DVS-Attacks: Adversarial Attacks on Dynamic Vision Sensors for Spiking Neural Networks

Alberto Marchisio[1,*], Giacomo Pira[2,*], Maurizio Martina[2], Guido Masera[2], Muhammad Shafique[3]
[1]*Technische Universität Wien, Vienna, Austria* [2]*Politecnico di Torino, Turin, Italy* [3]*New York University, Abu Dhabi, UAE*
Email: alberto.marchisio@tuwien.ac.at, giacomo.pira@studenti.polito.it
{*maurizio.martina, guido.masera*}*@polito.it, muhammad.shafique@nyu.edu*

Abstract—**Spiking Neural Networks (SNNs), despite being energy-efficient when implemented on neuromorphic hardware and coupled with event-based Dynamic Vision Sensors (DVS), are vulnerable to security threats, such as adversarial attacks, i.e., small perturbations added to the input for inducing a misclassification. Toward this, we propose *DVS-Attacks*, a set of stealthy yet efficient adversarial attack methodologies targeted to perturb the event sequences that compose the input of the SNNs. First, we show that noise filters for DVS can be used as defense mechanisms against adversarial attacks. Afterwards, we implement several attacks and test them in the presence of two types of noise filters for DVS cameras. The experimental results show that the filters can only partially defend the SNNs against our proposed *DVS-Attacks*. Using the best settings for the noise filters, our proposed *Mask Filter-Aware Dash Attack* reduces the accuracy by more than 20% on the DVS-Gesture dataset and by more than 65% on the MNIST dataset, compared to the original clean frames. The source code of all the proposed *DVS-Attacks* and noise filters is released at https://github.com/albertomarchisio/DVS-Attacks.**

Index Terms—**Spiking Neural Networks, SNNs, Deep Learning, Adversarial Attacks, Security, Robustness, Defense, Filter, Perturbation, Noise, Dynamic Vision Sensors, DVS, Neuromorphic, Event-Based.**

I. Introduction

Spiking Neural Networks (SNNs) represent energy-efficient learning models in a wide variety of machine learning applications, e.g., autonomous driving [1], healthcare [2], and robotics [3]. Unlike traditional (i.e., non-spiking) Deep Neural Networks (DNNs), the SNNs are more closely related to the human brain's processing [4]. Indeed, the event-based communication between neurons makes them biologically plausible. Moreover, SNNs are appealing for being implemented in resource-constrained embedded systems [5], due to a good combination of power/energy efficiency and real-time classification performance. In fact, compared to the equivalent DNN implementations, SNNs exhibit a lower computational load, as well as a reduction in the latency, by leveraging the spike-based communication between neurons [6].

Efficient SNNs are typically implemented on a specialized neuromorphic hardware [7], which is able to exploit the asynchronous communication mechanism between neurons and the event-based propagation of the information through layers. These characteristics led to an increasing interest in developing neuromorphic architectures like IBM TrueNorth [8] and Intel Loihi [9]. Another advancement in the field of neuromorphic hardware has come from the new generation of event-based camera sensors, such as the Dynamic Vision Sensor (DVS) [10]. Unlike a classical frame-based camera, the DVS emulates the behavior of the human retina, by recording the information in form of a sequence of spikes, which are generated every time a change of light intensity is detected. The event-based behavior of these sensors pairs well with SNNs implemented onto the neuromorphic hardware, i.e., the output of a DVS camera can be used as the direct input of an SNN to process events in real-time.

A. Target Research Problem and Scientific Challenges

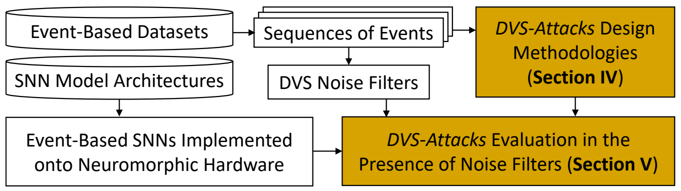
Different security threats challenge the correct functionality of DNNs and SNNs. The DNN trustworthiness has been extensively investigated in recent years [11], highlighting that one of the most critical issues is the adversarial attacks, i.e., small and imperceptible input perturbations to trigger misclassification [12]. Although some initial studies have been conducted [13][14][15][16], the SNN trustworthiness is a relatively new and unexplored problem. More specifically, DVS-based systems have not yet been investigated for SNN security. Moreover, the methods for defending SNNs against such adversarial attacks can be inspired from the recent advancements of the defense mechanisms for DNNs, where studies have focused on adversarial learning algorithms [17], loss/regularization functions [18], and image preprocessing [19]. The latter approach basically consists of suppressing the adversarial perturbation through dedicated filtering. Noteworthy, for the SNN-based systems feeded by DVS signals, the attacks and preprocessing-based defense techniques for frame-based sensors cannot be directly applied due to differences in the signal properties. Therefore, specialized noise filters for DVS sensors [20] must be employed.

As per our knowledge, the generation of adversarial attacks for DVS signals is an unexplored and open research problem. Towards this, we propose *DVS-Attacks*, a set of adversarial attack methodologies for DVS signals, and test them in scenarios where noise filters are employed as a defense mechanism against them. Since the DVS cameras contain also the temporal information, the generation of adversarial perturbation is technically different w.r.t. traditional adversarial attacks on images, where only the spatial information is considered. Hence, the temporal information needs to be leveraged for developing the attack and defense mechanisms. The steps involved in this work are visualized in Fig. 1.

B. Motivational Case Study

As a preliminary study for motivating our research in the above-discussed directions, we perform the following experiments. We trained a 4-layer SNN with 2 convolutional layers and 2 fully-connected layers, for the DVS-Gesture dataset [21] using the SLAYER method [22] in a DL-

*These authors contributed equally to this work.



Event-Based Datasets
Sequences of Events
DVS-Attacks Design Methodologies (Section IV)
SNN Model Architectures
DVS Noise Filters
Event-Based SNNs Implemented onto Neuromorphic Hardware
DVS-Attacks Evaluation in the Presence of Noise Filters (Section V)

Fig. 1: Overview of the steps involved in this work. Our novel contributions are highlighted in colored boxes.

workstation equipped with two Nvidia GeForce RTX 2080 Ti GPUs. For each frame of events, we perturb the testing dataset by injecting normally-distributed random noise and measure the classification accuracy. Moreover, to mitigate the effect of the perturbations, the *Background Activity Filter* (*BAF*) and the *Mask Filter* (*MF*) of [20] are applied, with various filter parameters. The accuracy results w.r.t. different noise magnitude are shown in Fig. 2. As indicated by pointer ① in Fig. 2, the filter may potentially reduce the accuracy of the SNN when no noise is applied. More specifically, more than 20% drop is noticed on the *MF* with $T = 25$, and lower differences for the other filters. However, in the presence of noise, the SNN becomes much more robust when the filter is applied. For example, when considering normal noise with a magnitude of 0.55, the *BAF* with $s = 1$ and $t = 5$ contributes to 64% accuracy improvement (see pointer ②). On the other hand, *BAFs* with $s \geq 2$ do not increase the accuracy much, compared to the unfiltered SNN. Moreover, *MFs* with $T \geq 100$ work even better than the BAFs in the presence of large perturbations. Indeed, the perturbations with magnitude of 1.0 are filtered out relatively well by the *MFs* with large T (see pointer ③), while, for the same noise magnitude, both the *MFs* with $T \leq 50$ and the *BAF* with $s = 1$ and $t = 5$ achieve an accuracy of only ≈33-34% (see pointer ④). The key message learnt from the above case study is that the noise filters for DVS can potentially restore a large portion of accuracy that would have been dropped due to the perturbations. Therefore, this motivates us to employ such filters as defense methods against adversarial attacks.

