PROGRESS IN NEUROMORPHIC COMPUTING

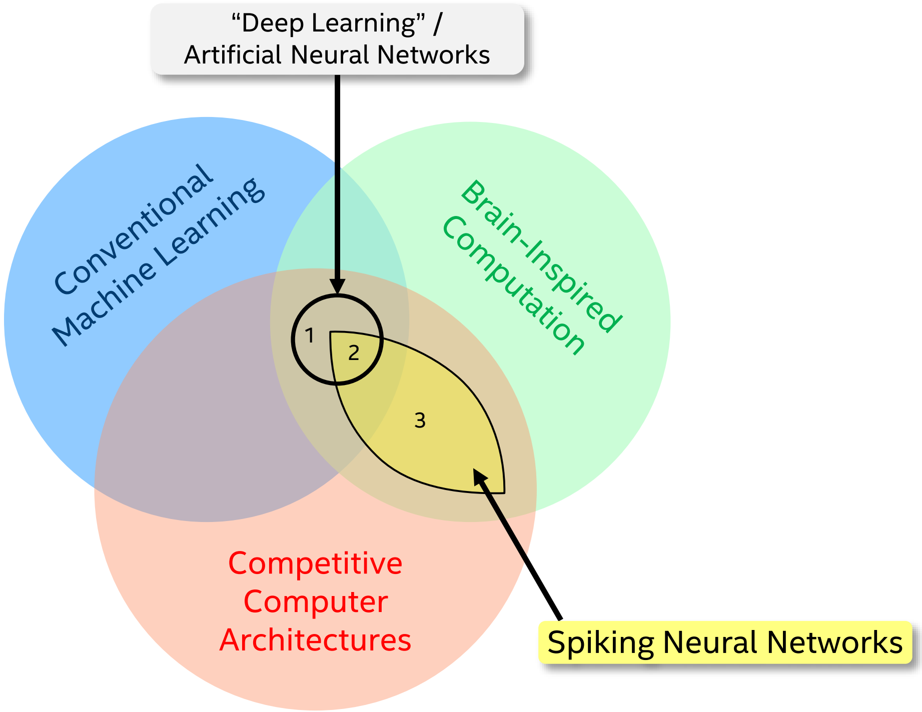
Drawing Inspiration from Nature for Gains in AI and Computing

Mike Davies
Director, Neuromorphic Computing Lab | Intel Labs

June 13, 2019
Nengo Summer School

LEGAL INFORMATION

Neuromorphic Computing Exploration Space



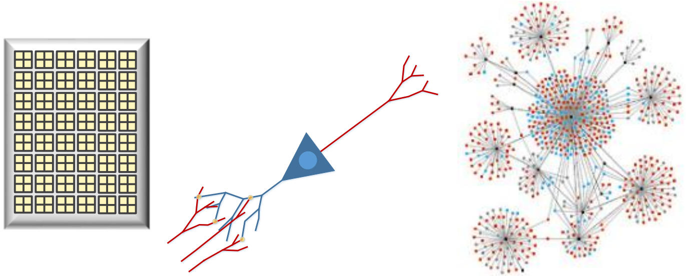
Research Goals:

- **Broad class** of brain-inspired computation
- **Efficient** hardware implementations
- **Scalable** from small to large problems and systems

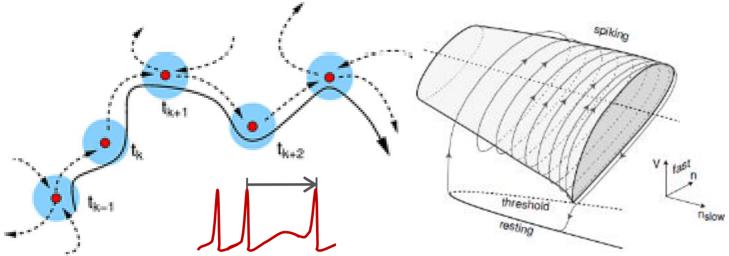
Examples:

- Learning without cloud assistance
- Learning with sparse supervision
- Online and lifelong learning
- Probabilistic inference and learning
- Sparse coding
- Associative memory, similarity matching
- Nonlinear adaptive control (robotics)
- SLAM and path planning
- Constraint satisfaction
- Dynamical systems modeling

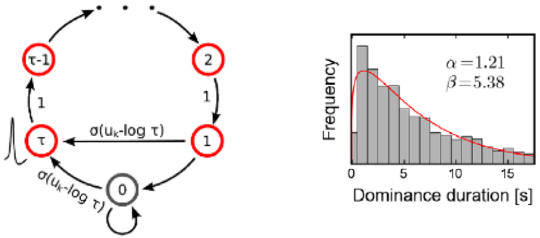
Some Principles of Neural Computation



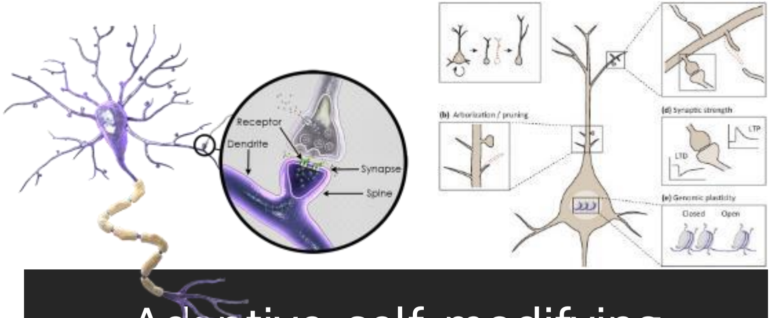
Fine-grained parallelism with massive fanout



