Deep Spiking Q Networks

Masterthesis - Endterm Presentation

Christoph Hahn

January 29, 2020

Outline

1. Recap Midterm
   - Research Goal
   - Motivation
   - Theoretical Background
   - State-of-the-Art
2. Experiments
3. Results
4. Future Work
5. Conclusion

Outline

1. **Recap Midterm**
   - Research Goal
   - Motivation
   - Theoretical Background
   - State-of-the-Art
2. Experiments
3. Results
4. Future Work
5. Conclusion

Research Goal

Research goal: **Explore** different methods to obtain a **spiking neural network** that can solve complex **reinforcement learning** tasks and **compare** these methods.

Motivation

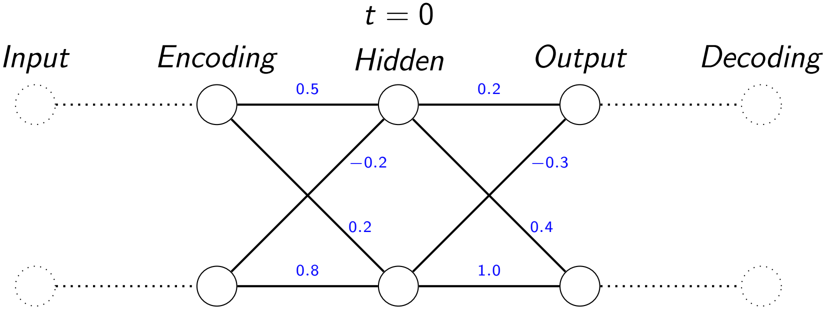
- \+ **Energy-Efficiency** and **fast inference** on dedicated hardware (**neuromorphic hardware**)[1]
- \+ **Brain-like** computation
- ? Exploit inherent **temporal** mechanics
- ? Naturally suited for **event-based inputs**

Theoretical Background: Reinforcement Learning

- An **agent** learns by moving through an **environment** and receiving **rewards**.
- **Deep Q-Learning**[2]: A neural network (**DQN**) is trained which approximates the **value function** $Q(s, a)$.
- **Policy Gradient**[3]: An agent learns to estimate (with a NN) a **stochastic policy directly** by using gradient ascent.

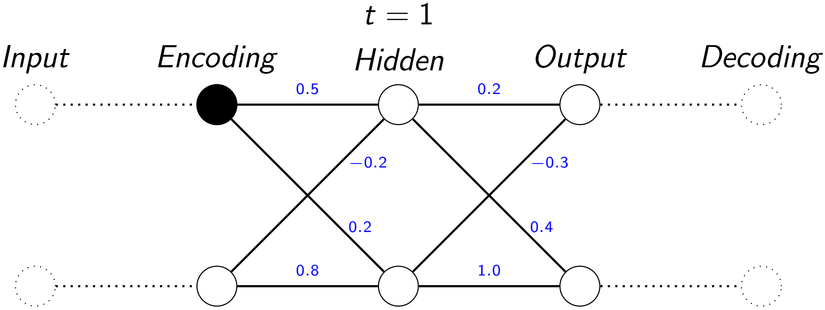
Theoretical Background: Spiking Neural Networks (SNNs)

- Inspired by neurons in the brain
- **Binary Outputs** (spike or no spike)
- Simulation over a **time period** (discrete or continuous)



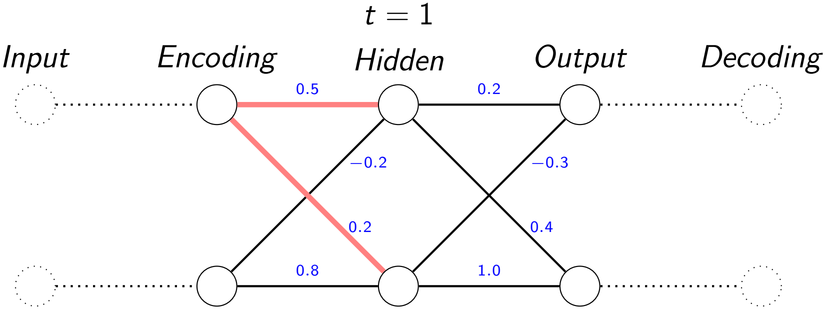
Theoretical Background: Spiking Neural Networks (SNNs)

