NEUROMORPHIC VISION SENSING FOR CNN-BASED ACTION RECOGNITION

A. Chadha, Y. Bi, A. Abbas, Y. Andreopoulos

Dept. of Electronic & Electrical Engineering
University College London
London, U.K.

ABSTRACT

Neuromorphic vision sensing (NVS) hardware is now gaining traction as a low-power/high-speed visual sensing technology that circumvents the limitations of conventional active pixel sensing (APS) cameras. While object detection and tracking models have been investigated in conjunction with NVS, there is currently little work on NVS for higher-level semantic tasks, such as action recognition. Contrary to recent work that considers homogeneous transfer between flow domains (optical flow to motion vectors), we propose to embed an NVS emulator into a multi-modal transfer learning framework that carries out heterogeneous transfer from optical flow to NVS. The potential of our framework is showcased by the fact that, for the first time, our NVS-based results achieve comparable action recognition performance to motion-vector or optical-flow based methods (i.e., accuracy on UCF-101 within 8.8% of I3D with optical flow), with the NVS emulator and NVS camera hardware offering 3 to 6 orders of magnitude faster frame generation (respectively) compared to standard Brox optical flow. Beyond this significant advantage, our CNN processing is found to have the lowest total GFLOP count against all competing methods (up to 7.7 times complexity saving compared to I3D with optical flow).

***Index Terms*—** neuromorphic vision sensing, transfer learning, knowledge distillation

1. INTRODUCTION

Machine learning with visual data has been described as the means to translate "pixels to concepts" [1], e.g., classify active pixel sensor (APS) video according to its illustrated human activity ("tennis match", "cooking", "people marching",...). However, APS-based video representations are known to be cumbersome for machine learning systems, due to [2]: limited frame rate, too much redundancy between successive frames, calibration problems under irregular camera motion, blurriness due to shutter adjustment under varying illumination, and very high power requirements. Inspired by these observations, hardware designs of neuromorphic sensors, a.k.a., silicon retinas [3, 4], have been proposed recently. Silicon retinas mimic the photoreceptor-bipolar-ganglion cell information flow of biological retinas by producing coordinates and timestamps of on/off spikes in an asynchronous manner, i.e., when the logarithm of the intensity value of a CMOS sensor grid position changes beyond a threshold due to scene luminance changes. Unlike conventional frame-based cameras that tend to blur the image due to slow shutter speed, silicon retinas capture the illumination changes caused by fast object motion and are inherently differential in nature. In practice, this means that neuromorphic vision sensing (NVS) data from hardware like the iniLabs DAVIS and the Pixium Vision ATIS cameras [4, 5, 6, 7] can be rendered to representations comprising up to 2000 frames-per-second (fps), whilst operating with robustness to changes in lighting and at low power, on the order of 10mW. Conversely, a typical APS video camera only captures (up to) 60 fps at more than 20 times the active power consumption and with shutter-induced blurriness artifacts when rapid illumination changes take place. The combination of these advantages makes NVS-based sensing particularly appealing within Internet-of-Things (IoT) and robotics contexts [8], where NVS data would be gathered at very low power and streamed to cloud computing servers for back-end analysis with deep convolutional neural networks (CNNs).

Despite these practical advantages, it has been widely recognized [2, 3, 4] that:

1. Progress in neuromorphic-based action recognition algorithms is severely hampered by the lack of NVS data for both training and inference.

2. There are currently no NVS-based recognition frameworks for large-scale multi-class human action recognition problems corresponding to datasets like UCF-101 and HMDB [9, 10].

