU ANN merely serves as a differentiable proxy, enabling effective network training that would otherwise be unstable to apply directly in TTFS SNNs. Both the supernet and the subnet are trained using the cross-entropy loss. The pseudocode of TNAS-ER is provided in Algorithm 1.

3.3 Search Space and Search Algorithm

Search Space. We adopt an effective hierarchical search space inspired by prior SNN NAS studies [20]. As shown in Figure 6, it consists of a macro-level backbone and a micro-level spiking block. At the macro level, the backbone includes a stem, multiple searchable blocks separated by max-pooling, and a fully connected head. At the micro level, each searchable block contains convolution blocks with different $k \times k$ kernel sizes or a skip operation. Residual shortcuts are excluded since TTFS SNNs process layerwise, non-overlapping temporal windows, where cross-layer links would misalign spike timing. Both skip operations and residual shortcuts introduce identity-like connections. However, skip operations are temporary during search, and any resulting inconsistency can be mitigated via ANN-assisted training, whereas residual shortcuts introduce permanent cross-path aggregation that disrupts the layer-wise temporal structure of TTFS SNNs. Each spiking convolution block comprises two convolution layers with searched kernel sizes interleaved with spiking activations. In the ANN counterpart, the activations are replaced by ReLU functions. BN layers can be applied during training but are fused into the weights when transferring to the TTFS SNN, ensuring compatibility with the mapping.

Algorithm 1 TNAS-ER Framework

Supernet Training Phase

Require: Training data D_{train}, supernet $\mathcal{S}_{\text{ANN}}$, max train iteration $\mathcal{T}_{train}$

Ensure: Trained supernet parameters Θ

for $r = 1$ to $\mathcal{T}_{\text{train}}$ **do**
  Sample a subnet $(\mathcal{A}) \sim \text{Uniform}(\mathcal{S}_{\text{ANN}})$
  $\hat{y} \leftarrow \mathcal{A}(D_{\text{train}}; \Theta)$
  Compute loss and update the shared parameters Θ
end for
return Θ

Evolutionary Search Phase

Require: Trained supernet $\mathcal{S}$ with parameters Θ, validation data D_{val}, evaluation pool size N_{eval}, elite pool size N_{top}, max search iteration $\mathcal{T}_{search}$

Ensure: Architecture $\mathcal{A}^*$ with the highest fitness

Initialize evaluation pool P_{eval}
for $r = 1$ to $\mathcal{T}_{\text{search}}$ **do**
  $P_{\text{top}} \leftarrow \text{TopK}(P_{\text{eval}}, N_{\text{top}})$
  $P_{\text{eval}} \leftarrow \text{Mutation}(P_{\text{top}}) \cup \text{Crossover}(P_{\text{top}})$
  $P_{\text{eval}} \leftarrow P_{\text{eval}} \cup \text{Uniform}(\mathcal{S}, N_{\text{eval}} - |P_{\text{eval}}|)$
  fitness score $\leftarrow \mathcal{A}(D_{\text{val}}; \Theta)$ for each $\mathcal{A}$ in P_{eval}
end for
return Top-1 architecture $\mathcal{A}^*$ in P_{top}

Subnet Retraining with Finetuning Phase

Require: Searched architecture $\mathcal{A}^*$, trained supernet $\mathcal{S}_{ANN}$, $\mathcal{S}_{SNN}$ with parameters Θ, training data D_{train}, max retrain and finetune iteration $\mathcal{T}_{retrain}$, $\mathcal{T}_{ft}$

Ensure: Trained SNN parameters θ_{SNN}

for $r = 1$ to $\mathcal{T}_{\text{retrain}}$ **do**
  $y \leftarrow \mathcal{A}^*(D_{\text{train}}; \theta)$
  Compute loss and update parameters θ
end for
Convert $\mathcal{A}^*(\theta)$ to TTFS SNN $\mathcal{A}^*_{\mathcal{SNN}}(\theta_{SNN})$
for $r = 1$ to $\mathcal{T}_{\text{ft}}$ **do**
  $y \leftarrow \mathcal{A}^*_{\text{SNN}}(D_{\text{train}}; \theta_{\text{SNN}})$
  Compute loss and update θ_{SNN}
end for
return θ_{SNN}

Table 1: Comparison of different methods on the SEE dataset. The best and the second-best results are highlighted with number and number, respectively. The abbreviations are defined as Ha→ Happiness, Sa → Sadness, An → Anger, Di → Disgust, Su → Surprise, Fe → Fear, Ne → Neutral, Nor → Normal, Over → Overexposure, Low → Low-light. "Hybrid" denotes architectures combining both SNN and ANN components.