- Inspired by neurons in the brain
- **Binary Outputs** (spike or no spike)
- Simulation over a **time period** (discrete or continuous)



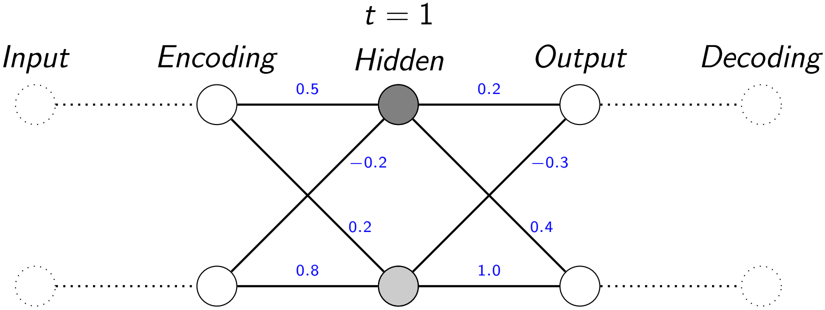
Theoretical Background: Spiking Neural Networks (SNNs)

- Inspired by neurons in the brain
- **Binary Outputs** (spike or no spike)
- Simulation over a **time period** (discrete or continuous)



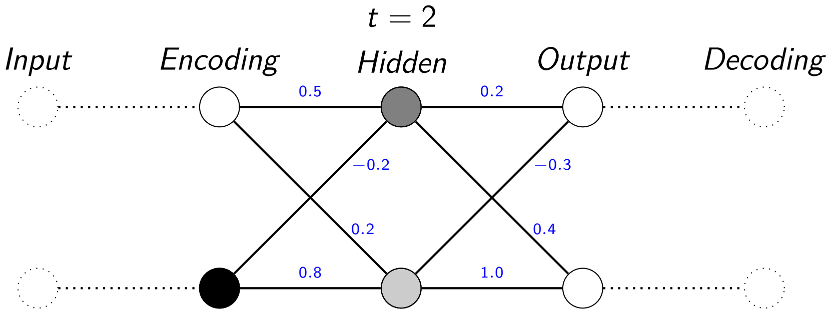
Theoretical Background: Spiking Neural Networks (SNNs)

- Inspired by neurons in the brain
- **Binary Outputs** (spike or no spike)
- Simulation over a **time period** (discrete or continuous)



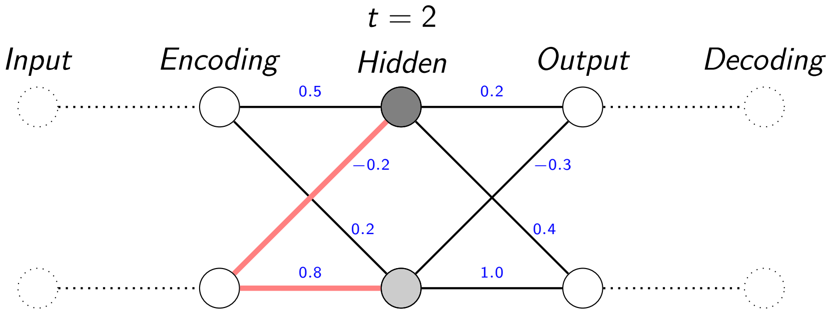
Theoretical Background: Spiking Neural Networks (SNNs)

- Inspired by neurons in the brain
- **Binary Outputs** (spike or no spike)
- Simulation over a **time period** (discrete or continuous)