This paper addresses these two issues. By converting pixel-domain video frames directly to the spike domain with an NVS emulator, we are able to provide an abundance of data

We acknowledge support from: EPSRC (projects EP/P02243X/1 and EP/R025290/1), The Leverhulme Trust (RAEng/Leverhulme Senior Research Fellowship of Y. Andreopoulos), the Royal Commission for the Exhibition of 1851 (Fellowship of A. Chadha) and EPSRC CASE (PhD studentship of A. Chadha, cosponsored by BAFTA).

on which to train a CNN in the NVS domain. For the action recognition task, we show how any such emulator can be embedded into a larger multi-modal teacher-student framework, where we capitalize on the availability of optical flow labelled data by employing a pre-trained optical flow stream as a teacher network to transfer knowledge to the emulated NVS student network. Our experimental results show that, for the first time, NVS-based CNNs can approach the accuracy of complex methods based on optical flow extraction from APS video, thus making NVS inputs a very competitive alternative to APS-based cameras and optical flow extraction for low-power IoT and robotics applications requiring action recognition.

2. RELATED WORK

Recent work [3, 8], has advocated possibilities for ingesting NVS data into frame-based deep CNN architectures deployed with state-of-the-art libraries like TensorFlow, in order to gain from the lower power and high frame-rate inherently available with an NVS camera. Research in this area is now beginning to consider deep CNNs. However, most activities relate to low-cost on-board processing for robotics and guidance systems and do not consider higher-level tasks like human action recognition or semantic scene analysis. In addition, all existing work is hampered by the lack of annotated NVS training data [3]. To mitigate the latter issue, several proposals recorded annotated APS video datasets under controlled conditions [2, 11], i.e., video frames displayed in a monitor under controlled frame-rate and brightness/contrast conditions and recorded with an NVS device like iniLabs dynamic vision sensor (DVS). Such experimental approaches are very valuable as they provided the first available annotated video spike recordings for human action recognition. However, their scale-up to larger datasets is hampered by natural variations in recordings from environmental and monitor conditions (e.g., lighting, monitor flicker, vibrations, etc.) or variations in NVS camera hardware. In addition, highly-accurate synchronization is required between the played-out video frames and the corresponding NVS because of drift between the playout device timing and the DVS camera timestamping (especially as the dataset grows in size).

With regards to the lack of training data, the recently proposed PIX2NVS framework [12] and work by Furber *et al.* [13] provide for parametric software-based solutions for converting pixel domain video frames directly to neuromorphic spike events. However, such emulation frameworks require complex tuning of their parameters in order to match the behaviour of NVS hardware. In this paper, we hypothesize that can potentially be resolved by embedding such an emulator into a teacher-student framework based on knowledge distillation [14, 15]. Our proposal considers heterogeneous transfer from optical flow to NVS, showcasing that this can bridge the performance gap between APS and NVS-based CNNs.

3. TEACHER-STUDENT FRAMEWORK

In order to validate NVS inputs as a low-cost activity-based alternative to optical flow that is traditionally used for action recognition, we propose to embed the PIX2NVS emulator [12] into a teacher-student framework based on knowledge distillation [14], where we essentially transfer knowledge from a pre-trained optical flow teacher network to the NVS student network by drawing pairwise correspondences and minimizing the cross entropy between the output distributions. The overall framework is illustrated in Figure 1 for the training and inference stages. After training the student model, the emulator component of the framework can potentially be replaced by an NVS camera during inference, to perform action recognition on either real or emulated events directly on the student, without the use of optical flow.

3.1. Frame generation

We use the PIX2NVS framework to extract the emulated NVS events from RGB video frames, which provides us with training data correspondences for the NVS domain. The emulator generates a set of event tuples $\boldsymbol{E}_e = \{\langle x_e, y_e, t_e, P_e\rangle\}$ over the video sequence, where the event polarity $P_e = \pm 1$ (i.e., representing ON or OFF). Let us define:

$$P(x,y,t) = \begin{cases} P_e, & \text{if } \langle x,y,t,P_e\rangle \in \boldsymbol{E}_e \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

We can aggregate the polarities into a single NVS frame $\boldsymbol{f}[n]$ corresponding to the n-th RGB video frame by summing the polarities at each spatial position (x, y) for events with timestamp t falling in the $n-1$-th to n-th video frame time interval I. For the n-th video frame, the summed polarity at position (x, y), $f[n](x, y)$ is denoted as:

$$f[n](x,y) = \sum_{t \in I} P(x,y,t) \quad (2)$$

This enables spatio-temporal correspondence with the video frames. While this frame grouping is artificial, it allows for the use of CNNs for recognition [8], and we can now aggregate consecutive NVS frames into training volumes $\boldsymbol{v}_s$.