Methods	Network	Input region	Accuracy for different emotions (%)							Accuracy for different lightings (%)				Metrics (%)	
			Ha	Sa	An	Di	Su	Fe	Ne	Nor	Over	Low	HDR	WAR↑	UAR↑
Resnet18 + LSTM	ANN	Face	37.0	81.2	38.9	44.7	39.0	48.5	56.4	53.2	49.5	50.4	40.8	48.6	49.4
Resnet50 + GRU	ANN	Face	31.9	70.1	42.0	48.0	49.2	46.0	58.7	52.1	51.6	51.6	41.1	49.3	49.4
3D Resnet18 [10]	ANN	Face	79.6	93.7	72.9	77.1	58.5	75.3	83.6	75.6	75.3	80.2	73.9	76.3	77.2
R(2+1)D [27]	ANN	Face	82.0	93.6	77.9	72.5	61.5	70.4	84.1	76.6	76.2	79.6	73.1	76.4	77.4
Former DFER [41]	ANN	Face	74.2	89.4	80.0	68.1	61.7	72.5	74.5	75.3	74.3	74.0	71.3	73.8	74.3
Eyemotion [11]	ANN	Eye	63.7	78.0	59.5	57.9	44.7	60.1	71.0	60.8	57.8	64.4	61.1	61.0	62.0
EMO [33]	ANN	Eye	72.6	91.2	79.8	65.5	57.7	77.8	76.6	73.3	74.0	77.8	69.6	73.7	74.5
SEEN [37]	Hybrid	Eye	58.5	76.9	67.4	57.1	50.7	60.9	64.7	61.8	64.1	65.9	55.1	61.8	62.3
MSKD [30]	SNN	Eye	64.4	82.3	71.0	60.9	52.8	67.1	70.1	65.7	67.7	68.4	63.1	66.3	66.9
Wan *et al.* [29]	ANN	Eye	84.2	92.8	79.9	76.7	66.2	75.8	84.2	78.4	80.6	83.1	74.4	79.2	80.0
TNAS-ER	SNN	Eye	77.8	95.0	81.3	72.1	71.2	82.3	81.2	78.1	80.5	83.8	75.8	79.6	80.1

Search Algorithm. As shown in Figure 3, TNAS-ER adopts a one-shot weight-sharing strategy based on the evolutionary algorithm. During supernet training, single-path uniform sampling is used to randomly sample subnets, train them, and share their updated weights back to the supernet. Afterward, the evolutionary algorithm is applied to explore the search space and identify the architecture with the highest fitness value. We define the fitness value score of an architecture $\mathcal{A}$ as:

$$F(\mathcal{A}) = \frac{WAR + UAR}{2}, \tag{6}$$

where UAR (Unweighted Average Recall) = $\frac{1}{C}\sum_{i=1}^{C}\frac{TP_i}{TP_i+FN_i}$ measures the average recall across all emotion classes, treating each class equally regardless of frequency, and WAR (Weighted Average Recall) = $\frac{TP+TN}{TP+TN+FP+FN}$ reflects the overall recognition accuracy across all samples. Combining both metrics balances recognition accuracy and robustness to class imbalance. The evolutionary process begins by randomly sampling N_{eval} candidate architectures to form an evaluation pool P_{eval}. We compute the fitness score of each architecture in P_{eval} and select the top N_{top} candidates as the elite set P_{top}. From P_{top}, we generate a new set of N_{eval} architectures through mutation and crossover, forming the new P_{eval}. We then combine P_{eval} and P_{top} and retain the top N_{top} architectures to update the elite pool. This process iterates until convergence, yielding the architecture with the highest fitness score.