Event-driven computation *with* time



Low precision and stochastic

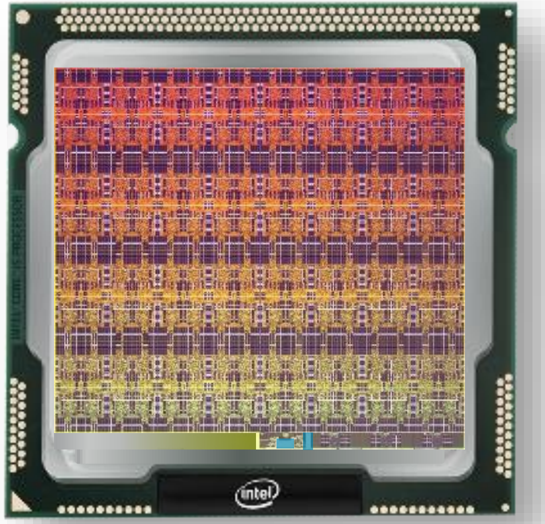


Adaptive, self-modifying

OUR LOIHI RESEARCH CHIP

KEY PROPERTIES

- 128 neuromorphic cores supporting up to 128k neurons and 128M synapses with an **advanced spiking neural network feature set**.
- Supports **highly complex neural network topologies**
- **Scalable on-chip learning** capabilities to support an unprecedented range of learning algorithms
- Fully digital **asynchronous** implementation
- Fabricated in Intel's **14nm FinFET process** technology



**Integrated
Memory + Compute
Neuromorphic Architecture**

Davies et al, "Loihi: A Neuromorphic Manycore Processor with On-Chip Learning." IEEE Micro, Jan/Feb 2018.

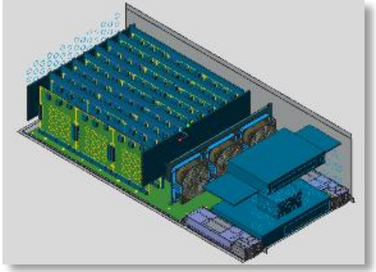
Loihi Systems

Q4 2017
Wolf Mountain
Remote Access
4 Loihi/Board

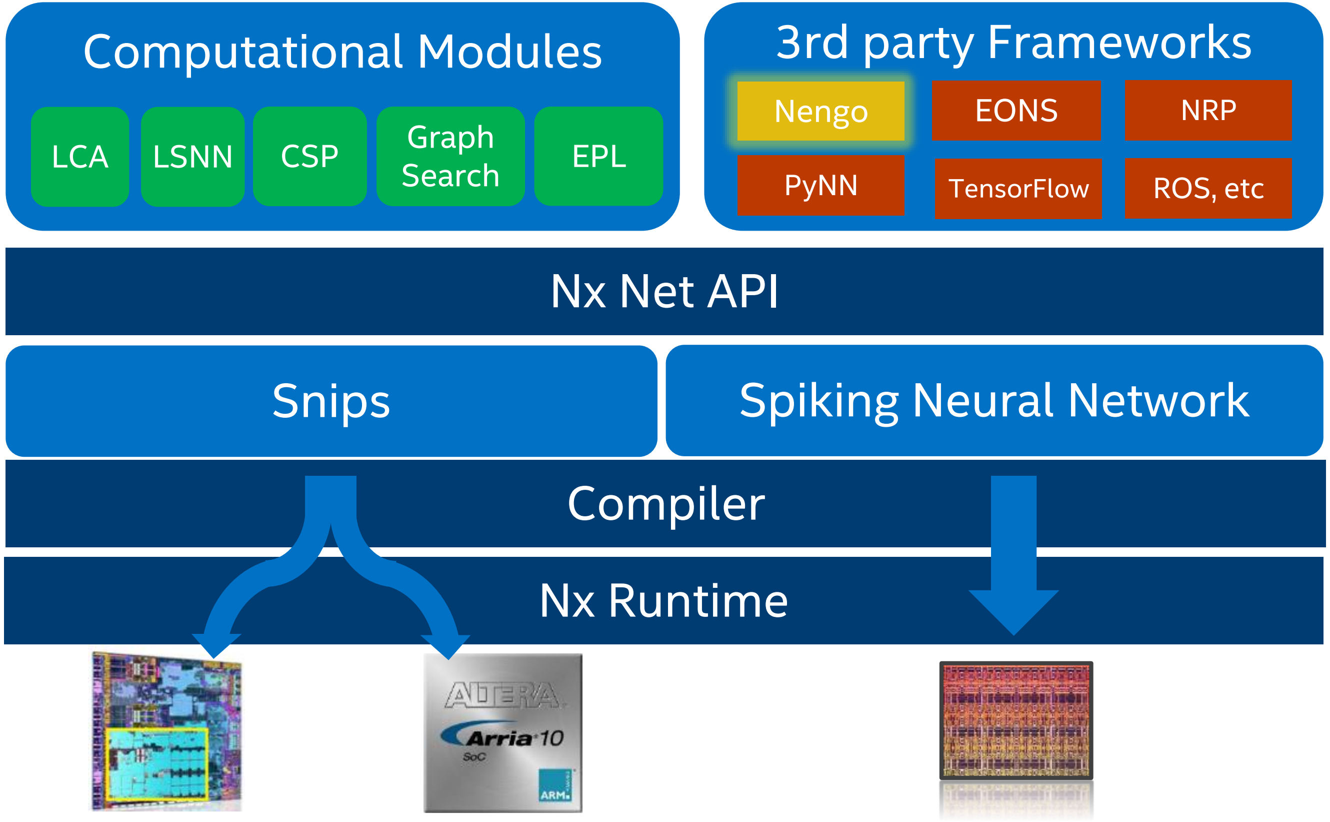
Q2 2018
Nahuku
Arria10 Expansion Board
For cloud & local use
8-32 Loihi/Board

Q3 2018
Kapoho Bay
1-2 Loihi
DVS interface
USB host interface

Q2 2019
Pohoiki Springs
Remote Access
Up to 768 chips
(100M neurons)