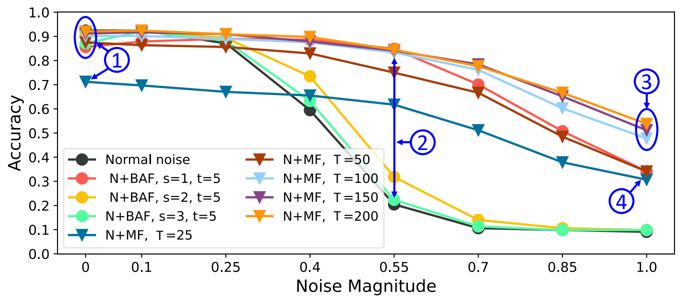


Fig. 2: Analyzing the impact of applying the normally-distributed noise to the DVS-Gesture dataset, in the presence of *BAF* and *MF* noise filters.

C. Our Novel Contributions

- We propose *DVS-Attacks*, a set of different adversarial attack methodologies generating perturbations for DVS signals; (**Section IV**). As per our knowledge, *these are the first proposed attack algorithms for event-based neuromorphic systems*.
- In particular, the *MF-Aware Dash Attack* is specifically designed to be resistant against the *Mask Filter* defense, by generating perturbations only on a limited set of frames; (**Section IV-E**).
- The experimental results on the DVS-Gesture and NMNIST datasets show that all the attacks are successful when no filter is applied. Moreover, the noise filters cannot fully defend against the *DVS-Attacks*, which represent a serious security threat for SNN-based neuromorphic systems; (**Section V**).
- For reproducible research, we released the source code of all the proposed *DVS-Attacks* methodologies, and filters for DVS-based SNNs at https://github.com/albertomarchisio/DVS-Attacks.

Before proceeding to the technical details, **Section II** presents an overview of SNNs, noise filters for DVS signal, adversarial attacks, and security threats for SNNs. Moreover, **Section III** discusses the threat model employed in this work.

II. Background and Related Work

A. Spiking Neural Networks (SNNs)

SNNs are considered as the third generation neural networks [23]. Compared to the traditional DNNs, they exhibit better biological plausibility [4] and high resilience [24][25][26] compared to the traditional DNNs [27][28]. Another key advantage of SNNs over the traditional DNNs is their improved energy-efficiency [29][30] when implemented on Neuromorphic chips like Intel Loihi [9] or IBM TrueNorth [8]. Moreover, the recent development of DVS sensors [10] has further reduced the energy requirements of the complete system [31][32].

In SNNs, the input is encoded using spikes, which propagate to the output through neurons and synapses. In a Leaky-Integrate-and-Fire (LIF) neuron, which is the most commonly adopted spiking neuron model, each input spike contributes to increasing the neuron membrane potential V over time. As shown in Fig. 3, when V overcomes a threshold V_{th}, an output spike is released by the neuron, and propagated to the neurons of the following layer.

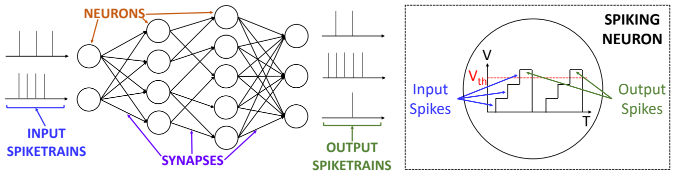


Fig. 3: Overview of an SNN's functionality, focusing on the evolution over time of the membrane potential of a spiking neuron.

Event-based cameras [10] are the new generations of bio-inspired sensors for the acquisition of visual information, directly related to the light variations in the scene. Instead of recording frames with a precise timing, the DVS cameras work asynchronously, recording only positive and negative brightness variations in the scene. Each event is encoded with four components (x, y, p, t), which represent the x-coordinate, the y-coordinate, the polarity, and the timestamp, respectively. Compared to classical frame-based image sensors, the event-based sensors consume significantly lower power, since the events are recorded only when a brightness variation in the scene is detected. This means that, in the absence of light changes, no information is recorded, leading close to zero power consumption. *Hence, DVS sensors can be efficiently deployed at the edge and directly coupled to neuromorphic hardware for low-power SNN-based applications.*

B. Noise Filters for Dynamic Vision Sensors

DVS sensors are mainly affected by background activity noise, caused by thermal noise and junction leakage current [33]. When the DVS is stimulated, a neighborhood of pixels is usually active at the same time, generating events. Therefore, the real events show a higher spatio-temporal correlation than the noise-related events. This empirical observation is exploited for generating the **Background Activity Filter (BAF)** [20]. The events are associated with a spatio-temporal neighborhood, within which the correlation between them is calculated. If the correlation is lower than a certain threshold, the events are likely due to noise and thus are filtered out; otherwise they are kept. The procedure is reported in Algorithm 1, where S and T are the only parameters of the filter and are used to set the dimensions of the spatio-temporal neighborhood. The larger S and T are, the lower the number of events are filtered out. The decision of the filter is made by the comparison between $t_e - M[x_e][y_e]$ and T (lines 15-16 of Algorithm 1). If the first term is lower, then the event is filtered out.

Algorithm 1 : *Background Activity Filter* for event-based sensors.

```
1: Being $E$ a list of events of the form $(x, y, p, t)$
2: Being $(x_e, y_e, p_e, t_e)$ the x-coordinate, the y-coordinate, the
   polarity and the timestamp of the event $e$ respectively
3: Being $M$ a $128 \times 128$ matrix
4: Being S and T the spatial and temporal filter's parameters
5: Initialize $M$ to zero
6: Order $E$ from the oldest to the newest event
7: for $e$ in $E$ do
8:    for $i$ in ($x_e - S$,$x_e + S$) do
9:       for $j$ in ($y_e - S$, $y_e + S$) do
10:         if not ($i == x_e$ and $j == y_e$) then
11:            $M[i][j] = t_e$
12:         end if
13:      end for
14:   end for
15:   if $t_e - M[x_e][y_e] > T$ then
16:      Remove $e$ from $E$
17:   end if
18: end for
```

Another type of scenario in which spontaneous noise activity is generated on the pixels which have low temporal contrast. In this case, a **Mask Filter (MF)** is required to filter-out such noise [20]. The procedure reported in Algorithm 2 shows that, compared to the *BAF*, the *MF* has only the temporal parameter T. If the activity of a pixel exceeds T, the mask is activated (lines 14-15 of Algorithm 2). After all the pixel coordinates of the mask are set, each event generated on a coordinate in which the mask is active is removed (lines 20-21). *Both the BAF and MF have been implemented and evaluated in the presence of intrinsic and parasitic noise of DVS sensors, while their application as a defense mechanism against adversarial attacks is still unexplored.*