4 Experiments

4.1 Datasets

The SEE dataset [37] was collected using a helmet-mounted DAVIS346 event camera positioned in front of the participant's right eye. It contains 71.5 minutes of recordings from 111 volunteers, covering seven emotions under four lighting conditions. Following prior works [30, 37], it is split into 1,638 training and 767 testing samples.

The DSEE dataset [30] extends SEE with samples converted from five video-based emotion datasets [1, 18, 22, 28, 39] using the video-to-event simulator [12]. It contains 6,235 samples from 394 subjects with greater age and ethnic diversity, making it a more comprehensive and challenging benchmark. Following [30], it is split into 4,362 training and 1,873 testing samples.

For both datasets, 80% of the training samples are used for supernet training and 20% for evolutionary search. All training samples are used for retraining, while the testing samplesremain strictly held out throughout the search process.

4.2 Experimental Settings

Implementation Details. All experiments are implemented in PyTorch and conducted on an NVIDIA GeForce RTX 4090 GPU (24 GB). The learning rates for supernet training, subnet retraining, and SNN fine-tuning are set to 5×10^{-3}, 5×10^{-4}, and 5×10^{-5}, respectively. The supernet and subnet are trained for 600 epochs, evolutionary search runs for 18 rounds, and fine-tuning is performed for 100 epochs. The networks are optimized using Adam, and the batch size is 96. Following prior works [30, 37], performance is evaluated using UAR and WAR, averaged over 20 inference runs.

Preprocessing. Event preprocessing follows prior works [29, 30, 37]. Events are aggregated into segments using the E4-S3 setting, where Ex-Sy denotes x event frames separated by a skip interval of y in units of 1/30 s, yielding a total test duration of $(x+(x-1)\times y)/30$ seconds. All methods use this setting, with multiple trials using random start points to ensure robustness. The event frames are concatenated into 4 channels and fed into the network.

Architecture The stem block contains two 3×3 convolutional layers. The searched backbone includes $n_l = 2$ macro-block groups, each with two searchable blocks. The first stem convolution takes 4 input channels and outputs 32 channels, and the channel width doubles after each max-pooling layer. During the search, N_{eval} and N_{top} are set to 12. The factor ζ in the temporal window is set to 0.5.

4.3 Performance Comparison

Table 1 and Table 2 compare the proposed TNAS-ER with state-of-the-art methods on the SEE and DSEE datasets, following the same experimental settings as prior works [29, 30] for fair evaluation.

TNAS-ER consistently achieves the highest performance on both datasets, outperforming all facial and eye-region methods. On the SEE dataset, TNAS-ER reaches a WAR of 79.6% and a UAR of 80.1%,

Table 2: Comparison of different methods on the DSEE dataset. The best and the second best results are highlighted in number and number, respectively. The abbreviations are defined as Ha→ Happiness, Sa → Sadness, An → Anger, Di → Disgust, Su → Surprise, Fe → Fear, Ne → Neutral, Nor → Normal, Over → Overexposure, Low → Low-light. "Hybrid" denotes architectures combining both SNN and ANN components.

Methods	Network	Input region	Accuracy for different emotions (%)							Accuracy for different lightings (%)				Metrics (%)	
			Ha	Sa	An	Di	Su	Fe	Ne	Nor	Over	Low	HDR	WAR↑	UAR↑
R(2+1)D [27]	ANN	Face	68.5	67.0	72.0	75.0	68.7	62.4	73.0	68.9	66.3	68.4	78.7	69.4	69.5
Former DFER [41]	ANN	Face	65.4	68.6	71.4	70.9	67.7	57.0	77.6	68.3	66.4	66.2	70.9	67.9	68.4
Eyemotion [11]	ANN	Eye	63.0	50.2	57.3	52.2	48.9	56.6	61.8	55.5	49.8	54.0	64.5	55.2	55.7
EMO [33]	ANN	Eye	57.7	61.0	67.2	61.1	61.0	48.0	71.8	60.1	58.0	61.7	65.1	60.5	61.1
SEEN [37]	Hybrid	Eye	54.8	55.5	62.6	58.8	63.1	47.6	69.8	58.5	59.0	56.6	58.5	58.3	58.9
MSKD [30]	SNN	Eye	61.3	59.3	65.6	60.9	63.3	47.4	67.1	60.1	59.1	61.4	63.0	60.4	60.7
Wan *et al.* [29]	ANN	Eye	75.1	70.6	72.9	76.1	73.0	56.2	80.6	72.3	69.4	67.8	77.8	71.6	72.1
TNAS-ER	SNN	Eye	70.4	70.7	78.2	74.5	74.7	62.4	79.3	72.1	70.2	72.2	79.7	72.6	72.9

Table 3: Computational efficiency comparison of different eye-based emotion recognition methods. For ANN-based methods, OPs refer to FLOPs, while for SNN-based methods, OPs are measured as SynOPs.

