THE OHIO STATE UNIVERSITY

AFRL
THE AIR FORCE RESEARCH LABORATORY

Dataset Augmentation for Robust Spiking Neural Networks

Anthony Baietto[1], Chris Stewart[1], Trevor J. Bihl[2]

Department of Computer Science and Engineering, Ohio State University[1]
Sensors Directorate, U.S. Air Force Research Laboratory[2]

Outline

- Spiking Neural Networks (SNNs) Introduction and Training Considerations
- SNN Spike Distribution Dependencies
- Our Dataset Augmentation Procedure
- Results
- Conclusions and Future Work

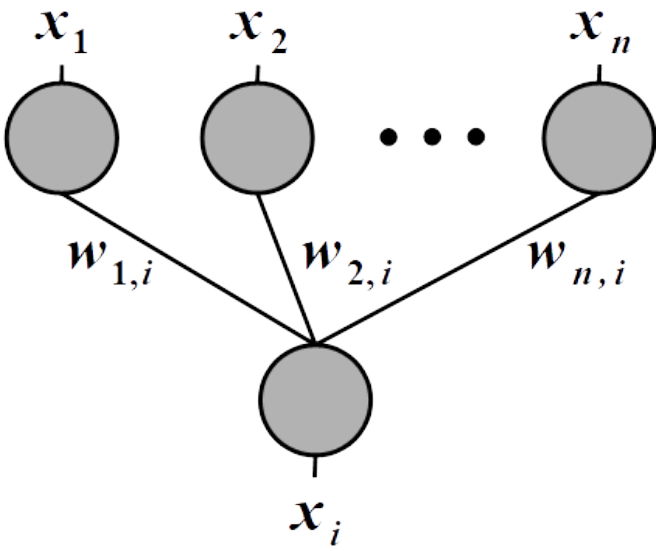
Spiking Neural Networks (SNNs)

- Biologically inspired 3rd generation neural networks
- Neurons communicate via discrete pulses over time
- Great for time-series data
- SNN processing consumes less power when realized on neuromorphic hardware such as Intel Loihi

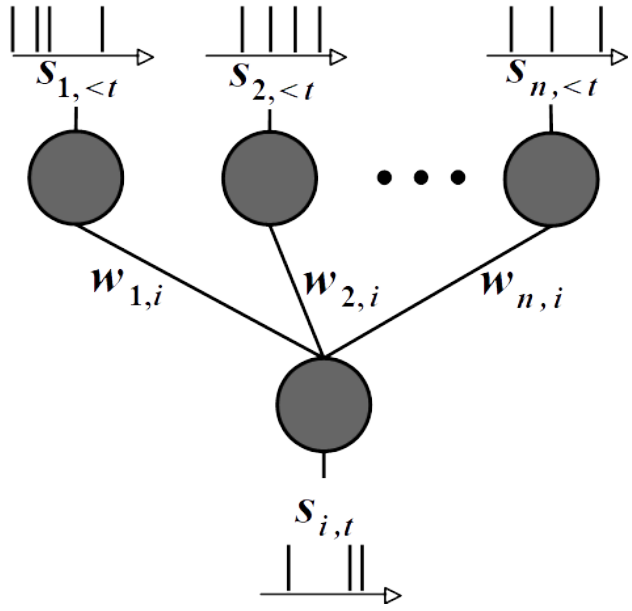
https://www.intel.com/content/www/us/en/newsroom/news/intel-unveils-neuromorphic-loihi-2-lava-software.html#gs.4ve63w

ANNs vs SNNs

- Operate on continuous values $x_1, x_2, \ldots, x_n$
- Information propagates instantaneously



- Operate on discrete spike trains $S_1, S_2, \ldots, S_n$
- Must be run over a period of time



ANNs vs SNNs

$$E = L(y, \hat{y})$$

$$net_j = \sum_i w_{ij} x_i + b$$

$$o_j = \varphi(net_j)$$

$$\delta_j = \begin{cases} \frac{\partial L(y, o_j)}{\partial o_j} \frac{d\varphi(net_j)}{dnet_j} & j \text{ output} \\ \left(\sum_k w_{jk} \delta_k\right) \frac{d\varphi(net_j)}{dnet_j} & j \text{ hidden} \end{cases}$$

$$\Delta w_{ij} = -\eta o_j \delta_j$$

Input $s_i(t) = \sum_f \delta\left(t - t_i^{(f)}\right)$

$a_i(t) = (\epsilon * s_i)(t)$ $\quad \epsilon(t) = \frac{t}{\tau_s} exp\left(1 - \frac{t}{\tau_s}\right) \Theta(t)$

