Object Detection for Embedded Systems Using Tiny Spiking Neural Networks: Filtering Noise Through Visual Attention

Hugo Bulzomi
i3S, Université Côte d'Azur, IMRA Europe
Sophia Antipolis, France
bulzomi@i3s.unice.fr

Amélie Gruel, Jean Martinet
CNRS, i3S, Université Côte d'Azur
Sophia Antipolis, France
{amelie.gruel, jean.martinet}@univ-cotedazur.fr

Takeshi Fujita
AISIN Corporation
Kariya, Japan
takeshi.fujita01@aisin.co.jp

Yuta Nakano, Rémy Bendahan
IMRA Europe
Sophia-Antipolis, France
{nakano, bendahan}@imra-europe.com

Abstract

Object detection is an important task becoming increasingly common in numerous applications for embedded systems. The traditional state-of-the-art deep neural networks (DNNs) tend to be incompatible with the limitations of many of those systems: their large size and high computational cost make them hard to deploy on hardware with limited resources. Spiking Neural Networks (SNNs) have been attracting attention in recent years because of their potential as energy-efficient alternatives when implemented on specialized hardware, and their smooth integration with energy-efficient event cameras. In this paper, we present a lightweight SNN architecture for efficient object detection in embedded systems using event camera data. We show that by applying visual attention mechanisms, we can ignore most of the noise from the input and thus reduce the number of neurons and activations since additional noise-filtering layers are not needed. Our proposed SNN is 24 times smaller than a previous similar method for our input resolution and maintains similar overall detection performances, while being more robust to noise. We finally demonstrate the energy efficiency of our network during runtime with an implementation on SpiNNaker chip, showing the applicability of our approach.

1 Introduction

Since their introduction, event cameras have gained popularity thanks to the way these bio-inspired devices process visual information [7]. Rather than capturing entire frames of images at fixed intervals, event cameras produce asynchronous events: time-stamped pixel-level and independent brightness changes. Overall, the unique characteristics of the captured event-based data such as a high temporal solution, robustness to motion blur, asynchronous operation and low-power consumption are especially valuable for embedded computer vision tasks [16].

The asynchronous, binary nature of the data produced by those cameras also makes them a perfect fit for spiking neural networks (SNNs). For the same reasons event cameras saw increased interest in the past few years, neuromorphic computing with SNNs is becoming an attractive energy-efficient, low-latency option to process information. Those networks operate using time-based processing, similarly to how biological neurons communicate through spikes, and have been applied to a variety of different visual tasks such as image classification [5], gesture recognition [2], and more recently object recognition [3]. Communication through spikes also makes the output of neurons undifferentiable: derivative of the neurons cannot be computed, and regular gradient descent is thus impossible to train SNNs. Various methods for error backpropagation in SNNs have been proposed [15] [13], but these tend to suffer from slow convergence and inherit limitations similar to those found in conventional ANNs, such as the unidirectional flow of information through network layers. Another axis of research for SNN aims at using bio-inspired unsupervised learning mechanisms, such as STDP[10]. Networks produced by those methods are typically smaller in size, and are not subject to the same limitation as regular artificial neural networks.

Unlike traditional approaches that require pre-training the neural network, the methodology presented in this paper does not involve a separate training phase. Instead, synaptic weights are dynamically adjusted during inference, in a manner similar to what is described by Gruel et al. [9]. These authors introduced various attentional mechanisms, enabling their network to focus solely on specific areas of the input. By preprocessing their data with this attentional network, they were able to maintain high classifier accuracy despite a drastically reduced number of events in

their samples, indicating the network's ability to select crucial information. Originally proposed as a way to preprocess event data in order to only select crucial information, these attentional mechanisms have not yet been explored within the realm of object detection.

In our implementation, visual attention allows our network to focus on specific parts of the data coming from an event camera. This allows to ignore a lot of noise, which removes the need for an additional refractory layer, as described by Acharya et al. [1] in a similar use-case. Regular hardware architectures aren't well suited for the simulation of SNNs due to their inability to compute sparsely and asynchronously. Instead, different neuromorphic systems [11] should be used to better exploit SNNs's potential. We show actual time and energy consumption measurements with an implementation on a SpiNNaker neuromorphic chip [12] to prove the applicability of our approach.

2 Material and method

We implemented our network using the PyNN framework [4]. Fig. 2 shows the global architecture of our network.

2.1 Dataset description

Compared to a previous similar work [1] that used data recorded at a single traffic junction, we used multiple recordings in both interior and exterior scenes. Objects captured are pedestrians, using a Prophesee Gen 3 of resolution 640×480. Whereas some interior and simple exterior recordings involve relatively low amounts of noise, other exterior recordings in front of busy supermarkets feature large quantities of noise over the whole sensor.

