Asynchronous Convolutional Networks for Object Detection in Neuromorphic Cameras Supplementary material

Marco Cannici Marco Ciccone Andrea Romanoni Matteo Matteucci
Politecnico di Milano, Italy
{marco.cannici,marco.ciccone,andrea.romanoni,matteo.matteucci}@polimi.it

In this document we describe our novel event-based datasets adopted in the paper "Asynchronous Convolutional Network for Object Detection in Neuromorphic Cameras".

1. Event-based object detection datasets

Due to the lack of object detection datasets with event cameras, we extended the publicly available N-MNIST, MNIST-DVS, Poker-DVS and we propose a novel dataset based on MNIST, i.e., Blackboard MNIST. They will be soon released, however, in Figure 1 we reported some example from the four datasets.

1.1. Shifted N-MNIST

The original N-MNIST [5] extends the well-known MNIST [3]: it provides an event-based representation of both the full training set ($60,000$ samples) and the full testing set ($10,000$ samples) to evaluate object classification algorithms. The dataset has been recorded by means of event camera in front of an LCD screen and moved to detect static MNIST digits displayed on the monitor. For further details we refer the reader to [5].

Starting from the N-MNIST dataset, we built a more complex set of recordings that we used to train the object detection network to detect multiple objects in the same scene. We created two versions of the dataset, Shifted N-MNIST v1 and Shifted N-MNIST v2, that contains respectively one or two non overlapping 34×34 N-MNIST digits per sample randomly positioned on a bigger surface. We used different surface dimensions in our tests which vary from double the original size, 68×68, up to 124×124. The dimension and structure of the resulting dataset is the same of the original N-MNIST collection.

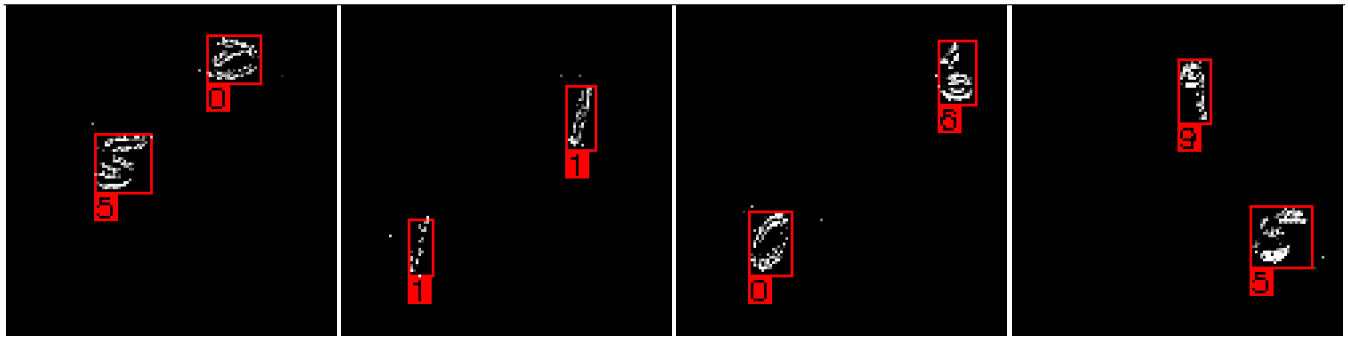
To extend the dataset for object detection evaluation, the bounding boxes ground truth are required. To estimate them we first integrate events into a single frame as described in Section 2 of the original paper. We remove the noise by considering only non-zero pixels having at least other ρ non-zero pixels around them within a circle of radius R. All the other pixels are considered noise. Then, with a custom version of the DBSCAN [2] density-based clustering algorithm we group pixels into a single cluster. A threshold min_{area} is used to filter out small bounding boxes extracted in correspondence of low events activities. This condition usually happens during the transition from a saccade to the next one as the camera remains still for a small fraction of time and no events are generated. We used $\rho = 3$, $R = 2$ and $min_{area} = 10$. The coordinates of these bounding boxes are then shifted based on the final position the digit has in the bigger field of view.

For each N-MNIST sample, another digit was randomly selected in the same portion of the dataset (training, testing or validation) to form a new example. The final dataset contains $60,000$ training samples and $10,000$ testing samples, as for the original N-MNSIT dataset. In Figure 2 we illustrate one example for v1 and the three variants of v2 we adopted (and described) in the paper.