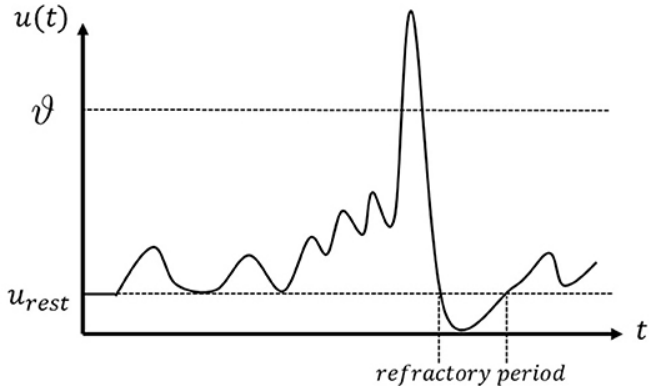
$v_i(t) = (v * s)(t)$ $\quad v(t) = -2\vartheta exp\left(1 - \frac{t}{\tau_r}\right) \Theta(t)$

$$u(t) = \sum_i w_i a_i(t) + v_i(t)$$

$$f_s(u) : u \to s$$

$$s(t) := s(t) + \delta\left(t - t^{(f+1)}\right)$$

$$t^{(f+1)} = min\left\{t : u(t) = \vartheta, t > t^{(f)}\right\}$$



Outline

- Spiking Neural Networks (SNNs) Introduction and Training Considerations
- SNN Spike Distribution Dependencies
- Our Dataset Augmentation Procedure
- Results
- Conclusions and Future Work

SNN Data

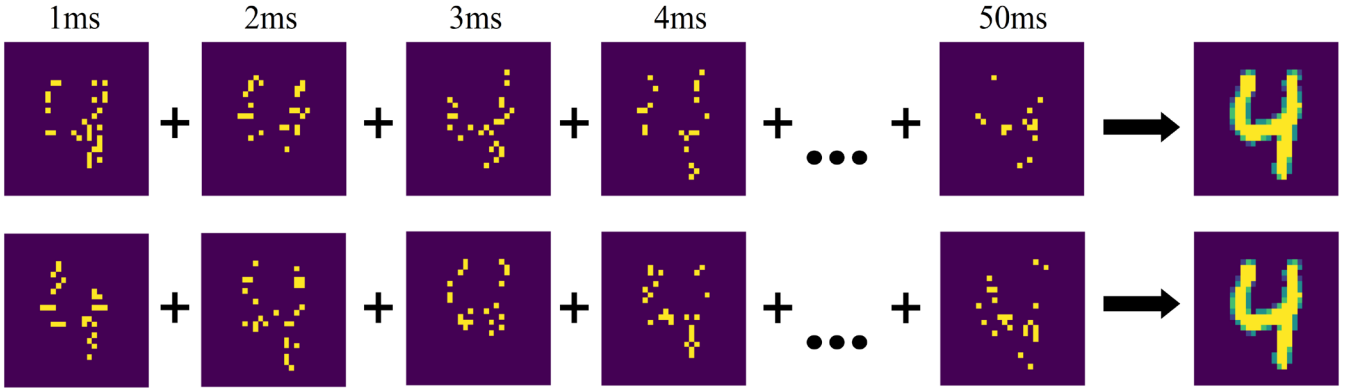
- SNNs operate on discrete spike trains
- Can be either generated from static data using integrate-and-fire (IF) neurons or captured directly using a Dynamic Vision Sensor (DVS) camera which produces event data:

https://inilabs.com/products/

$$[\boldsymbol{x}\ \text{coordinate}, \boldsymbol{y}\ \text{coordinate}, \boldsymbol{t}\ \text{timestep}, \boldsymbol{p}\ \text{polarity of light} - \text{intensity change}]$$

