Automotive Object Detection via Learning Sparse Events by Spiking Neurons

Hu Zhang, Yanchen Li, Luziwei Leng *Member, IEEE*, Kaiwei Che, Qian Liu, Qinghai Guo, Jianxing Liao, and Ran Cheng *Senior Member, IEEE*

***Abstract*—Event-based sensors, distinguished by their high temporal resolution of 1 µs and a dynamic range of 120 dB, stand out as ideal tools for deployment in fast-paced settings like vehicles and drones. Traditional object detection techniques that utilize Artificial Neural Networks (ANNs) face challenges due to the sparse and asynchronous nature of the events these sensors capture. In contrast, Spiking Neural Networks (SNNs) offer a promising alternative, providing a temporal representation that is inherently aligned with event-based data. This paper explores the unique membrane potential dynamics of SNNs and their ability to modulate sparse events. We introduce an innovative spike-triggered adaptive threshold mechanism designed for stable training. Building on these insights, we present a specialized spiking feature pyramid network (SpikeFPN) optimized for automotive event-based object detection. Comprehensive evaluations demonstrate that SpikeFPN surpasses both traditional SNNs and advanced ANNs enhanced with attention mechanisms. Evidently, SpikeFPN achieves a mean Average Precision (mAP) of 0.477 on the GEN1 Automotive Detection (GAD) benchmark dataset, marking significant increases over the selected SNN baselines. Moreover, the efficient design of SpikeFPN ensures robust performance while optimizing computational resources, attributed to its innate sparse computation capabilities. Source codes are publicly accessible at https://github.com/EMI-Group/spikefpn.**

***Index Terms*—Object detection, dynamical vision sensor, deep learning, spiking neural networks.**

I. Introduction

OBJECT detection is fundamental in computer vision, with applications ranging from face recognition to vehicle tracking and the identification of smaller objects [1], [2]. Traditionally, such tasks rely on frame-based cameras. However, these often produce images that are blurred or poorly exposed, especially in high-speed or challenging lighting conditions, common in scenarios like emergency vehicle detection. Addressing this gap, the recently advanced Dynamical Vision Sensor (DVS), a.k.a. the event-based sensor [3]–[8], offers a compelling alternative. Unlike traditional sensors, the DVS draws inspiration from retinal functionalities and is adept at registering pixel intensity variations. With its remarkable temporal resolution of 1 µs and a dynamic range of 120 dB, it is able to effectively record asynchronous events instigated by shifts in pixel brightness.

Despite the appealing characteristics, the unique nature of DVS also brings challenges. Its event-driven data is sparse and unpredictable, making traditional object detection methods based on Artificial Neural Networks (ANNs) less effective. While there have been efforts to preprocess such events using various architectures and algorithms [9], [10], they often come with computational and latency costs. By contrast, the Spiking Neural Networks (SNNs) [11], [12] introduce a new paradigm, resembling the brain's functioning by transmitting information through discrete spikes. Their inherent efficiency, marked by sparse activations and multiplication-free inferences, makes them ideal for managing the dynamic, sparse data from event-driven sensors.

Recent breakthroughs in direct training methods have enabled more efficient development of deep SNN architectures [13]–[15]. These advancements have paved the way to improve SNN performance in image classification tasks [16]–[19]. However, when it comes to more intricate vision tasks like automotive object detection [20]–[22], SNNs still find themselves trailing behind the established prowess of ANNs. Though significant strides have been made in enhancing ANNs for challenging object detection tasks [23]–[27], directly transposing these refined ANN architectures onto the SNN framework often leads to compromised performance and heightened latency. Such performance deficits are glaring in tasks that necessitate sophisticated neural network adjustments [20], [28], [29]. Moreover, while current efforts focus on converting ANNs to SNNs [30]–[34], these approaches grapple with inherent challenges. These primarily arise when integrating the dynamic nature of SNNs with the foundational attributes of ANNs, which frequently results in the underutilization of the unique temporal and sparse characteristics inherent to SNNs — essential for handling event-driven data efficiently. The inherent differences in data representation and processing between the two neural architectures often result in inconsistencies, undermining both the accuracy and computational performance of the neural system.

In response to these challenges, we delve deeper into the temporal dynamics inherent to spiking neurons, aiming to develop an approach characterized by minimal computational requirements and efficient response times, especially crucial for swiftly varying events. Leveraging our findings, we design a tailored spiking feature pyramid network (SpikeFPN) for event-based automotive object detection, which harmoniously blends the unique neuronal properties with surrogate gradient

This work was supported in part by Guangdong Natural Science Funds for Distinguished Young Scholar under Grant 2024B1515020019. (*Hu Zhang and Yanchen Li contributed equally to this work.*) (*Corresponding authors: Luziwei Leng and Ran Cheng.*)

Hu Zhang, Yanchen Li, Kaiwei Che and Ran Cheng are with the Southern University of Science and Technology, Shenzhen 518055, China. (e-mail: ranchengcn@gmail.com)

Luziwei Leng, Qian Liu, Qinghai Guo and Jianxing Liao are with the Advanced Computing and Storage Lab, Huawei Technologies Co., Ltd., Shenzhen 518055, China. (e-mail: lengluziwei@huawei.com)

training to achieve optimal performance. Our primary contributions are delineated as follows:

1) We explore the inherent temporal dynamics of spiking neurons, particularly focusing on membrane potential dynamics and adaptive membrane thresholds. By retaining the adaptive feedback mechanism and involving the adaptive spiking neuron model in training, our comprehensive study not only reveals their intrinsic strengths in managing rapid event modulations but also enhances feature robustness in sparse event data. The insights gained ensure stable and efficient training phases, optimizing the network's performance in real-world scenarios.
2) We design an event-driven SpikeFPN tailored specifically for automotive object detection, leveraging the self-adaptive mechanism inspired by neocortex neuron adaptation. This design foundation captures the unique temporal dynamics of spiking neurons, presenting a novel approach to object detection in automotive scenarios. We directly train our SpikeFPN using a surrogate gradient to facilitate model design, with the selected surrogate gradient function optimizing the development process.
3) Our proposed SpikeFPN demonstrates promising performance, surpassing previous SNN models and advanced ANNs with attention mechanisms in the realms of current event-based automotive object detection. It attains a noteworthy mean Average Precision (mAP) of 0.477 on the GEN1 Automotive Detection (GAD) benchmark dataset, marking a substantial advancement by outperforming the selected SNN baselines, exceeding the best baseline by 9.7%. Additionally, our design is a testament to efficiency — combining a streamlined architecture with impressive accuracy, all while significantly reducing computation costs.

The structure of this paper is outlined as follows: Section II provides the foundational background relevant to our study; Section III delves into the intricacies of the proposed SpikeFPN, detailing its architectural components and elaborating on the neural behavior and adaptive strategies employed; Section IV showcases our experimental approaches, the methodologies adopted, and the corresponding results, affirming the effectiveness of our design; finally, the paper culminates with conclusions drawn in Section V.

II. Background

A. Event-based Object Detection

Event cameras, possessing outstanding temporal resolution and dynamic range, are highly suited for scenarios demanding rapid object tracking, variable lighting conditions, and minimal latency. Nonetheless, the intrinsic dynamism of event camera data conflicts with conventional deep learning paradigms grounded in frame-based methodologies. Typically, to counter this incongruence, events are preprocessed to transform into denser representations before network processing. Various methodologies, both handcrafted and automated, have been developed for this transformation, encompassing event frames [35]–[40], time or event-number stacking [41], voxel grids [42], the event queue approach [43], LSTM grids [44], and discrete time convolutions [45]. For sparse event data, asynchronous convolutions have also been discussed [46], [47], with the latter notably applied to the GEN1 Automotive Detection (GAD) dataset acquired with the GEN1 sensor for event-driven automotive object detection tasks [48]. The work by Perot *et al.* [9] emphasized a convolutional LSTM network coupled with a temporal consistency loss, amplifying training outcomes. ASTMNet [10] showcased a progressive adaptive sampling protocol, fusing temporal attention and memory modules, and clocked significant accuracy levels.

B. Deep SNNs for Vision Tasks

Neuromorphic hardware evolution has increased interest in employing SNNs in challenging vision-based deep learning tasks [49]–[55]. When combined with event cameras, these neuromorphic ecosystems are expected to offer power efficiency coupled with minimal latency. To this end, previous studies have opened vital avenues of research [12].

The ANN-to-SNN conversion strategy revolves around approximating real-valued activations with spiking dynamics [33], [56]. This strategy has achieved high accuracy and provides a viable path for training and using SNN models. Kim *et al.* [29] introduced a spiking version of YOLO for object detection and achieved comparable performance on the PASCAL VOC [57] and MS COCO datasets [58], although it required many time steps to converge. Recent endeavors have managed to reduce latency after conversion [30], [31], yet the scalability of these methodologies to more intricate architectures beyond mere image classification is unclear.

Direct training for SNNs, on the other hand, uses surrogate gradient (SG) functions to mimic backpropagation gradients [13], [15], [59]. Owing to advancements like tailored encoding, SG, and loss function crafting [18], [19], [60], directly trained SNNs have been attaining competitive accuracies on strenuous benchmark tasks, demanding minimal simulation steps for convergence. These milestones have catalyzed their expansion into other event-centric vision tasks, from optical flow assessment [28] to video reconstruction [61] and object detection [20]. Recent breakthroughs [62]–[64] underscore that spike-centric differentiable hierarchical searches can significantly improve SNN performance across a spectrum of event-driven vision tasks, like deep stereo and semantic segmentation.

C. Feature Pyramid Networks

Feature Pyramid Networks (FPNs) are central components of numerous object detection systems, with their capacity to exploit multi-scale feature information enhancing overall system performance [65]–[67]. The original FPN design was introduced by Lin *et al.* [66], which built on traditional pyramid methods by efficiently leveraging single-scale image input, forming a foundation for subsequent advancements in the field.

Among these advancements, the FaPN by Huang *et al.* [68] incorporated a feature alignment module, achieving a significant refinement over the conventional FPN design. In

contrast to the fusion-based approach, the Single Shot Detector (SSD) method, proposed by Liu *et al.* [69], delegated different stages of feature maps to detect objects of distinct scales. Several integrated techniques, such as those found in previous work [70]–[74], fused features from diverse stages, predominantly using a top-down unidirectional fusion method — a prevailing FPN fusion mode in modern object detection models.

Liu *et al.* [75] made strides by introducing the concept of bottom-up secondary bi-directional fusion, shedding light on the potential benefits of bi-directional fusion. Further expanding on the capabilities of FPNs, the ASFF [76] incorporated an attention mechanism, while complex bidirectional fusion techniques [77], [78] ventured into more intricate fusion dynamics. A novel approach was taken as the Recursive-FPN [79], wherein the fused output of the FPN was reintegrated into the backbone, initiating an additional loop and marking new progress. Further bridging the gap between conventional neural networks and the emerging field of SNNs, Fu *et al.* [80] endeavored to combine feature pyramid structures with SNNs.

III. Method

This section begins with the representation of event data. Subsequently, it introduces the proposed feature-pyramid-centric spiking neural network supported by a self-adaptive spiking neuron model. To clarify the network design, the section describes the fundamental network architecture, highlighting modules such as the primary backbone, and further presents the formulation of the adaptive spiking neuron model intrinsic to the network. Ultimately, the training strategy for addressing non-derivable neuron models using approximate information is detailed.