3.2. Cost Function and Training

We leverage on the recently introduced large-scale Kinetics [16] action recognition dataset. Our choice of architecture is a variant of the Inception-3D (I3D) [16] CNN, which is essentially Inception-Vl with inflated spatio-temporal filters Notably, we replace the final pooling layer with a spatio-temporal global average pooling in order to minimize complexity further. Our implementation first involves initializing the optical flow I3D with the Kinetics trained weights and then fine-tuning on the target action recognition dataset, such as UCF-101. Next, we initialize the student NVS CNN with

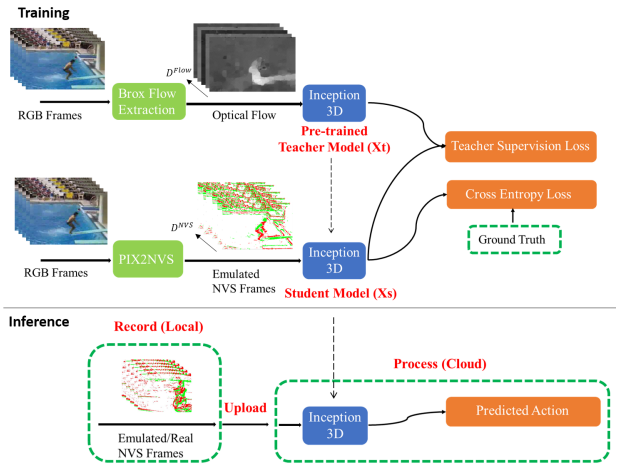

Fig. 1. Teacher-student framework using the PIX2NVS emulator.

the teacher weights. It is worth noting that our NVS inputs are only single-channel, whereas flow is dual-channel for the δx and δy components respectively; in order to apply an exact copy of our teacher network as the student, we simply replicate the NVS input channel-wise. The teacher network is now fixed and we train the student using a two-term cross-entropy loss for NVS frame volumes $\boldsymbol{v}_s \sim \mathbb{V}_s$, teacher flow volumes $\boldsymbol{v}_t \sim \mathbb{V}_t$ and labels $l \sim \mathbb{L}$:

$$\begin{aligned} L = &- \beta \mathbb{E}_{(\boldsymbol{v}_s, l) \sim (\mathbb{V}_s, \mathbb{L})} \sum_{k=1}^{K} 1_{[k=l]} \log(p(\boldsymbol{v}_s)) \\ &- \alpha T^2 \mathbb{E}_{(\boldsymbol{v}_s, \boldsymbol{v}_t) \sim (\mathbb{V}_s, \mathbb{V}_t)} q(\boldsymbol{v}_t, T) \log(p(\boldsymbol{v}_s)) \end{aligned} \tag{3}$$

The first term represents the standard cross-entropy loss between the softmax output of the student network $p(\boldsymbol{v}_s)$ and a one-hot encoded vector derived on the ground truth labels l. The second term is the teacher supervision cross-entropy loss between the teacher softmax output $q(\boldsymbol{v}_t, T)$ and $p(\boldsymbol{v}_s)$. The temperature T is the parameter that scales the logits z_i of the teacher network, such that $q_i(\boldsymbol{v}_t, T) = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}$. This softens the teacher distribution over classes, which can exemplify the class inter-dependencies for the student to learn a more informative representation. We treat the parameters α and β as simply binary coefficients on the loss terms, responsible for enabling/disabling the label cross entropy loss and teacher supervision loss respectively.