Spike Distribution Dependencies

- For a given static image, there are a copious number of valid spike trains which can be created/captured depending on IF neuron parameters, DVS camera settings, or lighting properties of the subject
- Surrogate gradient SNN training can fixate on the intervals of training spikes leading to generalization issues



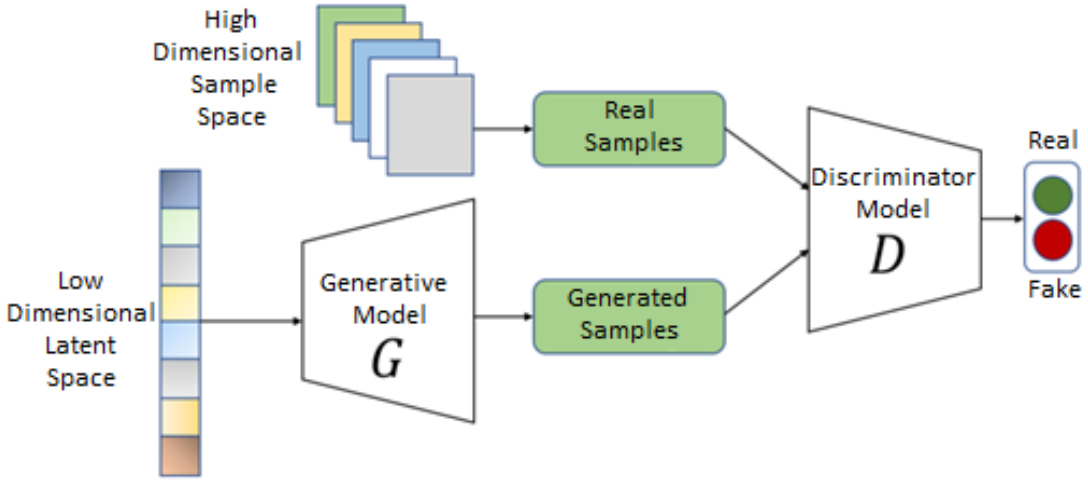
Example of differing spike distributions generated from identical static image

Outline

- Spiking Neural Networks (SNNs) Introduction and Training Considerations
- SNN Spike Distribution Dependencies
- Our Dataset Augmentation Procedure
- Results
- Conclusions and Future Work

Generative Adversarial Networks (GANs)

- Adversarial learning paradigm in which a generator model G synthesizes artificial samples, and a discriminator model D classifies samples as either real or generated
- Once converged, the generator model can be used to create an arbitrary number of realistic samples