A. Event Data Representation

Traditional cameras capture visual information in the form of images. In contrast, a DVS detects individual pixel changes in their receptive fields, registering changes in luminance and labeling these as "events". Typically, an event is recorded in a tuple format (t, x, y, p), where t represents the timestamp of the event occurrence; x and y denote the two-dimensional pixel coordinates where the event is registered; and p indicates the polarity of the event, reflecting the direction of luminance change.

In order to encode the discrete event data while retaining its temporal nuances, we employ the Stacking Based on Time (SBT) method for event preprocessing as presented by Wang *et al.* [41]. The SBT approach, recognized for its real-time processing and low computational overhead, has been integrated into the accumulator modules of mainstream event sensors. Beyond mere data compression, the SBT preserves the granular temporal information of the event stream. It amalgamates events into temporally contiguous frames, facilitating the downstream Spiking Neural Network (SNN) to learn from this rich temporal dataset. Within a single stack input into the detection model, the SBT method partitions the event data uniformly based on the time interval Δt. It then compresses this data to yield a set of frames. Assuming the event data in the stack can be subdivided into n equal time intervals, the value of each pixel in the i-th frame is characterized by the aggregated polarity of events as:

$$P(x, y) = \text{sign}\left(\sum_{t \in T} p(x, y, t) \right), \tag{1}$$

where P designates the pixel value at coordinates (x, y); t represents the timestamp; p denotes the event's polarity; and $T \in \left[{}^{(i-1)\Delta t}/_{n}, {}^{i\Delta t}/_{n} \right]$ specifies the timeframe of events consolidated into a single frame.

While the SBT approach offers substantial utility, alternative event data representation techniques exist, some of which closely resemble our encoding method. Although not integrated into our model, these techniques occasionally offer superior performance in specific contexts. For instance, in scenarios characterized by infrequent events during the consolidation interval, the SBT might yield a frame with notably sparse values. In such cases, the Stacking Based on the number of Events (SBE) method, as introduced by Wang *et al.* [41], emerges as a more apt choice. The SBE methodology builds a frame using a predetermined number of events, providing an effective solution to the challenge of sparse event representation. To comprehensively evaluate the influence of various event encoding strategies on object detection tasks, we have compared multiple encoding modalities in the ablation studies detailed in Section IV-C3.

B. Network Design

In response to the challenges of event-based object detection, we propose the SpikeFPN — a spiking neural network built upon the threshold-adaptive spiking unit mechanism with the feature pyramid network architecture as the core principle. The foundational neuron unit, characterized by self-adaptive and spiking attributes, flexibly adapts to intricate scenarios. Coupled with the feature pyramid network module, it provides a robust architecture. A multi-head prediction mechanism layered on top further enhances detection accuracy. This section discusses the overall network architecture, encapsulating the design intricacies of both the backbone and the feature pyramid module. Subsequently, it illustrates the behavior and propagation mechanisms of the spiking neuron unit incorporated within our approach.

1) Network Architecture: The network architecture adopts a feature pyramid module which is based on a downsampling scheme, employing foundational cell-based units to construct the encoding backbone. Within such a design, the primary focus shares similarities with the thoughts of YOLOv3 [74], which draws on the pyramid feature map handling process, with small-sized feature maps being employed to detect large-sized objects while large-sized feature maps detect small-sized objects. Spiking feature maps, derived from the concluding three stages of the backbone, combine to create the spiking feature pyramid — a focal element of the primary network design as depicted in Fig. 1. This component plays a pivotal role in feature extraction. A comprehensive explanation of the backbone's output feature map, in conjunction with the entire network, can be found in Table I. The first two spiking stem

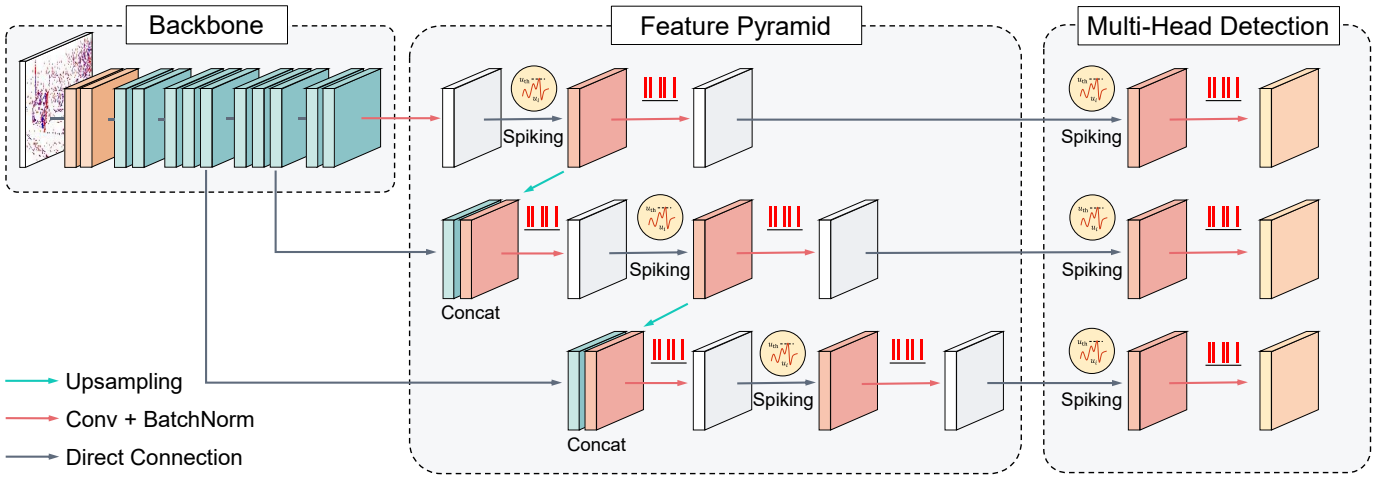

Fig. 1. Overview of the proposed spiking feature pyramid network. The design encompasses an encoder backbone facilitated by a multi-stage spiking network. Different stages of the backbone contribute to the integrated spiking feature pyramid. Subsequent to this, a multi-head prediction module processes the feature pyramid's output through parallel spiking convolution layers, culminating in the generation of multiple prediction boxes. These are ultimately refined using the non-maximum suppression (NMS) method. Notably, the entire network operates on spike-based computation, allowing the advantage of multiplication-free inference.

TABLE I
DETAILED ARCHITECTURE SETTINGS OF THE PROPOSED SPIKING FEATURE PYRAMID NETWORK. THE ILLUSTRATION ENCOMPASSES THE ENCODER BACKBONE, THE FEATURE PYRAMID, AND THE MULTI-HEAD PREDICTION MODULE. HERE, C REPRESENTS THE NUMBER OF CLASSES, WHILE K SIGNIFIES THE NUMBER OF ANCHORS.

Module	**Layer**	**Output Feature Map** $c \times h \times w$
Backbone	Stem 0	$48 \times 128 \times 128$
	Stem 1	$96 \times 64 \times 64$
	Cell 0	$96 \times 64 \times 64$
	Cell 1	$96 \times 64 \times 64$
	Cell 2	$192 \times 32 \times 32$
	Cell 3	$192 \times 32 \times 32$
	Cell 4	$192 \times 32 \times 32$
	Cell 5	$384 \times 16 \times 16$
	Cell 6	$384 \times 16 \times 16$
	Cell 7	$384 \times 16 \times 16$
	Cell 8	$768 \times 8 \times 8$
	Cell 9	$768 \times 8 \times 8$
Feature Pyramid	Cell 4 $\rightarrow$ p1	$96 \times 32 \times 32$
	Cell 7 $\rightarrow$ p2	$192 \times 16 \times 16$
	Cell 9 $\rightarrow$ p3	$384 \times 8 \times 8$
Multi-Head Prediction	p1 $\rightarrow$ d1	$K \times (C+5) \times 32 \times 32$
	p2 $\rightarrow$ d2	$K \times (C+5) \times 16 \times 16$
	p3 $\rightarrow$ d3	$K \times (C+5) \times 8 \times 8$

layers of the architecture facilitate double downsampling of the feature map. This is succeeded by 10 spiking cells, which together forge a four-stage downsampling hierarchy. Outputs from the terminal cells across stages 2, 3, and 4 are fed into the feature pyramid module. The full architecture achieves a maximum downsampling ratio of 32. In the following, we further explain the design elements of the backbone, complemented by the subsequent spiking feature pyramid network.

As illustrated in Fig. 2, the encoder backbone adopts a multi-stage downsampling strategy. This architecture consists of two initial spiking stem layers, followed by a sequence of ten spiking cells. Through the downsampling process, the spatial resolution of the feature map is reduced by half, while the channel size is doubled. Each spiking stem layer comprises a convolution layer, a batch normalization layer (which can be integrated into convolution weights during test inference, as suggested by [81]), and a spiking activation function. These layers are specially designed to propel initial feature extraction and channel modulation. The local design of the architecture is based on the inspiration of SpikeDHS [62]. We adopt tailored cells and nodes to compose this architecture. In SpikeFPN, a cell is defined as a repeating unit with an internally directed acyclic graph structure consisting of nodes, each of which receives the results of the previous two cells (if any) as input; a node is a pre-defined block of spiking neural network which is the basic component of a cell, the specific architecture of the node and cell is represented as in the underside of Fig. 2. Each spiking cell is composed of three spiking nodes, with this arrangement being repeated across multiple layers. The initial pair of nodes acquire inputs from the preceding two cells, while the third node receives input from its immediate two predecessors. The feature maps across all nodes maintain a consistent size. Their respective outputs are converged, forming the collective output of the cell. Within each node, operations stemming from various levels aggregate at the membrane potential level before the spiking activation generation. The interconnections within the spiking cell are heuristically determined to harmonize inference speed with accuracy.