The events produced by the camera are tuples in the form $e_i = (x_i, y_i, p_i, t_i)$, where x_i and y_i are the spatial coordinates of the event on the sensor, p_i is the polarity change with $p_i = \{0, 1\}$, and t_i the time of the event in μs. From this event representation, an implementation of Firenet [14] was used to generate grayscale video frames that were then annotated by hand with the bounding boxes of each pedestrian. Fig. 1 shows some examples from our dataset.

2.2 Event data processing

Before sending the events to our network, we first apply spatial funnelling in order to downscale the video resolution by a factor of 8. As presented in [8], spatial funnelling is a simple and fast downscaling method, where the spatial coordinates of incoming events are divided by a factor in order to make them fit into a smaller window of a target size. The polarities of the downscaled events are then merged and directly used as input spikes for our network, which results in a downscaled input window of size 80×60.

2.3 Input layer

The input spikes are then fed to a convolution that further downscales the input. The convolution uses a kernel of size $S \times S$ with a stride of S to create patches without overlap over the input window. In our experiments, we chose $\mathbf{S = 5}$, resulting in 16×12 individual patches. These patches of input spikes are sent to the next layer, with each patch connected to one neuron via excitatory synapses, in a manner similar to the ROI layer described by Gruel et al. [9]. It should be mentioned that no Leaky-Integrate-and-Fire (LIF) neurons have to be simulated on SpiNNaker here: we are merely making a convolution over the input window which is simply a spike source without membrane dynamics.

2.4 ROI layer

Our regions of interest (ROI) layer is composed of simple LIF neurons, whose membrane potential evolves following the differential equation Eq 1:

$$\tau_m \frac{dV}{dt} = -(V - V_{rest}) + RI \tag{1}$$

with V the membrane potential of the neuron, V_{reset} its resting potential, I the neuron's input, R the membrane resistance, and τ_m the membrane time constant governing the speed at which potential will leak. Table. 1 shows the neuron parameters used.

Each neuron is connected to a single square patch from the input, thus having a $S \times S$ receptive field over the input window. Both lateral excitation and inhibition mechanisms are used respectively in order to create smoother areas of activation and to force the layer activity on specific parts of the input. Each neuron is connected to its neighbourhood via excitatory synapses as described by Acharya et al. [1], in addition to being connected to all other neurons via exponential inhibitory connections as described by Gruel et al. [9] and by Eq 2:

$$w_{inhib} = min(\frac{e^d}{w \times h}, w_{max}) \tag{2}$$

where the weight of the inhibitory connection w_{inhib} is modulated by the Euclidean distance d between neurons on the layer. From Eq. 2, we can see that the farther apart two neurons are on the RoI layer, the stronger their inhibition towards each other. These inhibitory lateral connections limit the amount of activity in this layer: the more neurons get activated, the harder it will become for other neurons to fire. Dynamic adaptive weights are also used to further advantage parts of the input window with a lot of activity.

Type	τ_m	$V_{threshold}$	V_{reset}	V_{rest}	$\tau_{refraction}$
LIF	2.5ms	-25mV	-100mV	-65mV	4.0ms

Table 1. RoI neurons parameters.

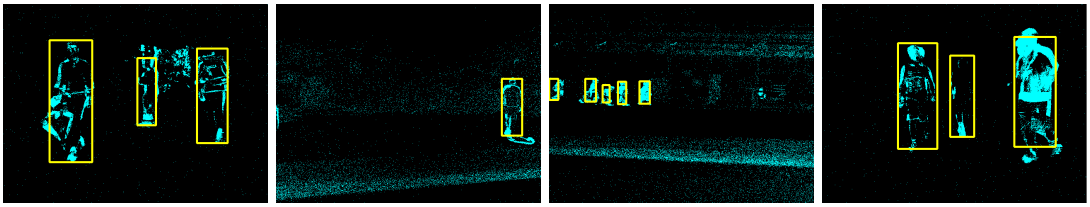

Figure 1. Examples from our dataset. Yellow squares indicate pedestrians bounding boxes.

The weights of synapses that have recently fired are increased, whereas the weights of synapses that are not used are decreased. Active areas will then trigger more lateral inhibition, and thus out-compete parts of the input with less consistent activity, such as noise. Additional noise-filtering layers such as the one described by Acharya et al [1] are then not needed.

This final layer features an extremely low neuron count (192 neurons for our input resolution) as it comes after two distinct downscaling phases (spatial funnelling and convolution).

2.5 Clustering

To detect objects on screen and create their bounding boxes, we aggregate the spikes of the final ROI layer over a fixed period of time. Since each neuron in the ROI layer corresponds to a patch of size $S \times S$ on the input, we can map their activation to the detection of objects over different parts of the input window. Patches whose corresponding neurons spiked enough times are considered to contain an object. Adjacent activated patches are finally grouped together to form bounding boxes.