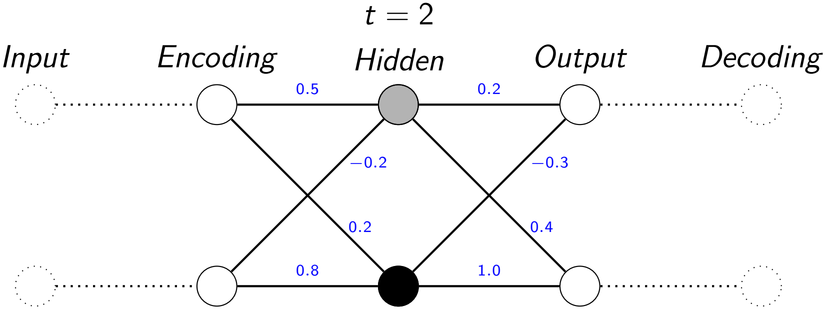
Theoretical Background: Spiking Neural Networks (SNNs)

- Inspired by neurons in the brain
- **Binary Outputs** (spike or no spike)
- Simulation over a **time period** (discrete or continuous)



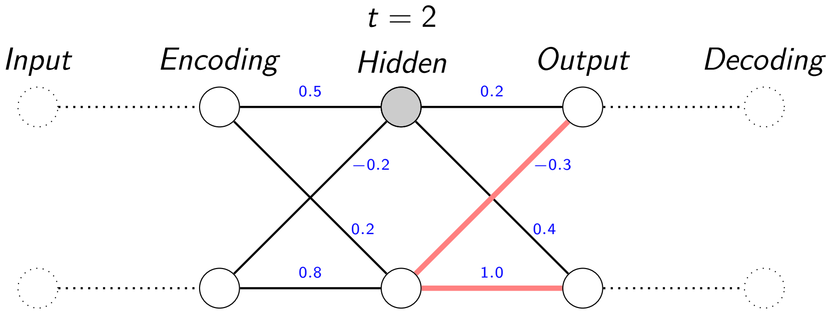
Theoretical Background: Spiking Neural Networks (SNNs)

- Inspired by neurons in the brain
- **Binary Outputs** (spike or no spike)
- Simulation over a **time period** (discrete or continuous)



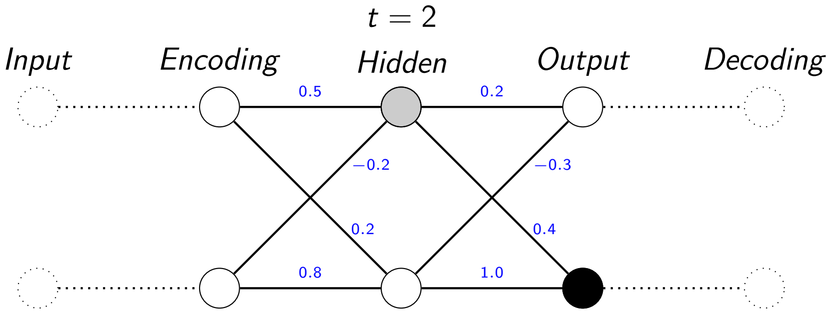
Theoretical Background: Spiking Neural Networks (SNNs)

- Inspired by neurons in the brain
- **Binary Outputs** (spike or no spike)
- Simulation over a **time period** (discrete or continuous)



Theoretical Background: Spiking Neural Networks (SNNs)

- Inspired by neurons in the brain
- **Binary Outputs** (spike or no spike)
- Simulation over a **time period** (discrete or continuous)