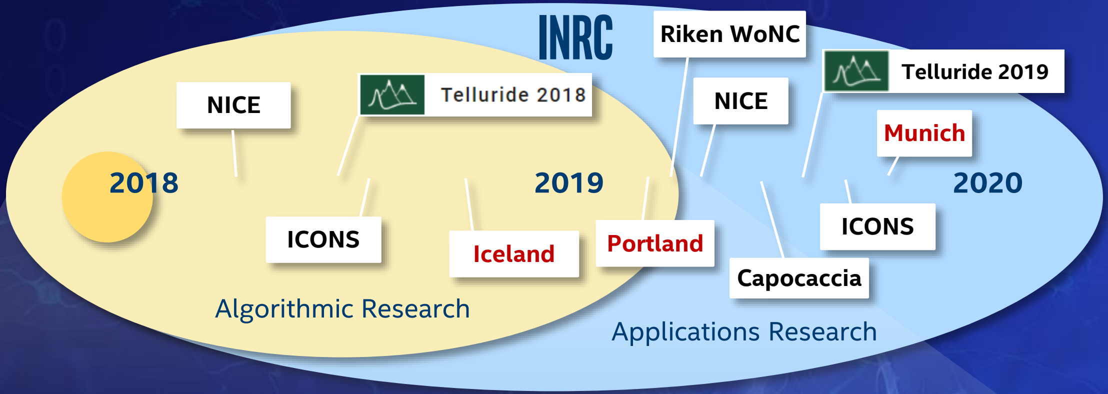


Nx SDK Software Architecture



INTEL NEUROMORPHIC RESEARCH COMMUNITY

Collaborating to Accelerate Progress

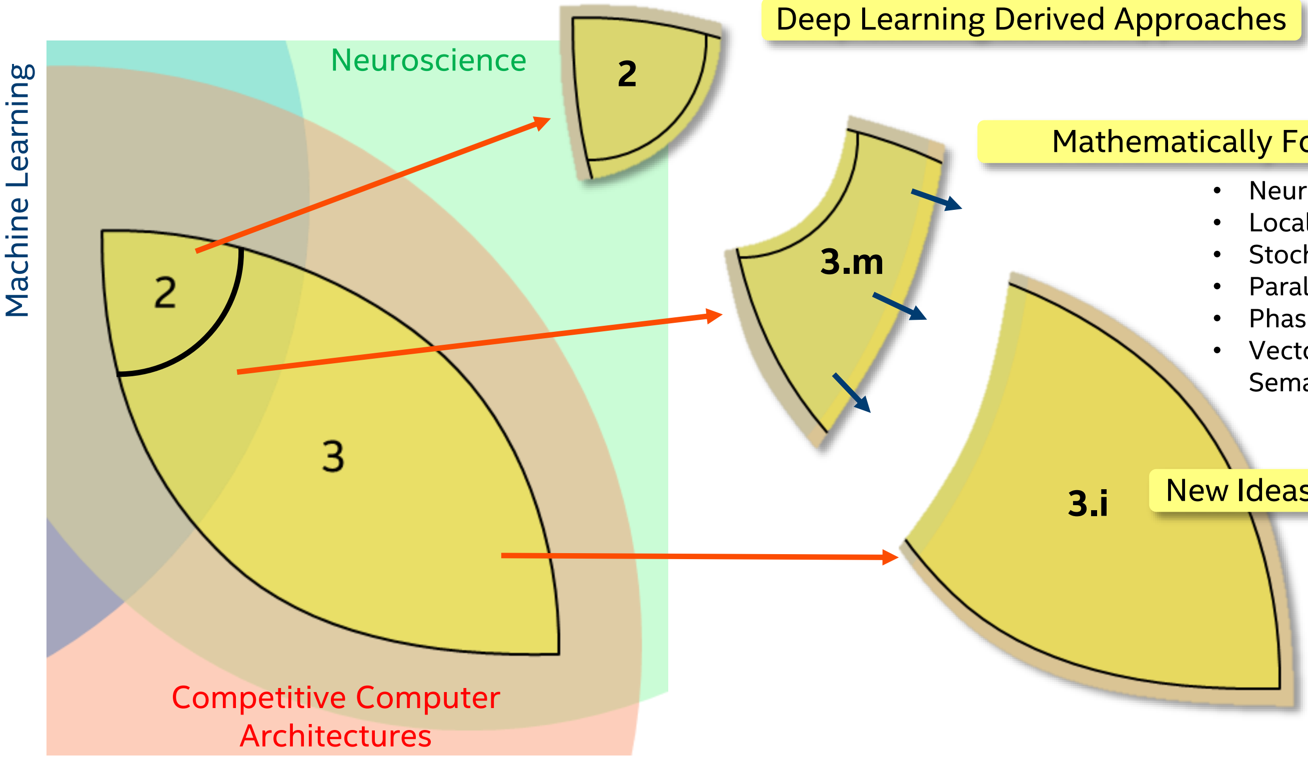


Over 50 active projects

Iceland Workshop (Sep 28 – Oct 2) attended by 62 researchers

Winter Workshop (Feb 11-15) attended by 90+ researchers

The Challenge: SNN Algorithm Discovery



Deep Learning Derived Approaches

- DNN -> SNN conversion
- SNN backpropagation
- Online SNN pseudo-backprop

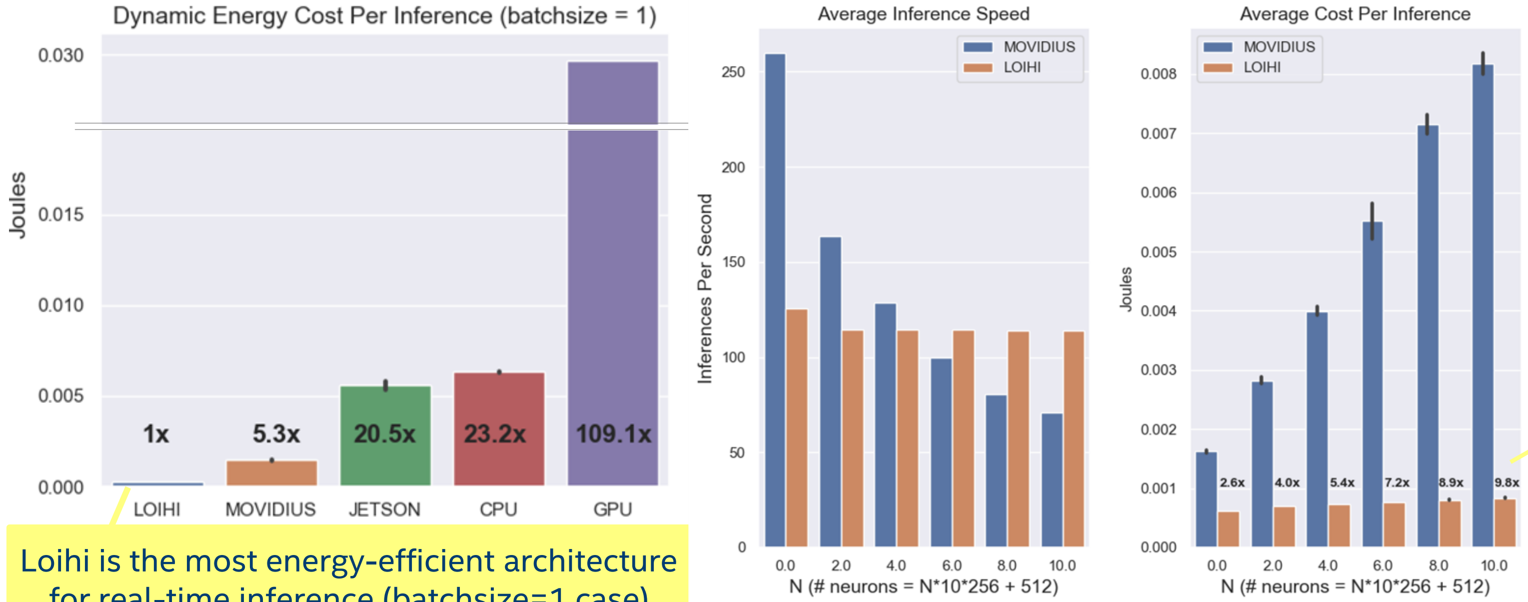
Mathematically Formalized

- Neural Engineering Framework (NEF)
- Locally Competitive Algorithm for LASSO
- Stochastic SNNs for solving CSPs
- Parallel graph search
- Phasor associative memories
- Vector symbolic architectures (VSA) *aka* Semantic pointer architecture (SPA)