3 Performance comparison

For comparison, we will focus on two other different approaches: Hynna's detection system [16] that simply applies a connected component labelling algorithm over the input events, and is meant for object detection in a similar use-case; and another SNN called RPN-SNN that follows the work of Acharya et al. [1]. This latter network is closer to our approach, but does not use the presented visual attention mechanisms, and instead features an additional refractory layer of the same size as the input window to filter out the noise. For the presented approaches, we measured precision and recall based on Intersection over Union (IoU) between predicted and ground-truth bounding boxes. One predicted bounding box can match one or several ground-truth bounding boxes as long as the IoU is over an arbitrary threshold. Tab. 2 presents overall precision and recall of all approaches over the entire dataset. While this table alone suggests that our approach performs similarly to the one described by Acharya et al. [1], these results do not differentiate performances in low and high noise conditions. To better evaluate if the attention mechanisms we presented are able to efficiently filter out noise, we evaluated each video of our dataset separately and measured the noise ratio of each video with the formula of Eq 3.

$$Noise\_ratio = \frac{\sum Events\_outside\_GT\_bounding\_boxes}{\sum Events} \tag{3}$$

The closer to 1 the noise ratio is, the moise is present on the video. Figure 3 presents performances relative to this ratio of noise. We can observe in Figure 3 that RPN-SNN [1] performs best in relatively low-noise conditions: a ratio in $[0, 0.4]$ yields better results than our approach. For higher noise values however, our networks greatly surpass RPN-SNN both in terms of precision and recall.

4 Network analysis and implementation on neuromorphic hardware

Tab. 3 presents a size comparison between our network and the RPN-SNN from Acharya et al. [1], and the mean spike count for both networks for 150ms segments of videos with an 80×60 input window (640×480 input downscaled by a factor 8). We can see our network uses 26 times fewer neurons and approximately 4.3 times more synapses while producing almost 10 times less spikes. This trade-off is explained by the fact that we do not use an additional refractory layer to filter-out noise, and instead rely on visual attention mechanisms to ignore it, which requires fewer neurons and more synapses. A lower number of neurons and spike count, as shown in Tab. 3, should theoretically lead to lower energy needs : neurons needs to have

Method	Precision	Recall	F1
Hynna's detection [16]	0.252	0.896	0.393
RPN-SNN [1]	0.513	0.752	0.610
Ours	0.550	0.739	0.631

Table 2. Recall, accuracy, and F1 score of compared methods over the whole dataset.

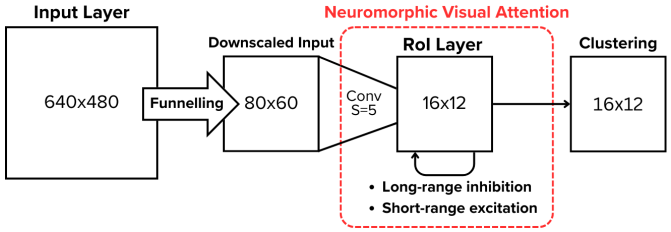

Figure 2. Network architecture.

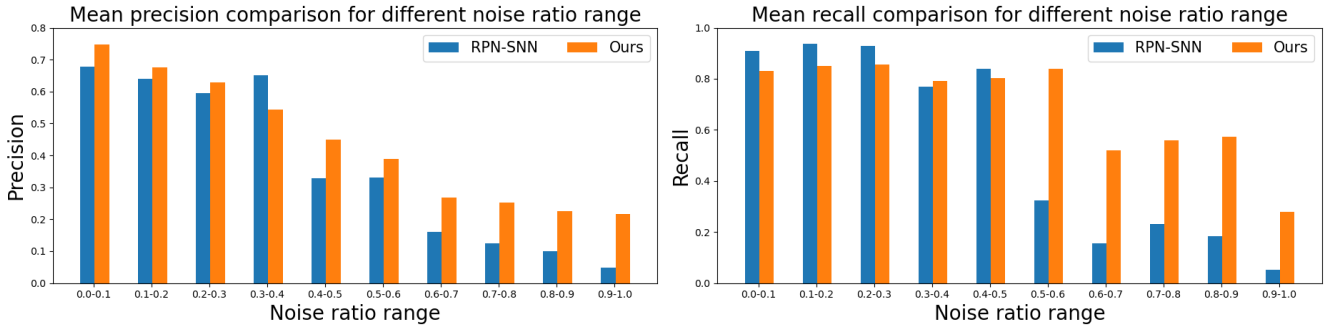

Figure 3. Mean precision-recall comparison in different noise ratio intervals.