Methods	WAR (%)	UAR (%)	OPs (G)	Params (M)	Latency (ms)
Eyemotion	61.0	62.0	5.73	25.13	45
EMO	73.7	74.5	0.35	1.68	19
SEEN	61.8	62.3	0.73	6.08	9
MSKD	66.3	66.9	0.17	4.04	6
Wan *et al.*	79.2	80.0	8.44	51.94	19
TNAS-ER	79.6	80.1	0.03	1.04	1.03

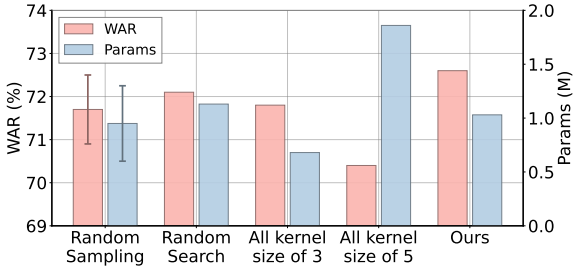


Figure 7: Performance comparison with SNNs obtained from different settings on DSEE dataset.

and achieves top scores in four of the seven emotion categories (sadness, anger, surprise, and fear) and in all four lighting conditions (normal, overexposure, low-light, and HDR). Notably, compared with other SNN-based approaches such as MSKD and SEEN, TNAS-ER achieves substantially stronger performance, with improvements exceeding 10% in both WAR and UAR. A similar trend is observed on the DSEE dataset, where our TNAS-ER obtains a WAR of 72.6% and a UAR of 72.9%, outperforming the best-performing baseline [29] by 1.0% in WAR and 0.8% in UAR. Furthermore, TNAS-ER achieves top scores across six of seven emotion categories (happiness, sadness, anger, surprise, fear, and neutral) and all lighting conditions. Since DSEE includes more subjects with a broader range of ages and ethnic backgrounds, the larger performance gain compared to SEE highlights the strong generalization and robustness of TNAS-ER across diverse and challenging scenarios.

4.4 Computational Efficiency

This section evaluates the computational efficiency of TNAS-ER for practical deployment. We compare it with existing eye-based approaches on the SEE dataset in terms of recognition performance (WAR and UAR) and computational cost (the number of operations, parameters, and inference latency).

As shown in Table 3, TNAS-ER achieves the best balance between accuracy and efficiency. It surpasses all baseline methods with the highest recognition performance, while reducing operations, parameters, and latency by over 5.6×, 1.6×, and 5.8×, respectively. Compared to the most accurate baseline Wan *et al.* [29], TNAS-ER reduces the number of operations by over 280×, parameters by nearly 50× and inference latency by over 17×. Even when compared to the lightweight EMO model, TNAS-ER delivers nearly 6% higher accuracy with 11× fewer operations, 1.6× fewer parameters and 17× lower latency. We further observe that both SNN-based methods, MSKD and ours, exhibit substantially fewer operations than ANN-based models, confirming that the intrinsic sparsity of SNNs contributes to improved computational efficiency. Owing to its compact architecture and TTFS coding, TNAS-ER achieves lower SynOPs than MSKD. These results collectively highlight the superior effectiveness and efficiency of TNAS-ER, making it particularly suitable for low-power, real-world deployments.

4.5 Effectiveness of the Searched Architecture

Comparison with Random Sampling. We randomly sample 10 architectures from the search space, where each searchable block independently selects a 3×3 or 5×5 convolution, or a skip connection. Each sampled architecture is trained and evaluated independently, and the reported WAR and UAR values are averaged among 10 architectures. As shown in Figure 7, the architectures obtained through random sampling achieve lower performance than TNAS-ER, highlighting the superiority of the searched architecture.