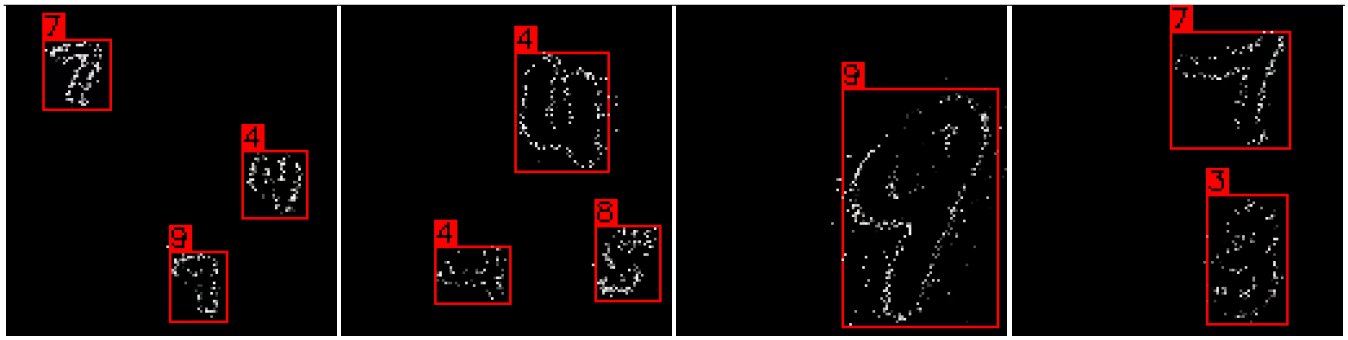
1.2. Shifted MNIST-DVS

The MNIST-DVS dataset [6] is another collection of event-based recordings that extends MNIST [3]. It consists of $30,000$ samples recorded by displaying digits on an screen in front of a event camera, but differently from N-MNIST, they move digits on the screen instead of the sensors, and they use the digits at three different scales, i.e., *scale4*, *scale8* and *scale16*. The resulting dataset is composed of $30,000$ event-based recordings showing each one of the selected $10,000$ MNIST digits on thee different dimensions. Examples of these recordings are shown in Figure 3.

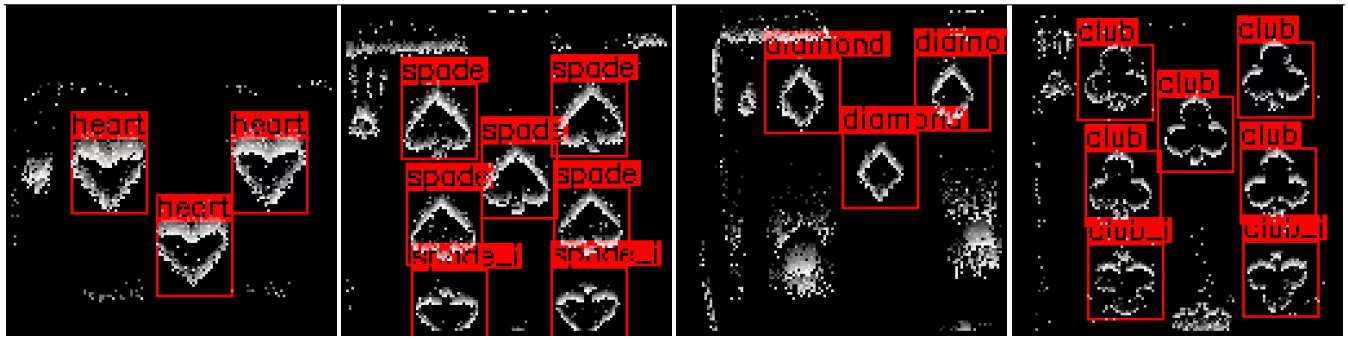
We used MNIST-DVS recordings to build a detection dataset by means of a procedure similar to the one we used to create the Shifted N-MNIST dataset. However in this case we mix together digits of multiple scales. All the MNIST-DVS samples, despite of the actual dimensions of the digits being recorded, are contained within a fixed 128×128 field of view. Digits are placed centered inside