After processing through the backbone module, fundamental features are discerned. However, subsequent operations are also necessary to achieve the anticipated results. The lower spatial resolution features from the pyramid are upsampled via the nearest interpolation method, which doubles both width and height dimensions. These are subsequently concatenated with feature maps (of congruent spatial dimensions) generated from the backbone. Within the overall network architecture,

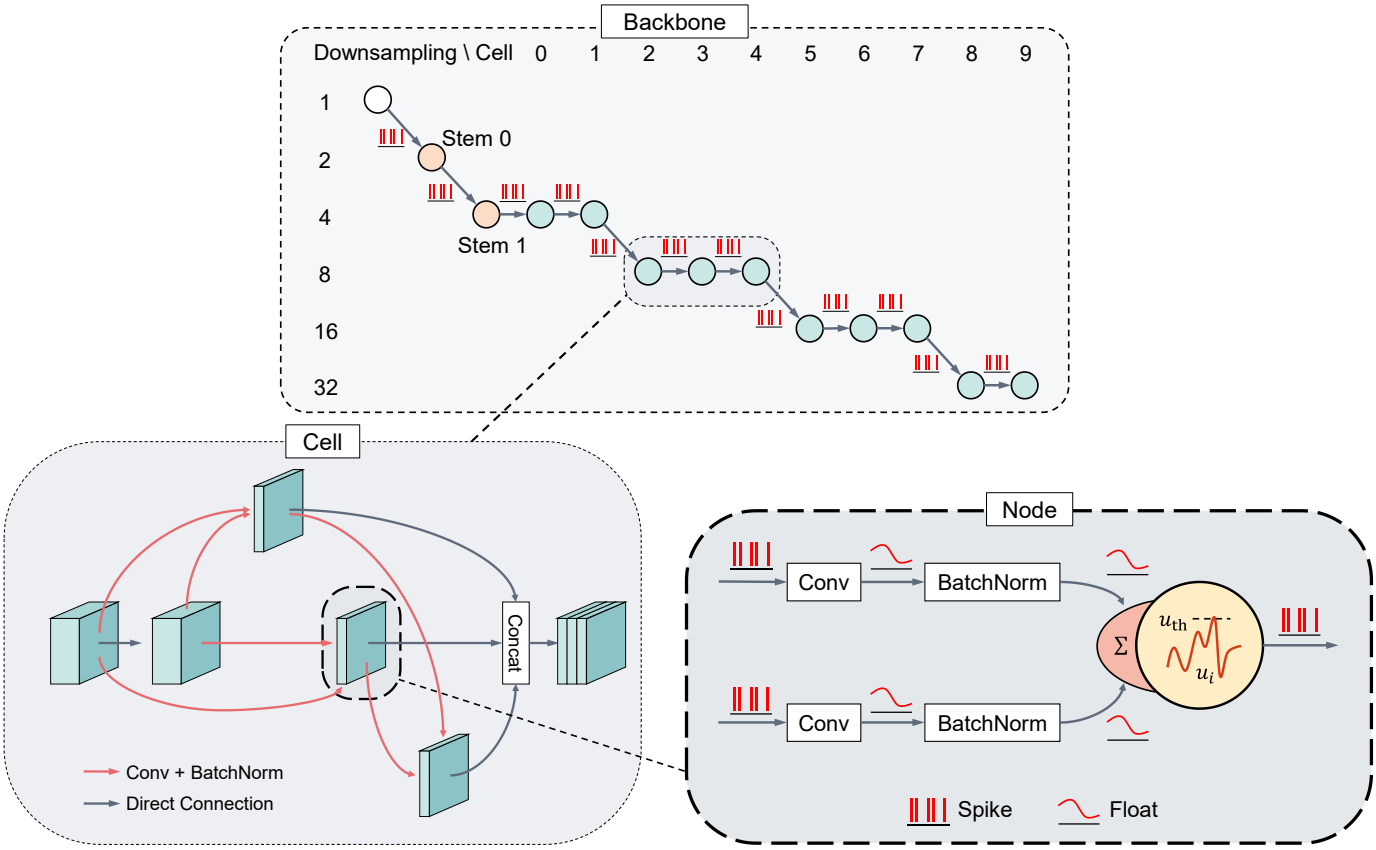

Fig. 2. Overview of the proposed encoder backbone. The structure adopts a multi-stage downsampling scheme, incorporating multiple cells interlinked via directed acyclic graphs. The vertical annotations on the left delineate the downsampling scale associated with each cell structure, while the numbers atop the left side correspond to the cell's subscript, aligning with the detailed architecture showcased in Table I. Two spiking convolution layers, termed as stem layers, facilitate initial channel variations. The unique cell connection topology recurs across different layers. Operations within each cell, executed at the node level, accumulate on the membrane potential tier, culminating in the form of spikes.

the feature pyramid module enhances the features, synergizing the backbone components, thereby promoting good adaptability for intricate tasks at hand. These enhanced features are processed by a 1×1 convolution layer, followed by a batch normalization layer and a spike activation function, constituting the succeeding module of the feature pyramid. The spiking feature pyramid then interfaces with a multi-head prediction module through convolution layers, batch normalization layers, and spike activation functions. The resultant features are processed by a 1×1 convolution, yielding floating-point outcomes with the shape of $K \times (C+5) \times h \times w$. Here, h and w represent the height and width of the corresponding feature maps, respectively; K, the number of anchors, is set to 3; C denotes the number of classes; and the additional 5 accounts for four bounding box coordinate predictions and one confidence prediction. The method of non-maximum suppression (NMS) [82], [83] is employed to finalize the bounding boxes.

2) Neural Behavior and Adaptation: In the realm of SNNs, the choice of the basic neuron unit emerges as a pivotal aspect of network design. In event-based real-time scenarios, adaptability to the conveyed information is a core concern. As such, we adopt an adaptively tunable spiking neuron model. The iterative Leaky Integrate-and-Fire (LIF) neuron [14] is adopted as the fundamental unit, which is then augmented with a firing threshold-adaptive mechanism, enhancing its representational capabilities for event data, as discussed below.

Leveraging its inherent temporal dynamics and sparse activation, the LIF neuron model's propagation pattern can be articulated as:

$$u_i^{(t,n)} = \tau \cdot u_i^{(t-1,n)} \cdot \left(1 - y_i^{(t-1,n)}\right) + I_i^{(t,n)}, \tag{2a}$$

$$I_i^{(t,n)} = \sum_j w_{ji} \cdot y_j^{(t,n-1)}, \tag{2b}$$

with the spike generation process formulated with:

$$y_i^{(t,n)} = H\left(u_i^{(t,n)} - u_{\text{th}}\right) = \begin{cases} 1, & \text{if } u_i^{(t,n)} \geq u_{\text{th}}, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

where the $u_i^{(t,n)}$ represents the membrane potential of the neuron i of the n-th layer at time t, τ is a constant which represents the membrane time attenuation factor consequently configures the membrane leakage of the neuron, $I_i^{(t,n)}$ is the influx currency, which is essentially the weighted sum of the spiking activation $y_j^{(t,n-1)}$ generated from the connected neurons from the previous layer $n-1$. The spiking activation $y_i^{(t,n)}$ is defined by a Heaviside function $H(\cdot)$ which is calculated to be 1 only when $u_i^{(t,n)}$ reaches a membrane threshold u_{th}, otherwise remains 0.

Drawing inspiration from the adaptive behaviors of neurons in the neocortex [84], [85], our design incorporates the self-adaptive mechanism into the LIF spiking neuron. This self-adaptive mechanism has demonstrated adaptability across various experimental setups and applications. Liu *et al.* [86] and Chen *et al.* [87] have enhanced spiking neurons with adaptive properties, verifying the effectiveness of these adaptations from both biological and application perspectives. Furthermore, Yang *et al.* [88], Chen *et al.* [89], and Wei *et al.* [90] developed dynamic mechanisms for adjusting the membrane threshold over time. Meanwhile, Sun *et al.* [91] explored making the membrane threshold a trainable parameter, synergistically training it with synaptic weights to enhance model performance.

Our enhancement module incorporates an adaptive threshold mechanism previously utilized in recurrent SNNs for temporal sequence learning [92], [93], enabling longer memory retention. Unlike previous models, our approach maintains a mechanism where the membrane threshold is influenced by outputs while simplifying the overall model. This provides specific feedback properties that differ from those where the threshold changes only over time. To maximize this module's effectiveness, we made its hyper-parameters trainable, allowing it to actively participate in the training and modulation processes. We strategically apply this module at the first layer of SpikeFPN to balance flexibility and stability. Specifically, the adaptive threshold mechanism's propagation pattern can be described as:

$$y^{(t)} = H\big(u^{(t)} - A_{\text{th}}^{(t)}\big), \tag{4a}$$

$$A_{\text{th}}^{(t)} = u_{\text{th}} + \beta \cdot a^{(t)}, \tag{4b}$$

$$a^{(t)} = \tau_a \cdot a^{(t-1)} + y^{(t-1)}, \tag{4c}$$

where $H(\cdot)$ is the Heaviside step function. $A_{\text{th}}^{(t)}$ signifies the modifiable threshold at time t. Meanwhile, $a^{(t)}$ represents the evolving threshold increment, which is modulated by the neuron's spiking history. Here, β is a scaling coefficient, and τ_a is a trainable time constant for a.

Crucially, the adjustable threshold A_{th} dynamically influences the spiking rate. When faced with intensive input, this threshold rises, thereby inhibiting the neuron's firing propensity. On the other hand, sparse input leads to a decrease in this threshold, making the neuron more prone to firing. By examining the limits of this adaptive mechanism under various input scenarios, we can discern its bounds. Under conditions where the neuron remains consistently inactive, as t approaches infinity, $a^{(t)}$ converges to 0. This implies that the lowest bound for $A_{\text{th}}^{(t)}$ is the base threshold u_{th}. However, in a contrasting scenario where the neuron is incessantly active (with initial conditions $y^{(0)} = 1$ and $a^{(0)} = 0$), the evolution of $a^{(t)}$ can be portrayed as:

$$a^{(t)} = \sum_{i=1}^{t} \tau_a^{(i-1)}, \tag{5}$$

and its upper limit, as t grows indefinitely, is determined to be $1/(1-\tau_a)$. Hence, the adaptive threshold A_{th} spans from u_{th} to $u_{\text{th}} + \beta/(1-\tau_a)$. This event-driven adaptivity augments the network's resilience and stability, especially in the face of swift alterations in event density.

C. Network Training

To capitalize on the temporal dynamics of spiking neurons, the initial obstacle is the intricacy of training SNNs. This challenge stems from the discontinuous nature of spike generation, such that the absence of a gradient renders backpropagation unfeasible.

Early strides in the application of deep SNNs to machine learning tasks were made by transferring parameters from ANN-trained models to corresponding SNNs [33], [34]. However, this method is confined to feed-forward architectures devoid of recurrency, making it suitable only for tasks with limited temporal dependencies. To overcome this limitation, we embrace the surrogate gradient-based direct training method. This approach retains the discontinuous activation mechanism for forward propagation while replacing the derivative of the spike-generating function (see Eq. 3) with an approximation. This substitution enables backpropagation to proceed effectively. During this training phase, temporal neural dynamics, as detailed in Eq. (2), are meticulously modeled, and the resulting errors are backpropagated through time. Consequently, surrogate gradient approximations not only facilitate the training of spiking neurons while considering their temporal intricacies but are also inherently apt for tasks with pronounced temporal dependencies, like the continuous object detection challenge we address. As a stand-in for the gradient of the spiking function, we utilize the Dspike surrogate function [18], given by:

$$\mathrm{Dspike}(u) = a \cdot \tanh\big(b \cdot (u - c)\big) + d, \quad 0 \le u \le 1, \tag{6a}$$

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}, \tag{6b}$$

where u denotes the membrane potential. The parameters (a, b, c, d) modify the intrinsic hyperbolic tangent function, $\tanh$, ensuring its output remains bounded within $[0, 1]$, while varying in shape. The parameter b, termed the temperature factor, regulates the smoothness of the function. In our chosen variant of the Dspike function, adjustments ensure that $\mathrm{Dspike}(0) = 0$ and $\mathrm{Dspike}(1) = 1$. By selecting different hyper-parameters, the Dspike function can represent a broader range of surrogate gradients. As it encompasses a family of more stable spike functions, the estimation of gradients remains within a desirable range. During model design, only minimal adjustments to the hyper-parameters are required to achieve favorable outcomes. This flexibility provides valuable insights for adapting the model to various scenarios. With a wide array of surrogate functions at our disposal, we can effectively cater to diverse training needs.