C. Adversarial Attacks and Security Threats for SNNs in the Spatio-Temporal Domain

Currently, adversarial attacks are deployed on a wide range of deep learning applications [11]. They represent a serious threat for safety-critical applications, like surveillance, medicine,

Algorithm 2 : *Mask Filter* for event-based sensors.

```
1: Being $E$ a list of events of the form $(x, y, p, t)$,
2: Being $(x_e, y_e, p_e, t_e)$ the x-coordinate, the y-coordinate, the
   polarity and the timestamp of the event $e$ respectively,
3: Being $M$ a $N \times N$ matrix, where $N$ is the size of the frames,
4: Being $activity$ a $N \times N$ matrix, representing the number of event
   produced by each pixel,
5: Being T, the temporal threshold passed to the filter as a parameter,

6: Initialize $activity$ to zero
7: for x in range(N) do
8:    for y in range(N) do
9:       for e in E do
10:         if $(x, y) == (x_e, y_e)$ then
11:            $activity[x][y] += 1$
12:         end if
13:      end for
14:      if $activity[x][y] > T$ then
15:         $M[x][y] = 1$
16:      end if
17:   end for
18: end for
19: for e in E do
20:   if $M[x_e][y_e] == 1$ then
21:      Remove e from E
22:   end if
23: end for
```

and autonomous driving [34]. The objective of a successful attack is to generate small imperceptible perturbations to fool the network. Recently, adversarial attacks for SNNs have been explored, working in black-box [14] and white-box settings [13]. Sharmin et al. [15] proposed a methodology to attack (non-spiking) DNNs, and then the adversarial examples mislead the equivalent converted SNNs. Liang et al. [16] proposed a gradient-based adversarial attack methodology for SNNs. Venceslai et al. [35] proposed a methodology to attack SNNs through bit-flips triggered by adversarial perturbations. Towards adversarial robustness, recent works demonstrated that SNNs are inherently more robust than DNNs, due to the effect of effects of discrete input encoding, non-linear activations, and structural parameters [36][37]. *However, none of these previous works analyze attacks or defenses on frames of events, coming from DVS cameras.*

While in adversarial attack algorithms for images the perturbations are only added in the spatial domain, an attack on the DVS signal must introduce perturbations also in the temporal domain. As per our knowledge, there are no existing adversarial attack methodologies for event-based cameras coupled with SNN processing hardware. Some related works can be found in the field of attacks on video signals, i.e., sequences of events. State-of-the-art adversarial attacks on videos include, among others, sparse adversarial perturbations [38], where only a small subset of frames are perturbed. In this way, the attack is stealthy, because only a few frames are perturbed, and effective, due to the temporal interaction between consecutive frames. Another state-of-the-art method is represented by the adversarial framing [39], where the perturbation is added to the border of the frames and the misclassification is achieved. *However, frames of events cannot be treated as videos, since the latter contain the information*

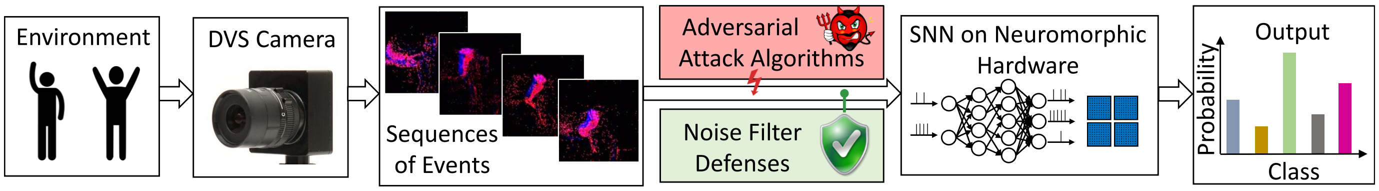


Fig. 4: Threat model considered in this work. Different types of adversarial attacks are considered, and different types of noise filters are applied as defense.

of pixel intensities for every frame, while do not contain other types of information, such as the polarity. Hence, the adversarial attacks for videos cannot be directly applied to DVS signals.

III. Threat Model

The system that we use in our experiments is composed of a DVS camera, for recording the scenes of the environment as sequences of events, and a given SNN implemented onto the neuromorphic hardware. As shown in Fig. 4, the adversarial attacks and noise filters for defense are located at the input of the SNN, and have access to modify the sequences of events. We conduct several experiments with different combinations of attacks and defenses. The noise filters described in Section II-B have been employed as defense methods. For the combinations in which both attacks and defenses are present in the system, the modifications generated by the attack are applied before the filter operation. In this way, the filter has the ability to filter out any events that have been generated or modified by the attack algorithm, thus aiming at making the defense stronger. The detailed description of the adversarial attack methodologies is discussed in the following Section IV.

IV. Our Proposed DVS-Attacks Methodologies

A. Sparse Attack

The proposed *Sparse Attack* is an iterative algorithm, which progressively updates the perturbation values based on the loss function (lines 6-12 of Algorithm 3), for each frame series of the dataset D. A mask M determines in which subset of the frames of events the perturbation should be added (line 7). Then, the output probability and the respective loss, obtained in the presence of the perturbation, are respectively computed in lines 9 and 10. Finally, the perturbation values are updated based on the gradients of the inputs w.r.t. the loss (line 11).

Algorithm 3 : *Sparse Attack* Methodology.

```
1: Being $M$ a mask able to select only certain frames
2: Being $D$ an event-based dataset
3: Being $P$ a perturbation to be added to the images
4: Being $prob$ the output probability of a certain class
5: for $d$ in $D$ do
6:     for $i$ in $max\_iteration$ do
7:         Add $P$ to $d$ only on the frames selected by $M$
8:         Calculate the prevision on the perturbed input
9:         Extract $prob$ for the actual class of $d$
10:        Update the loss value as $loss = -log(1 - prob)$
11:        Calculate the gradients and update $P$
12:    end for
13: end for
```

B. Frame Attack

The *Frame Attack* is a simple yet effective attack methodology, which consists of adding a frame around the sample (lines 6-8 of Algorithm 4). It does not require any expensive calculations, because the same perturbation (which coincides with the frame) is added to all the samples. In a dataset made of large images, such as the DVS-Gesture (128×128) it is also not so easy to spot, while with the perturbations on the NMNIST dataset (34×34) result more evident. One drawback is due to the overhead added to the samples in terms of events. In fact, since the attack targets every pixel of the boundary, for every frame, the number of events dramatically increases. Therefore, the size of the samples and the inference latency to process the events with the SNN and the filters increase as well.