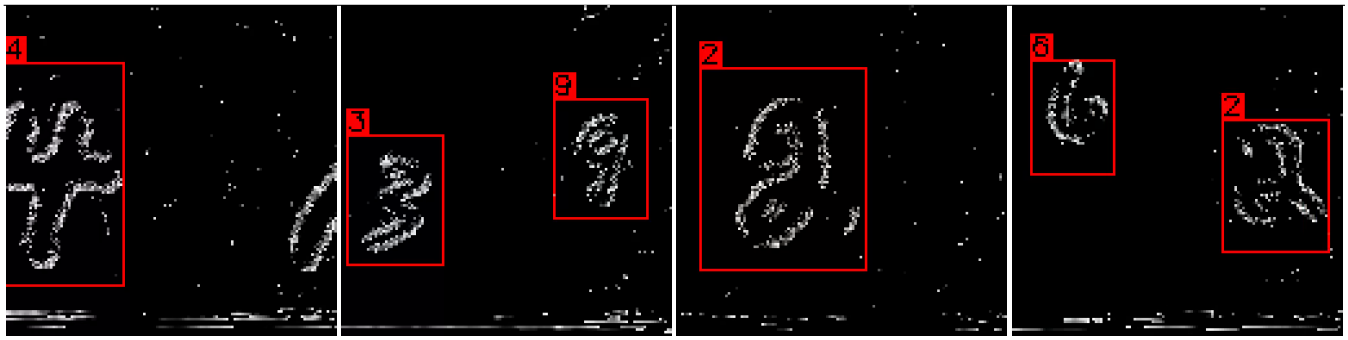
Shifted N-MNIST



Shifted MNIST-DVS

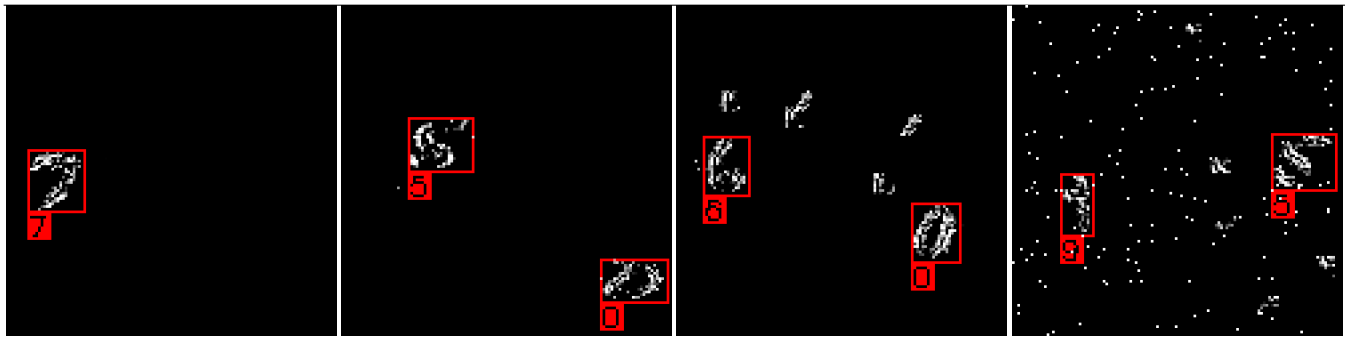


OD-Poker-DVS



Blackboard-MNIST

Figure 1: Examples of samples from the proposed datasets.



v1 v2 v2fr v2fr+ns

Figure 2: Different versions of Shifted N-MNIST.

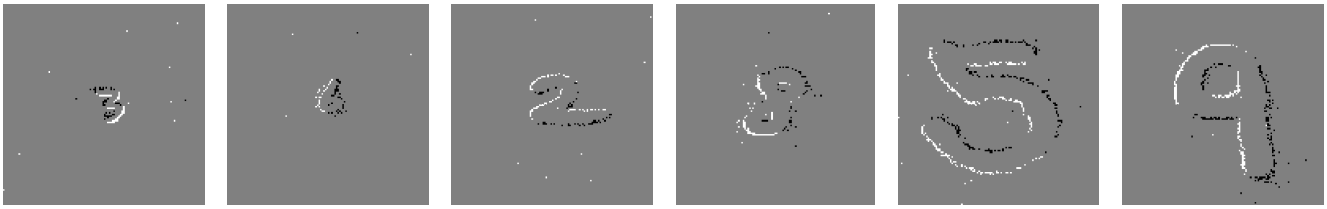


Figure 3: Examples of the three different scales of MNIST-DVS digits. Two samples at scale *scale4*, two at *scale8* and two at *scale16*.

