Live Demonstration: ANN vs SNN vs Hybrid Architectures for Event-based Real-time Gesture Recognition and Optical Flow Estimation

Adarsh Kumar Kosta Marco Paul E. Apolinario Kaushik Roy
Purdue University, West Lafayette, IN, USA
{akosta@purdue.edu, mapolina@purdue.edu, kaushik@purdue.edu}

Abstract

Spiking Neural Networks (SNNs) have recently emerged as a promising solution to handle asynchronous data from event-based cameras. Their inherent recurrence allows temporal information in events to be effectively captured unlike widely used non-spiking artificial neural networks (so-called ANNs). However, SNNs are not suitable to run on GPUs and still require specialized neuromorphic hardware to process events efficiently. Hybrid SNN-ANN architectures aim to obtain the best of both worlds with initial SNN layers capturing input temporal information followed by standard ANN layers for ease of training and deployment on GPUs. In this work, we implement ANN, SNN, and hybrid architectures for real-time gesture recognition and optical flow estimation on standard GPUs. We compare different architectures in terms of prediction accuracy, number of parameters, latency, and computational power when executing them in real time on a standard laptop. Our implementation suggests that the hybrid architecture offers the best trade-off in terms of accuracy, compute efficiency, and latency on readily available GPU platforms.

1. Introduction and Motivation

Event-based cameras [3] have recently shown great potential for capturing motion information in high-speed applications due to their high temporal resolution while having an extremely low power consumption. The sparse and asynchronous data stream produced by event-based cameras is incompatible with frame-driven processing in ANNs. Spiking Neural Networks (SNNs) with their event-driven computations serve as a promising candidates for handling event data. However, fully-spiking models are not suitable for deployment on standard GPUs. Hybrid SNN-ANN architectures solve this by providing the temporal information capturing ability of SNNs while still allowing deployment on GPUs. We evaluate a variety of these models and report their compute requirements, thereby determining their suitability for real-time applications.

2. Experiments and Results

We consider two event-based tasks: (1) A simple classification task of gesture recognition, and (2) A complex regression task of optical flow estimation.

Architecture: For gesture recognition, we use a simple SNN model with three convolutional layers with 8, 16, 32 output channels followed by two fully-connected layers of sizes 32, 11 leading to just ∼16K parameters. We also evaluate a larger model (SNN_{Large}) from a recent state-of-the-art implementation in [2] (∼52K parameters).For optical flow estimation, we utilize the encoder-decoder SNN models proposed in [4]. Specifically the Base (∼13M parameters) and Nano (∼270K parameters) models were evaluated from [4]. The inputs were passed to the network over several timesteps and the output at the last layer was accumulated to generate the prediction. For all SNN models, the ANN and Hybrid counterparts were also evaluated. The ANN model (ANN_P) involved passing the timesteps parallely as inputs channels. A sequential version of the ANN model **(ANN_S)** was also evaluated for the gesture recognition task with inputs passed into the model over timesteps. The hybrid models consisted of the first layer being spiking for both tasks (conv-1 for gesture recognition, encoder-1 for optical flow). The ANN layers used a rectified lin-

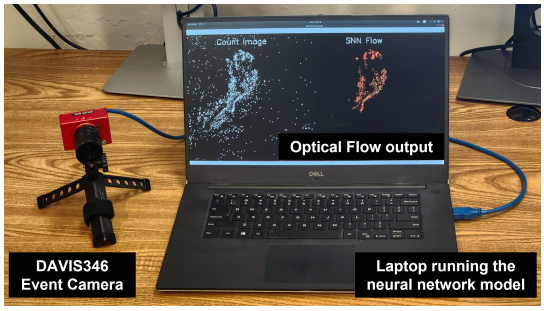

Figure 1. Experimental setup for gesture recognition and optical flow estimation using a DAVIS346 camera and a standard laptop. Optical flow output with SNN model shown.

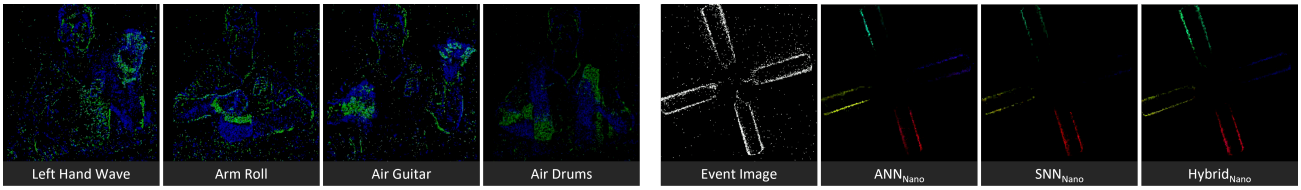

Figure 2. (left) Some gestures from the IBM DVSGesture dataset. (right) Optical Flow estimations by ANN$_{Nano}$, SNN$_{Nano}$ and Hybrid$_{Nano}$.

ear unit (ReLU) activation while the spiking layers used a leaky-integrate and fire (LIF) unit with learnable leak and threshold as activation. The input size was 64×64 for gesture recognition and 256×256 for flow estimation.