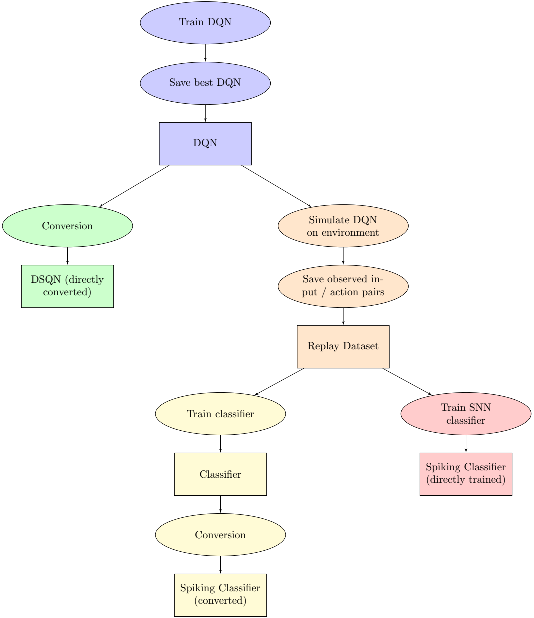


Theoretical Background: Spiking Neural Networks (SNNs)

- Inspired by neurons in the brain
- **Binary outputs** (spike or no spike)
- Simulation over a **time period** (discrete or continuous)
- Neuron model:
  - **Non-Leaky Integrate-and-Fire**
  - Reset mechanisms: **Reset-to-Zero**, **Reset-by-Subtraction**

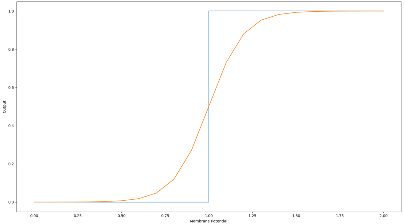
Theoretical Background: Conversion

- Train a **conventional Neural Network** with certain constraints: ReLu activation in hidden layers
- **Rate-based Encoding**
- **Weight conversion** based on observed data
- **Rate-based Decoding**


Train DQN
Save best DQN
DQN
Conversion
DSQN (directly converted)
Simulate DQN on environment
Save observed input / action pairs
Replay Dataset
Train classifier
Classifier
Conversion
Spiking Classifier (converted)
Train SNN classifier
Spiking Classifier (directly trained)

Theoretical Background: Backpropagation with Surrogate Gradients

- SNN is trained on conventional hardware using **Backpropagation**
- **Gradients** of binary function are **replaced** with a steep sigmoid function[4]
- **SpyTorch** Framework[5]



State-of-the-Art: Training of SNNs

Conversion

- Strong performance on **classification** tasks
- Patel et al.[6] **convert a DQN directly** for BreakOut and report similar performance after conversion with better robustness
- Meschede[7] **converts a DQN indirectly** for a lane following task

State-of-the-Art: Training of SNNs

Conversion

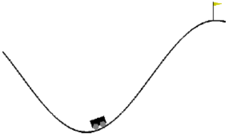
- Strong performance on **classification** tasks
- Patel et al.[6] **convert a DQN directly** for BreakOut and report similar performance after conversion with better robustness
- Meschede[7] **converts a DQN indirectly** for a lane following task

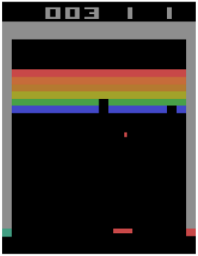
Other methods (Surrogate gradients, STDP, eRBP)

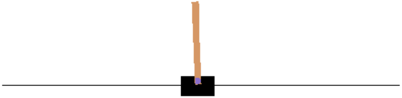
- Strong performance on **classification** tasks
- Few (STDP) to no (all others) papers on deep reinforcement learning
- Fremaux et al.[8] propose **STDP based actor-critic** framework which relies on **smart preprocessing** and **network layout**

Outline

1. Recap Midterm
   - Research Goal
   - Motivation
   - Theoretical Background
   - State-of-the-Art
2. **Experiments**
3. Results
4. Future Work
5. Conclusion