the scene and occupy a limited portion of the frame, especially those belonging to the smallest and middle scales. In order to place multiple examples on the same scene we first cropped the three scales of samples into smaller recordings occupying 35×35, 65×65 and 105×105 spatial regions respectively. The bounding boxes annotations and the final examples were obtained by means of the same procedure we used to construct the Shifted N-MNIST dataset. These recordings were built by mixing digits of different dimensions in the same sample. Based on the original samples dimensions, we decided to use the following four configurations (which specify the number of samples of each category used to build a single Shifted MNIST-DVS example): (i) three *scale4* digits, (ii) two *scale8* digits, (iii) two *scale4* digits mixed with one *scale8* digit (iv) one *scale16* digit placed in random locations of the field of view. The overall dataset is composed of $30,000$ samples containing these four possible configurations.

1.3. OD-Poker-DVS

The original Poker-DVS [6] have been proposed to test object recognition algorithms; it is a small collection of neuromorphic recordings obtained by quickly browsing custom made poker card decks in front of a DVS camera for 2-4 seconds. The dataset is composed of 131 samples containing centered pips of the four possible categories (spades, hearts, diamonds or clubs) extracted from three decks recordings. Single pips were extracted by means of an event-based tracking algorithms which was used to follow symbols inside the scene and to extract 31×31 pixels examples.

With OD-Poker-DVS we extend its scope to test also object detection. To do so we used the event-based tracking algorithm provided with the original dataset to follow the movement of the 31×31 samples in the uncut recordings and extract their bounding boxes. The final dataset was obtained using a procedure similar to the one used in [7]. Indeed, we divided the sections of the three original decks recordings containing visible digits into a set of shorter examples, each of which about 1.5ms long. Examples were split in order to ensure approximately the same number of objects (i.e., ground truth bounding boxes) in each example. The final detection dataset is composed of 292 small examples which we divided into 218 training and 74 testing samples.

Even if composed of a limited amount of samples, this dataset represents an interesting real-world application that highlights the potential of event-based vision sensors. The nature of the data acquisition is indeed particularly well suited to neuromorphic cameras due to their very high temporal resolution. Symbols are clearly visible inside the recordings even if they move at very high speed. Each pip, indeed, takes from 10 to 30 ms to cross the screen but it can

be easily recognized within the first 1-2 ms.

1.4. Blackboard MNIST

The two dataset based on MNIST presented in Section 1.1 and 1.2 have the drawback of recording digits at predefined sizes. Therefore, in Blackboard MNIST we propose a more challenging scenario that consists of a number of samples showing digits (from the original MNIST dataset) written on a blackboard in random positions and with different scales.

We used the DAVIS simulator released by [4] to build our own set of synthetic recordings. Given a three-dimensional virtual scene and the trajectory of a moving camera within it, the simulator is able to generate a stream of events describing the visual information captured by the virtual camera. The system uses Blender [1], an open-source 3D modeling tool, to generate thousands of rendered frames along a predefined camera trajectory which are then used to reconstruct the corresponding neuromorphic recording. The intensity value of each single pixel inside the sequence of rendered frames, captured at a constant frame-rate, is tracked. As Figure 4a shows, an event is generated whenever the log-intensity of a pixel crosses an intensity threshold, as in a real event-based camera. A piecewise linear time interpolation mechanism is used to determine brightness changes in the time between frames in order to simulate the microseconds timestamp resolution of a real sensor. We extended the simulator to output bounding boxes annotations associated to every visible digit.

We used Blender APIs to place MNIST digits in random locations of a blackboard and to record their position with respect to the camera point of view. Original MNIST images depict black handwritten digits on a white background. To mimic the chalk on the blackboard, we removed the background, we turned digits in white and we roughen their contours to make them look like if their were written with a chalk. An example is shown in Figure 4b.