New Ideas Guided by Neuroscience

- Olfaction-inspired rapid learning
- Dynamic Neural Fields
- SLAM
- Evolutionary search
- Cortical models

DNN-to-SNN conversion for keyword spotting



Loihi is the most energy-efficient architecture for real-time inference (batchsize=1 case)

Loihi provides extremely good scaling vs conventional architectures as network size grows by 50x

- Loihi provides 5-10x lower energy than closest conventional DNN architecture
- Caveats: batchsize=1 and reduced accuracy (90.6% SNN vs 92.7% DNN)

Results from: Blouw et al, “Benchmarking Keyword Spotting Efficiency on Neuromorphic Hardware.” arXiv:1812.01739

LASSO Sparse Coding

The *Spiking Locally Competitive Algorithm (S-LCA)*

Problem

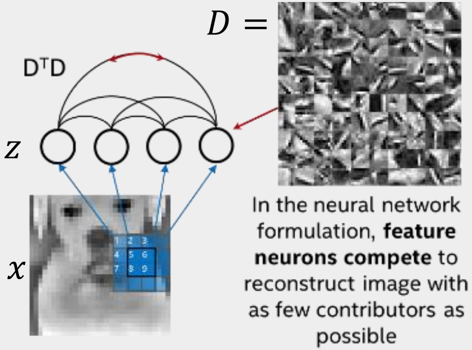
$$\min_z \frac{1}{2}\|x - Dz\|_2^2 + \lambda\|z\|_1$$