Experimental Setup: The gesture recognition models were trained on the IBM DVS Gesture dataset [1] with 11 hand gesture classes recorded using a DVS128 camera. Models for optical flow estimation were trained on the $outdoor\_day2$ driving sequence from the Multi-Vehicle Stereo Event Camera (MVSEC) dataset [5] recorded using a DAVIS346 [3] camera. Testing was done on the $outdoor\_day1$ sequence. The demo setup involved a DAVIS346 [3] camera capturing events in real-time using $libcaer$ and $pyaer$ libraries. It was then filtered for noise and converted to appropriate input representations. The inference ran on a laptop running Ubuntu 18.04 with a i7-11800H CPU and a Nvidia RTX 3080 GPU.

Model	Accuracy(%)	Compute Power (W)	Compute Latency (ms)
SNN$_{Large}$	**94.76**	50.51	62.53
ANN$_{P}$	92.49	**10.98**	**0.84**
ANN$_{S}$	91.87	**10.22**	7.82
SNN	92.58	11.25	18.16
Hybrid	**92.78**	10.78	**4.92**

Table 1. Evaluated metrics for for Gesture Recognition

Model	AEE	Compute Power (W)	Compute Latency (ms)
ANN$_{Base}$	0.48	**15.51**	**30.41**
SNN$_{Base}$	**0.44**	44.78	123.18
ANN$_{Nano}$	0.62	**5.18**	40.18
SNN$_{Nano}$	**0.52**	13.26	42.12
Hybrid$_{Nano}$	0.53	5.95	**14.02**

Table 2. Evaluated metrics for for Optical Flow Estimation

Performance metrics: The performance was evaluated on the testing sets in terms of prediction accuracy for gesture recognition and average endpoint error (AEE) from [4] for optical flow estimation. We see that the lightweight SNN and Hybrid models perform competitively with larger models. The ANN models have the lowest performance (lowest accuracy and highest AEE). Table-1 and Table-2 show the evaluated metrics for the two tasks.

Power and Latency: We use the python binding for Nvidia Management Library ($pynvml$) for power measurement and python $time$ library to compute latency. Among lightweight models, SNNs incur the highest power and latency due to the more complex LIF activation evaluated over several timesteps. This is because a GPU is unable to take advantage of the event driven sparse processing in SNNs. This is more evident in flow estimation task due to many skip connections in the model. The ANN models are most efficient due to single timestep processing in GPUs but are suboptimal in performance as they do not capture timing information. Hybrid models lead to an optimal trade-off.

3. Conclusion

We present a real-world comparison of ANN, SNN and Hybrid models for gesture recognition and optical flow estimation tasks. We show that lightweight SNN and Hybrid models lead to respectable accuracy while having high computational efficiency compared to ANNs.

Visitors Experience: During the demo, there will be two setups running gesture recognition and optical flow estimation using DAVIS346 event cameras. Visitors will be able to interactively perform gestures and visualize the corresponding outputs in a graphical user interface on a monitor.

Acknowledgement: This work was supported by the National Science Foundation, IARPA Microe4AI, and the Center for the Co-Design of Cognitive Systems (CoCoSys) a JUMP 2.0 center, funded by SRC and DARPA.

References

[1] Arnon Amir, Brian Taba, David Berg, et al. A low power, fully event-based gesture recognition system. *CVPR*, 2017. 2

[2] Marco Paul E. Apolinario, Adarsh Kumar Kosta, Utkarsh Saxena, and Kaushik Roy. Hardware/software co-design with adc-less in-memory computing hardware for spiking neural networks. *arXiv*, 2022. 1

[3] Christian Brandli, Raphael Berner, Minhao Yang, Shih-Chii Liu, and Tobi Delbruck. A 240 × 180 130 db 3 µs latency global shutter spatiotemporal vision sensor. *IEEE Journal of Solid-State Circuits*, 2014. 1, 2

[4] Adarsh Kumar Kosta and Kaushik Roy. Adaptive-spikenet: Event-based optical flow estimation using spiking neural networks with learnable neuronal dynamics. *arXiv*, 2023. 1, 2

[5] Alex Zihao Zhu, Dinesh Thakur, Tolga Özaslan, Bernd Pfrommer, Vijay Kumar, and Kostas Daniilidis. The multivehicle stereo event camera dataset: An event camera dataset for 3d perception. *IEEE Robotics and Automation Letters*, 2018. 2