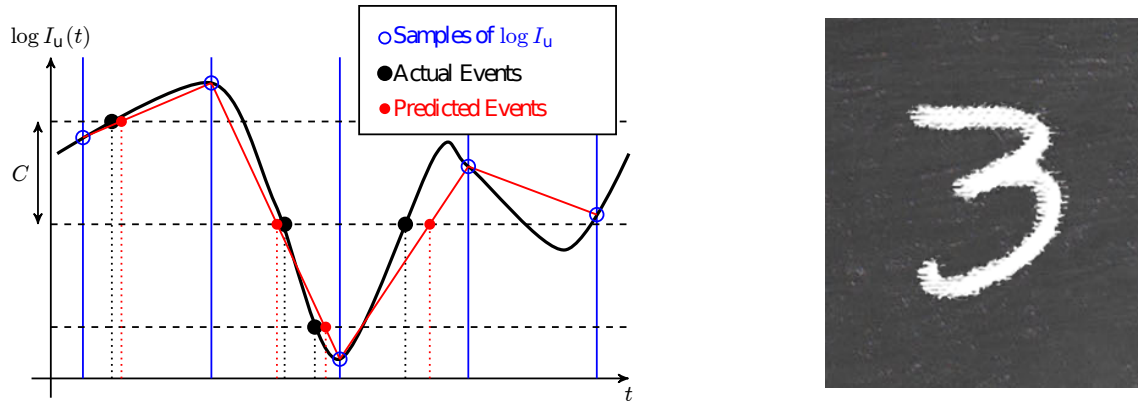
The scene contains the image of a blackboard on a vertical plane and a virtual camera with 128×128 resolution that moves horizontally on a predefined trajectory parallel to the blackboard plane (see Figure 5). The camera points a hidden object that moves on the blackboard surface, synchronized with the camera, following a given trajectory. To introduce variability in the camera movement, and to allow all the digits outline to be seen (and possibly detected), we used different trajectories that vary from a straight path to a smooth or triangular undulating path that makes the camera tilt along the transverse axis while moving (Figure 5b).

Before starting the simulation, we randomly select a number of preprocessed MNIST digits and place them in a random portion of the blackboard. The camera moves so that all the digits will be framed during the camera movement. The simulation is then finally started on this modified scene to generate neuromorphic recordings. Every time a frame is rendered during the simulation, the bounding boxes of all the visible digits inside the frame are also extracted. This operation is performed by computing the camera space coordinates (or normalized device coordinates) of the top-left and bottom-right vertex of all the images inside the field of view. Since images are slightly larger than the actual digits they contain, we cropped every bounding box to better enclose each digit and also to compensate the small offset in the pixels position introduced by the camera motion and by the linear interpolation mechanism. In addition, bounding boxes corresponding to objects which are only partially visible are also filtered out. In order to build the final detection dataset, this generation process is executed multiple times, each time with different digits.

We built three sub-collections of recordings with increasing level of complexity which we merged together to obtain our final dataset: *Blackboard MNIST EASY*, *Blackboard MNIST MEDIUM*, *Blackboard MNIST HARD*. In Blackboard MNIST EASY, we used digits of only one dimension (roughly corresponding to the middle scale of MNIST-DVS samples) and a single type of camera trajectory which moves the camera from right to left with the focus object moving in a straight line. In addition, only three objects were placed on the blackboard using only a fixed portion of its surface. We collected a total of $1,200$ samples ($1,000$ training, 100 testing, 100 validation).

Blackboard MNIST MEDIUM features more variability in the number and dimensions of the digits and in the types of camera movements. Moreover, the portion of the blackboard on which digits were added varies and may cover any region of the blackboard, even those near its edges. The camera motions were also extended to the set of all possible trajectories that combine either left-to-right or right-to-left movements with variable paths of the focus object. We used three types of trajectories for this object: a straight line, a triangular path or a smooth curved trajectory, all parallel to the camera trajectory and placed around the position of the digits on the blackboard. One of these path was selected randomly for every generated sample. Triangular and curved trajectories were introduced as we noticed that sudden movements of the camera produce burst of events that we wanted our detection system to be able to handle. The number and dimensions of the digits were chosen following three possible configurations, similarly to the Shift MNIST-DVS dataset: either six small digits (with sizes comparable to *scale4* MNIST-DVS digits), three intermediate-size digits (comparable to the MNIST-DVS *scale8*) or two big digits (comparable to the biggest scale of the MNIST-DVS dataset, *scale16*). A set of $1,200$ recordings was generated using the same splits of the first variant and with equal amount of samples in each one of the three configurations.