Input

Reconstruction

Sparse regularization

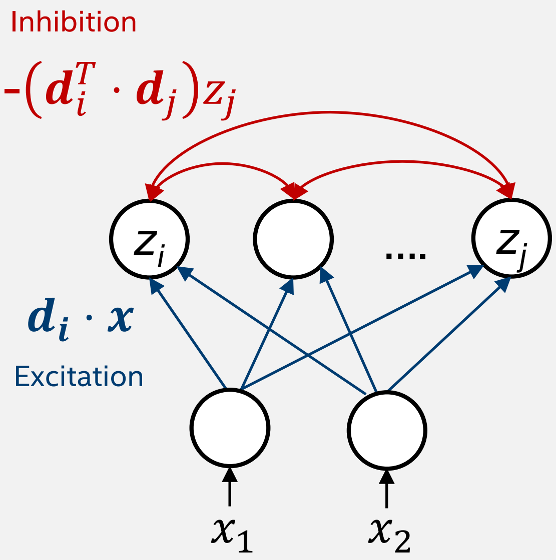
Implementation



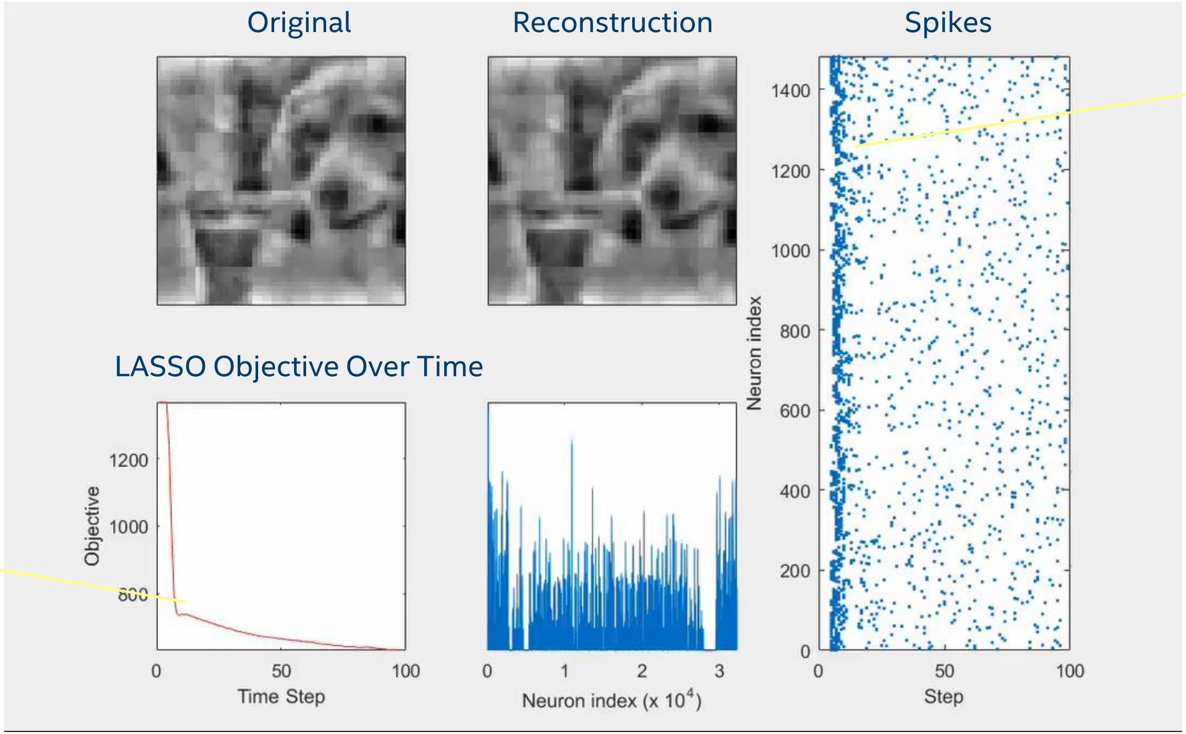
In the neural network formulation, **feature neurons compete** to reconstruct image with as few contributors as possible

Tang et al, arxiv: 1705:05475

Neural Network Structure



Spiking LCA dynamics on Loihi



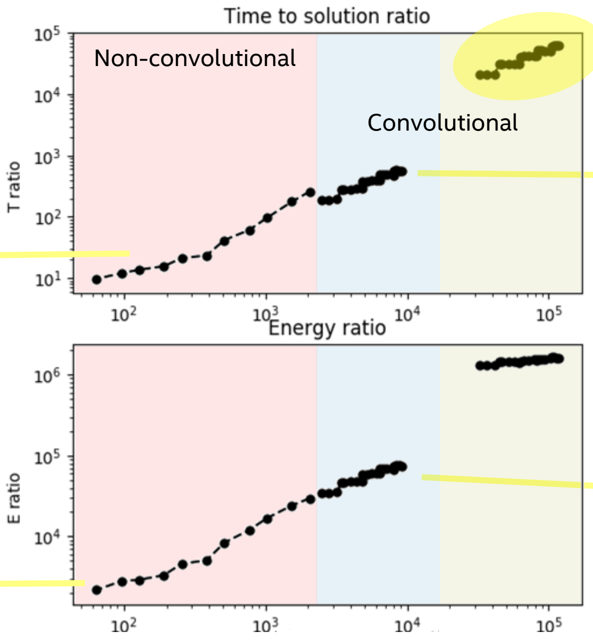
Intense but very brief period of competition

Much faster convergence on a neuromorphic architecture

LCA on Loihi compared to FISTA on Core i7 CPU

Clear, compelling scaling trend across both non-convolutional and convolutional examples.

CPU/Loihi Ratios



unknowns

>10,000x faster

(Possibly unfair to CPU since SPAMS is not optimized for convolutional LASSO.)

100-1000x faster

10-50x faster

10,000-100,000x lower energy

1,000-10,000x lower energy

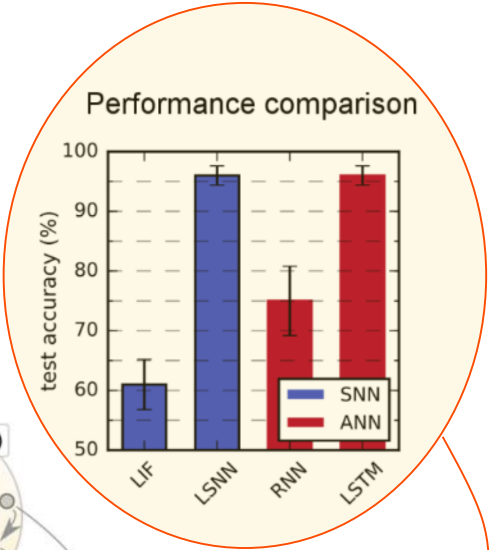
* **Intel Core i7-4790 3.6GHz w/ 32GB RAM.** FISTA solver: **SPAMS** http://spams-devel.gforge.inria.fr/
Performance results are based on testing as of December 2018 and may not reflect all publicly available security updates. No product can be absolutely secure.

Spike-based LSTMs – “LSNNs”

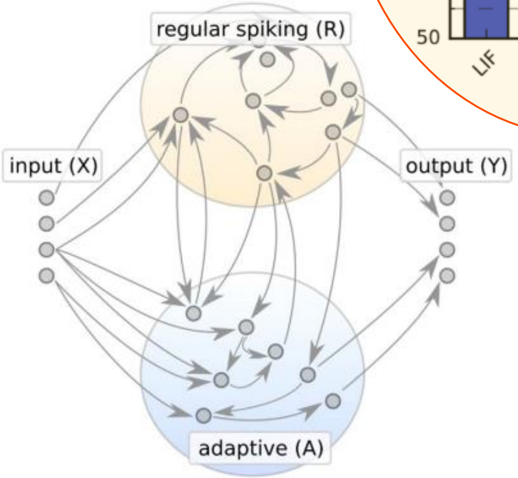
Simple adaptive spiking model achieves LSTM-level accuracy

- SNN reservoir augmented with adaptive neurons
- Thresholds rise on each spike, decay exponentially
  ☞ *Highly energy-efficient adaptation*
- Trained offline with BPTT (TensorFlow)
- For Sequential MNIST dataset:
  - **Loihi achieves 94% accuracy**
  - LSTM: **98%** (simple RNNs: **68-89%**)

[Bellec et al, arXiv preprint arXiv:1803.09574]