Comparison with Random Search. We implement random search following [20] by selecting the architecture with the highest fitness scores among 100 randomly sampled architectures based on the trained supernet. As shown in Figure 7, this approach yields better performance than random sampling but still falls short of TNAS-ER. The result indicates that while fitness-based selection

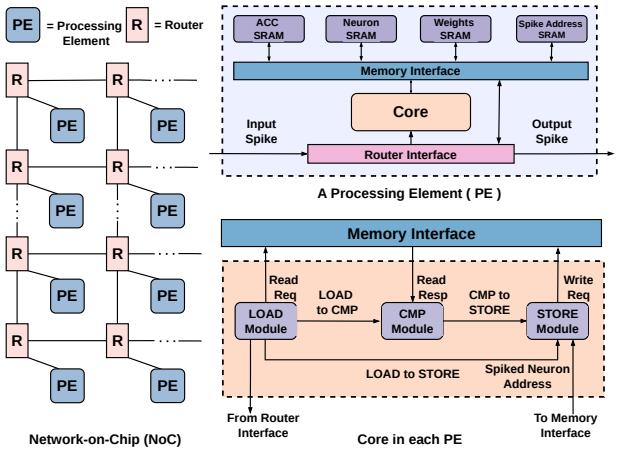


Figure 8: Overview of YOSO, a TTFS-based neuromorphic hardware [36].

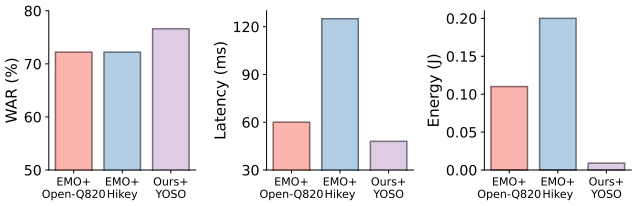


Figure 9: Hardware performance of different eye-based emotion recognition systems.

helps identify stronger TTFS SNN architectures, the evolutionary strategy in TNAS-ER further enhances search effectiveness.

Comparison with Fixed Backbones. We further compare our searched architecture with fixed-backbone architectures, where all searchable blocks are replaced with convolution blocks of uniform kernel sizes (3×3 or 5×5). As shown in Figure 7, both architectures with fixed configurations perform worse than the architecture discovered by TNAS-ER. We observe that the architecture using 3×3 kernels outperforms that using 5×5 kernels, indicating that a larger network does not necessarily lead to higher performance. These results demonstrate that the proposed evolutionary search can automatically discover an optimal balance between network capacity and efficiency, yielding compact yet high-performing architectures beyond what manual design can achieve.

Architecture Transferability. We evaluate the transferability of the searched architecture across datasets. Transferability is particularly important when users need to rapidly deploy an existing architecture to new tasks, avoiding the cost of searching from scratch. Since the DSEE dataset contains a broader range of samples, we first search for the optimal architecture on SEE and then train and test this architecture on DSEE. The transferred model achieves 72.1% WAR and 72.3% UAR, with less than a 1% performance drop compared to the architecture searched directly on DSEE. Remarkably, it attains competitive results with Wan et al. [29] while using only 1.04 M parameters, compared to their 51 M, demonstrating strong generalization and efficiency of the searched architecture.

4.6 Evaluation on Neuromorphic Hardware

We evaluated the deployability of TNAS-ER by deploying it on YOSO [6, 36], a general-purpose neuromorphic hardware platform for TTFS SNN. This experiment aims to validate whether the searched architecture can be effectively executed on neuromorphic hardware under realistic system constraints. As shown in Figure 8, YOSO consists of multiple Processing Elements (PEs) interconnected through a Network-on-Chip (NoC). Each PE integrates a compute core capable of parallel load, write, and compute operations, with clock-gated SRAMs for storing weights, neuron states, accumulated values, and spike information. The system operates at 54 MHz.