We train both the teacher and student CNNs with momentum and a decay rate of 0.9. Every convolutional layer is followed by batch normalization with a decay rate of 0.9. and an initial learning rate of 0.01. Our pre-processing constitutes a per-frame normalization and extracting a distorted bounding box crop from the original resized frame with a random horizontal flip. Importantly, we only infer on a single shot, extracted at the point of maximum motion activity, in order to minimize the temporal footprint during inference. This point n^* is localized by using the emulated NVS frames as an activity sensor and finding the n^*-th frame with the maximum sum over absolute summed polarities:

$$n^* = \underset{n}{\operatorname{argmax}} \left(\sum_{x,y} |f[n](x,y)| \right) \tag{4}$$

3.3. Ablation Study

Table 1 represents a basic exploration over the parameters α, β and T in (3). We note that we only ingest inputs $\boldsymbol{v} \in \mathbb{R}^{H \times W \times D \times C}$ of size $224 \times 224 \times 8 \times 2$ for both the flow and NVS streams, in order to speed up convergence. As configured, the optical flow CNN achieves 84.4% on a single input. The emulated NVS CNN, when trained without the teacher supervision loss and inferring on a single shot of emulated NVS frames, achieves 71.0%. This shows that incorporating the teacher supervision loss results in a substantial accuracy increase, with optimum accuracy attainable when the teacher logits are additionally softened with $T = 2$. Increasing T beyond 2 causes a decrease in accuracy, as the output begins to converge to a more uniform distribution. For the remainder of the chapter, we fix $\alpha = 1$, $\beta = 1$ and $T = 2$.

	α	β	T	Accuracy(%)
Teacher	-	-	-	84.4
Student	0	1	-	71.0
	1	0	1	73.1
	1	1	1	75.9
	1	1	2	77.0

Table 1. Recognition accuracy on UCF-101 (split 1) for the teacher and student, when varying parameters α, β and T of (3). Accuracies are reported on a *single shot* of 8 frames. For the student, both training and inference is performed on emulated NVS events generated by the PIX2NVS emulator.

4. COMPARISON WITH APS-BASED METHODS

Finally, we compare the proposed framework against current state-of-the-art APS-based methods. Given the combination of global average pooling and 3D convolutional layers in our proposal, this inherently means that the number of weights per CNN layer is not a function of the input temporal resolution D. Therefore, we initially train the optical flow teacher CNN on a larger input temporal resolution $D^{\text{Flow}} = 32$. We then initialize NVS student CNN with the flow pre-trained weights, but reduce the input temporal resolution to $D = 16$ when training with teacher supervision and during inference, in order to minimize the student complexity whilst transferring longer temporal dependencies. For our final implemen-

tation of the trained NVS student, we therefore infer only on a single shot of emulated frames with size $224 \times 224 \times 16 \times 2$.

We note that conventional APS-based methods typically use a combination of RGB frame and highly complex optical flow inputs during training and inference. Conventional NVS cameras, such as the iniLabs DAVIS240C are equipped with an onboard RGB camera. The DAVIS240C camera has an array size of 240×180 pixels with an APS bandwidth of 35 FPS. Therefore, we propose to combine the trained NVS student with an RGB stream during inference, which we restrict to ingesting inputs $\boldsymbol{v}^{\mathrm{RGB}} \in \mathbb{R}^{H \times W \times D \times C}$ of size $224 \times 224 \times 8 \times 3$. Our choice of architecture is I3D, pre-trained on Kinetics and fine-tuned on the target action recognition dataset. During inference, we infer on a single shot of RGB frames only; we seek the video segment with the maximum motion activity, again by employing the NVS stream as an activity sensor [see (4)]. We cross-reference the timestamp of the maximum motion NVS frame to the RGB stream and extract a single shot of frames at this point for inference, which can be uploaded and processed on the cloud, as with the NVS recorded events. In order to minimize the latency in uploading the RGB frames compared to the NVS frames, we downsample captured RGB frames by a factor of 2, and only upsample to the original resolution prior to CNN processing. Subsequently, we fuse the NVS and RGB modalities by following Simonyan *et al.* [17] and simply average the scores per video instance, thus generating our prediction.