Experiments

- Experiments on three different **Open AI Gym**[9] environments: **CartPole**, **MountainCar**, **Breakout**
- Compare **conversion methods**
- Compare conventional DQN training and DSQN training using **surrogate gradients**
- Run networks in **NEST**[10] and on **SpiNNaker** hardware[11]

Outline

1. Recap Midterm
   - Research Goal
   - Motivation
   - Theoretical Background
   - State-of-the-Art
2. Experiments
3. **Results**
4. Future Work
5. Conclusion

Reproducibility

Problems:

- Results are **difficult to reproduce**
- Reported **metrics vary greatly**[12]

Reproducibility

Problems:

- Results are **difficult to reproduce**
- Reported **metrics vary greatly**[12]

Approaches in this thesis:

- Report **hyperparameters**, **software versions**, **seeds**
- **Average performance** of single agent + **standard deviation**
- **Conversion Accuracy**/Similarity as additional metric
- **Multiple training runs** to compare Backpropagation with and without surrogate gradients

Reproducibility

Problems:

- Results are **difficult to reproduce**
- Reported **metrics vary greatly**[12]

Approaches in this thesis:

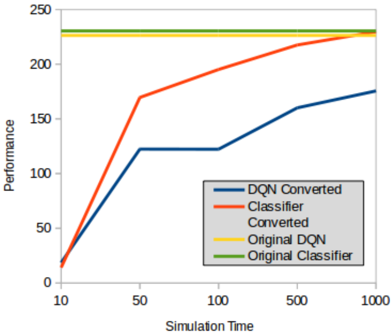
- Report **hyperparameters**, **software versions**, **seeds**
- **Average performance** of single agent + **standard deviation**
- **Conversion Accuracy**/Similarity as additional metric
- **Multiple training runs** to compare Backpropagation with and without surrogate gradients

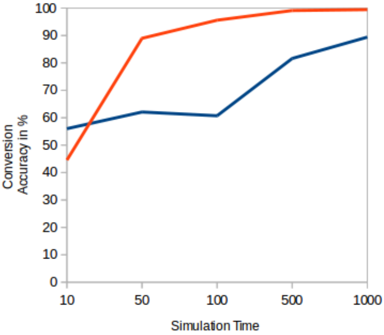
Further improvements:

- **Average performance of multiple trained agents** and **confidence intervals** (see Henderson et al.[12])

Direct vs Indirect Conversion

CartPole Results

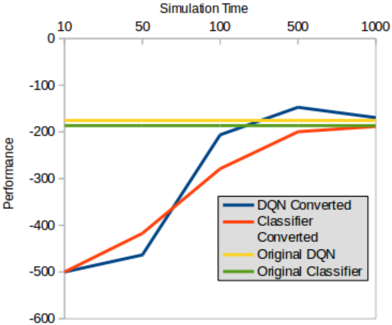


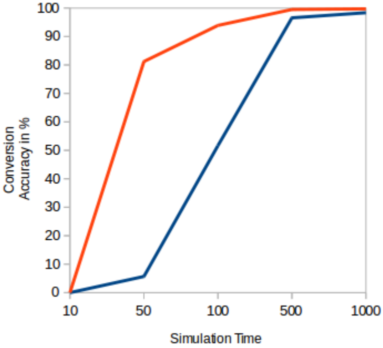


=> Classifier conversion works with smaller simulation time

Direct vs Indirect Conversion

MountainCar Results

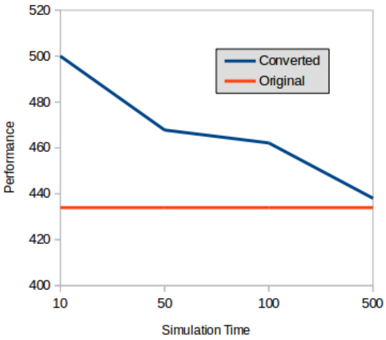


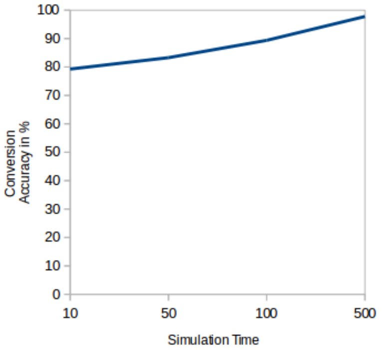


=> **Classifier can have worse performance than DQN**
Converting can improve performance