First case of an **SNN matching LSTM accuracy**



LSNN Benchmarking Results

Algorithm	Dataset	Training	#Param	Best Accuracy
LSNN	Sequential MNIST	SNN Backprop + DEEP-R w/ TensorFlow (Adam Optimizer)	68210	94.1%
LSTM	Sequential MNIST	Standard Backprop w/ TensorFlow (Adam Optimizer)	67850	98.5%

Loihi is **best on all metrics**, *including throughput* (with batch size = 1)

Architecture	Batch size	Energy per inference (mJ)		Latency per inference (ms)		Inference Throughput (1/s)	
Loihi[1]	1	2.68	1x	21.5	1x	47	1x
Intel Core i5-7440HQ[2]	1	1740	649x	83.2	3.9x	12	1/4x
Intel Core i7-7700HQ[2]	1	2510	937x	77.7	3.6x	13	1/3.6x
NVIDIA GeForce GTX 1050 Ti[3]	1	n/a	n/a	66.8	3.1x	15	1/3.1x
NVIDIA Tesla P100[4]	1	3480	1298x	94.9	4.4x	11	1/4.3x
NVIDIA Tesla P100[4]	64	171	64x	148	6.9x	435	9.3x

Best GPU is **worst** on all metrics except **throughput w/ large batch size**

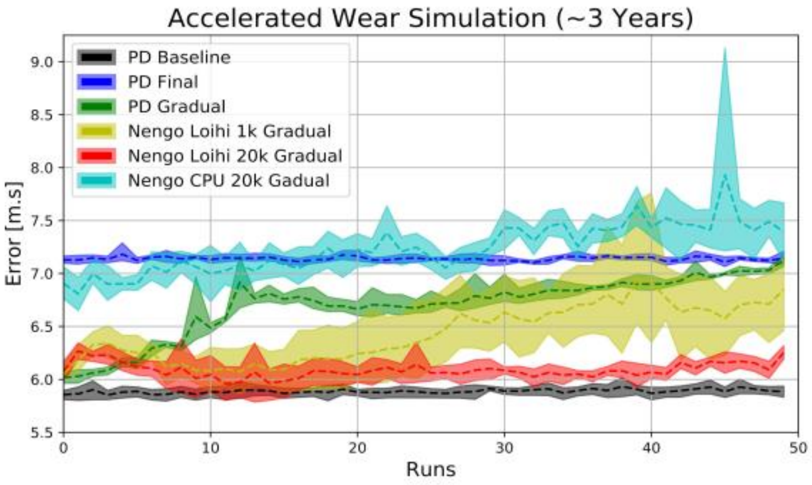
[1] Loihi Wolf Mountain running NxSDK 0.85
[2] 2.8-3.8 GHz CPU with 16 GB RAM. TensorFlow v1.14.1 on Windows 10.
[3] 4GB RAM, CUDA v10.0. Driver v419.17. TensorFlow v1.13.1
[4] 16GB RAM, CUDA v10.0. Driver v410.104. TensorFlow v1.10.1

Performance results are based on testing as of December 2018 and may not reflect all publicly available security updates. No product can be absolutely secure.

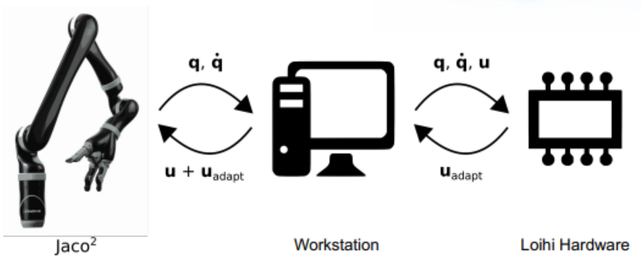
Adaptive Control of a Robot Arm Using Loihi

SNN adaptive dynamic controller implemented on Loihi allows a robot arm to adjust in real time to nonlinear, unpredictable changes in system mechanics[1][2].

Result outperforms standard PD & PID control algorithms.



Different control methods adapting to a gradual, linear increase in friction, over the course of 50 runs. This simulates ~3 years of wear over the course of 16.67 minutes of run time, a 90K times speed up. Only 20K neurons on Loihi is able to successfully cope with this perturbation.



[1] DeWolf, T., Stewart, T. C., Slotine, J. J., & Eliasmith, C. (2016, November). A spiking neural model of adaptive arm control. In *Proc. R. Soc. B* (Vol. 283, No. 1843, p. 20162134). The Royal Society.
[2] Eliasmith, "Building applications with next generation neuromorphic hardware." *NICE Workshop 2018*

Graph Search – Path Planning

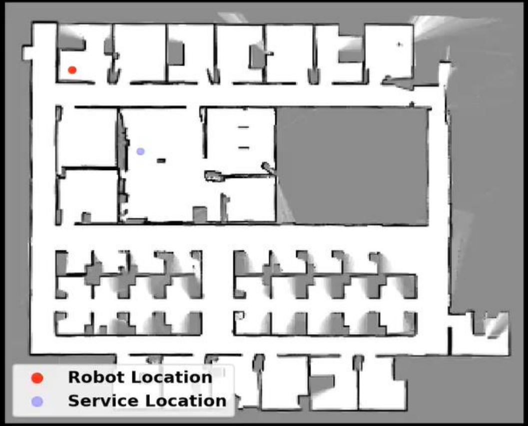
Runtime comparison to best Djikstra optimizations:
- Neuromorphic: $O(L \cdot \sqrt{V})$
- Standard: $O(E)$

For most nontrivial problems:
- L<<E
- V<<E

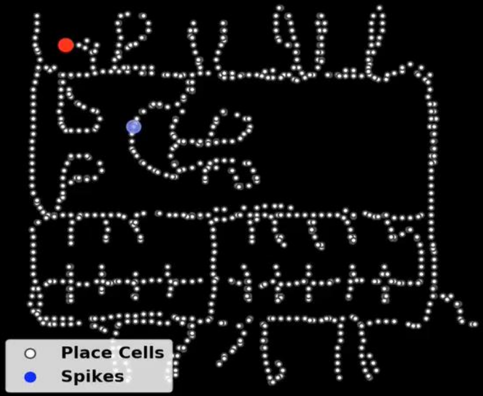
Neuromorphic solution uses *fine-grain parallelism* an *temporal wavefront-driven computation* to potentially provide great performance gains for large problems.