Algorithm 4 : *Frame Attack* Methodology.

```
1: Being $D$ an event-based dataset
2: Being $d \subset D$ a $(C \times N \times N \times T)$ tensor, where C represents the
   channels, N represents the frame dimensions, and T the sample
   duration
3: for d in D do
4:     for x in range(N) do
5:         for y in range(N) do
6:             if $x == 0$ or $x == N-1$ or $y == 0$ or $y == N-1$
               then
7:                 $d[:, x, y, :] = 1$
8:             end if
9:         end for
10:    end for
11: end for
```

C. Corner Attack

The *Corner Attack*, as the name suggests, targets the corner of the images. It starts by modifying only two pixels at the top-left corner (lines 10-11 of Algorithm 5) and then, if it is not successful in fooling the SNN (line 16), it moves to the other corners. If some samples remain correctly classified, after hitting all 4 corners, the size of the perturbation increases and the algorithm resumes from the first corner. Before the updating phase, both when it changes corner or when it increase its size, the attack is applied to every sample in the dataset that was not yet corrupted. In this way, as the algorithm proceeds, the number of samples reduces and the the process is sped up. The main feature of this attack is that not all the samples are modified by the same amount of perturbation. For example, while the majority of the samples are misinterpreted by the SNN after few iterations, other samples are perturbed for longer time, thus making the attack easier to spot.

D. Dash Attack

The *Dash Attack* methodology is designed taking inspiration from the *Corner Attack*. Indeed, the two algorithms are quite

Algorithm 5 : *Corner Attack* Methodology.

```
1: Being $D$ an event-based dataset made of $(C \times N \times N \times T)$ tensors, where C represents the number of channels, N the size, and T the duration of the sample
2: $S$ is a list of the samples that compose $D$
3: $x = 0$
4: $y = 2$
5: $left = True$
6: while $S$ is not empty do
7:   for s in S do
8:     for i in range(N) do
9:       for j in range(N) do
10:        if $i == x$ and ($left$ and $j < y$ or $\overline{left}$ and $j \geq N - y - 1$) then
11:          $s[:, i, j, :] = 1$
12:        end if
13:      end for
14:    end for
15:    The perturbed sample s is fed to the SNN, which produces a prediction P
16:    if P is incorrect then
17:      Remove s from S
18:    end if
19:  end for
20:  if $x == 0$ then
21:    $x = N - 1$
22:  else
23:    $left = left$ xor 1
24:    $x = 0$
25:    if $\overline{left}$ then
26:      $y = y + 1$
27:    end if
28:  end if
29: end while
```

Algorithm 6 : *Dash Attack* Methodology.

```
1: Being $D$ an event-based dataset made of $(C \times N \times N \times T)$ tensors, where C represents the number of channels, N the size, and T the duration of the sample
2: $S$ is a list of the samples that compose $D$
3: $x_{min} = 0$, $x = 0$, $y = 2$
4: $left = True$
5: while $S$ is not empty do
6:   for s in S do
7:     for i in range(N) do
8:       for j in range(N) do
9:         if $i == x$ and ($left$ and ($j == y$ or $j == y - 1$) or $\overline{left}$ and ($j == N - y$ or $j == N - y + 1$)) then
10:          $s[:, i, j, :] = 1$
11:        end if
12:      end for
13:    end for
14:    The perturbed sample s is fed to the SNN, which produces a prediction P
15:    if P is incorrect then
16:      Remove s from S
17:    end if
18:  end for
19:  if $x == x_{min}$ then
20:    $x = N - x_{min} - 1$
21:  else
22:    $left = left$ xor 1
23:    $x = x_{min}$
24:    if $\overline{left}$ then
25:      $y = y + 1$
26:    end if
27:  end if
28:  if $y > N/2$ then
29:    $x_{min} = x_{min} + 1$
30:  end if
31: end while
```

similar. The main difference is that in the *Dash Attack* only two pixels are targeted every time. The main structure of the algorithm is the same as for the *Corner Attack*, as the *Dash Attack* starts by targeting the top-left corner and by modifying the first two pixels. Moreover, the x, y coordinates are updated, in order for the attack to hit only two consecutive pixels (see lines 19-29 of Algorithm 6). Hence, this attack results to be very difficult to spot, and the introduced perturbations do not cause a large overhead of events on the samples. Moreover, all the samples under the *Dash Attack* are modified by the same amount of perturbations.

E. MF-Aware Dash Attack

The main issue of the above-discussed attacks, as will be demonstrated in Section V, is their intrinsic weakness against the *MF*. In fact, they targeted both channels ('on' and 'off' events) of the same pixels for all the duration of the sample. This leads to a clear distinction between the pixels affected by the attack and those that are not. In fact, the number of events produced by the targeted pixels is significantly higher than the events associated to the other pixel coordinates that were not hit by the attack. In addition, we have to consider the fact that the proposed attacks mainly focus on the boundaries of the images, thus they do not tend to overlap with useful information. In other words, in the datasets that we used the subject is typically centered. Hence, by hitting the perimeter or the corners, the risk of superimposing adversarial noise to the main subject is low. These considerations explain why the *MF* is successful for restoring the original SNN accuracy. Indeed, the targets are easily identifiable given their high number of events, and the filter does not remove useful information, since modifications are mainly conducted at the edge of the image.

Based on these premises, we have designed an attack aiming at being resistant to the *MF*, which we call *MF-Aware Dash Attack*. It receives as a parameter th, which is correlated to the T parameter of the *MF* (recall from Algorithm 2), and it uses it to set a limit on the number of frames that can be changed for each pixel (line 11 of Algorithm 7). Therefore, the algorithm targets a couple of pixels to be perturbed, as in case of the *Dash Attack*. However, after modifying th frames, it moves to the following ones (lines 16-18). The visual effect generated by the *MF-Aware Dash Attack* is that of a dash advancing along a line. The smaller the parameter th is, the faster will the dash seem moving along the image.

V. Evaluation of the DVS-Attacks in the Presence of Noise Filters

A. Experimental Setup

We conducted experiments on two datasets, the DVS-gesture [21] and the NMNIST [40]. The former is a collection of of 1077 samples for training and 264 for testing, divided into 11 classes, while the latter is a spiking version of the original frame-based MNIST dataset [41]. It is generated by an ATIS event-based sensor [42] that is moved while capturing

