Denosing Event Data with Neural Networks

Tianshu Kuai, Yan Ma, and Yihan Ni

Abstract—Event cameras are bio-inspired sensors that generate sequences of events representing pixel-wise binary brightness changes. The output event sequence can be inherently noisy because of random perturbations in its brightness threshold and camera movement. Recent works in event data learning often use deep-learning based methods on voxelized event stream data. In this work, we propose an event-denoising network trained in a self-supervised fashion to restore clean data from random shifts and reject outlier events. Concretely, we employ a 3D U-Net architecture to denoise the voxelized event stream. Using our pre-trained denoiser as a pre-processing module, we observe large performance gains in multi-class classification tasks for both training the classifier from scratch and directly running inference on two different datasets, with huge potential to generalize to other downstream tasks in event-based vision.

Index Terms—Event-based Vision, Denoising, 3D U-Net, Multi-class Classification

1 Introduction

Event cameras are bio-inspired sensors that mimic the working principle of human retinas. Unlike traditional cameras with rolling shutters that capture image frames at regular frequencies, event cameras detect pixel-wise binary brightness changes in the scene and output asynchronous sequences of "events". The advantages of event cameras compare to RGB cameras include a high dynamic range, microsecond-level temporal resolution, and no motion blur. Therefore, event cameras are well-suited for high-speed applications such as driving scenarios.

Since event cameras produce events based on light intensity change, there could be noise in the events output if the captured scene is changing fast or the camera is not steady. There are plenty of methods to denoise image data sampled from RGB cameras, however, they cannot be directly applied to event data due to the difference in data representations. In event data learning, deep-learning-based approaches usually convert event streams to image-like data and use image models like CNNs to do further processing. In this work, we explore deep neural networks' capabilities in terms of denoising event data. We propose a 3D U-Net structure that denoises voxelized event data. We train the network in a self-supervised fashion as the model learns to reconstruct the original event data from noise-augmented event data. The trained denoiser can recover objects in the noisy data, and can be used as a pre-process module to boost classification performance. It shows robust performance in the multi-class classification tasks on both dynamic and static object datasets.

Our contribution in this work can be summarized as:

1) Event Classifiers trained on clean datasets become much more robust to noise with our pre-trained denoiser.
2) Event Classifiers can achieve higher test accuracy training on noisy datasets if combined with our pre-trained denoiser.
3) The training of the denoiser is a fully self-supervised process. No labels are needed for training, and the model can take any event data as training input.

• *Tianshu Kuai, Yan Ma, and Yihan Ni are with the Department of Computer Science, University of Toronto*
E-mail: {tianshu.kuai, yankuai.ma, nicky.ni}@mail.utoronto.ca

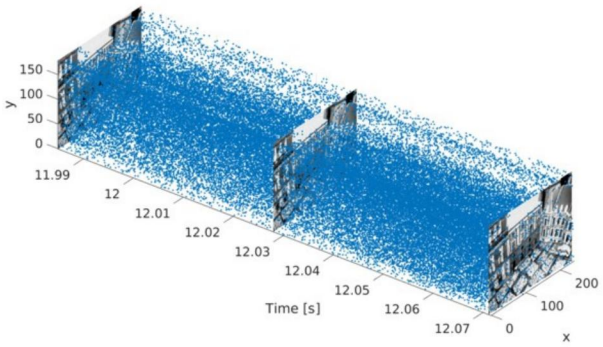

Fig. 1. An example of a typical event stream visualized in 3D. The events are an asynchronous and unordered set of points.

2 Related Work

2.1 Event Voxel Representation

Since the event data samples are streams of events, existing methods usually transform the raw event streams into image-like data that can be processed by CNNs. [1] proposed a way to convert the event data from a four-dimensional (x, y, t, p) structure, to a three-dimensional voxel grid by projecting or summing one of the four dimensions. It shows that the Event Voxel Grid representation performs well in classification and optical flow tasks. Specifically, the Event Voxel Grid is obtained by dividing event streams into a number of portions with equal temporal length and projecting all events within each portion onto individual image channels. If multiple events are projected onto the same pixel in a time channel, the intensity of that pixel accumulates based on the number of occurred events. Data of opposite polarities are handled separately and concatenated at the end of the processing step, doubling

the total number of channels. Therefore, for an event camera with a spatial resolution of m by n, and an event stream of ΔT seconds, we can turn it into an Event Voxel Grid with $2k$ channels of size m by n, each corresponds to its polarity and time channel for $\Delta T/k$ seconds. [1] also proposed another event data representation named the Event Spike Tensor (EST), which is similar to the Event Voxel Grid structure. In the EST representation, the pixel intensities of each time channel are replaced by the normalized timestamps of the corresponding events. This representation produces the same data dimensions as the Event Voxel Grid method and preserves as much temporal information within each time bin as possible.