Before deployment, the trained TNAS-ER model is quantized, achieving 76.6% WAR and 77.1% UAR. During hardware mapping, different layers are assigned to distinct PE groups to maximize parallelism. Each output feature map is partitioned into multiple regions according to the number of neurons, with each region processed by a single PE. This mapping strategy minimizes inter-PE data transactions, thereby reducing overall energy consumption. More details are provided in the Supplementary. Following the evaluation protocol of EMO [33], we measure only the resource usage during the emotion recognition phase, as the overhead of eye-region video acquisition lies outside network computation.

The TNAS-ER + YOSO system achieves an energy consumption of 9.0 mJ and a latency of 48.0 ms. Since the latency is much shorter than the 0.4 s duration of event input, the system satisfies real-time processing requirements. Due to the lack of hardware evaluations in eye-based emotion recognition models, and none of them are TTFS-based, direct hardware performance comparisons are not feasible. The EMO system [33], which has been deployed on a wearable headset, is the only prior eye emotion recognition work that reports hardware performance. As shown in Figure 9, our system exhibits higher accuracy, lower energy consumption and shorter latency than the EMO system [33]. These results indicate that the proposed method holds strong potential for practical deployment in future wearable devices and provide a reference baseline for future research on event-based eye emotion recognition.

5 Conclusion

In this paper, we present TNAS-ER, a NAS framework tailored to TTFS SNNs for efficient eye-based emotion recognition. TNAS-ER introduces an ANN-assisted search strategy that couples the TTFS SNN with its ReLU ANN counterpart to enable stable training and effective architecture exploration. Extensive experimental results demonstrate that TNAS-ER achieves state-of-the-art recognition performance with 1.6× fewer parameters, 5.6× fewer operations and 5.8 × shorter inference latency compared to existing ANN and SNN baselines. We further validate that TNAS-ER outperforms both manually designed and randomly searched architectures. We also deploy TNAS-ER on the YOSO neuromorphic hardware and show the strong potential of the system for practical deployment. Overall, this work underscores the potential of architecture-level optimization in advancing spiking neural computing and paves the way for next-generation wearable and edge-intelligent systems.

Acknowledgments

This work was supported by the Natural Science Foundation of China (62372275), the foundation of Jinan (JNSX2025004, 202534077), the Key R&D Program of Shandong Province (2025CXGC010112), Qingdao Key Laboratory of Trustworthy Artificial Intelligence, Shandong Province Key Laboratory of Network Integration Theory and Technology, and Shandong Provincial Laboratory for Humanities and Natural Sciences Collaborative Innovation.

References

[1] Niki Aifanti, Christos Papachristou, and Anastasios Delopoulos. 2010. The MUG facial expression database. In *11th International Workshop on Image Analysis for Multimedia Interactive Services WIAMIS 10*. IEEE, 1–4.

[2] Irwan Bello, Barret Zoph, Vijay Vasudevan, and Quoc V Le. 2017. Neural optimizer search with reinforcement learning. In *International Conference on Machine Learning*. PMLR, 459–468.

[3] Han Cai, Ligeng Zhu, and Song Han. 2019. ProxylessNAS: Direct Neural Architecture Search on Target Task and Hardware. In *International Conference on Learning Representations*.

[4] Kaiwei Che, Luziwei Leng, Kaixuan Zhang, Jianguo Zhang, Qinghu Meng, Jie Cheng, Qinghai Guo, and Jianxing Liao. 2022. Differentiable hierarchical and surrogate gradient search for spiking neural networks. *Advances in Neural Information Processing Systems* 35 (2022), 24975–24990.

[5] Kaiwei Che, Zhaokun Zhou, Jun Niu, Zhengyu Ma, Wei Fang, Yanqi Chen, Shuaijie Shen, Li Yuan, and Yonghong Tian. 2024. Auto-spikformer: Spikformer architecture search. *Frontiers in Neuroscience* 18 (2024), 1372257.

[6] Kyle Timothy Ng Chu, Burin Amornpaisannon, Yaswanth Tavva, Venkata Pavan Kumar Miriyala, Jibin Wu, Malu Zhang, Haizhou Li, Trevor E Carlson, et al. 2020. You only spike once: Improving energy-efficient neuromorphic inference to ann-level accuracy. *arXiv preprint arXiv:2006.09982* (2020).

[7] Roddy Cowie, Ellen Douglas-Cowie, Nicolas Tsapatsoulis, George Votsis, Stefanos Kollias, Winfried Fellenz, and John G Taylor. 2001. Emotion recognition in human-computer interaction. *IEEE Signal Processing Magazine* 18, 1 (2001), 32–80.