Based on Ponulak F., Hopfield J.J. *Rapid, parallel path planning by propagating wavefronts of spiking neural activity. Front. Comput. Neurosci. 2013. V. 7. Article № e98.*

Robot Motion



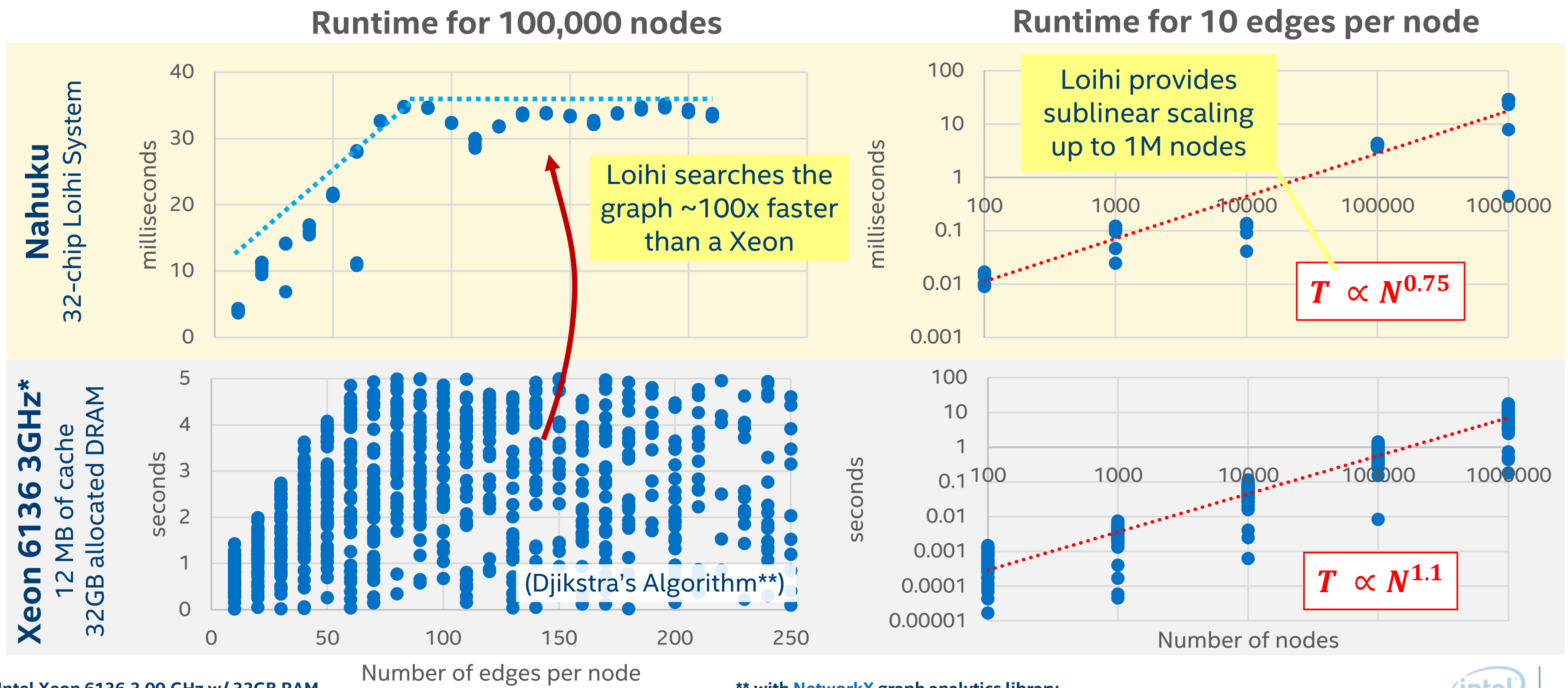
Loihi Representation



DARPA SDR Site B
(Data from Radish Robotics Dataset)

Searching Small World Networks with Loihi

Watts-Strogatz network model with rewiring probability 20%.



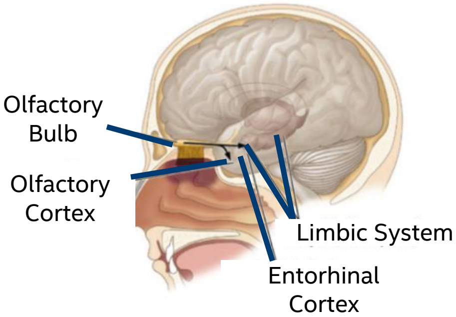
* Intel Xeon 6136 3.00 GHz w/ 32GB RAM. ** with NetworkX graph analytics library

Performance results are based on testing as of December 2018 and may not reflect all publicly available security updates. No product can be absolutely secure.