IV. Experiments

This section delves into a comprehensive examination of our proposed SpikeFPN, underscoring its advantages through rigorous experimentation. Initially, we detail the training regimen of SpikeFPN on the GEN1 Automotive Detection (GAD)

dataset [48], encompassing facets such as data preparation, hyper-parameter optimization, and comparative performance analysis. Subsequently, validation experiments of SpikeFPN on the N-CARS dataset [94] are provided to demonstrate the generalizability of the model on different automotive tasks. Afterwards, ablation studies elucidate the design decisions underpinning our model — highlighting the nuances of event data preprocessing and architectural considerations. Finally, capitalizing on the inherent sparse propagation properties of SNNs, we present a comparative analysis of SpikeFPN's computational efficiency and energy consumption.

A. Experiment on GAD

A comprehensive procedure encompassing data preparation, training, and testing has been executed on the GAD dataset, yielding noteworthy results. This section delineates the entire experimental procedure, beginning with the details of the input dataset and its encoding settings. This is followed by a discussion on the requisite hyper-parameters and performance evaluation metrics used during the training and testing phases, culminating in an evaluation of the outcomes.

1) Input Data Representation: The GAD dataset, obtained using a Prophesee GEN1 ATIS sensor, is an expansive event-based automotive object detection collection. It consists of over 39 hours of automotive recordings (with a resolution of 304×240). Alongside approximately 255,000 manually labeled bounding boxes at 1 Hz, 2 Hz, and 4 Hz denoting cars and pedestrians, the data is divided into training, validation, and test sets. The recordings are segmented into 60-second intervals, with each set containing 1460, 429, and 470 videos, respectively. During this experiment's training and testing phases, the data partitioning in GAD was meticulously followed to ensure accurate representation of results.

In real-world settings, the event data from the sensor is sequential and varies in length. For a more efficient implementation, we processed the variable-length event data through fixed-length segmentation. After encoding the event data, $S \times C$ frames preceding the label were created to be fed into the network for each sample in the following training. Here, S represents the number of stacks from the SBT framing process, and C is the number of frames in each stack. This frame-compression approach corresponds to what is presented in Eq. (1). With $\Delta t = 60\,\mathrm{ms}$, $n = C = 3$, and $S = 3$, the segmented event data is transformed into tensor-form data frames, preserving the time information between frames. The minimum temporal resolution here is $T = {}^{\Delta t}/_{n} = 20\,\mathrm{ms}$.

2) Hyper-Parameters Setting: For the training phase, we employed the AdamW [95] optimizer with an initial learning rate of 0.001 and a weight decay of 0.0005. Pre-training was deemed unnecessary; hence, all models underwent direct training for 30 epochs with a batch size of 32. A warm-up policy was applied using the first epoch's learning rate, which initially starts at a nominal value and then gradually ascends to the prescribed learning rate. The LIF and ALIF neurons have a membrane time constant τ of 0.2 and a membrane threshold u_{th} of 0.3. The temperature b of the Dspike function is set at 3. For the ALIF neuron, β is 0.07, while τ_a is a learnable parameter constrained to [0.2, 0.4], initialized at 0.3. To assess the network's training error and gauge the model's performance, we opted for the mAP, setting the prediction score to 0.3 as the accuracy metric. For a more comprehensive evaluation, we employed two mAP-based metrics: mAP_{50} and $\mathrm{mAP}_{50:95}$. The former, mAP_{50}, represents the widely-accepted mean Average Precision with an overlap threshold of 0.5 [57], [58]. In contrast, $\mathrm{mAP}_{50:95}$ serves as an alternative metric, indicating the mean Average Precision across 10 IoU ranges ([.50:.05:.95]).

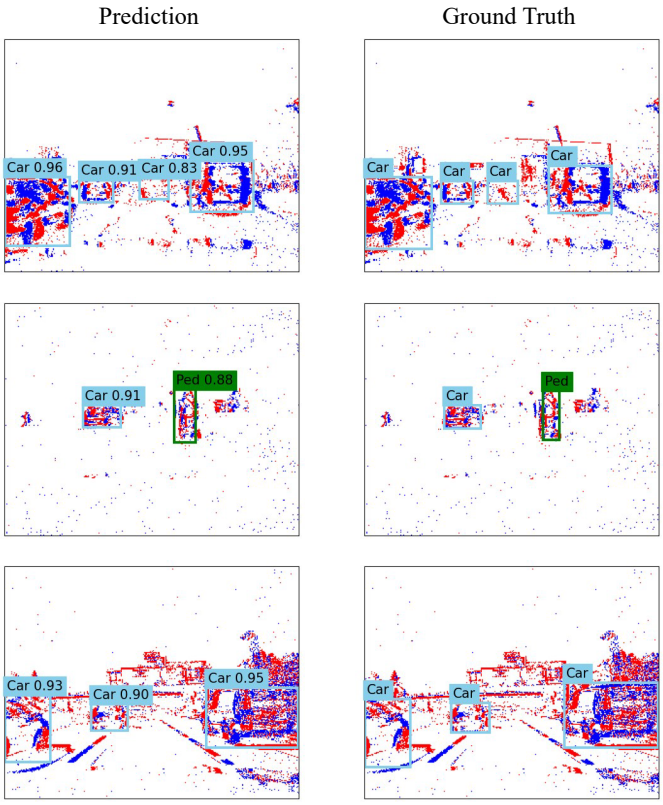

Fig. 3. The prediction results of SpikeFPN (left column) and the corresponding ground truth (right column). The "Ped" in each cell stands for the category of pedestrian.

3) Results: Throughout the training phase, the final stack output from the model serves as the prediction result. This approach ensures the preservation and accumulation of temporal information, crucial for the model's learning process and subsequent calculation of evaluation metrics. For clarity, we juxtaposed our methods against various SNN and ANN models tested on the GAD dataset. These comparative results are tabulated in Table II, with values for other models sourced from existing literature.

From an SNN perspective, our proposed SpikeFPN markedly outperforms three preceding models discussed in [20]. It exhibits an improvement ranging from 9.7% to 12.7% for mAP_{50} and between 3.4% to 7.6% for $\mathrm{mAP}_{50:95}$. Remarkably, this is achieved with fewer requisite steps to convergence. An examination of the training loss between SNNs, especially when comparing LIF and ALIF neurons in the first layer (depicted in Fig. 4), reveals that the ALIF

TABLE II
COMPARISON OF THE TWO MEAN AVERAGE PRECISION METRICS AND THE NETWORK PARAMETER QUANTITIES FOR DIFFERENT TYPES OF NETWORKS ON THE GEN1 AUTOMOTIVE DETECTION DATASET.

Model	Network Type	Params (M)	Time Steps	mAP_{50}	$\text{mAP}_{50:95}$
SparseConv [47]	ANN	133	-	0.149	-
Events-RetinaNet [9]	ANN	33	-	0.340	-
E2Vid-RetinaNet [9]	ANN	44	-	0.270	-
RED [9]	ANN	24	-	0.400	-
Gray-RetinaNet [9]	ANN	33	-	0.440	-
ASTMNet [10]	ANN	-	-	0.467	-
VGG-11 + SDD [20]	SNN	13	5	0.37*	0.174
MobileNet-64 + SSD [20]	SNN	24	5	0.35*	0.147
DenseNet121-24 + SSD [20]	SNN	8	5	0.38*	0.189
SpikeFPN (ours)	SNN	22	3	**0.477**	**0.223**

* Provided by the authors.

neuron fosters a more stable training trajectory than its LIF counterpart. Moreover, our SpikeFPN's streamlined architecture modestly outpaces the ASTMNet [10], a state-of-the-art ANN incorporating attention mechanisms and bespoke adaptive sampling event preprocessing schemes.

To assess our model's real-time capabilities, we continuously input the entire test split into the network. This process mirrors the length of steps, generating sequential labels. The average inference velocity for a single stack stands at 54 frames per second when operating on a single GPU. Such outcomes underscore both the efficacy of the SpikeFPN and the broader promise of SNNs in handling dynamic events coupled with sparse propagation. For further illustration, a selection of our model's prediction results can be perused in Fig. 3.

B. Experiment on N-CARS

In this subsection, to verify the generalizability of the model proposed in our work, the N-CARS dataset [94] is adopted to serve as an additional validation experiment for SpikeFPN. The dataset information and the experimental setup are first presented, followed by the experimental results and their corresponding analysis.

The N-CARS dataset is proposed based on an event-based vehicle recognition task, which falls into a type of binary classification. In an automotive task scenario, vehicle recognition is one of the major tasks, verifying the performance of SpikeFPN on the N-CARS dataset helps to determine the real-world performance of the model, as well as its generalizability.

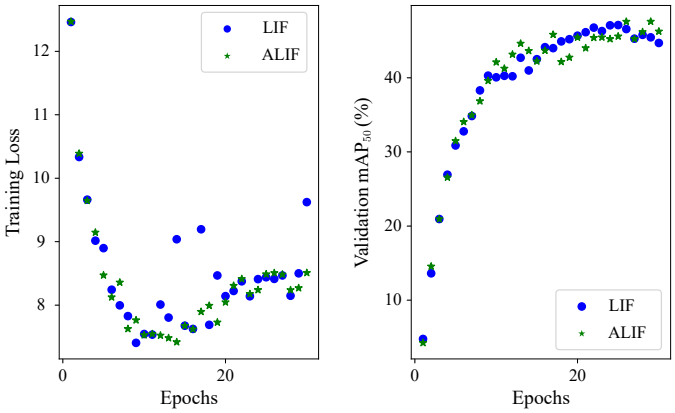

Fig. 4. Comparisons of training losses and validation mAPs of SNNs using LIF and ALIF neurons for the first layer.

TABLE III
COMPARISON OF SPIKEFPN ACCURACY ON N-CARS TEST DATASET VERSUS EXISTING BASELINES. THE "AUC" STANDS FOR "AREA UNDER THE CURVE".

Model	Accuracy	AUC Score
H-First [96]	0.561*	0.408*
HOTS [35]	0.624*	0.568*
Gabor-SNN [94]	0.789*	0.735*
HATS [94]	**0.902***	**0.945***
SpikeFPN (ours)	0.8694	0.8689

* Provided by [94].

In terms of dataset processing, 20% of the training dataset is divided for validation, the same as the GAD dataset, the SBT method is adopted as preprocessing, whereas the minimum temporal resolution $T = {}^{\Delta t}/_{n}$ is set to 10 ms. The number of frames within each stack (C) is set to 1 with varied numbers of stacks $S \in [1, 10]$ depending on the data as the N-CARS max time 100 ms for each piece of data. The learning rate is set to 0.0001 to ensure the stability of the training process.

During the training process, the best performing outcome on the validation set is selected as the final weights yielding the testing accuracy and Area Under the Curve (AUC) score as shown in Table III. As can be seen from the comparison in the table, the SpikeFPN model proposed in this work, although not specifically designed for classification tasks, still outperforms some of the baselines in the N-CARS dataset and is capable of obtaining comparable performance.

C. Ablation Study

In this subsection, we conduct ablation experiments from two aspects: the event coding paired with a basic neuron computation scheme, and the structure of the network itself. Through these, we aim to highlight the enhancements the proposed SpikeFPN brings to the table.