	Accuracy(%)	
	UCF-101	HMDB-51
NVS (emulated) CNN	78.6	51.6
RGB CNN	84.0	55.9
NVS (emulated)-RGB CNN	89.0	62.0

Table 2. Recognition accuracy on UCF-101 and HMDB-51 for stream modalities utilized during inference. The NVS stream is trained in the teacher-student framework with $\{\alpha, \beta, T\} = \{1, 1, 2\}$.

In order to isolate the performance of each stream utilized during inference, we first report the accuracy of each modality separately in Table 2 and compare this to the accuracy when the streams are fused for the UCF-101 [9] and HMDB-51 [10] action recognition datasets. Evidently, the performance of the RGB CNN is boosted by the NVS CNN, despite complexity savings in opting for a short temporal extent for both streams and only inferring on a single shot at maximum motion activity. Our final results for the NVS-RGB CNN compared to recent external methods are reported in Table 3. We report complexity for CNN processing in terms of total GFLOPs by multiplying the GFLOPs per input to the CNN with the number of inputs required during inference. Our proposed NVS-RGB CNN is able to achieve a competitive accuracy-complexity trade-off when compared to state-of-the-art methods utilizing highly complex optical flow or motion vectors.

Method	Σ GFLOPs	UCF-101	HMDB-51
inc. optical flow			
Two-Stream [17]	150	88.0	59.4
3D Conv Fusion [18]	153	92.5	65.4
Action-VLAD [19]	-	92.7	66.9
ST-ResNet [20]	-	93.4	66.4
Two-Stream I3D [16]	648	97.8	80.9
no optical flow			
EMV-CNN [15]	150	86.4	-
CoViAR[21]	110	90.4	59.1
C3D [22]	385	82.3	51.6
Res3D [23]	193	85.8	54.9
Proposed, NVS (emulated)-RGB CNN	84	89.0	62.0

Table 3. Accuracy and complexity versus the state-of-the-art (results reported where available) for UCF-101 and HMDB-51. Results are reported after averaging over the three splits per dataset. We also report the theoretical GFLOPs for CNN processing of all inputs during inference.

The only method substantially outperforming our approach in terms of accuracy is I3D [16]; however, I3D requires optical flow for both training and inference (and substantial APS activity for I3D (RGB-only)) and comprises a substantially larger input, resulting in 3.7 to 7.7 times more GFLOPs than our NVS-RGB CNN. Additionally, the speed in generating conventional Brox optical flow [24] compared to emulated and real NVS frames is reported in Table 4 in terms of framerate; the emulator can generate NVS frames at 357 FPS whilst the real NVS camera can output NVS frames at 2000 FPS (3 to 6 orders of magnitude faster than Brox flow). We therefore note that any RGB-derived model can easily be fused with our NVS stream for performance gain with minimal computation in input generation and processing.

	Framerate (FPS)
Emulated	357
DAVIS240C NVS camera [25]	2000
Brox optical flow [24]	0.314

Table 4. Average framerate over 4600 video frames for Brox optical flow and emulated (PIX2NVS) and real (DAVIS240C) NVS frame generation.

5. CONCLUSION

We propose the first method for NVS-based action recognition that performs competitively to state-of-the-art APS methods on standard datasets. We embed an NVS emulator into a large multi-modal teacher-student framework, in order leverage on the availability of labelled data for training with highly complex optical flow. Given its promising recognition performance and low complexity, our framework could become a viable solution within IoT and robotics contexts, where, for example, NVS data would be gathered at very low power and streamed to cloud computing servers for the back-end CNN processing.

6. REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] Y. Hu, H. Liu, M. Pfeiffer, and T. Delbruck, "DVS benchmark datasets for object tracking, action recognition, and object recognition," *Front. Neurosc.*, vol. 10, 2016.

[3] T. Delbruck, "Neuromorphic vision sensing and processing," in *Proc. Europ. Solid-State Dev. Res. Conf.* IEEE, 2016, pp. 7–14.

[4] T. Delbrück, B. Linares-Barranco, E. Culurciello, and C. Posch, "Activity-driven, event-based vision sensors," in *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*. IEEE, 2010, pp. 2426–2429.

[5] E Neftci, C Posch, and E Chicca, "Neuromorphic engineering," *Comput. Intel.-Vol. II*, p. 278, 2015.

[6] R. Benosman, C. Clercq, X. Lagorce, S. Ieng, and C. Bartolozzi, "Event-based visual flow," *IEEE Trans. Neural Netw. and Learn. Syst.*, vol. 25, no. 2, pp. 407–417, 2014.

[7] Q. Sabatier, S. Ieng, and R. Benosman, "Asynchronous event-based fourier analysis," *IEEE Trans. on Image Processing*, vol. 26, no. 5, pp. 2192–2202, 2017.

[8] D. P. Moeys, F. Corradi, E. Kerr, P. Vance, G. Das, D. Neil, D. Kerr, and T. Delbrück, "Steering a predator robot using a mixed frame/event-driven convolutional neural network," in *Proc. Int. Conf. Event-based Contr., Comm., and Sig. Process., EBCCSP*. IEEE, 2016, pp. 1–8.

[9] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.

[10] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: a large video database for human motion recognition," in *IEEE Int. Conf. Comp. Vis., ICCV*. IEEE, 2011, pp. 2556–2563.

[11] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, "Converting static image datasets to spiking neuromorphic datasets using saccades," *Front. Neurosc.*, vol. 9, 2015.

[12] Y. Bi and Y. Andreopoulos, "PIX2NVS: Parameterized conversion of pixel-domain video frames to neuromorphic vision streams," *Proc. IEEE Int. Conf. Image Process., ICIP*, 2017.

[13] G. Pineda García, P. Camilleri, Q. Liu, and S. Furber, "pyDVS: An extensible, real-time dynamic vision sensor emulator using off-the-shelf hardware," in *Proc. IEEE Symp. Ser. Comp. Intel. (SSCI), 2016*. IEEE, 2016, pp. 1–7.

[14] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[15] B. Zhang, L. Wang, Z. Wang, Y. Qiao, and H. Wang, "Real-time action recognition with enhanced motion vector cnns," in *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*. IEEE, 2016, pp. 2718–2726.

[16] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 4724–4733.

[17] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Adv. Neur. Inf. Process. Syst., NIPS*, 2014, pp. 568–576.

[18] C. Feichtenhofer, A. Pinz, and A. P. Zisserman, "Convolutional two-stream network fusion for video action recognition," 2016.

[19] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell, "Actionvlad: Learning spatio-temporal aggregation for action classification," in *CVPR*, 2017, vol. 2, p. 3.

[20] C. Feichtenhofer, A. Pinz, and R. Wildes, "Spatiotemporal residual networks for video action recognition," in *Advances in neural information processing systems*, 2016, pp. 3468–3476.

[21] C. Wu, M. Zaheer, H. Hu, R. Manmatha, A. J. Smola, and P. Krähenbühl, "Compressed video action recognition," *arXiv preprint arXiv:1712.00636*, 2017.

[22] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comp. Vis., ICCV*, 2015, pp. 4489–4497.

[23] D. Tran, J. Ray, Z. Shou, S. Chang, and M. Paluri, "Convnet architecture search for spatiotemporal feature learning," *arXiv preprint arXiv:1708.05038*, 2017.

[24] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *European conference on computer vision*. Springer, 2004, pp. 25–36.

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