[8] Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J Davison, Jörg Conradt, Kostas Daniilidis, et al. 2020. Event-based vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 1 (2020), 154–180.

[9] Runduo Han, Xiuping Liu, Yi Zhang, Jun Zhou, Hongchen Tan, and Xin Li. 2025. Hierarchical Event-RGB Interaction Network for single-eye expression recognition. *Information Sciences* 690 (2025), 121539.

[10] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. 2018. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 6546–6555.

[11] Steven Hickson, Nick Dufour, Avneesh Sud, Vivek Kwatra, and Irfan Essa. 2019. Eyemotion: Classifying facial expressions in VR using eye-tracking cameras. In *2019 IEEE Winter Conference on Applications of Computer Vision*. IEEE, 1626–1635.

[12] Yuhuang Hu, Shih-Chii Liu, and Tobi Delbruck. 2021. v2e: From video frames to realistic DVS events. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1312–1321.

[13] Yesmina Jaafra, Jean Luc Laurent, Aline Deruyver, and Mohamed Saber Naceur. 2019. Reinforcement learning for neural architecture search: A review. *Image and Vision Computing* 89 (2019), 57–66.

[14] Saeed Reza Kheradpisheh and Timothée Masquelier. 2020. Temporal backpropagation for spiking neural networks with one spike per neuron. *International Journal of Neural Systems* 30, 06 (2020), 2050027.

[15] Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2019. DARTS: Differentiable Architecture Search. In *International Conference on Learning Representations*.

[16] Qianhui Liu, Jiaqi Yan, Malu Zhang, Gang Pan, and Haizhou Li. 2024. LitE-SNN: designing lightweight and efficient spiking neural network through spatial-temporal compressive network search and joint optimization. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*. 3097–3105.

[17] Yuqiao Liu, Yanan Sun, Bing Xue, Mengjie Zhang, Gary G Yen, and Kay Chen Tan. 2021. A survey on evolutionary neural architecture search. *IEEE Transactions on Neural Networks and Learning Systems* 34, 2 (2021), 550–570.

[18] Patrick Lucey, Jeffrey F Cohn, Takeo Kanade, Jason Saragih, Zara Ambadar, and Iain Matthews. 2010. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*. IEEE, 94–101.

[19] Wolfgang Maass. 1997. Networks of spiking neurons: the third generation of neural network models. *Neural Networks* 10, 9 (1997), 1659–1671.

[20] Byunggook Na, Jisoo Mok, Seongsik Park, Dongjin Lee, Hyeokjun Choe, and Sungroh Yoon. 2022. Autosnn: Towards energy-efficient spiking neural networks. In *International Conference on Machine Learning*. PMLR, 16253–16269.

[21] Wenxuan Pan, Feifei Zhao, Guobin Shen, Bing Han, and Yi Zeng. 2024. Brain-inspired multi-scale evolutionary neural architecture search for deep spiking neural networks. *IEEE Transactions on Evolutionary Computation* (2024).

[22] Maja Pantic, Michel Valstar, Ron Rademaker, and Ludo Maat. 2005. Web-based database for facial expression analysis. In *2005 IEEE International Conference on Multimedia and Expo*. IEEE, 5–pp.

[23] Seongsik Park, Seijoon Kim, Byunggook Na, and Sungroh Yoon. 2020. T2FSNN: deep spiking neural networks with time-to-first-spike coding. In *2020 57th ACM/IEEE Design Automation Conference*. IEEE, 1–6.

[24] Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. 2019. Towards spike-based machine intelligence with neuromorphic computing. *Nature* 575, 7784 (2019), 607–617.

[25] Bodo Rueckauer and Shih-Chii Liu. 2018. Conversion of analog to spiking neural networks using sparse temporal coding. In *2018 IEEE International Symposium on Circuits and Systems*. IEEE, 1–5.

[26] Ana Stanojevic, Stanisław Woźniak, Guillaume Bellec, Giovanni Cherubini, Angeliki Pantazi, and Wulfram Gerstner. 2024. High-performance deep spiking neural networks with 0.3 spikes per neuron. *Nature Communications* 15, 1 (2024), 6793.