1) Event Input Configuration: Our goal here is to gauge the influence of varying preprocessing methods and input

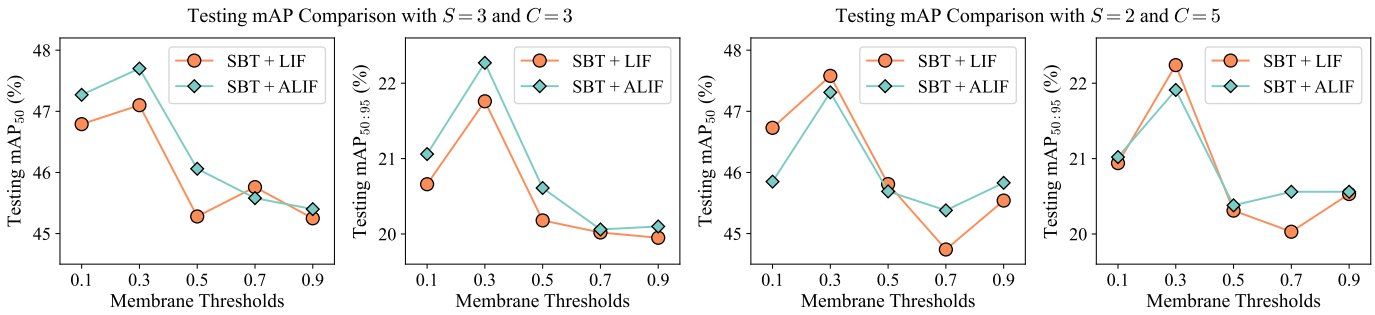

Fig. 5. The performance comparison of the "SBT + LIF" versus "SBT + ALIF" design solutions on the GEN1 Automotive Detection testing dataset with different membrane thresholds.

configurations on model performance. To this end, we juxtapose results obtained using SBE and SBT, both with LIF neurons. For each method, two input configurations, which are defined by different numbers of stacks (S) and frames within each stack (C), are employed: $S = 3$, $C = 3$ and $S = 2$, $C = 5$. With the SBT method, events within 20 ms are housed in each frame, while the SBE method's frame encapsulates 5000 events. Additionally, we introduce the Adaptive-LIF (ALIF) neuron to the initial layer in the SBT process, facilitating a comparison of neuron types across various input configurations. The findings are tabulated in Table IV.

TABLE IV
RESULTS WITH DIFFERENT INPUT CONFIGURATIONS. S AND C REPRESENT THE NUMBER OF STACKS AND THE NUMBER OF FRAMES IN EACH STACK RESPECTIVELY.

Method	S	C	**mAP**$_{50}$	**mAP**$_{50:95}$
SBE + LIF	2	5	0.4356	0.1970
SBE + LIF	3	3	0.4324	0.1944
SBT + LIF	2	5	0.4758	0.2224
SBT + LIF	3	3	0.4710	0.2176
SBT + ALIF	2	5	0.4731	0.2191
SBT + ALIF	3	3	**0.4770**	**0.2227**

As seen from the table IV, it is difficult to guarantee that "SBT + ALIF" is always the optimal due to the relatively complex correlation between the hyper-parameters, (e.g., at $S = 2, C = 5$), however, in our proposed design, this method is demonstrated experimentally to be the better solution in most cases. Further experimental analyses are provided as shown in Fig. 5. It can be seen that when selecting $S = 3, C = 3$, both metrics mAP$_{50}$ and mAP$_{50:95}$ are better with "SBT + ALIF" than "SBT + LIF" in most cases; when selecting $S = 2, C = 5$, while "SBT + ALIF" performs slightly worse in the mAP$_{50}$ metric, it also shows better performance on mAP$_{50:95}$ in the majority of situations.

Preliminary observations indicate that the LIF neuron, when coupled with the SBE process, does not fare as well in terms of accuracy compared to its performance with the SBT process. Interestingly, the configurations $S = 3, C = 3$ underperform relative to $S = 2, C = 5$ for both SBE and SBT. However, when SBT is paired with the ALIF neuron, the performance is optimized at $S = 3, C = 3$. In essence, the SBT process outperforms SBE in our model setup, and the ALIF neuron, when presented with a larger stack number, better modulates its threshold in the SBT process.

TABLE V
RESULTS WITH DIFFERENT HYPERPARAMETER SETTINGS OF THE ADAPTIVE THRESHOLD MECHANISM.

State of τ_a	**Initial τ_a**	β	**mAP**$_{50}$	**mAP**$_{50:95}$
non-trainable	0.3	0.07	0.470	0.219
trainable	0.3	0.09	0.471	0.220
trainable	0.3	0.05	0.473	0.217
trainable	0.3	0.07	**0.477**	**0.223**

2) ALIF Performance: We further delve into the sensitivity of the adaptive threshold in ALIF concerning hyper-parameters and the merits of training τ_a. Our findings, summarized in Table V, unequivocally show the robustness of performance across a spectrum of β values. Additionally, networks furnished with a trainable τ_a consistently outshine those with fixed counterparts.

According to Eq. (2), the membrane time constant, τ, dictates the proportion of the prior time step's membrane potential to retain. This attenuated accumulation addresses the sparse propagation issue. Fig. 6 depicts a sequence of input event density compared to the first layer activation with different τ values. Layers with LIF neurons function akin to low-pass filters,

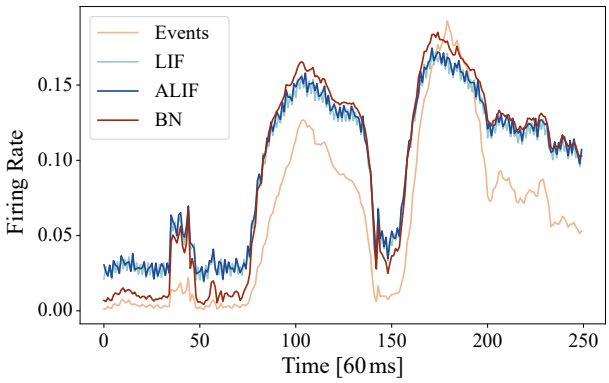

Fig. 6. Illustration of a sample sequence depicting the neuron firing rates alongside the first layer's activation for LIF $\tau = 0.2$, ALIF $\tau = 0.2$, and binary neurons (denoted as "BN" in the legend). Each point on the "Events" curve represents the average event density of an SBT stack over a duration of 60 ms. The layer activation is derived by averaging the resulting spiking feature map.

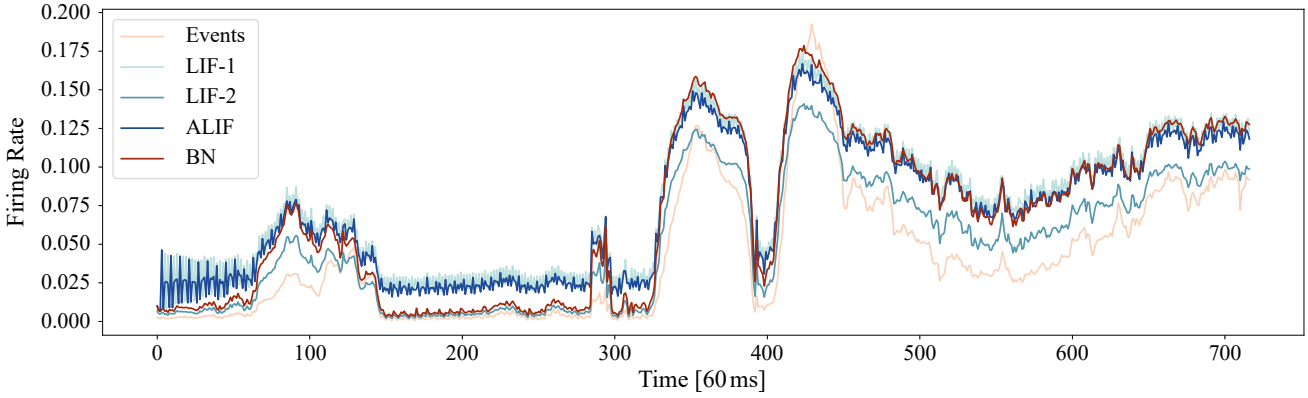

Fig. 7. Comparison of event density and layer firing rate across LIF, ALIF, and binary neurons (denoted as "BN" in the legend). LIF-1 and LIF-2 denote LIF neurons with threshold values set to 0.3 and 0.4, respectively. Each data point on the "Events" curve represents the mean event density of an SBT stack spanning 60 ms. Layer activation is determined by averaging the spiking feature map.

especially when event density experiences sharp variations. In contrast, layers with binary neurons (where $\tau = 0$) tend to mirror their input closely, displaying synchronous changes. An exploration into the effect of the membrane time constant was undertaken, testing τ values from 0 to 1 at 0.1 intervals. With a set random seed for LIF neurons, mAP_{50} values at τ of 0, 0.1, 0.2, and 0.3 were 0.4252, 0.4455, 0.4710, and 0.4696, respectively. This emphasizes the membrane potential constant's impact on overall accuracy. Retaining prior time step data, attenuated during temporal accumulation, proves beneficial. To corroborate the findings, additional experiments with three random seeds were executed for varying τ values. The resultant mAP_{50} values were averaged and presented in Fig. 8, underscoring the advantages of self-adaptive thresholds during event encoding.

When using different random seeds, the network with the first layer utilizing ALIF neurons consistently surpassed networks with LIF neurons at 0.3 and 0.4 thresholds, as detailed in Section IV-A. This accentuates the adaptive threshold's capability in managing dynamic events. In the context of firing rate, the ALIF neuron leads to decreased layer firing rate, as seen in Fig. 9, emphasizing the merits of self-adjusting thresholds in event encoding.

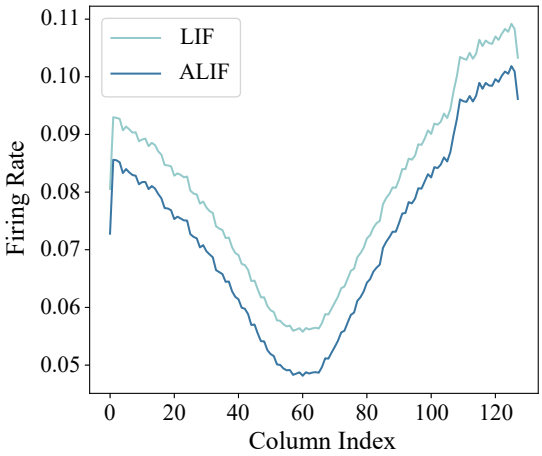

Fig. 9. Averaged column firing rate of the first layer on the GAD test set for LIF and ALIF neurons. Data was pooled from four experiments, each initiated with a different random seed. The mean mAP_{50} values for LIF and ALIF neurons stand at 0.4662 ± 0.0029 and 0.4738 ± 0.0034, respectively. The V-shaped pattern reflects fewer events from distant objects, particularly at the valley.

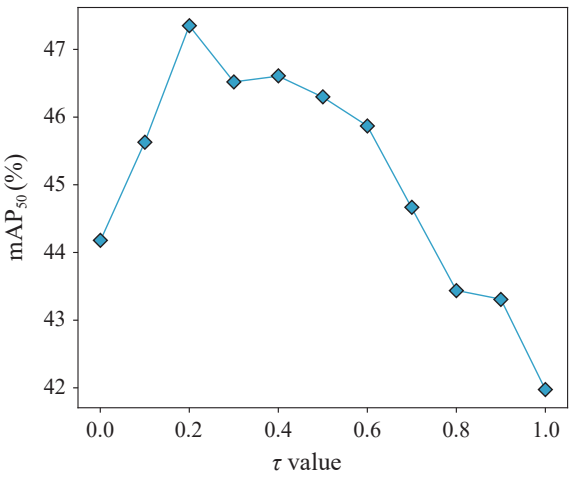

Fig. 8. A graph detailing the mean average precision for LIF neurons at varying τ values. The results, averaged across three distinct experiments with varied random seeds, emphasize the influence of the membrane time constant on performance.