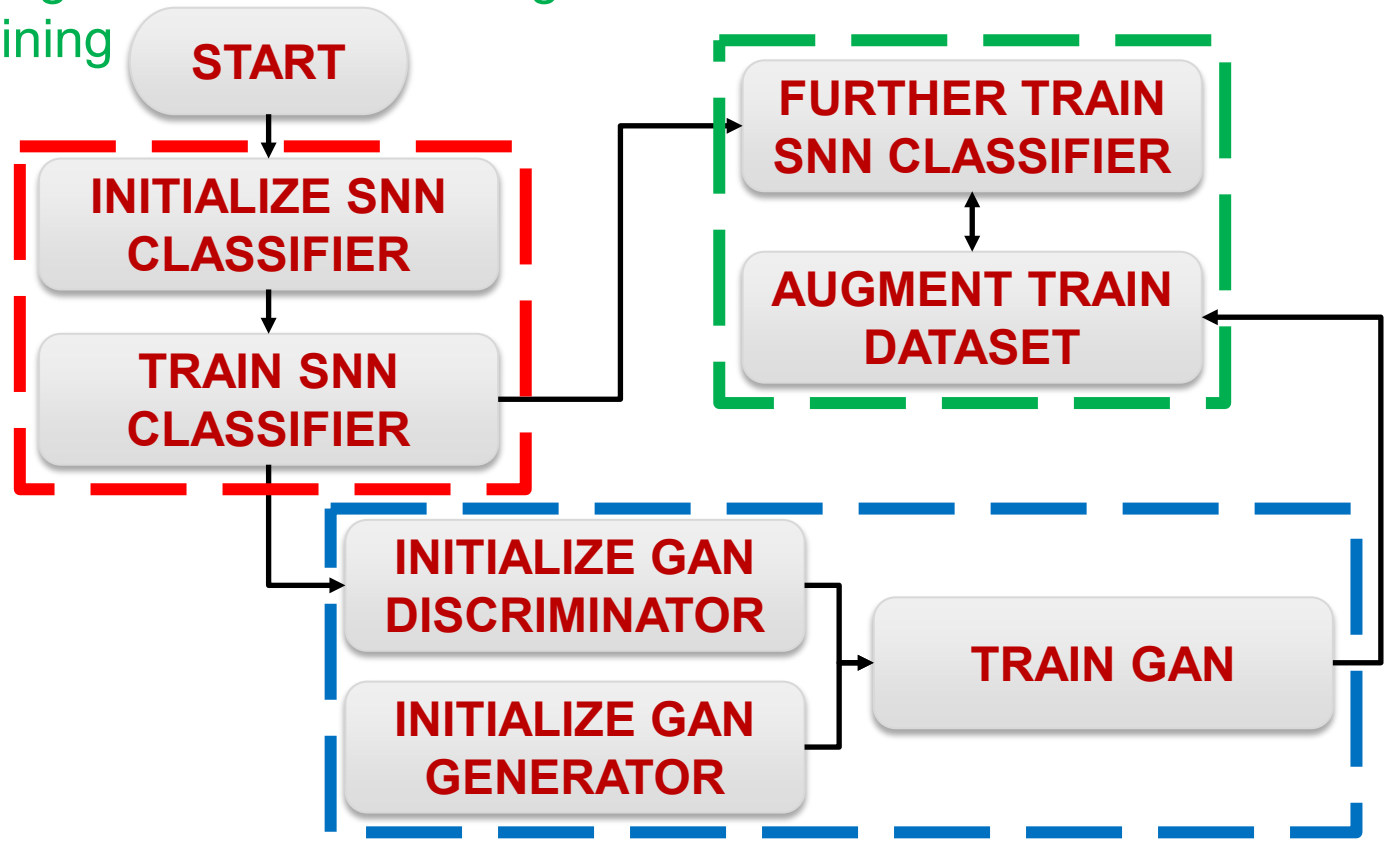
Our approach

- Using a spiking GAN, generate **valid** samples of **varying** spike distributions
- Augmented dataset provides additional robustness against samples different from the original training set
- Generated samples enrichen dataset without additional manual collection of data

(1) SNN classifier trained to convergence

(2) GAN trained using classifier weights to seed discriminator

(3) Trained GAN generator used to augment train dataset for further classifier training



Our approach (cont'd)

- During augmentation, samples are generated on an as-needed basis determined by the relative class performances
- Difficulty of correct classification is not uniform across all classes of data → disproportionate number of samples can achieve same overall accuracy

Our approach (cont'd)

- Three schemes used to determine the number of additional samples needed for the next iteration:
  1) **equal**: same number of samples across all classes
  2) **adhoc**: only samples from the 3 worst performing classes added
  3) **scale**: number of samples added correlated to relative performance of each class