2.2 Event Denoising

Recent work in denoising event data incorporates deep-learning approaches. Event Probability Mask (EPM) [2] presents a method for labeling real-world neuromorphic camera sensor data by calculating the likelihood of generating an event at each pixel within a short time window. It also proposed an event denoising CNN (EdCNN) which extracts features and does binary classfication on noise rejection.

EventZoom [3] is a recently proposed neural network approach for event denoising and super-resolution. It's able to effectively remove noisy events and achieves SOTA super-resolution image reconstruction while being 10x faster. EventZoom was built upon the 3D U-Net backbone which first downsamples the event tensor with convolution layers and then upsamples it with deconvolution layers until obtains the 2x feature map size. It also incorporated an E2I module, as a combination of an event-to-image reconstruction network E2VID [4] and an image super-resolution (SR) network FSRCNN, to leverage the RGB image information.

3 Theory

3.1 Background

An event-based camera is an advanced sensor that detects pixel-wise brightness change at microsecond level. Unlike a standard camera, the output of an event-based camera can be expressed as:

$$\mathbf{e}_t = \mathbf{\Phi}_{\boldsymbol{\epsilon}} \left(\log(\frac{I^t_{(x,y)}}{I^{t-1}_{(x,y)}}) \right), \tag{1}$$

where $\mathbf{e}_t$ is the event triggered at the pixel of interest (x, y) with intensity $I^t_{(x,y)}$ at time t due to $\log$ brightness change exceeding a threshold ϵ, and the function $\mathbf{\Phi}_{\boldsymbol{\epsilon}}$ is defined as:

$$\mathbf{\Phi}_{\boldsymbol{\epsilon}}(r_t) = \begin{cases} 1 & \text{if } r_t \geq \boldsymbol{\epsilon} \\ -1 & \text{else if } r_t \leq -\boldsymbol{\epsilon} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

In general, we consider the event with value $\mathbf{\Phi}_{\boldsymbol{\epsilon}}(r_t) = -1$ to be a negative event, and the event with value $\mathbf{\Phi}_{\boldsymbol{\epsilon}}(r_t) = 1$ to be positive event, which corresponds to decreasing and increasing brightness changes at the pixel of interest. We also refer this property as the "polarity" of an event in subsequent sections.

Practically, we collect a set of asynchronous events as:

$$\mathbf{E} = \{\mathbf{e}_i\}_{i=1}^N = \{(\mathbf{x}_i, \mathbf{y}_i, \mathbf{t}_i, \mathbf{p}_i)\}_{i=1}^N, \tag{3}$$

where $\mathbf{x}_i$ and $\mathbf{y}_i$ are the pixel coordinates, $\mathbf{t}_i$ is the timestamp, and $\mathbf{p}_i$ is the polarity of i-th event in an event stream.

A set of events can be converted to Event Voxels by dividing the spatial and temporal dimensions into evenly spaced bins. The value of each voxel is the number of events that fall into it. It is common practice to handle the two polarities separately, which leads to two set of voxels given an event set where one set of voxels contain the information only on the positive events, and the other set contains the information only on the negative events.

Due to potential camera shakings and random perturbations in the $\log$ brightness threshold, noise usually occurs in real-world captures and can be modeled as:

$$\tilde{\mathbf{e}}_t = \mathbf{\Phi}_{\boldsymbol{\epsilon}+n_{\boldsymbol{\epsilon}}} \left(\log(\frac{I^{t+n_t}_{(x,y)+n_s}}{I^{t-1}_{(x,y)+n_s}}) \right), \tag{4}$$

where n_s corresponds to the random spatial shifts of the events, n_t corresponds to the random temporal shifts of the events, and $n_{\boldsymbol{\epsilon}}$ corresponds to the random perturbations in the $\log$ brightness threshold. The random shifts often cause the captures to be blurry, while the random perturbations to the $\log$ brightness threshold will create new random events or miss some events from the captures.