Model	Neurons	Synapses	Mean spikes
RPN-SNN [1]	4992	**10972**	7092.6
Ours	**192**	47644	**750.5**

Table 3. Size and mean spike counts comparison for an 640×480 input downscaled to 80×60.

their potential constantly updated whereas synapses are only used sparsely during the propagation of a spike. To validate this assumption, we will now present measurements directly taken on a neuromorphic hardware implementation of the presented networks.

We chose to implement our model on SpiNNaker [6] to better assess the cost of our approach. The very small size of our network largely allowed it to fit on the small 4-chip SpiNN-3 board which can simulate up to 18K neurons. Tab. 4 presents energy consumption measurements for the different processes of simulation for a 150ms video segment that have been slowed down by a factor 10 as the SpiNN-3 could not keep-up with the flow of events.

Process	Our network	RPN-SNN[1]
Chips during runtime (continuous cost)	**22.3J**	22.7J
Loading+mapping model (one-time cost)	49.01J	**36.76J**

Table 4. Energy used by the SpiNNaker implementation. The video have been slowed 10 folds to allow the SpiNN-3 to process every event.

It should be stressed that the values from Tab. 4 are only meant to provide approximate energy measurements for the sake of comparison. Still, from this last table we can see that though our higher number of synapses increases the cost of building and loading the model, our lower neuron count leads to a lower energy consumption by the chips during runtime.

5 Conclusion

This paper presented a neuromorphic architecture for object detection for embedded systems that leverages bio-inspired visual attention mechanisms in order to ignore the noise from the input. Using these mechanisms allowed our model to largely surpass the performances of the detection system of Hynna [16], and showed to be more robust to noise than a previous similar SNN model [1], despite the smaller size of our network. We also presented preliminary measurements on neuromorphic hardware to better assess the cost of our approach and its applicability potential. Our contribution is thus two-fold: we both propose a low-energy neuromorphic detection system with actual hardware cost assessment, and further demonstrate how newly theorized bio-inspired visual attention mechanisms can be used to improve other approaches.

Given the small size of our network, one extension of this work would be to integrate it into a larger object detection and classification architecture. A small end-to-end neuromorphic solution for this task could be beneficial to numerous embedded applications.

References

[1] J. Acharya, V. Padala, and A. Basu. Spiking neural network based region proposal networks for neuromorphic vision sensors. In *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE, 2019.

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

[3] L. Cordone, B. Miramond, and P. Thierion. Object detection with spiking neural networks on automotive event data. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2022.

[4] A. P. Davison, D. Brüderle, J. M. Eppler, J. Kremkow, E. Muller, D. Pecevski, L. Perrinet, and P. Yger. Pynn: a common interface for neuronal network simulators. *Frontiers in neuroinformatics*, page 11, 2009.

[5] P. U. Diehl and M. Cook. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in computational neuroscience*, 9:99, 2015.

[6] S. B. Furber, D. R. Lester, L. A. Plana, J. D. Garside, E. Painkras, S. Temple, and A. D. Brown. Overview of the spinnaker system architecture. *IEEE transactions on computers*, 62(12):2454–2467, 2012.

[7] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza. Event-based vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(1):154–180, 2022.

[8] A. Gruel, J. Martinet, T. Serrano-Gotarredona, and B. Linares-Barranco. Event data downscaling for embedded computer vision. In *17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP)*, 2022.

[9] A. Gruel, A. Vitale, J. Martinet, and M. Magno. Neuromorphic event-based spatio-temporal attention using adaptive mechanisms. In *2022 IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, pages 379–382. IEEE, 2022.

[10] S. Huang, C. Rozas, M. Trevino, J. Contreras, S. Yang, L. Song, T. Yoshioka, H.-K. Lee, and A. Kirkwood. Associative hebbian synaptic plasticity in primate visual cortex. *Journal of Neuroscience*, 34(22):7575–7579, 2014.

[11] D. Ivanov, A. Chezhegov, M. Kiselev, A. Grunin, and D. Larionov. Neuromorphic artificial intelligence systems. *Frontiers in Neuroscience*, 16:1513, 2022.

[12] C. Mayr, S. Hoeppner, and S. Furber. Spinnaker 2: A 10 million core processor system for brain simulation and machine learning. *arXiv preprint arXiv:1911.02385*, 2019.

[13] E. O. Neftci, H. Mostafa, and F. Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019.

[14] C. Scheerlinck, H. Rebecq, D. Gehrig, N. Barnes, R. Mahony, and D. Scaramuzza. Fast image reconstruction with an event camera. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 156–163, 2020.

[15] S. B. Shrestha and G. Orchard. Slayer: Spike layer error reassignment in time. *Advances in neural information processing systems*, 31, 2018.

[16] D. Singla, S. Chatterjee, L. Ramapantulu, A. Ussa, B. Ramesh, and A. Basu. Hynna: Improved performance for neuromorphic vision sensor based surveillance using hybrid neural network architecture. In *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE, 2020.