Algorithm 7 : *MF-Aware Dash Attack* Methodology.

```
1: Being $D$ an event-based Dataset made of $(2 \times N \times N \times T)$ tensors, where N represents the frame dimensions, and T the sample duration
2: $S$ is a list of the samples that compose $D$
3: $th$ is a parameter associated the activity threshold of the MF
4: $x = 0$ , $y_0 = 2$, $left = True$
5: while $S$ is not empty do
6:     for s in S do
7:         $th = th_0$, $y = y_0$
8:         for t in T do
9:             for i in range(N) do
10:                for j in range(N) do
11:                    if $i == x$ and $t < th$ and ($left$ and ($j == y$ or $j == y - 1$) or $\overline{left}$ and ($j == N - y$ or $j == N - y + 1$)) then
12:                        $s[0, i, j, t] = 1$
13:                    end if
14:                end for
15:            end for
16:            if $t == th$ then
17:                $th = th + th_0$ , $y = y + 2$
18:            end if
19:        end for
20:        The perturbed sample s is fed to the SNN, which produces a prediction P
21:        if P is incorrect then
22:            Remove s from S
23:        end if
24:    end for
25:    if $x == 0$ then
26:        $x = N - 1$
27:    else
28:        $left = left$ xor $1$ , $x = 0$
29:        if $\overline{left}$ then
30:            $y_0 = y_0 + 1$
31:        end if
32:    end if
33: end while
```

the MNIST images projected on a LCD screen. It consists of 60,000 training and 10,000 testing samples. As classifier for the DVS-gesture dataset, we employed the 4-layer SNN as described in [22], with two convolutional layers and two fully-connected layers, and trained it for 625 epochs with the SLAYER backpropagation method [22], using a batch size of 4 and learning rate equal to 0.01. We measured a test accuracy of 92.04% on clean inputs. As classifier for the NMNIST dataset, we employed a multilayer perceptron with two fully-connected layers [22], trained for 350 epochs with the SLAYER backpropagation method [22], using a batch size of 4 and learning rate equal to 0.01. The test accuracy on clean inputs is 95%. We implemented the SNNs on a DL-workstation with two Nvidia GeForce RTX 2080 Ti GPUs, using the PyTorch framework [43]. We also implemented the adversarial attack algorithms and the noise filters in PyTorch. The experimental setup and tool-flow in a real-world setting is shown in Fig. 5.

B. Results for the Sparse Attack

The *Sparse Attack* on DVS frames is successful on both benchmarks, as the accuracy is drastically decreased to 15.15% for the DVS-Gesture dataset (see pointer ① in Fig. 6), and to 4% for the NMNIST dataset (see pointer ②). By looking at the adversarial examples reported at the left side of Fig. 6, no

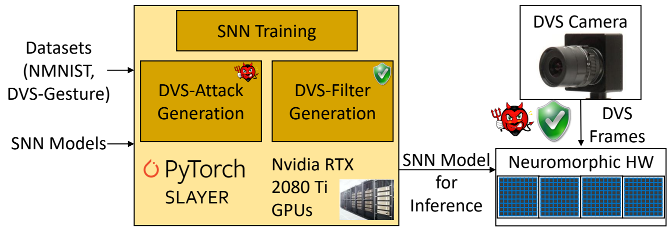


Fig. 5: Experimental setup, tool-flow, and integration with the system.

significant perturbations are perceived, thus making the *Sparse Attack* stealthy. However, the accuracy can be easily restored using a noise filter. When the *BAF* filter is employed, for a wide range of the (s, t) parameters the SNNs' accuracy overcomes 90% (see pointers ③). When the *MF* is used, a low T does not protect well against the *Sparse Attack*, but when $T \geq 50$ for the DVS-Gesture dataset (see pointer ④) and when $T \geq 25$ for the NMNIST dataset (see pointer ⑤), high robustness is achieved.

C. Results for the Frame Attack

The results for the experiments conducted on the *Frame Attack* are reported in Fig. 7. As expected, the perturbations are perceivable in the form of a line added to the border of the visualized shot. This feature is much more accentuated on the NMNIST dataset, where the resolution is of 34×34 pixels, while the perturbations are less distinguishable on the 128×128 examples of the DVS-Gesture dataset. The accuracy under attack drops to 9.85% and 8% for the two datasets, respectively. However, the *BAF* does not work well as a defense against the *Frame Attack*. As highlighted by pointers ① in Fig. 7, there exist no combinations of the (s, t) parameters of the *BAF* for which the SNNs' accuracy significantly increases. Indeed, the accuracy difference compared to the attack without filter is relatively low. On the other hand, the *MF* results to be a successful defense, because the SNNs' accuracy is high for large values of T (see pointer ②).

D. Results for the Corner Attack

The *Corner Attack* is visibly stealthier than the *Frame Attack*. Indeed, the perturbations are only added in the corner of the images. For example, the perturbation is noticeable in the top-left corner of the first example of the NMNIST dataset (see pointer ① in Fig. 8), or in the bottom-left corner of the second example (see pointer ②). Moreover, the SNNs are completely fooled by the *Corner Attack*, since the accuracy without noise drops to 0% (see pointers ③). The *BAF* works relatively better for the DVS-Gesture dataset, compared to the MNIST dataset. However, the accuracy in the presence of the *BAF* filter as defense remains very low. The peak reached with $s = 1$ and $t = 5$ has an accuracy of only 15.15% for the SNN on the DVS-Gesture dataset (pointer ④). Similarly to the *Frame Attack*, also the *Corner Attack* can be successfully mitigated when the *MF* with large T is applied (see pointers ⑤).

E. Results for the Dash Attack

The *Dash Attack* performs in a similar way as the *Corner Attack*, but the perturbations are not strictly confined in a corner. In this way, the perturbations introduced by the attack result very similar to the inherent background noise generated by

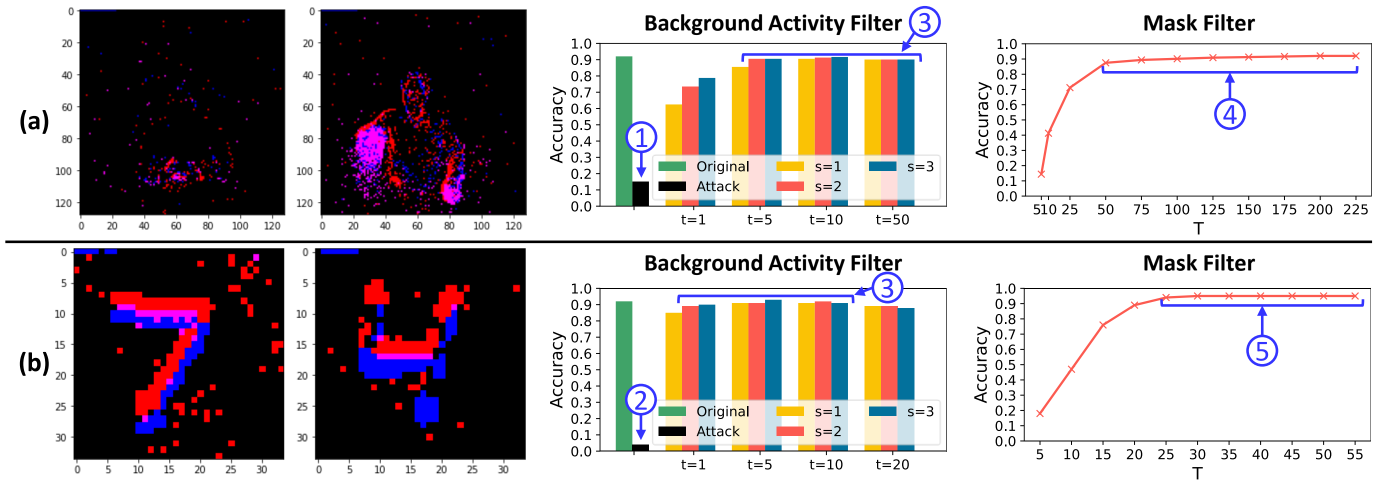


Fig. 6: Evaluation of the *Sparse Attack*: frame samples and accuracy when the *BAF* and *MF* are applied, for (a) DVS-Gesture and (b) NMNIST.