In pursuit of optimal neuron placement for the adjustable threshold, we compared the performance of models employing LIF and adaptive threshold LIF (ALIF) neurons. Experiments were conducted using ALIF neurons across the entire network and exclusively on the first layer, with subsequent layers using LIF neurons. During these trials, we treated τ_a as a trainable

TABLE VI
RESULTS ON THE GAD DATASET WHEN APPLYING DIFFERENT NEURON MODELS ON THE FIRST AND ALL LAYERS OF THE NETWORK.

Neuron	Layer	Threshold	mAP_{50}
LIF	All/First	0.3	0.470 ± 0.001
LIF	First	0.4	0.470
ALIF	All	0.3	0.400
ALIF	First	0.3	**0.476 ± 0.002**

variable, consistent across layers. Outcomes are summarized in Table VI. However, integrating ALIF neurons throughout the network led to subpar outcomes. Over-reliance on adjustable thresholds might infuse excessive flexibility, thereby hampering training precision. For clarity, Fig. 7 presents an input event stream sequence by event density, juxtaposed with the first layer's activation using distinct neuron models. The LIF layer with $u_{\text{th}} = 0.4$ exhibits the least activation, whereas the binary neuron layer with $u_{\text{th}} = 0.3$ shows significant fluctuations. In comparison, both LIF and ALIF layers provide a moderating effect over input variations. The ALIF layer's activation resides between the LIF layers' threshold bounds, attributed to the adaptive threshold properties, emphasizing ALIF's potential in decreasing layer firing rate and enhancing accuracy.

TABLE VII
DETAILED ARCHITECTURE OF THE MANUALLY CONSTRUCTED SPIKE-BASED RESNET, INCLUDING THE ENCODER BACKBONE OF RESNET-18, THE FEATURE PYRAMID AND THE MULTI-HEAD PREDICTION MODULE. C DENOTES THE NUMBER OF CLASSES AND K DENOTES THE NUMBER OF ANCHORS.

Module	**Layer**	**Output Feature Map** $c \times h \times w$
Backbone	Stem 0	$80 \times 128 \times 128$
	Stem 1	$80 \times 64 \times 64$
	Layer 1-1	$80 \times 64 \times 64$
	Layer 1-2	$80 \times 64 \times 64$
	Layer 2-1	$160 \times 32 \times 32$
	Layer 2-2	$160 \times 32 \times 32$
	Layer 3-1	$320 \times 16 \times 16$
	Layer 3-2	$320 \times 16 \times 16$
	Layer 4-1	$640 \times 8 \times 8$
	Layer 4-2	$640 \times 8 \times 8$
Feature Pyramid	Layer 2-2 $\rightarrow$ p1	$80 \times 32 \times 32$
	Layer 3-2 $\rightarrow$ p2	$160 \times 16 \times 16$
	Layer 4-2 $\rightarrow$ p3	$320 \times 8 \times 8$
Multi-Head Prediction	p1 $\rightarrow$ d1	$K \times (C+5) \times 32 \times 32$
	p2 $\rightarrow$ d2	$K \times (C+5) \times 16 \times 16$
	p3 $\rightarrow$ d3	$K \times (C+5) \times 8 \times 8$

3) Spiking ResNet: In this ablation study segment, we designed a spike-based ResNet-18 encoder backbone. Distinctive features include variable initial channel sizes, incorporation of the feature pyramid, a multi-head prediction module, and substitution of all activation functions with the Heaviside function $H(\cdot)$, referenced in Eq. (3). This network, adhering to a 4-stage downsampling blueprint, adopts its residual block from [19], [97]. Outputs from the last blocks of stages 2, 3, and 4 are relayed to the feature pyramid. The network's maximum downsampling ratio stands at 32. The architecture specifics, when initiated with 80 channels, are tabulated in Table VII. Assessing our encoder's design efficacy, we paralleled it with this variant, given its analogous downsampling approach. Different channel sizes were evaluated to represent diverse model capacities, with findings consolidated in Table VIII. Remarkably, when juxtaposed under similar model parameters, ResNet (initialized with 80 channels) marginally underperforms against SpikeFPN. A plausible explanation could be SpikeFPN's extensive intra and inter-cell connections, fostering gradient and information flow throughout the SNN training.

TABLE VIII
RESULTS ON GAD DATASET WITH DIFFERENT BACKBONE ARCHITECTURES. THE ACRONYM "ICS" STANDS FOR THE INITIAL CHANNEL SIZE.

Architecture	**ICS**	**Model Size** (M)	**mAP**$_{50}$	**mAP**$_{50:95}$
ResNet18	64	13.34	0.4163	0.1799
ResNet18	80	20.82	0.4208	0.1896
ResNet18	96	29.97	0.4429	0.1992
SpikeFPN (ours)	48	21.63	**0.4770**	**0.2227**

D. Network Firing Rate and Computation Cost

SNNs inherently harness the potential of spike-based sparse computations and eschew multiplicative inferences, ensuring a pronounced computational edge over their ANN counterparts, which are predicated on dense matrix multiplication. To elucidate this, we compared SpikeFPN's computational efficiency against existing ANN and SNN benchmarks.

As illustrated in Fig. 10, SpikeFPN is characterized by varied layer firing rates across layers. Comprehensive firing rate metrics and computational costs are detailed in Table IX. Evidently, our SpikeFPN outshines contemporaneous spiking neural networks in automotive object detection tasks, achieving lower firing rates and fewer operations.

Adopting methodologies of energy consumption calculation from [16], [62], SNN addition operations are quantified using $s \times T \times A$, where s signifies the average of the neuron firing rates of all time steps, T represents the time step, and A accounts for additions per iteration. Energy estimations are predicated on [98]'s examination of 45 nm CMOS technology, as adopted in works like [16], [18], [62]. Take results from [16], [99] as the reference values, SNN addition operations cost 0.9 pJ, whereas ANN multiply-accumulate (MAC) operations demand 4.6 pJ. We benchmarked against Events-RetinaNet [9] given its accessible codebase, facilitating a balanced comparison. Notably, compared to Events-RetinaNet, SpikeFPN demonstrates a nearly 6-fold reduction in operations and a 33-fold decrease in energy consumption. Such frugality underscores the SNN's potential in energy-efficient, event-driven vision tasks.

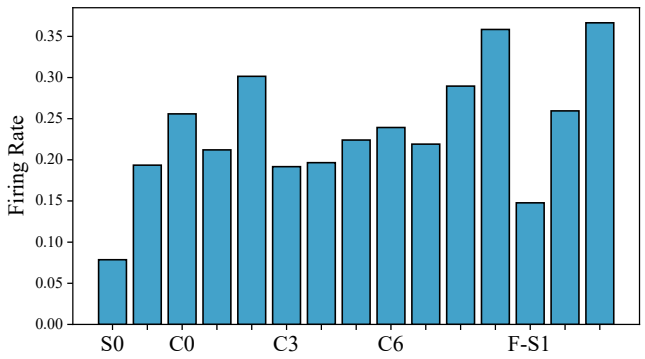

Fig. 10. The layer firing rates of SpikeFPN. The network exhibits an overall sparse activity with varying degrees across layers, with the lowest firing rate at the first layer due to highly sparse input event streams.

TABLE IX
COMPARISON OF OPERATION NUMBER, FIRING RATE AND ESTIMATED ENERGY COST. "#OP." DENOTES THE NUMBER OF OPERATIONS, WHICH IS ACCUMULATED IF THE MODEL IS AN SNN AND MULTIPLY-ACCUMULATE IF THE MODEL IS AN ANN.

Model	**#OP.** (G)	**Firing Rate**	**Energy** (mJ)
Events-RetinaNet (ANN)	18.73	-	86.16
VGG-11 + SDD	12.30	22.22%	11.07
MobileNet-64 + SSD	6.39	29.44%	5.74
DenseNet121-24 + SSD	4.33	37.20%	3.90
SpikeFPN (ours)	**2.83**	**19.10%**	**2.55**

V. CONCLUSIONS

The presented research introduces the SpikeFPN, a novel architecture that seamlessly combines a spiking feature pyramid network with a self-adaptive spiking neuron. This design specifically targets event-based automotive object detection tasks by harnessing the unique capability of the self-adaptive spiking neurons to encode sparse data effectively. The adoption of such architecture, an area of burgeoning interest, has delivered impressive outcomes in terms of both detection accuracy and computational efficiency.

Extensive experimentation using the GEN1 Automotive Detection dataset underscores the efficiency and validity of our proposed model. These experimental results depict not only a commendable mAP prediction accuracy, but also the sound reasoning behind the SpikeFPN's design. In drawing comparisons with similar network structures, inclusive of non-adaptive and various downsampling configurations, our SpikeFPN demonstrates consistent robustness across varying problem sizes. Theoretical insights further reveal the benefits of the spiking mechanism-based model, particularly highlighting its low power consumption and reduced computational demands, suggesting its potential utility for applications demanding efficient computational paradigms.

Notably, although the integration of self-adaptive spiking mechanisms within the feature pyramid network structure presents a promising avenue for advancing event-based visual tasks, there are still some limitations in the current model design. On the one hand, the current model design relies on the acquisition capability of the data sensors, which may prevent the model from achieving the desired effect when acting on other sensors, resulting in a model that does not satisfy the cross-sensor universality. On the other hand, in the current data preprocessing and model design, although the potential sparsity can be explored, the interface from data to model is still only a simple frame-pressing method, leaving a gap between the sparse data and the sparse model. The question of how to fully exploit the generality of models and how to effectively utilize the adaptability of models and data remains to be solved. Given the inherent discrete and sparse nature of events and the added advantage of preserving temporal information, aligning this data format with our unique neuron setup might pave the way for notable performance enhancements. The inherent low power consumption attributed to the sparse propagation properties of spiking neurons, when combined with our experimental outcomes, suggests a possible edge that SNNs could hold over traditional architectures in specific domains. Such potential merits further exploration and research.

REFERENCES

[1] L. Liu, W. Ouyang, X. Wang, P. W. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 261–318, 2020.

[2] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *Proc. IEEE*, vol. 111, no. 3, pp. 257–276, 2023.

[3] P. Lichtsteiner, C. Posch, and T. Delbrück, "A 128×128 120 db 15 μs latency asynchronous temporal contrast vision sensor," *IEEE J. Solid State Circuits*, vol. 43, no. 2, pp. 566–576, 2008.

[4] C. Posch, D. Matolin, and R. Wohlgenannt, "A QVGA 143 db dynamic range frame-free PWM image sensor with lossless pixel-level video compression and time-domain CDS," *IEEE J. Solid State Circuits*, vol. 46, no. 1, pp. 259–275, 2011.

[5] S. Chen, W. Tang, X. Zhang, and E. Culurciello, "A 64 × 64 pixels UWB wireless temporal-difference digital image sensor," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 20, no. 12, pp. 2232–2240, 2012.

[6] T. Serrano-Gotarredona and B. Linares-Barranco, "A 128×128 1.5% contrast sensitivity 0.9% FPN 3 μs latency 4 mw asynchronous frame-free dynamic vision sensor using transimpedance preamplifiers," *IEEE J. Solid State Circuits*, vol. 48, no. 3, pp. 827–838, 2013.