3.2 Proposed Denoiser

Following EventZoom [3], the denoising network is built upon the 3D U-Net [5] backbone as shown in Figure 2. The input event stream is first voxelized to four-dimensional tensors (P, T, W, H) with each dimension representing the polarity, time, width, and height, respectively. We choose the 3D U-Net instead of its 2D counterparts for its larger receptive fields to capture information in time channels, which effectively improves its denoising capability for 4D data.

Our U-Net architecture comprises a downsampling path and an upsampling path each with four resolution steps. In the beginning, the input's polarity dimension only has 2 channels. As it forwards through the downsampling path, this dimension piles up, and the exact number of channels is given by c in Figure 2. Meanwhile, the input's spatial size decided by the width and height dimension is downscaled by a stride of two, and the accumulated stride corresponding to the original input size is given by s in the Figure. Since the time dimension is not as large as the spatial dimensions, it is kept consistent through the network. The upsampling path is the reverse operation of the above. Each step, it does skip connections by concatenating the output from the previous step with the same-resolution-data in the downsampling steps. The output is therefore the same shape as the input, containing voxels from both polarities.

4 Experimental Results

4.1 Datasets

We validate our method for event data denoising using two datasets: the **DVS128 Gesture Dataset** [6] and the **N-Caltech 101 Dataset** [7]. The reason for choosing these two datasets is that they are representatives of capturing two different kinds of movement. The DVS128 Gesture Dataset

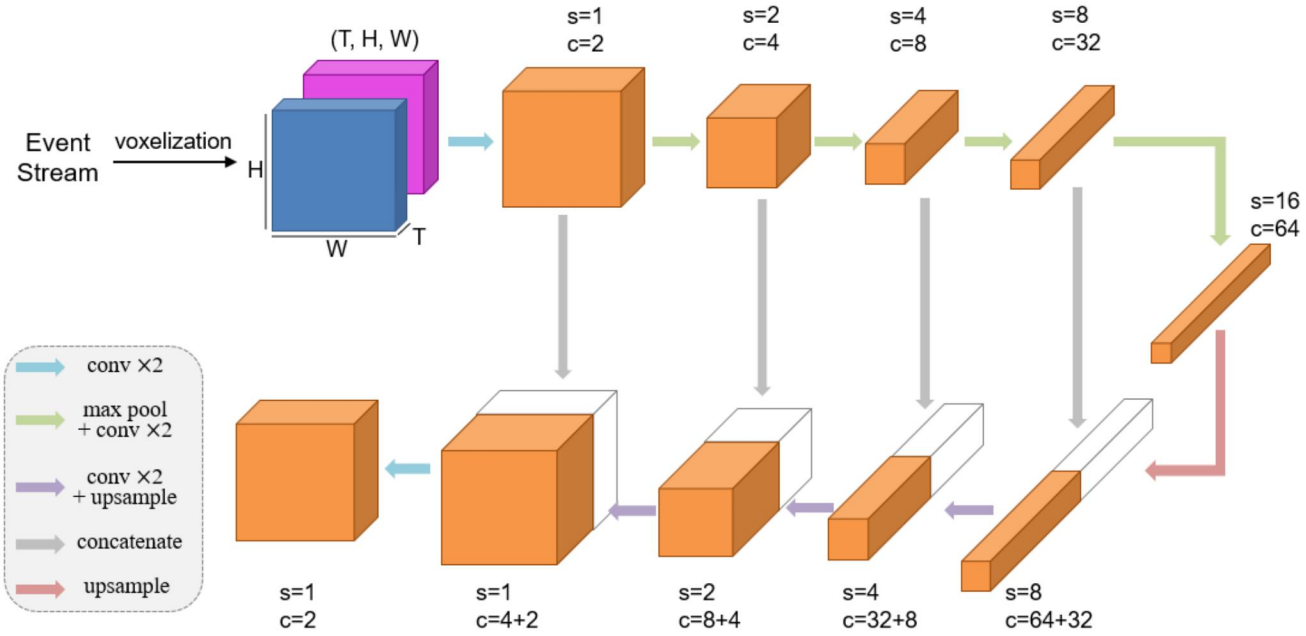

Fig. 2. The event denoising network employs a 3D U-Net backbone. First, the raw event stream is converted to 3D voxels for both polarities and then fed into the network as input. In each step of the downsampling path, the data's spatial dimension is halved, while the number of channels in the original polarity dimension doubles. The process is reversed during the upsampling path. Four skip connections concat the intermediate feature of the same resolution in each step. At last, the network outputs the denoised data as voxels with the same shape as the input for downstream tasks.

uses a fixed camera to record moving objects, while the N-Caltech 101 Dataset shifts the camera against a static picture to create relative movements.