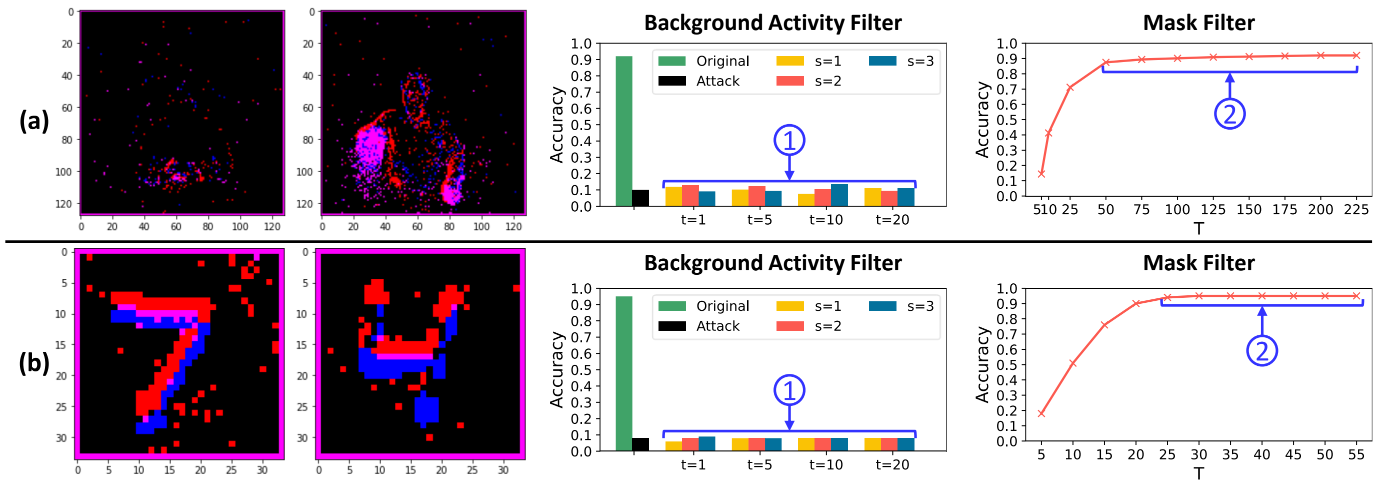


Fig. 7: Evaluation of the *Frame Attack*: frame samples and accuracy when the *BAF* and *MF* are applied, for (a) DVS-Gesture and (b) NMNIST.

the DVS camera recording the events. For instance, the attack perturbations on the examples for the NMNIST dataset (see pointers ① in Fig. 9) might be confused with the inherent background noise (see pointers ②). Compared to the *Corner Attack*, while the accuracy of the SNNs under the *Dash Attack* without filter drops to 0%, the *BAF* defense produces a slightly higher SNN accuracy for the DVS-Gesture dataset. However, the accuracy peak of 28.41% (see pointer ③), obtained in the presence of the *BAF* with $s = 1$ and $t = 10$, remains too low to consider the *BAF* as a good defense method against the *Dash Attack*. Once again, a good defense for robust SNNs is achieved by the *MF* with large T (see pointers ④).

F. Results for the MF-Aware Dash Attack

Fig. 10 shows the results for the experiments conducted on the *MF-Aware Dash Attack*, for different values of the parameter th. While the stealthiness of the adversarial examples (Fig. 10 reports the samples generated with $th = 150$ for the DVS-Gesture dataset and $th = 20$ for the NMNIST dataset) is similar to the *Corner* and *Dash Attacks*, the behavior of the *MF-Aware Dash Attack* in the presence of noise filters is much different. Moreover, the accuracy of the SNNs under attack without filter are different from 0, reaching up to 7.95% for $th = 50$ on the DVS-Gesture dataset (see pointer ① in Fig. 10). The SNNs defended by the *BAF* show discrete robustness, in particular when $s = 3$ and $t = 1$. In such scenario, the accuracy reaches 59.09% when the *MF-Aware Dash Attack* with $th = 50$ is applied to the SNN for the DVS-Gesture dataset (see pointer ②). However, when $t \geq 5$, the SNN accuracy is lower than 31.44% for the DVS-Gesture dataset (see pointer ③) and lower than 13% for the NMNIST dataset (see pointer ④). The key advantage compared to the above-discussed attacks resides in the behavior of the *MF-Aware Dash Attack* in the presence of the *MF*. If $T \geq th$, the SNN accuracy becomes lower than 23.5% for the the DVS-Gesture dataset (see pointer ⑤) and lower than 2% for the NMNIST dataset (see pointer ⑥). On the contrary, the behavior when $T < th$ is similar to the results obtained for the other attacks. For example, the curve relative to the *MF-Aware Dash Attack* with $th = 50$ for the DVS-Gesture dataset achieves 71.21% accuracy for $T = 25$ (see pointer ⑦), which is 20.83% lower than the original SNN accuracy.

G. Key Observations Derived from the Experiments

By analyzing in more detail the results for the different types of attacks, we can derive the following key observations:

- All the attack algorithms belonging to the *DVS-Attacks* set are successful when no filter is applied, since the SNNs' accuracy is significantly decreased.

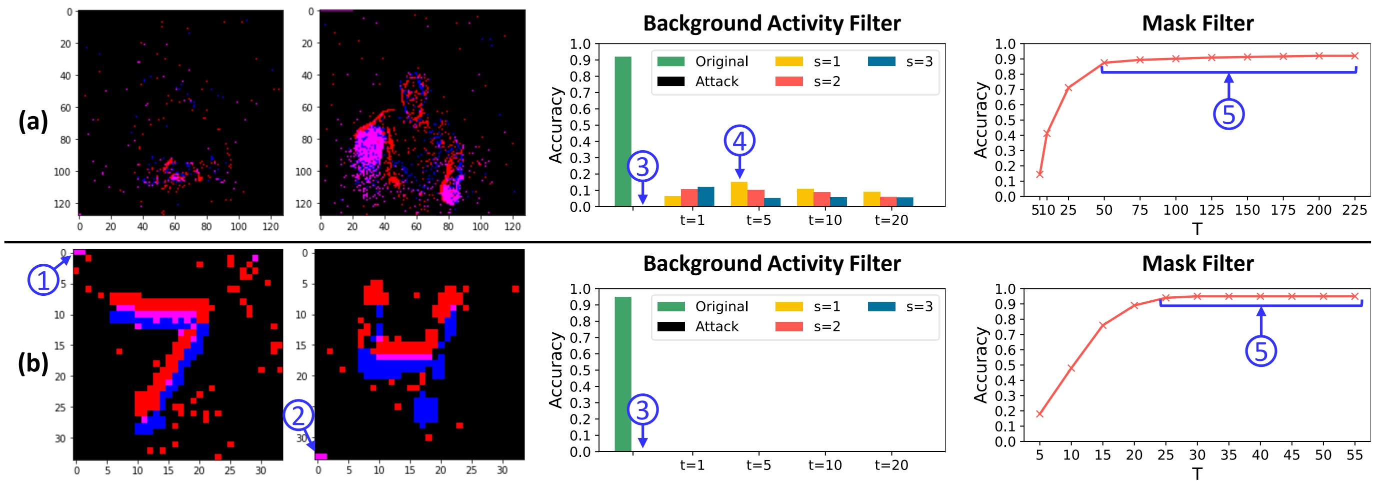


Fig. 8: Evaluation of the *Corner Attack*: frame samples and accuracy when the *BAF* and *MF* are applied, for (a) DVS-Gesture and (b) NMNIST.