Policy Gradient Conversion

CartPole Results





=> Policy Gradient Networks convert well

Comparison of Conversion Methods

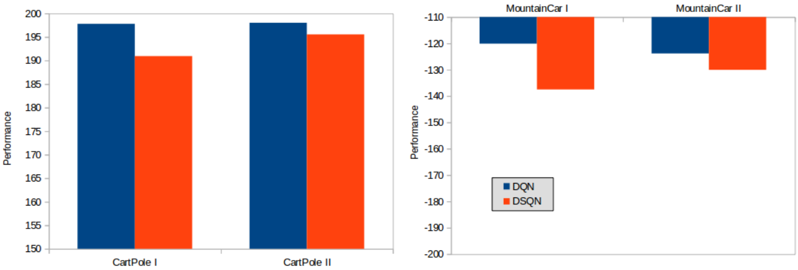
- Encoding: Poisson, Equidistant Spikes, **Constant Input Currents**
  - **Noise** from Poisson encoding can improve performance
- Decoding: Argmax of spikes, Argmax of **potentials**
- Resetting: Reset-to-Zero, **Reset-By-Subtraction**
- Weight Normalization: Model-based, Data-based, **Robust**

Backpropagation with Surrogate Gradients

Methodology

- **5 training runs** of DQN and DSQN each
- **Same hyperparameters**
- Two different learning rates

Backpropagation with Surrogate Gradients



=> **Less stable training** (of DSQN)
Longer training time (of DSQN)

Backpropagation with Surrogate Gradients

But:

Average performance when loading:

- CartPole: 454.84 (DSQN) vs 226.31 (DQN)
- MountainCar: -130.92 (DSQN) vs -175.4 (DQN)

=> **Better Performance** (of DSQN)
=> **Better Generalization**?

Surrogate Gradients vs Conversion

Conversion

+ More stable training
+ Less training time

Surrogate Gradients

+ Shorter inference time (except SNN classifier)
+ Better Generalization/Performance

NEST[10]

Challenge: Different neuron models, Simulation hyperparameters
Neuron Models: iaf_psc_delta, pp_psc_delta
Results: Classifier conversion is more robust against different neuron models

SpiNNaker[11]

Description: Neuromorphic hardware backend using PyNN[13]
Problem: Long simulation times
Results: Classifier conversion works, DQN conversion not yet

Outline

1. Recap Midterm
   - Research Goal
   - Motivation
   - Theoretical Background
   - State-of-the-Art
2. Experiments
3. Results
4. **Future Work**
5. Conclusion

Future Work

- More **complex environments**
- Convert **Actor-Critic** network
- Optimize algorithms for **SpiNNaker** or **NEST**
- **Event-based Random Backpropagation**[14]
- Facilitate the use of **state-of-the-art RL libraries**

Outline

1. Recap Midterm
   - Research Goal
   - Motivation
   - Theoretical Background
   - State-of-the-Art
2. Experiments
3. Results
4. Future Work
5. **Conclusion**

Conclusion

- **Conversion** of DQNs (direct, indirect, SNN-classifier) principally **works**
- **Direct conversion** of DQNs needs relatively **long simulation** times
- **Backpropagation with surrogate gradients** is **harder to train** (needs more time, more unstable)
- Networks trained with **surrogate gradients generalize better**
- Reproducing results is difficult
- Adapting networks to run on **NEST**, **PyNN**, or **SpiNNaker** is possible

Thank you!