4.1.1 *DVS128 Gesture Dataset*

The **DVS128 Gesture Dataset** [6] captures 11 different types of human gesture movements using a DVS128 (Dynamic Vision Sensor 128) Event Camera. The data was grouped in trials each recorded one subject stood against a stationary background and performed a sequence of the 11 gestures including 'arm roll', 'hand clap', 'air guitar', etc. The dataset was used to build the real-time, gesture recognition system described. We cut the event recordings into individual gesture performed by a user. Each gesture sample lasts for 6 seconds on average.

4.1.2 *N-Caltech 101 Dataset*

The **N-Caltech 101 Dataset** [7] is an event-based version of the original frame-based Caltech101 dataset. It contains 8,246 samples from 100 object classes plus a background class. The dataset was captured by moving the ATIS sensor in front of an LCD monitor projecting various samples from Caltech101. Each sample recording lasts for 1.2 seconds on average.

4.2 Noise Data Generation

As there is no existing dataset collected and designed for the task of event data denoising, we generate noisy event data by adding noises to the original data from both datasets. To simulate the noises n_ϵ on the log brightness threshold in equation 4, we randomly remove and add events from the original event stream using uniform distribution sampling across the feasible spatial event coordinates. We add random shifts in Gaussian distribution to the existing events in both spatial and temporal directions to model the noises n_s and n_t in equation 4. The events after random shifts can be expressed in the following form:

$$\mathbf{E} = \{\mathbf{e}_i\}_{i=1}^{N} = \{(\mathbf{x}_i + \delta_s, \mathbf{y}_i + \delta_s, \mathbf{t}_i + \delta_t, \mathbf{p}_i)\}_{i=1}^{N}, \quad (5)$$

where $\delta_s \sim \mathcal{N}(0, \sigma_s)$ is the noise sampled in spatial domain with a zero-mean Gaussian and standard deviation σ_s, as well as $\delta_t \sim \mathcal{N}(0, \sigma_t)$ corresponds to the noise in sampled temporal domain with zero-mean Gaussian and standard deviation σ_t.

4.3 Implementation Details

We train our 3D U-Net denoiser in a self-supervised fashion where we add noise to the original event data, and let the 3D U-Net reconstruct the clean event voxels. We use the Mean Squared Error (MSE) Loss in training. The 3D U-Net is trained for 30 epochs with the Adam [8] optimizer. As both datasets are designed for multi-class classification, we use the convolutional layers of a pre-trained ResNet-34 followed by a linear classifier as the baseline classifier for comparison and evaluation. Note that we replace the first layer in the pretrained ResNet-34 to accommodate for the change input channel dimension from 3 channels to the number of bins along in the temporal dimension. The classifiers are also trained for 30 epochs with the the Adam [8] optimizer. We use voxel size of $9 \times 180 \times 240$ and $9 \times 130 \times 130$ for the N-Caltech 101 Dataset and the DVS128 Gesture Dataset, respectively. For evaluation, we freeze the parameters of the denoiser and only do forward passes from the data input for all testing setups. The denoiser's output voxels

are treated as input to our ResNet-34-based classifiers. For noise generation, we specifically delete 20% of total events randomly and add 50% of total events by random sampling across the entire resolutions. Then we randomly select and shift 80% of total events with a spatial shift sampled from a zero-mean Gaussian of 20 pixels standard deviation, and a temporal shift sampled from a zero-mean Gaussian of 0.0001 seconds standard deviation.

4.4 Qualitative Results

We show qualitative results on both datasets in Figure 3 and 4. The denoised data exhibits a similar data distribution as the original data. While the noisy data can hardly be distinguished by human eyes, the denoised image can present the object shapes or moving trajectory quite clearly, showing a great de-blurring capability. An interesting observation is that the original data is not perfectly clean as there are noise events that do not fall on the actual objects in the scene. Our denoising model can also identify them as noisy events and remove them, although they are treated as ground-truth events in training.