[7] C. Brandli, R. Berner, M. Yang, S. Liu, and T. Delbrück, "A 240 × 180 130 db 3 μs latency global shutter spatiotemporal vision sensor," *IEEE J. Solid State Circuits*, vol. 49, no. 10, pp. 2333–2341, 2014.

[8] B. Son, Y. Suh, S. Kim, H. Jung, J. Kim, C. Shin, K. Park, K. Lee, J. M. Park, J. Woo, Y. Roh, H. Lee, Y. M. Wang, I. A. Ovsiannikov, and H. Ryu, "4.1 A 640×480 dynamic vision sensor with a 9μm pixel and 300meps address-event representation," in *2017 IEEE International Solid-State Circuits Conference, ISSCC*. IEEE, 2017, pp. 66–67.

[9] E. Perot, P. de Tournemire, D. Nitti, J. Masci, and A. Sironi, "Learning to detect objects with a 1 megapixel event camera," in *Advances in Neural Information Processing Systems, NeurIPS*, 2020.

[10] J. Li, J. Li, L. Zhu, X. Xiang, T. Huang, and Y. Tian, "Asynchronous spatio-temporal memory network for continuous event-based object detection," *IEEE Trans. Image Process.*, vol. 31, pp. 2975–2987, 2022.

[11] S. Ghosh-Dastidar and H. Adeli, "Spiking neural networks," *Int. J. Neural Syst.*, vol. 19, no. 4, pp. 295–308, 2009.

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

[13] F. Zenke and S. Ganguli, "SuperSpike: Supervised learning in multilayer spiking neural networks," *Neural Comput.*, vol. 30, no. 6, 2018.

[14] Y. Wu, L. Deng, G. Li, J. Zhu, Y. Xie, and L. Shi, "Direct training for spiking neural networks: Faster, larger, better," in *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI*, 2019, pp. 1311–1318.

[15] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," *IEEE Signal Process. Mag.*, vol. 36, no. 6, pp. 51–63, 2019.

[16] N. Rathi and K. Roy, "DIET-SNN: A low-latency spiking neural network with direct input encoding and leakage and threshold optimization," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 34, no. 6, pp. 3174–3182, 2023.

[17] H. Zheng, Y. Wu, L. Deng, Y. Hu, and G. Li, "Going deeper with directly-trained larger spiking neural networks," in *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI*, 2021, pp. 11 062–11 070.

[18] Y. Li, Y. Guo, S. Zhang, S. Deng, Y. Hai, and S. Gu, "Differentiable spike: Rethinking gradient-descent for training spiking neural networks," in *Advances in Neural Information Processing Systems, NeurIPS*, 2021, pp. 23 426–23 439.

[19] S. Deng, Y. Li, S. Zhang, and S. Gu, "Temporal efficient training of spiking neural network via gradient re-weighting," in *The Tenth International Conference on Learning Representations, ICLR*. OpenReview.net, 2022.

[20] L. Cordone, B. Miramond, and P. Thiérion, "Object detection with spiking neural networks on automotive event data," in *International Joint Conference on Neural Networks, IJCNN*. IEEE, 2022, pp. 1–8.

[21] R. Pérez, F. Schubert, R. Rasshofer, and E. Biebl, "Deep learning radar object detection and classification for urban automotive scenarios," in *2019 Kleinheubach Conference*, 2019, pp. 1–4.

[22] M. Ulrich, S. Braun, D. Köhler, D. Niederlöhner, F. Faion, C. Gläser, and H. Blume, "Improved orientation estimation and detection with hybrid object detection networks for automotive radar," in *25th IEEE International Conference on Intelligent Transportation Systems, ITSC*. IEEE, 2022, pp. 111–117.

[23] C. Chen, J. Song, C. Peng, G. Wang, and Y. Fang, "A novel video salient object detection method via semisupervised motion quality perception," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 5, pp. 2732–2745, 2022.

[24] G. Brazil and X. Liu, "M3D-RPN: Monocular 3D region proposal network for object detection," in *2019 IEEE/CVF International Conference on Computer Vision, ICCV*. IEEE, 2019, pp. 9286–9295.

[25] Y. Gong, X. Yu, Y. Ding, X. Peng, J. Zhao, and Z. Han, "Effective fusion factor in FPN for tiny object detection," in *IEEE Winter Conference on Applications of Computer Vision, WACV*. IEEE, 2021, pp. 1159–1167.

[26] D. Liang, Q. Geng, Z. Wei, D. A. Vorontsov, E. L. Kim, M. Wei, and H. Zhou, "Anchor retouching via model interaction for robust object detection in aerial images," *IEEE Trans. Geosci. Remote. Sens.*, vol. 60, pp. 1–13, 2022.

[27] C. Chen, J. Wei, C. Peng, and H. Qin, "Depth-quality-aware salient object detection," *IEEE Trans. Image Process.*, vol. 30, pp. 2350–2363, 2021.

[28] J. J. Hagenaars, F. Paredes-Vallés, and G. de Croon, "Self-supervised learning of event-based optical flow with spiking neural networks," in *Advances in Neural Information Processing Systems, NeurIPS*, 2021, pp. 7167–7179.

[29] S. J. Kim, S. Park, B. Na, and S. Yoon, "Spiking-YOLO: Spiking neural network for energy-efficient object detection," in *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI*, 2020, pp. 11 270–11 277.

[30] T. Bu, W. Fang, J. Ding, P. Dai, Z. Yu, and T. Huang, "Optimal ANN-SNN conversion for high-accuracy and ultra-low-latency spiking neural networks," in *The Tenth International Conference on Learning Representations, ICLR*. OpenReview.net, 2022.

[31] Y. Li, S. Deng, X. Dong, R. Gong, and S. Gu, "A free lunch from ANN: Towards efficient, accurate spiking neural networks calibration," in *Proceedings of the 38th International Conference on Machine Learning, ICML*, ser. Proceedings of Machine Learning Research, vol. 139. PMLR, 2021, pp. 6316–6325.

[32] J. Ding, Z. Yu, Y. Tian, and T. Huang, "Optimal ANN-SNN conversion for fast and accurate inference in deep spiking neural networks," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI*. ijcai.org, 2021, pp. 2328–2336.

[33] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Frontiers in Neuroscience*, vol. 11, 2017.

[34] P. U. Diehl, D. Neil, J. Binas, M. Cook, S. Liu, and M. Pfeiffer, "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," in *2015 International Joint Conference on Neural Networks, IJCNN*. IEEE, 2015, pp. 1–8.

[35] X. Lagorce, G. Orchard, F. Galluppi, B. E. Shi, and R. Benosman, "HOTS: A hierarchy of event-based time-surfaces for pattern recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 7, pp. 1346–1359, 2017.

[36] A. I. Maqueda, A. Loquercio, G. Gallego, N. García, and D. Scaramuzza, "Event-based vision meets deep learning on steering prediction for self-driving cars," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. Computer Vision Foundation / IEEE Computer Society, 2018, pp. 5419–5427.

[37] D. P. Moeys, F. Corradi, E. Kerr, P. J. Vance, G. P. Das, D. Neil, D. Kerr, and T. Delbrück, "Steering a predator robot using a mixed frame/event-driven convolutional neural network," in *Second International Conference on Event-based Control, Communication, and Signal Processing, EBCCSP*. IEEE, 2016, pp. 1–8.

[38] C. Ye, A. Mitrokhin, C. Parameshwara, C. Fermüller, J. A. Yorke, and Y. Aloimonos, "Unsupervised learning of dense optical flow and depth from sparse event data," *CoRR*, vol. abs/1809.08625, 2018.

[39] A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis, "EV-FlowNet: Self-supervised optical flow estimation for event-based cameras," in *Robotics: Science and Systems XIV, Carnegie Mellon University*, 2018.

[40] A. Nguyen, T. Do, D. G. Caldwell, and N. G. Tsagarakis, "Real-time 6dof pose relocalization for event cameras with stacked spatial LSTM networks," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops*. Computer Vision Foundation / IEEE, 2019, pp. 1638–1645.

[41] L. Wang, S. M. M. I., Y. Ho, and K. Yoon, "Event-based high dynamic range image and very high frame rate video generation using conditional generative adversarial networks," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. Computer Vision Foundation / IEEE, 2019, pp. 10 081–10 090.

[42] A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis, "Unsupervised event-based learning of optical flow, depth, and egomotion," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. Computer Vision Foundation / IEEE, 2019, pp. 989–997.

[43] S. Tulyakov, F. Fleuret, M. Kiefel, P. V. Gehler, and M. Hirsch, "Learning an event sequence embedding for dense event-based deep stereo," in *2019 IEEE/CVF International Conference on Computer Vision, ICCV*. IEEE, 2019, pp. 1527–1537.

[44] M. Cannici, M. Ciccone, A. Romanoni, and M. Matteucci, "A differentiable recurrent surface for asynchronous event-based data," in *Computer Vision - ECCV*, vol. 12365. Springer, 2020, pp. 136–152.

[45] K. Zhang, K. Che, J. Zhang, J. Cheng, Z. Zhang, Q. Guo, and L. Leng, "Discrete time convolution for fast event-based stereo," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*. IEEE, 2022, pp. 8666–8676.

[46] C. Scheerlinck, N. Barnes, and R. E. Mahony, "Asynchronous spatial image convolutions for event cameras," *IEEE Robotics Autom. Lett.*, vol. 4, no. 2, pp. 816–822, 2019.

[47] N. Messikommer, D. Gehrig, A. Loquercio, and D. Scaramuzza, "Event-based asynchronous sparse convolutional networks," in *Computer Vision - ECCV*, vol. 12353. Springer, 2020, pp. 415–431.

[48] P. de Tournemire, D. Nitti, E. Perot, D. Migliore, and A. Sironi, "A large scale event-based detection dataset for automotive," *CoRR*, vol. abs/2001.08499, 2020.

[49] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura, B. Brezzo, I. Vo, S. K. Esser, R. Appuswamy, B. Taba, A. Amir, M. D. Flickner, W. P. Risk, R. Manohar, and D. S. Modha, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[50] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The SpiNNaker project," *Proc. IEEE*, vol. 102, no. 5, pp. 652–665, 2014.

[51] L. Leng, M. A. Petrovici, R. Martel, I. Bytschok, O. Breitwieser, J. Bill, J. Schemmel, and K. Meier, "Spiking neural networks as superior generative and discriminative models," *Cosyne Abstracts, Salt Lake City USA*, 2016.

[52] L. Leng, R. Martel, O. Breitwieser, I. Bytschok, W. Senn, J. Schemmel, K. Meier, and M. A. Petrovici, "Spiking neurons with short-term synaptic plasticity form superior generative networks," *Scientific reports*, vol. 8, no. 1, p. 10651, 2018.

[53] A. F. Kungl, S. Schmitt, J. Klähn, P. Müller, A. Baumbach, D. Dold, A. Kugele, E. Müller, C. Koke, M. Kleider, C. Mauch, O. Breitwieser, L. Leng, N. Gürtler, M. Güttler, D. Husmann, K. Husmann, A. Hartel, V. Karasenko, A. Grübl, J. Schemmel, K. Meier, and M. A. Petrovici, "Accelerated physical emulation of bayesian inference in spiking neural networks," *Frontiers in Neuroscience*, vol. 13, 2019.