Finally, Blackboard MNIST HARD contains digits



(a) (b)

Figure 4: *(a)* The image shows in black the intensity, expresses as $\log I_u(t)$, of a single pixel $\mathbf{u} = (x, y)$. This curve is sampled at a constant rate when frames are generated by Blender, shown in figure as vertical blue lines. The sampled values thus obtained (blue circles) are used to approximate the pixel intensity by means of a simple piecewise linear time interpolation (red line). Whenever this curve crosses one of the threshold values (horizontal dashed lines) a new event is generated with the corresponding predicted timestamp. (Figure from [4]) **(b)** A preprocessed MNIST digit on top of the blackboard's background.

(a) (b)

Figure 5: **(a)** The 3D scene used to generate the Blackboard MNIST dataset. The camera moves in front of the blackboard along a straight trajectory while following a *focus object* that moves on the blackboard's surface, synchronized with the camera. The camera and its trajectory are depicted in green, the focus object is represented as a red cross and, finally, its trajectory is depicted as a yellow line. **(b)** The three types of focus trajectories.

recorded by using the second and third configuration of objects we described previously. However, in this case each image was resized to a variable size spanning from the original configuration size down to the previous scale. A total of 600 new samples (500 training, 50 testing, 50 validation) were generated, 300 of them containing three digits each and the remaining 300 consisting of two digits with variable size.

The three collections can be used individually or jointly; the whole Blackboard MNIST dataset contains $3,000$ samples in total (2500 training, 250 testing, 250 validation). Examples of different objects configurations are shown in Figure 6. Samples were saved by means of the AEDAT v3.1 file format for event-based recordings.

2. Results

Table 1 provides a comparison between the average precision of YOLE and fcYOLE on N-Caltech101 classes. We also provide a qualitative comparison between the two models in the video attachment.

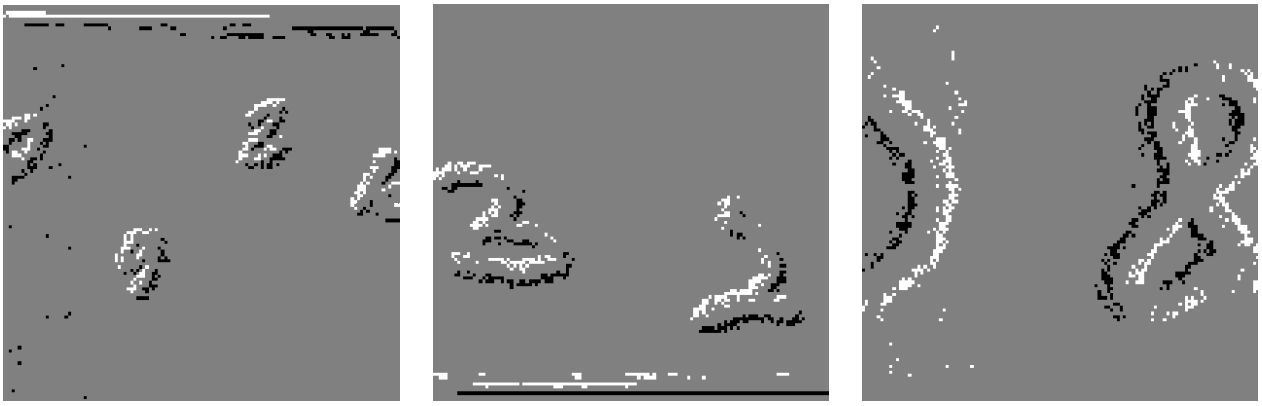


Figure 6: Examples of the three types of objects configurations used to generate the second collection of the Blackboard MNIST dataset.