4.5 Quantitative Results

For quantitative evaluations, we report the multi-class classification accuracies on both DVS128 Gesture [6] and N-Caltech 101 [7] datasets. As we generate the noises ourselves, we have the options of testing the trained classifier using either the original input data or the noisy input data. Therefore, we evaluate our denoising method under the following two scenarios.

4.5.1 Direct Inference on Noisy Test set

We first train a ResNet-34-based classifier using the original data, and directly run inference on the noisy test sets from both datasets. As shown in Table 1 and Table 2, we observe significant drops in classification accuracy on both datasets. The reason for such large performance degradations is that the classifier trained on the original event data cannot handle the out-of-distribution noisy input. As shown in the tables, if we run our trained denoiser first on the noisy data, and feed the denoised event voxels into the classifier trained on the original data, it successfully boosted the performance on classification back to the similar level as tested on the original data *without any additional training*. It indicates that our proposed denoiser model can successfully estimate the distribution of the original event voxels, which allows the classifier trained on the original data to achieve the much higher testing performance on the noisy test set. If we use our denoiser on the original test set and the trained classifier, we can also maintain competitive performance, which shows that the denoiser can also handle the original data. Overall, Table 1 and Table 2 show that our proposed 3D U-Net can be used directly to address the distribution shifts from the training data to the noisy data without the requirements of re-training the baseline classifiers.

4.5.2 Training on Noisy Training Set

We also test the effectiveness of our 3D U-Net denoiser in the classifier training using noisy data. Specifically, we compare ResNet-34 based classifier trained on the noisy data and ground-truth class label pairs, against the setup where we train the classifier of the same architecture with the denoised input data and ground-truth class labels pairs. Note that we use our trained denoiser to get the denoised data from noisy data, and during the training, the 3D U-Net denoiser is not updated. We show the results of the two setups in Table 3. Using the trained denoiser can improve the final classification accuracies on both datasets as it is able to provide cleaner input data for feature extraction and pattern recognition.

TABLE 1
Evaluation on N-Caltech 101 test set using a trained classifier and denoising 3D U-Net without additional training. The numbers are reported as average classification accuracy (%).

	Method	Original Test Set	Noisy Test Set
(1)	Baseline Classifier	85.01	5.89
(2)	Ours + (2)	66.89	66.96

TABLE 2
Evaluation on DVS128 Gesture test set using a trained classifier and denoising 3D U-Net without additional training. The numbers are reported as average classification accuracy (%).

	Method	Original Test Set	Noisy Test Set
(1)	Baseline Classifier	93.36	17.32
(2)	Ours + (2)	71.61	87.24

TABLE 3
Evaluation on N-Caltech 101 and DVS128 Gesture noisy test sets with training on the noisy training set. The numbers are reported as average classification accuracy (%).

Method	N-Caltech 101	DVS128 Gesture
Classifier only	71.63	89.45
Ours + Classifier	77.24	91.02

5 Discussion and Conclusion

5.1 Limitations and Future Work

Despite our proposed approach achieves great classification results in datasets with moving and static objects, further work is needed to evaluate the downstream performance on the tasks that require more fine-grained denoising such as optical flow estimation and depth estimation. In addition, our denoiser is trained from a fixed noise level. We could further augment the noise generation by adding different levels of noise so that the denoiser is generalized to various noise conditions. We are also interested in using the downsampling part of the 3D U-net as a feature extractor as it is trained to preserve event features from various noise scenarios, and potentially use it in a more complex scene such as in event driving recordings.