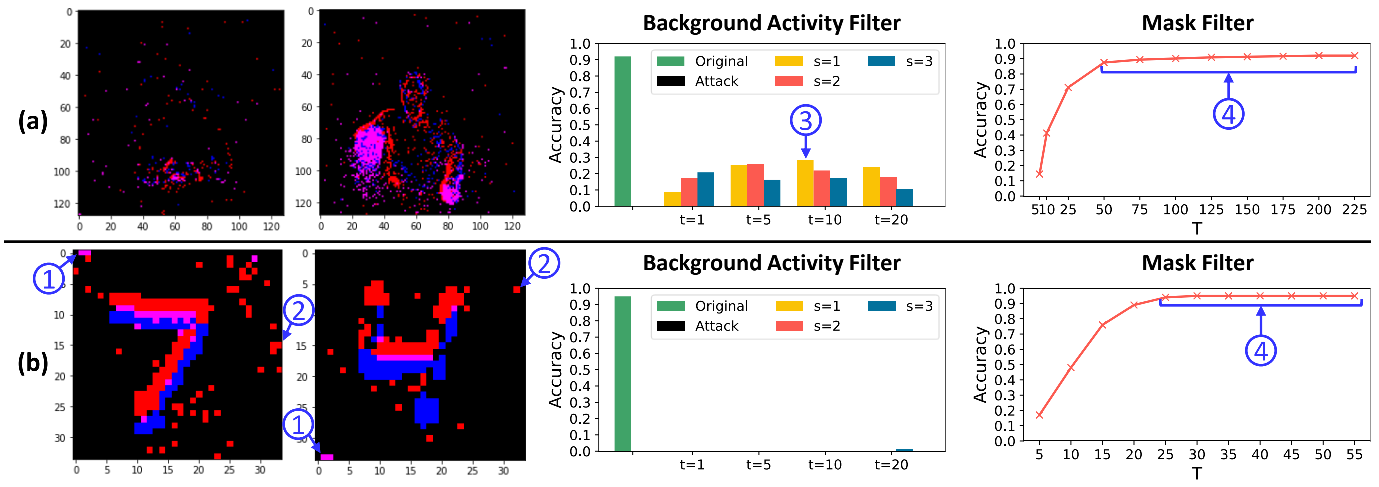


Fig. 9: Evaluation of the *Dash Attack*: frame samples and accuracy when the *BAF* and *MF* are applied, for (a) DVS-Gesture and (b) NMNIST.

- The *Sparse Attack* is the stealthiest attack, while *Corner*, *Dash* and *MF-Aware Dash Attacks* are stealtier than the *Frame Attack*.
- The *BAF* achieves good defense only for the *Sparse Attack*, while all the other attacks can fool SNNs defended by the *BAF*. Some accuracy is recovered for the *MF-Aware Dash Attack*, but a considerable accuracy loss is measured.
- Different (s, t) parameters of the *BAF* need to be evaluated for obtaining the highest accuracy, and the the combinations of these parameters can vary according to different attack algorithms.
- The *MF* with large T is a good defense for almost all the attacks, but it does not work well with the *MF-Aware Dash Attack*, since it is an adversarial attack specifically designed for being resistant to the *MF*.
- The best *MF-Aware Dash Attack*, that is with $th = 50$ for the DVS-Gesture dataset, and with $th = 10$ for the NMNIST dataset, can reduce the accuracy by at least 20% and 65% for the two datasets, respectively.

VI. Conclusion

In this paper, we designed *DVS-Attacks*, a set of adversarial attack methodologies for SNNs, which introduce the perturbations into the sequences of events. Therefore, they are suitable for neuromorphic systems supplied by DVS cameras. Moreover, two types of noise filters, namely the *Background Activity Filter* and the *Mask Filter*, are applied as defenses. The experimental results show the high success of the attacks, since the SNNs cannot be completely defended by the noise filters. Therefore, they represent critical security threats for SNN-based neuromorphic systems supplied by event-based sensors. We released the source code of the *DVS-Attacks* and noise filters at https://github.com/albertomarchisio/DVS-Attacks.

Acknowledgments

This work has been partially supported by the Doctoral College Resilient Embedded Systems, which is run jointly by the TU Wien's Faculty of Informatics and the UAS Technikum Wien. This work was also jointly supported by the NYUAD Center for Interacting Urban Networks (CITIES), funded by Tamkeen under the NYUAD Research Institute Award CG001 and by the Swiss Re Institute under the Quantum Cities™ initiative, and Center for CyberSecurity (CCS), funded by Tamkeen under the NYUAD Research Institute Award G1104.

References

[1] S. Zhou *et al.*, "Deep scnn-based real-time object detection for self-driving vehicles using lidar temporal data," *IEEE Access*, 2020.
[2] C. D. V. Gonzalez, J. H. S. Azuela, J. Antelis, and L. E. Falcón, "Spiking

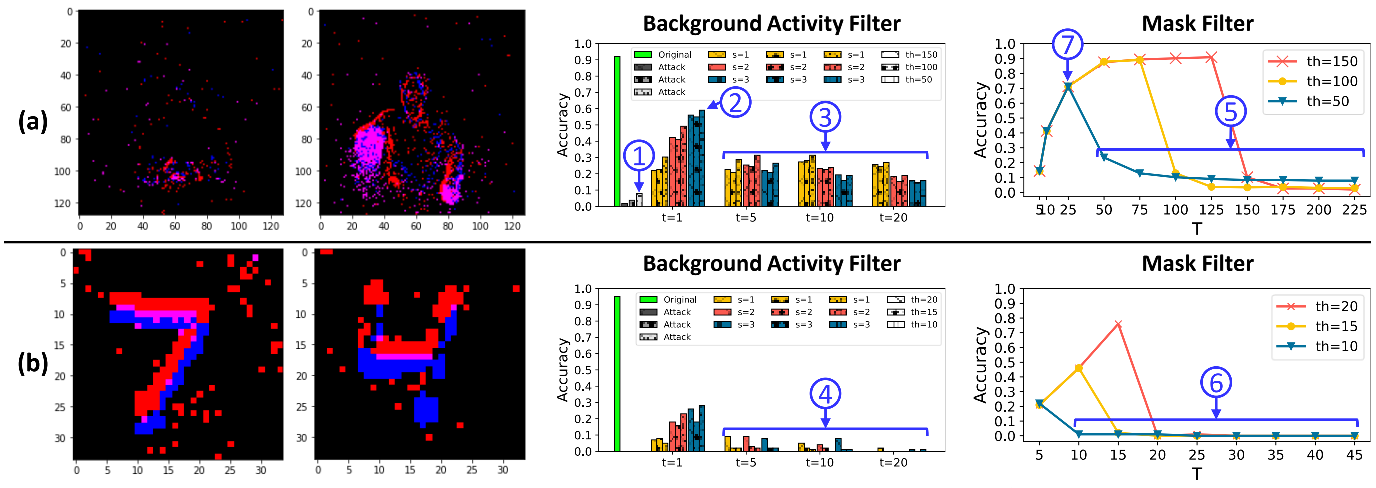


Fig. 10: Evaluation of the *MF-Aware Dash Attack*: frame samples and accuracy when the *BAF* and *MF* are applied, for (a) DVS-Gesture and (b) NMNIST. On the left side are reported two adversarial frame samples generated with $th = 150$ for the DVS-Gesture dataset, and with $th = 20$ for the NMNIST dataset.

neural networks applied to the classification of motor tasks in eeg signals," *Neural networks*, 2020.