Github repository:
https://github.com/vhris/Deep-Spiking-Q-Networks

[1] Michael Pfeiffer and Thomas Pfeil. Deep Learning With Spiking Neurons: Opportunities and Challenges. *Frontiers in NeuroScience*, 25 October 2018.

[2] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, and Shane Legg & Demis Hassabis. Human-level control through deep reinforcement learning. *Nature, Volume 518*, 26 February 2015.

[3] Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in Neural Information Processing Systems*, (April 2015):1057–1063, 2000. ISSN 10495258.

[4] Friedemann Zenke and Surya Ganguli. SuperSpike: Supervised learning in multi-layer spiking neural networks. October 14, 2017.

[5] Emre O. Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate Gradient Learning in Spiking Neural Networks. *IEEE SPM WHITE PAPER FOR THE SPECIAL ISSUE ON NEUROMORPHIC COMPUTING*, 28 January 2019.

[6] Devdhar Patel, Hananel Hazan, Daniel J. Saunders, Hava T. Siegelmann, and Robert Kozma. Improved robustness of reinforcement learning

policies upon conversion to spiking neuronal network platforms applied to Atari Breakout game. *Neural Networks*, 2019. ISSN 08936080. doi: 10.1016/j.neunet.2019.08.009. URL `https://doi.org/10.1016/j.neunet.2019.08.009`.

[7] Claus Meschede. Training Neural Networks for Event-Based End-to-End Robot Control. *Master thesis at the Technical University of Munich*, 28 July 2017.

[8] Nicolas Frémaux, Henning Sprekeler, and Wulfram Gerstner. Reinforcement Learning Using a Continuous Time Actor-Critic Framework with Spiking Neurons. *PLoS Computational Biology*, 9(4), 2013. ISSN 1553734X. doi: 10.1371/journal.pcbi.1003024.

[9] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Open AI Gym. *CoRR*, abs/1606.01540, 2016. URL `http://arxiv.org/abs/1606.01540`.

[10] Charl Linssen, Mikkel Elle Lepperød, Jessica Mitchell, Jari Pronold, Jochen Martin Eppler, Chrisitan Keup, Alexander Peyser, Susanne Kunkel, Philipp Weidel, Yannick Nodem, Dennis Terhorst, Rajalekshmi Deepu, Moritz Deger, Jan Hahne, Ankur Sinha, Alberto Antonietti, Maximilian Schmidt, Luciano Paz, Jesús Garrido, Tammo Ippen, Luis Riquelme, Alex Serenko, Tobias Kühn, Itaru Kitayama, Håkon Mørk, Sebastian Spreizer, Jakob Jordan, Jeyashree Krishnan, Mario Senden, Espen Hagen, Alexey Shusharin, Stine Brekke Vennemo, Dimitri Rodarie, Abigail Morrison,

Steffen Graber, Jannis Schuecker, Sandra Diaz, Barna Zajzon, and Hans Ekkehard Plesser. NEST 2.16.0, August 2018. URL https://doi.org/10.5281/zenodo.1400175.

[11] University of Manchester, University of Southampton, University of Cambridge, University of Sheffield, ARM Ltd., Silistix Ltd, and Thales. SpiNNaker webpage: http://apt.cs.manchester.ac.uk/projects/SpiNNaker/, retrieved on 04.10.2019.

[12] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pages 3207–3214, 2018.

[13] P. Davison Andrew, Daniel Brüderle, Jochen Eppler, Jens Kremkow, Eilif Muller, and Dejan Pecevski. PyNN : a common interface for neuronal network simulators. 2(January):1–10, 2009. doi: 10.3389/neuro.11.011.2008.

[14] Emre Neftci, Charles Augustine, Somnath Paul, and Georgios Detorakis. Neuromorphic Deep Learning Machines. pages 1–25, 2016. doi: 10.3389/fnins.2017.00324. URL http://arxiv.org/abs/1612.05596{%}0Ahttp://dx.doi.org/10.3389/fnins.2017.00324.