5.2 Conclusion

To summarize, we proposed a 3D U-Net denoiser that is trained in a self-supervised fashion and can effectively estimate clean data from a noisy event voxel. With the help of

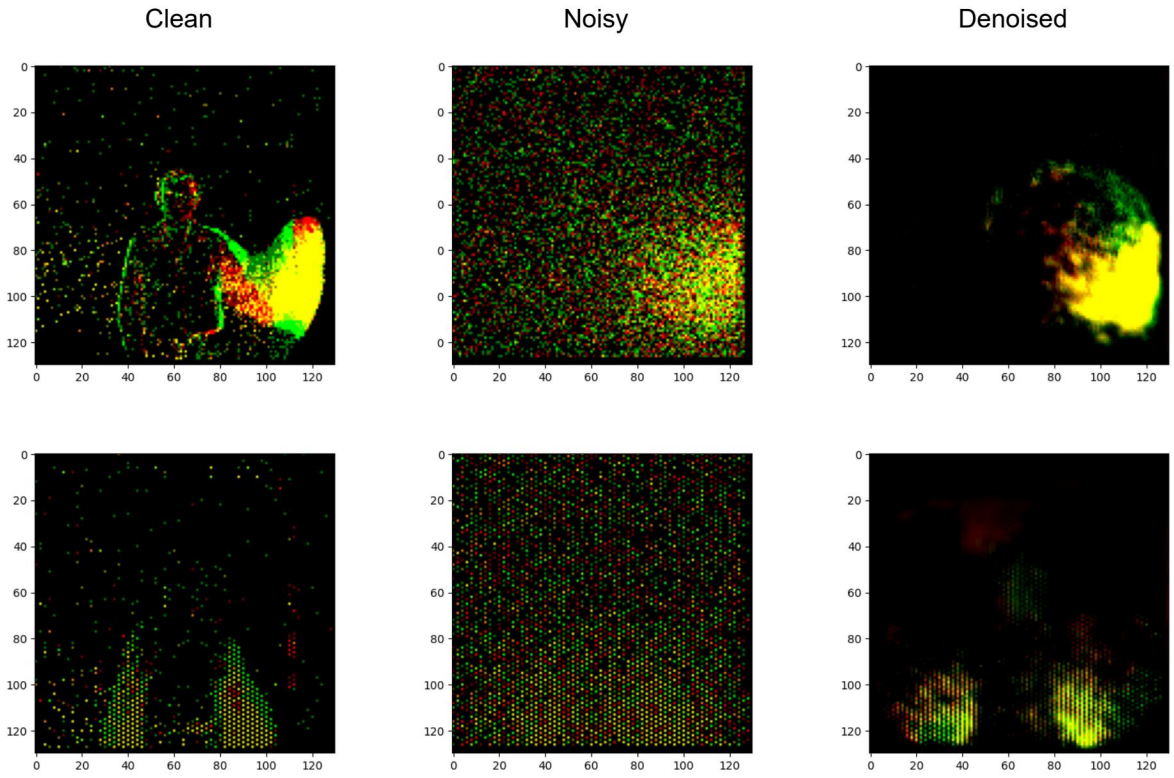

Fig. 3. Denoising results shown as a single time channel (1/39 slice of total temporal length) event image from DVS128 Gesture dataset. Red pixels represent positive events, green pixels represent negative events, and yellow pixels indicate that events of both polarities are present.