[3] G. Tang and K. P. Michmizos, "Gridbot: An autonomous robot controlled by a spiking neural network mimicking the brain's navigational system," *ArXiv*, vol. abs/1807.02155, 2018.

[4] F. Ponulak and A. Kasiński, "Introduction to spiking neural networks: Information processing, learning and applications," *Acta neurobiologiae experimentalis*, 2011.

[5] M. Capra *et al.*, "Hardware and software optimizations for accelerating deep neural networks: Survey of current trends, challenges, and the road ahead," *IEEE Access*, 2020.

[6] L. Deng *et al.*, "Rethinking the performance comparison between snns and anns," *Neural networks*, 2020.

[7] C. D. Schuman *et al.*, "A survey of neuromorphic computing and neural networks in hardware," *ArXiv:1705.06963*, 2017.

[8] P. A. Merolla *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, 2014.

[9] M. Davies *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, 2018.

[10] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 x 128 120db 30mw asynchronous vision sensor that responds to relative intensity change," in *ISSCC*, 2006.

[11] M. Shafique *et al.*, "Robust machine learning systems: Challenges, current trends, perspectives, and the road ahead," *IEEE D&T*, 2020.

[12] C. Szegedy *et al.*, "Intriguing properties of neural networks," in *ICLR*, 2014.

[13] A. Bagheri, O. Simeone, and B. Rajendran, "Adversarial training for probabilistic spiking neural networks," in *SPAWC*, 2018.

[14] A. Marchisio *et al.*, "Is spiking secure? a comparative study on the security vulnerabilities of spiking and deep neural networks," in *IJCNN*, 2020.

[15] S. Sharmin *et al.*, "A comprehensive analysis on adversarial robustness of spiking neural networks," in *IJCNN*, 2019.

[16] L. Liang *et al.*, "Exploring adversarial attack in spiking neural networks with spike-compatible gradient," *ArXiv*, vol. abs/2001.01587, 2020.

[17] A. Madry *et al.*, "Towards deep learning models resistant to adversarial attacks," in *ICLR*, 2018.

[18] H. Zhang *et al.*, "Theoretically principled trade-off between robustness and accuracy," in *ICML*, 2019.

[19] F. Khalid *et al.*, "Fademl: Understanding the impact of pre-processing noise filtering on adversarial machine learning," in *DATE*, 2019.

[20] A. Linares-Barranco *et al.*, "Low latency event-based filtering and feature extraction for dynamic vision sensors in real-time fpga applications," *IEEE Access*, 2019.

[21] A. Amir *et al.*, "A low power, fully event-based gesture recognition system," in *CVPR*, 2017.

[22] S. B. Shrestha and G. Orchard, "Slayer: Spike layer error reassignment in time," in *NeurIPS*, 2018.

[23] W. Maas, "Networks of spiking neurons: The third generation of neural network models," *Trans. Soc. Comput. Simul. Int.*, 1997.

[24] C. D. Schuman *et al.*, "Resilience and robustness of spiking neural networks for neuromorphic systems," in *IJCNN*, 2020.

[25] R. V. W. Putra and M. Shafique, "Q-spinn: A framework for quantizing spiking neural networks," in *IJCNN*, 2021.

[26] R. V. W. Putra, M. A. Hanif, and M. Shafique, "Sparkxd: A framework for resilient and energy-efficient spiking neural network inference using approximate dram," in *DAC*, 2021.

[27] A. Marchisio *et al.*, "Deep learning for edge computing: Current trends, cross-layer optimizations, and open research challenges," in *ISVLSI*, 2019.

[28] M. Capra *et al.*, "An updated survey of efficient hardware architectures for accelerating deep convolutional neural networks," *Future Internet*, 2020.

[29] R. V. W. Putra and M. Shafique, "Fspinn: An optimization framework for memory-efficient and energy-efficient spiking neural networks," *TCAD*, 2020.

[30] R. V. W. Putra and M. Shafique, "Spikedyn: A framework for energy-efficient spiking neural networks with continual and unsupervised learning capabilities in dynamic environments," in *DAC*, 2021.

[31] R. Massa, A. Marchisio, M. Martina, and M. Shafique, "An efficient spiking neural network for recognizing gestures with a dvs camera on the loihi neuromorphic processor," in *IJCNN*, 2020.

[32] A. Viale *et al.*, "Carsnn: An efficient spiking neural network for event-based autonomous cars on the loihi neuromorphic research processor," in *IJCNN*, 2021.

[33] Y. Nozaki and T. Delbruck, "Temperature and parasitic photocurrent effects in dynamic vision sensors," *IEEE Transactions on Electron Devices*, 2017.

[34] C.-H. Cheng *et al.*, "Neural networks for safety-critical applications — challenges, experiments and perspectives," in *DATE*, 2018.

[35] V. Venceslai *et al.*, "Neuroattack: Undermining spiking neural networks security through externally triggered bit-flips," in *IJCNN*, 2020.

[36] S. Sharmin, N. Rathi, P. Panda, and K. Roy, "Inherent adversarial robustness of deep spiking neural networks: Effects of discrete input encoding and non-linear activations," in *ECCV*, 2020.

[37] R. El-Allami, A. Marchisio, M. Shafique, and I. Alouani, "Securing deep spiking neural networks against adversarial attacks through inherent structural parameters," in *DATE*, 2021.

[38] X. Wei, J. Zhu, and H. Su, "Sparse adversarial perturbations for videos," in *AAAI*, 2019.

[39] M. Zajac, K. Zolna, N. Rostamzadeh, and P. H. O. Pinheiro, "Adversarial framing for image and video classification," in *AAAI*, 2019.

[40] G. Orchard, A. Jayawant, G. Cohen, and N. Thakor, "Converting static image datasets to spiking neuromorphic datasets using saccades," *Frontiers in Neuroscience*, 2015.

[41] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, 1998.

[42] 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 JSSC*, 2011.

[43] A. Paszke *et al.*, “Automatic differentiation in pytorch,” in *NIPS 2017 Workshop on Autodiff*, 2017.