[54] L. Leng, "Solving machine learning problems with biological principles," Ph.D. dissertation, 2020.

[55] M. Davies, A. Wild, G. Orchard, Y. Sandamirskaya, G. A. F. Guerra, P. Joshi, P. Plank, and S. R. Risbud, "Advancing neuromorphic computing with Loihi: A survey of results and outlook," *Proc. IEEE*, vol. 109, no. 5, pp. 911–934, 2021.

[56] A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy, "Going deeper in spiking neural networks: VGG and residual architectures," *CoRR*, vol. abs/1802.02627, 2018.

[57] M. Everingham, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010.

[58] T. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Computer Vision - ECCV*, vol. 8693. Springer, 2014, pp. 740–755.

[59] S. B. Shrestha and G. Orchard, "SLAYER: Spike layer error reassignment in time," in *Advances in Neural Information Processing Systems, NeurIPS*, 2018, pp. 1419–1428.

[60] W. Zhang and P. Li, "Temporal spike sequence learning via backpropagation for deep spiking neural networks," in *Advances in Neural Information Processing Systems, NeurIPS*, 2020.

[61] L. Zhu, X. Wang, Y. Chang, J. Li, T. Huang, and Y. Tian, "Event-based video reconstruction via potential-assisted spiking neural network," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*. IEEE, 2022, pp. 3584–3594.

[62] K. Che, L. Leng, K. Zhang, J. Zhang, Q. Meng, J. Cheng, Q. Guo, and J. Liao, "Differentiable hierarchical and surrogate gradient search for

spiking neural networks," in *Advances in Neural Information Processing Systems, NeurIPS*, 2022.

[63] R. Zhang, L. Leng, K. Che, H. Zhang, J. Cheng, Q. Guo, J. Liao, and R. Cheng, "Accurate and efficient event-based semantic segmentation using adaptive spiking encoder-decoder network," *CoRR*, vol. abs/2304.11857, 2023.

[64] B. Li, L. Leng, S. Shen, K. Zhang, J. Zhang, J. Liao, and R. Cheng, "Efficient deep spiking multilayer perceptrons with multiplication-free inference," *IEEE Trans. Neural Networks Learn. Syst.*, pp. 1–13, 2024.

[65] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, 2015.

[66] T. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. IEEE Computer Society, 2017, pp. 936–944.

[67] Z. Zhao, P. Zheng, S. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 30, no. 11, pp. 3212–3232, 2019.

[68] S. Huang, Z. Lu, R. Cheng, and C. He, "FaPN: Feature-aligned pyramid network for dense image prediction," in *2021 IEEE/CVF International Conference on Computer Vision, ICCV*. IEEE, 2021, pp. 844–853.

[69] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Computer Vision - ECCV*, vol. 9905. Springer, 2016, pp. 21–37.

[70] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, 2017.

[71] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," in *IEEE International Conference on Computer Vision, ICCV*. IEEE Computer Society, 2017, pp. 2980–2988.

[72] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. Computer Vision Foundation / IEEE Computer Society, 2018, pp. 6154–6162.

[73] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *IEEE International Conference on Computer Vision, ICCV*. IEEE Computer Society, 2017, pp. 2999–3007.

[74] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *CoRR*, vol. abs/1804.02767, 2018.

[75] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. Computer Vision Foundation / IEEE Computer Society, 2018, pp. 8759–8768.

[76] S. Liu, D. Huang, and Y. Wang, "Learning spatial fusion for single-shot object detection," *CoRR*, vol. abs/1911.09516, 2019.

[77] G. Ghiasi, T. Lin, and Q. V. Le, "NAS-FPN: Learning scalable feature pyramid architecture for object detection," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. Computer Vision Foundation / IEEE, 2019, pp. 7036–7045.

[78] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*. Computer Vision Foundation / IEEE, 2020, pp. 10 778–10 787.

[79] S. Qiao, L. Chen, and A. L. Yuille, "DetectoRS: Detecting objects with recursive feature pyramid and switchable atrous convolution," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. Computer Vision Foundation / IEEE, 2021, pp. 10 213–10 224.

[80] Q. Fu and H. Dong, "Spiking neural network based on multi-scale saliency fusion for breast cancer detection," *Entropy*, vol. 24, no. 11, p. 1543, 2022.

[81] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning, ICML*, vol. 37. JMLR.org, 2015, pp. 448–456.

[82] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. IEEE Computer Society, 2017, pp. 6517–6525.

[83] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. IEEE Computer Society, 2014, pp. 580–587.

[84] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

[85] N. W. Gouwens, J. Berg, D. Feng, S. A. Sorensen, H. Zeng, M. J. Hawrylycz, C. Koch, and A. Arkhipov, "Systematic generation of biophysically detailed models for diverse cortical neuron types," *Nature communications*, vol. 9, no. 1, p. 710, 2018.

[86] Y. Liu, R. Liu, J. Wang, W. Chen, Y. Wang, and C. Sun, "A cerebellum-inspired spiking neural model with adapting rate neurons," *IEEE Trans. Cogn. Dev. Syst.*, vol. 15, no. 3, pp. 1628–1638, 2023.

[87] Y. Chen, Y. Mai, R. Feng, and J. Xiao, "An adaptive threshold mechanism for accurate and efficient deep spiking convolutional neural networks," *Neurocomputing*, vol. 469, pp. 189–197, 2022.

[88] Q. Yang, M. Zhang, J. Wu, K. C. Tan, and H. Li, "LC-TTFS: Towards lossless network conversion for spiking neural networks with TTFS coding," *IEEE Trans. Cogn. Dev. Syst.*, 2023.

[89] T. Chen, L. Wang, J. Li, S. Duan, and T. Huang, "Improving spiking neural network with frequency adaptation for image classification," *IEEE Trans. Cogn. Dev. Syst.*, 2023.

[90] W. Wei, M. Zhang, H. Qu, A. Belatreche, J. Zhang, and H. Chen, "Temporal-coded spiking neural networks with dynamic firing threshold: Learning with event-driven backpropagation," in *IEEE/CVF International Conference on Computer Vision, ICCV*. IEEE, 2023, pp. 10 518–10 528.

[91] H. Sun, W. Cai, B. Yang, Y. Cui, Y. Xia, D. Yao, and D. Guo, "A synapse-threshold synergistic learning approach for spiking neural networks," *IEEE Trans. Cogn. Dev. Syst.*, vol. 16, no. 2, pp. 544–558, 2024.

[92] G. Bellec, D. Salaj, A. Subramoney, R. Legenstein, and W. Maass, "Long short-term memory and learning-to-learn in networks of spiking neurons," in *Advances in Neural Information Processing Systems, NeurIPS*, 2018, pp. 795–805.

[93] G. Bellec, F. Scherr, A. Subramoney, E. Hajek, D. Salaj, R. Legenstein, and W. Maass, "A solution to the learning dilemma for recurrent networks of spiking neurons," *Nature communications*, vol. 11, no. 1, p. 3625, 2020.

[94] A. Sironi, M. Brambilla, N. Bourdis, X. Lagorce, and R. Benosman, "HATS: Histograms of averaged time surfaces for robust event-based object classification," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. Computer Vision Foundation / IEEE Computer Society, 2018, pp. 1731–1740.

[95] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *7th International Conference on Learning Representations, ICLR*. OpenReview.net, 2019.

[96] G. Orchard, C. Meyer, R. Etienne-Cummings, C. Posch, N. V. Thakor, and R. Benosman, "HFirst: A temporal approach to object recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 10, pp. 2028–2040, 2015.

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

[98] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE International Conference on Solid-State Circuits Conference, ISSCC*. IEEE, 2014, pp. 10–14.

[99] Y. Kim, J. Chough, and P. Panda, "Beyond classification: Directly training spiking neural networks for semantic segmentation," *Neuromorph. Comput. Eng.*, vol. 2, no. 4, p. 44015, 2022.

Hu Zhang received the B.Eng. degree from Huazhong University of Science and Technology, Wuhan, China, in 2018, and the M.Eng. degree from Southern University of Science and Technology, Shenzhen, China, in 2023. He is currently working at BYD Co., Ltd. as a performance development engineer. His research interests include computer vision, deep learning, spiking neural networks and intelligent driving.

Yanchen Li received the B.Eng. degree from the China University of Petroleum (East China), Qingdao, China, in 2021, and the M.Phil. degree from the Southern University of Science and Technology, Shenzhen, China, in 2024. He is currently a Research Assistant at the Southern University of Science and Technology, Shenzhen, China. His research focuses on brain-inspired computing and machine learning systems.

Luziwei Leng (Member, IEEE) received the B.Sc. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2011, and the Master and the Doctoral degree from the Department of Physics, Heidelberg University, Germany in 2014 and 2019, respectively. He is currently a Principal Engineer at the Advanced Computing and Storage Lab of Central Research Institute, Huawei Technologies Co., Ltd. and an Industry Supervisor of the joint-master program between Huawei and the Southern University of Science and Technology, Shenzhen, China. His research focuses on brain-inspired computing and machine learning, including spiking neural networks, event-based vision, bio-inspired learning rules and evolutionary algorithms, etc.

Kaiwei Che received the B.Eng. degree from Shenzhen University, in 2020, and the M.Eng. degree from the Southern University of Science and Technology, in 2023. He is currently a Ph.D. student at Peking University. His research interests include deep learning and brain-inspired algorithms.

Qian Liu received the B.Eng. degree from the Beijing University of Technology, Beijing, China, in 2008, and the Ph.D. degree in Computer Science from the University of Manchester, Manchester, U.K., in 2018. Currently, she works at Huawei Zurich Research Lab as a senior researcher. Her research focuses on algorithm-hardware co-design on neuromorphic architecture, aiming to enable cognitive capabilities in computer systems with optimal energy efficiency.

Qinghai Guo is currently a Principal Engineer at the Advanced Computing and Storage Lab of Huawei Technologies Co., Ltd. and the Team Leader of the Brain-inspired Computing Team. He received his Doctoral degree from the Institute of Mathematical Stochastics, University of Goettingen, Germany in 2017. His research focuses on brain-inspired computing and machine learning, including spiking neural networks, biological plausible and efficient learning theories, neuromorphic computing architecture, etc.

Jianxing Liao received the Bachelor and the Master degree from Xidian University, Xi'an, China, in 1997 and 2000, respectively. He is currently an Expert at the Advanced Computing and Storage Lab of Central Research Institute, Huawei Technologies Co., Ltd. and an Industry Supervisor of the joint-master program between Huawei and the Southern University of Science and Technology, Shenzhen, China. His research focuses on brain-inspired computing and neuromorphic chips, including event-based vision, brain-computer interface, etc.

Ran Cheng (Senior Member, IEEE) received the B.Sc. degree from the Northeastern University, Shenyang, China, in 2010, and the Ph.D. degree from the University of Surrey, Guildford, U.K., in 2016. He is currently an Associate Professor with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. He is a recipient of the 2018 and 2021 IEEE Transactions on Evolutionary Computation Outstanding Paper Award, the 2019 IEEE Computational Intelligence Society (CIS) Outstanding Ph.D. Dissertation Award, and the 2020 IEEE Computational Intelligence Magazine Outstanding Paper Award. He is the founding Chair of the IEEE CIS Shenzhen Chapter. He is an Associate Editor of IEEE TEVC, IEEE TAI, IEEE TETCI, and IEEE TCDS.