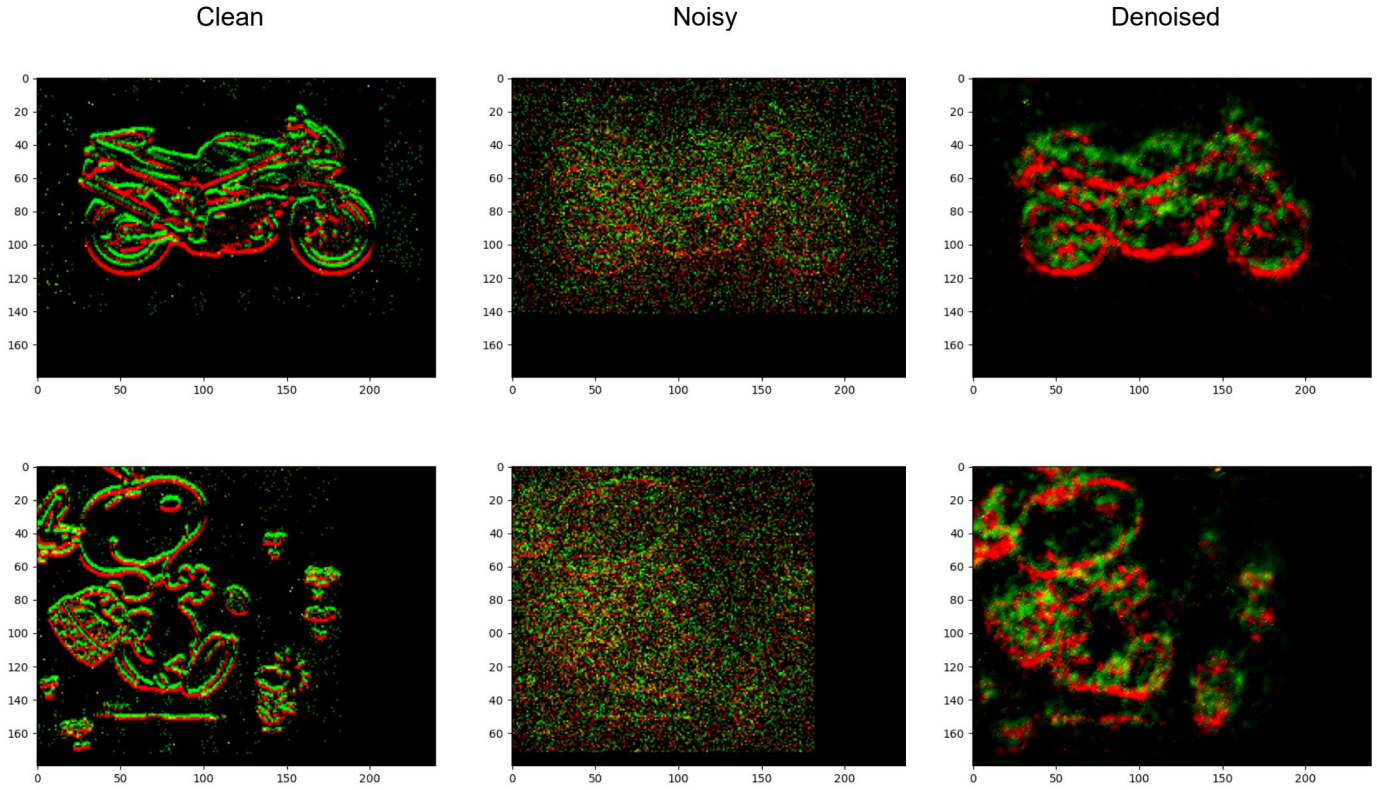

Fig. 4. Denoising results shown as a single time channel (1/9 slice of total temporal length) event image from N-Caltech101 dataset. Red pixels represent positive events, green pixels reprsent negative events, and yellow pixels indicate that events of both polarities are present.

our pre-trained denoiser, classifiers trained on clean datasets are able to gain much higher test accuracy on unseen noisy datasets. Moreover, classifiers trained on noisy datasets can also achieve marginal performance boost with our denoiser. In addition, our denoiser shows robust performance in multi-class classification tasks on both dynamic and static object datasets. We can improve our denoiser by adding various levels of noise during training, and we leave the test of the denoiser on more downstream tasks to future work.

Acknowledgments

We would like to thank Parsa Mirdehghan for his valuable suggestions on our project. We also thank Professor David Lindell and all the teaching assistants for putting together this great course.

References

[1] D. Gehrig, A. Loquercio, K. G. Derpanis, and D. Scaramuzza, "End-to-end learning of representations for asynchronous event-based data," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 5633–5643.

[2] R. Baldwin, M. Almatrafi, V. Asari, and K. Hirakawa, "Event probability mask (epm) and event denoising convolutional neural network (edncnn) for neuromorphic cameras," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1701–1710.

[3] P. Duan, Z. W. Wang, X. Zhou, Y. Ma, and B. Shi, "Eventzoom: Learning to denoise and super resolve neuromorphic events," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 824–12 833.

[4] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza, "Events-to-video: Bringing modern computer vision to event cameras," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3857–3866.

[5] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: learning dense volumetric segmentation from sparse annotation," in *International conference on medical image computing and computer-assisted intervention*. Springer, 2016, pp. 424–432.

[6] A. Amir, B. Taba, D. J. Berg, T. Melano, J. L. McKinstry, C. di Nolfo, T. K. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza, J. A. Kusnitz, M. V. DeBole, S. K. Esser, T. Delbrück, M. Flickner, and D. S. Modha, "A low power, fully event-based gesture recognition system," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7388–7397, 2017.

[7] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, "Converting static image datasets to spiking neuromorphic datasets using saccades," *Frontiers in neuroscience*, vol. 9, p. 437, 2015.

[8] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.