Outline

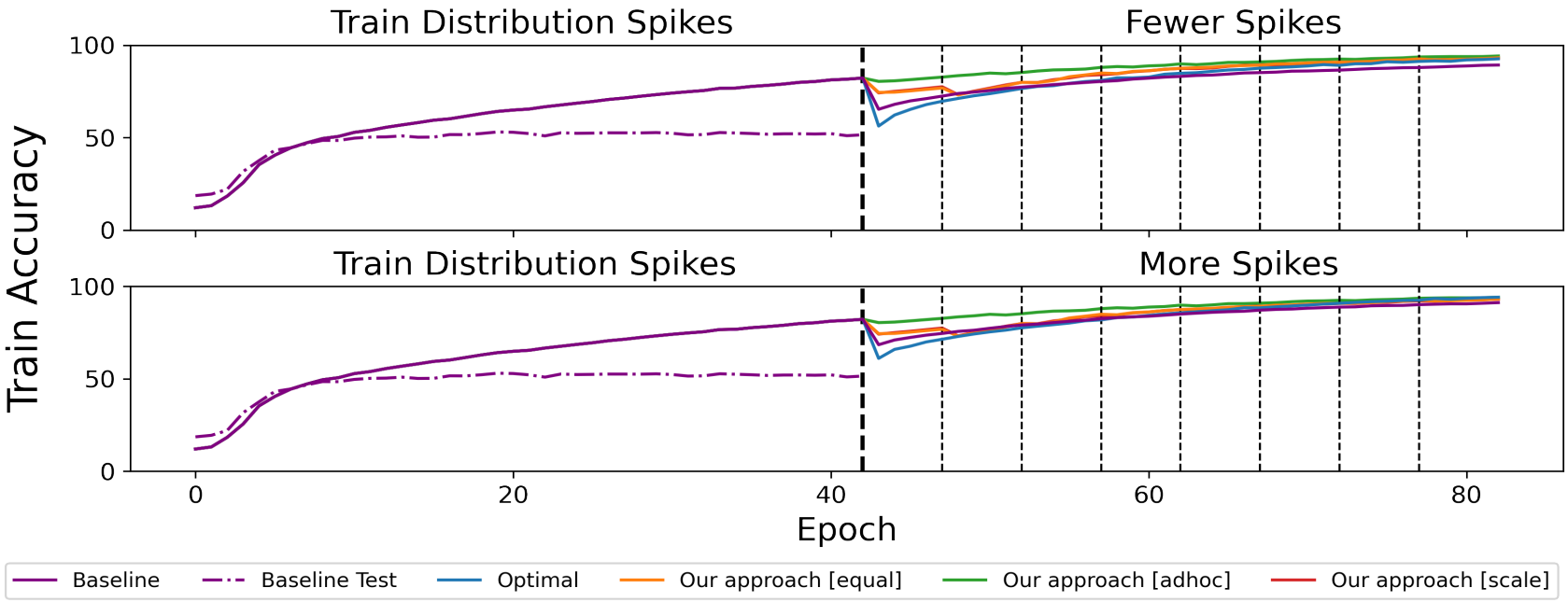
- Spiking Neural Networks (SNNs) Introduction and Training Considerations
- SNN Spike Distribution Dependencies
- Our Dataset Augmentation Procedure
- Results
- Conclusions and Future Work

Setup

- SLAYER SNN training platform used
  (S. B. Shrestha and G. Orchard, "Slayer: Spike layer error reassignment in time," 2018. [Online]. Available: https://arxiv.org/abs/1810.08646)
- CIFAR-10 training spike trains generated from $X \sim U(100, 200)$ firing rate distribution using LIF (leaky integrate-and-fire neuron) in Nengo simulator
- Models evaluated on **fewer** spikes and **more** spikes distributions → half ($X \sim U(50, 100)$) and double ($X \sim U(200, 400)$) the number of spikes compared to training distribution

Training

- Our models quickly responded to the changing spike distribution



Testing

- All models perform worse as samples drifted further from the training distribution
- Our models outperformed baseline classifier by an average **1.80%** and had an average **1.02%** lesser reduction in accuracy

Model	Testing Spike Distribution		
	Fewer Spikes	Train Dist. Spikes	More Spikes
Baseline	37.76 ± 0.34	52.67 ± 0.31	42.73 ± 0.52
Our approach [equal]	39.09 ± 0.25	**54.57 ± 0.27**	44.07 ± 0.52
Our approach [adhoc]	**39.33 ± 0.28**	54.25 ± 0.30	**44.67 ± 0.52**
Our approach [scale]	38.52 ± 0.39	54.05 ± 0.32	43.51 ± 0.29

Outline

- Spiking Neural Networks (SNNs) Introduction and Training Considerations
- SNN Spike Distribution Dependencies
- Our Dataset Augmentation Procedure
- Results
- Conclusions and Future Work

Conclusions and Future Work

- Conventional SNN training methods do **not ensure generalization capabilities** for temporal data
- Our results show **improvements in model robustness** against dissimilar samples from the training data
- Next steps:
    a) Evaluate performance on different datasets and SNN training platforms
    b) Model minimum number of additional samples needed to achieve target accuracy

Questions?

baietto.2@osu.edu