Spiking Neural Networks for Event-based Vision

Garrick Orchard

CVPR Event-based Vision Workshop

17th June 2019, Long Beach, CA

Intel’s Loihi Neuromorphic Research Chip
Legal Information

This presentation contains the general insights and opinions of Intel Corporation (“Intel”). The information in this presentation is provided for information only and is not to be relied upon for any other purpose than educational. Intel makes no representations or warranties regarding the accuracy or completeness of the information in this presentation. Intel accepts no duty to update this presentation based on more current information. Intel is not liable for any damages, direct or indirect, consequential or otherwise, that may arise, directly or indirectly, from the use or misuse of the information in this presentation.

Intel technologies’ features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer.

No computer system can be absolutely secure. No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document. Intel, the Intel logo, Movidius, Core, and Xeon are trademarks of Intel Corporation in the United States and other countries.

*Other names and brands may be claimed as the property of others

Copyright © 2019 Intel Corporation.
Visual Recognition

Spatial filters using neurons as coincidence detectors

Visual Recognition

Basic HMAX (frames)

Classifier

SVM

Pool

Max

Convolution

Templates

Pool

Max

Convolution

Gabor

Objects: Cat

HFirst (spikes)

Pool

Max (first spike)

Convolution

Templates

Pool

Max (first spike)

Convolution

Gabor

Objects: Fish


Visual Motion Estimation

Creating coincidence using delays

5 pixel region

Neuron sensitive to rightward motion at a fixed speed

threshold
rest
Neuron Activation

Visual Motion Estimation

Turning attention to Datasets

HFirst and SAVME models were both hand-tuned on very small self-captured datasets

There was a clear need for larger better datasets

Applying Deep Learning to SNNs

- Two main obstacles to using backpropagation for SNNs
  1) Non-differentiability of spike generation
  2) Error assignment

1) Non-differentiability of spike generation
   - A proxy function can be used in place of the derivative
   - Easily and successfully implemented by many works

2) Error assignment
   - A neuron’s state depends on its previous inputs
   - Error at the current time should be distributed to these previous inputs
   - Largely ignored by most works
SLAYER

Spike LAYer Error Reassignment

Error Signal
Soma voltage
Synapse B PSP
Synapse A PSP
Synapse B spike
Synapse A spike

Forward Propagation
spike \uparrow
PSP \varepsilon \ast s

Back Propagation
\uparrow error
temporal error
credit assignment \varepsilon \otimes e
A proxy function for the derivative of the spike function (s)

- If at time $\tau$ a small perturbation $\pm \zeta$ of the Soma voltage $u$ causes the spike $s$ to change
- The “derivative” can be approximated as

$$\frac{\Delta s}{\Delta u} = \frac{\delta(t-\tau)}{\Delta \zeta}$$

$$E[f_s'(\tau)] = E[\frac{\Delta s}{\Delta u}] = \rho(u(\tau) - \vartheta)$$

$$\rho(t) = \alpha \exp(-\beta |u(t) - \vartheta|)$$

Shrestha, S. B.; and Orchard, G.; "SLAYER: Spike Layer Error Reassignment in Time", Neural Information Processing Systems, Montreal, Canada, Dec 2018
SLAYER: Forward Model

\[ z^{(l+1)} = W^{(l)} a^{(l)} + b^{(l)} \]

\[ a^{(l)} = f(z^{(l)}) \]

Shrestha, S. B.; and Orchard, G.; "SLAYER: Spike Layer Error Reassignment in Time", *Neural Information Processing Systems, Montreal, Canada, Dec 2018*
SLAYER: Backpropagation

ANN

\[
\begin{align*}
    e^{(l)} &= \begin{cases} 
    \frac{\partial E}{\partial a^{(n_l)}} 
    \left( W^{(l)} \right)^T \delta^{(l+1)} & \text{if } l = n_l \\
    \delta^{(l+1)} & \text{otherwise}
    \end{cases} \\
    \delta^{(l)} &= f'(z^{(l)}) \cdot e^{(l)} \\
    \nabla_{W^{(l)}} E &= \delta^{(l+1)} \left( a^{(l)} \right)^T
\end{align*}
\]

SNN

\[
\begin{align*}
    e^{(l)}(t) &= \begin{cases} 
    \frac{\partial L(t)}{\partial a^{(n_l)}} 
    \left( W^{(l)} \right)^T \delta^{(l+1)}(t) & \text{if } l = n_l \\
    \delta^{(l+1)}(t) & \text{otherwise}
    \end{cases}
\end{align*}
\]

Shrestha, S. B.; and Orchard, G.; "SLAYER: Spike Layer Error Reassignment in Time", Neural Information Processing Systems, Montreal, Canada, Dec 2018

Data licensed under GNU GPL v3
Shrestha, S. B.; and Orchard, G.; "SLAYER: Spike Layer Error Reassignment in Time", Neural Information Processing Systems, Montreal, Canada, Dec 2018

Dataset licensed under CC BY 4.0:
## SLAYER: Results

First results on Loihi!

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>Architecture</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>Lee et al.</td>
<td>28x28–800–10</td>
<td>99.31%</td>
</tr>
<tr>
<td></td>
<td>Rueckauer et al.</td>
<td>SNN converted from standard ANN</td>
<td><strong>99.44%</strong></td>
</tr>
<tr>
<td></td>
<td>SLAYER</td>
<td>28x28–12c5–2a–64c5–2a–10o</td>
<td>99.35 ± 0.07%</td>
</tr>
</tbody>
</table>

Shrestha, S. B.; and Orchard, G.; "SLAYER: Spike Layer Error Reassignment in Time", *Neural Information Processing Systems, Montreal, Canada, Dec 2018*
Loihi DVS Gesture Implementation

128x128x2 - 4a - 16c5z - 2a - 32c3z - 2a - 512 – 11

1.05M weights to learn

0.47M weights are zero

2 weights (8,9) are outside range of a 4 bit signed number
Summary

We have come a long way from hand tuning weights and delays in tiny models on tiny datasets!

SLAYER is a thoroughly derived powerful framework for configuring SNNs

- Enables learning in deep spiking networks (10+ layers)
- Can learn both weights and delays
- Allows learning of precise spike times and/or rates (depending on loss function)
- Preliminary results indicate good translation to limited precision hardware
Thanks!

https://nusneuromorphic.github.io/labmembers/index.html

Sumit Shrestha

Luca Della Vedova
Resources

PyTorch implementation of SLAYER (use this one!)
- https://github.com/bamsumit/slayerPytorch

Singapore lab website
- https://nusneuromorphic.github.io/

Links to code (SLAYER/Hfirst/SAVME), datasets (NMNIST/NCaltech101), and papers
- www.garrickorchard.com