[27] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. 2018. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 6450–6459.

[28] Job Van Der Schalk, Skyler T Hawk, Agneta H Fischer, and Bertjan Doosje. 2011. Moving faces, looking places: validation of the Amsterdam Dynamic Facial Expression Set (ADFES). *Emotion* 11, 4 (2011), 907.

[29] Zixuan Wan, Jiqing Zhang, Yushan Wang, Hu Lin, Yafei Wang, Zetian Mi, Xin Yang, Xianping Fu, and Huibing Wang. 2025. Eye-based Emotion Recognition via Event-Driven Sparse Transformers. In *Proceedings of the 33rd ACM International Conference on Multimedia*. 4659–4668.

[30] Yang Wang, Haiyang Mei, Qirui Bao, Ziqi Wei, Mike Zheng Shou, Haizhou Li, Bo Dong, and Xin Yang. 2024. Apprenticeship-inspired elegance: synergistic knowledge distillation empowers spiking neural networks for efficient single-eye emotion recognition. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*. 3160–3168.

[31] Wenjie Wei, Malu Zhang, Hong Qu, Ammar Belatreche, Jian Zhang, and Hong Chen. 2023. Temporal-coded spiking neural networks with dynamic firing threshold: Learning with event-driven backpropagation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 10552–10562.

[32] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. 2019. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10734–10742.

[33] Hao Wu, Jinghao Feng, Xuejin Tian, Edward Sun, Yunxin Liu, Bo Dong, Fengyuan Xu, and Sheng Zhong. 2020. EMO: Real-time emotion recognition from single-eye images for resource-constrained eyewear devices. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*. 448–461.

[34] Jiaqi Yan, Qianhui Liu, Malu Zhang, Lang Feng, De Ma, Haizhou Li, and Gang Pan. 2024. Efficient spiking neural network design via neural architecture search. *Neural Networks* 173 (2024), 106172.

[35] Qu Yang, Malu Zhang, Jibin Wu, Kay Chen Tan, and Haizhou Li. 2023. LC-TTFS: Toward Lossless Network Conversion for Spiking Neural Networks With TTFS Coding. *IEEE Transactions on Cognitive and Developmental Systems* 16, 5 (2023), 1626–1639.

[36] Miao Yu, Tingting Xiang, Srivatsa P, Kyle Timothy Ng Chu, Burin Amornpaisannon, Yaswanth Tavva, Venkata Pavan Kumar Miriyala, and Trevor E Carlson. 2023. A TTFS-based energy and utilization efficient neuromorphic CNN accelerator. *Frontiers in Neuroscience* 17 (2023), 1121592.

[37] Haiwei Zhang, Jiqing Zhang, Bo Dong, Pieter Peers, Wenwei Wu, Xiaopeng Wei, Felix Heide, and Xin Yang. 2023. In the blink of an eye: Event-based emotion recognition. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–11.

[38] Lei Zhang, Shengyuan Zhou, Tian Zhi, Zidong Du, and Yunji Chen. 2019. Tdsnn: From deep neural networks to deep spike neural networks with temporal-coding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 1319–1326.

[39] Guoying Zhao, Xiaohua Huang, Matti Taini, Stan Z Li, and Matti PietikäInen. 2011. Facial expression recognition from near-infrared videos. *Image and Vision Computing* 29, 9 (2011), 607–619.

[40] Lusen Zhao, Zihan Huang, Jianhao Ding, and Zhaofei Yu. 2025. TTFSFormer: A TTFS-based Lossless Conversion of Spiking Transformer. In *Forty-Second International Conference on Machine Learning*.

[41] Zengqun Zhao and Qingshan Liu. 2021. Former-dfer: Dynamic facial expression recognition transformer. In *Proceedings of the 29th ACM International Conference on Multimedia*. 1553–1561.

[42] Hangyu Zhu and Yaochu Jin. 2021. Real-time federated evolutionary neural architecture search. *IEEE Transactions on Evolutionary Computation* 26, 2 (2021), 364–378.

[43] Barret Zoph and Quoc V Le. 2017. Neural architecture search with reinforcement learning. In *International Conference on Learning Representations*.