Table 1: YOLE and fcYOLE average precisions on N-Caltech101

	Motorbikes	airplanes	Faces_easy	watch	Leopards	chair	bonsai	car_side	ketch	flamingo	ant	chandelier	crocodile	grand_piano	brain	hawksbill	butterfly	helicopter	menorah	starfish
AP fcYOLE	97.5	96.8	92.2	75.7	57.2	7.5	30.2	70.3	42.3	2.3	2.4	34.8	0.0	69.5	35.3	19.6	33.5	8.6	67.7	23.2
AP YOLE	97.8	95.8	94.7	84.2	62.9	17.3	59.3	61.7	52.9	10.0	25.8	55.7	1.6	81.3	53.3	29.1	46.3	14.9	80.7	32.7
N_{train}	480	480	261	145	120	109	78	75	70	68	66	65	61	61	60	60	55	54	53	52
	scorpion	kangaroo	trilobite	sunflower	buddha	ewer	revolver	laptop	llama	ibis	minaret	umbrella	crab	electric_guitar	cougar_face	dragonfly	crayfish	dalmatian	ferry	euphonium
AP fcYOLE	2.5	3.4	41.2	29.1	46.5	35.3	20.3	40.0	1.4	1.5	59.5	61.0	5.0	23.2	21.8	55.6	7.3	24.9	29.7	43.5
AP YOLE	6.9	5.0	62.5	43.3	57.2	51.3	57.4	88.1	10.2	6.5	81.3	85.9	19.5	29.7	39.8	59.9	9.5	33.0	34.0	53.6
N_{train}	52	52	52	51	51	51	50	49	48	48	46	45	45	45	43	42	42	41	41	40
	lotus	stop_sign	joshua_tree	soccer_ball	elephant	schooner	dolphin	lamp	stegosaurus	rhino	wheelchair	cellphone	yin_yang	cup	sea_horse	pyramid	windsor_chair	hedgehog	bass	nautilus
AP fcYOLE	18.1	55.1	30.2	57.3	11.4	37.3	6.5	17.7	34.5	5.8	25.8	46.5	60.4	5.6	1.9	41.1	52.3	13.3	4.2	13.0
AP YOLE	27.6	61.7	29.8	51.5	6.0	56.8	11.5	45.3	44.6	11.3	25.3	54.6	63.3	17.5	8.8	48.6	65.2	9.8	4.0	50.4
N_{train}	40	40	40	40	40	39	39	37	37	37	37	37	36	35	35	35	34	34	34	33
	pizza	emu	accordion	dollar_bill	tick	crocodile_head	gramophone	rooster	camera	pagoda	cougar_body	barrel	ceiling_fan	beaver	cannon	mandolin	flamingo_head	brontosaurus	stapler	pigeon
AP fcYOLE	17.4	5.6	48.3	74.4	28.2	9.8	30.6	26.6	15.1	34.0	0.0	30.7	40.8	0.5	0.0	6.6	2.7	0.2	46.0	6.2
AP YOLE	54.5	5.0	52.7	86.5	55.2	10.0	48.4	64.5	32.7	33.3	12.2	41.2	50.1	0.0	0.3	43.4	9.3	25.8	68.1	43.1
N_{train}	33	33	33	32	31	31	31	31	30	29	29	29	29	28	27	27	27	27	27	27
	headphone	anchor	scissors	wrench	okapi	lobster	panda	saxophone	mayfly	water_lilly	garfield	wild_cat	gerenuk	platypus	binocular	octopus	strawberry	snoopy	metronome	inline_skate
AP fcYOLE	10.7	14.6	28.2	14.7	10.0	0.0	4.7	59.7	0.5	3.1	23.4	0.0	4.8	13.1	0.4	14.0	0.5	8.3	63.4	37.2
AP YOLE	21.1	17.3	47.5	29.7	44.6	0.0	12.2	68.4	0.7	14.7	62.3	0.0	7.2	34.7	11.8	13.8	29.4	53.1	88.3	75.1
N_{train}	26	26	25	25	25	25	24	24	24	23	22	22	22	22	21	21	21	21	20	19

References

[1] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam, 2017. 4

[2] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages

226–231. AAAI Press, 1996. 1

[3] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, Nov 1998. 1

[4] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza. The Event-Camera Dataset and Simulator: Event-based Data for Pose Estimation, Visual Odometry, and SLAM. *arXiv*, Oct 2016. 4, 5

[5] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor. Converting Static Image Datasets to Spiking Neuromorphic Datasets Using Saccades. *Front. Neurosci.*, 9, Nov 2015. 1

[6] T. Serrano-Gotarredona and B. Linares-Barranco. Poker-DVS and MNIST-DVS. Their History, How They Were Made, and Other Details. *Front. Neurosci.*, 9, Dec 2015. 1, 3

[7] E. Stromatias, M. Soto, T. Serrano-Gotarredona, and B. Linares-Barranco. An Event-Driven Classifier for Spiking Neural Networks Fed with Synthetic or Dynamic Vision Sensor Data. *Front. Neurosci.*, 11, Jun 2017. 3