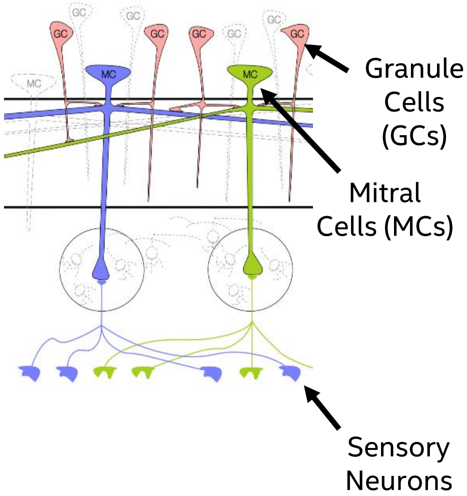
3.i

Olfaction-Inspired Pattern Matching

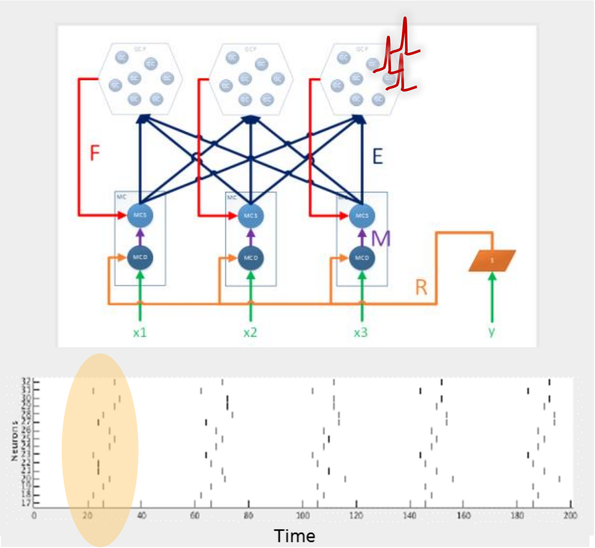
Olfactory System



Olfactory Bulb Neural Circuit



Spatiotemporal Attractor Model



Example of a novel ML algorithm abstracted from detailed systems neuroscience model

[Nabil Imam (Intel) with Thomas Cleland (Cornell) – in review]

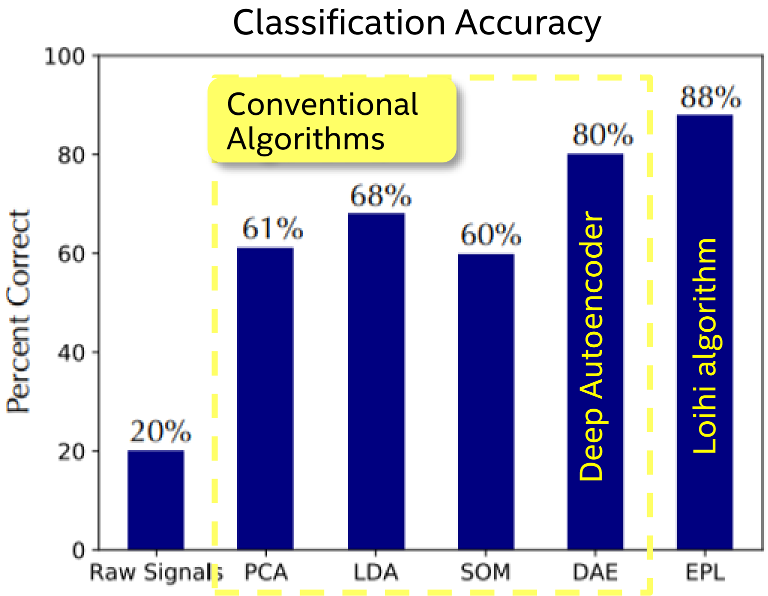
Olfaction-Inspired Pattern Matching and Learning

Supports one-shot learning, outperforms conventional algorithms

Provides average of **8% accuracy improvement** vs deep autoencoder

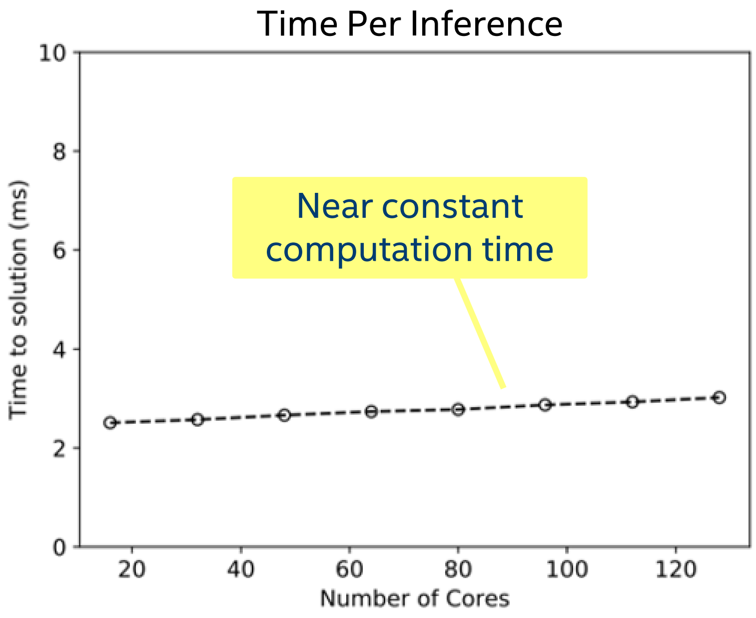
40x more data efficient learning vs backpropagation

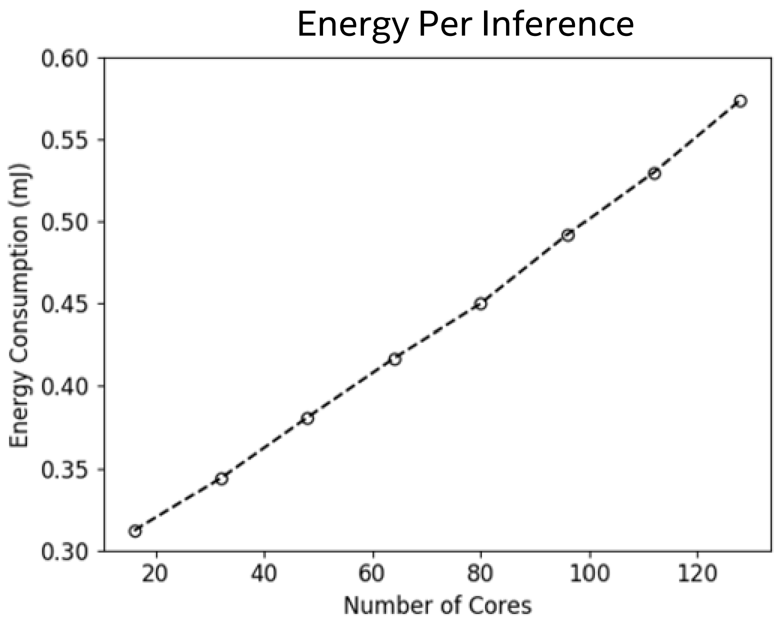
Supports **online learning** (robust to catastrophic forgetting)



Olfaction-Inspired Pattern Matching and Learning

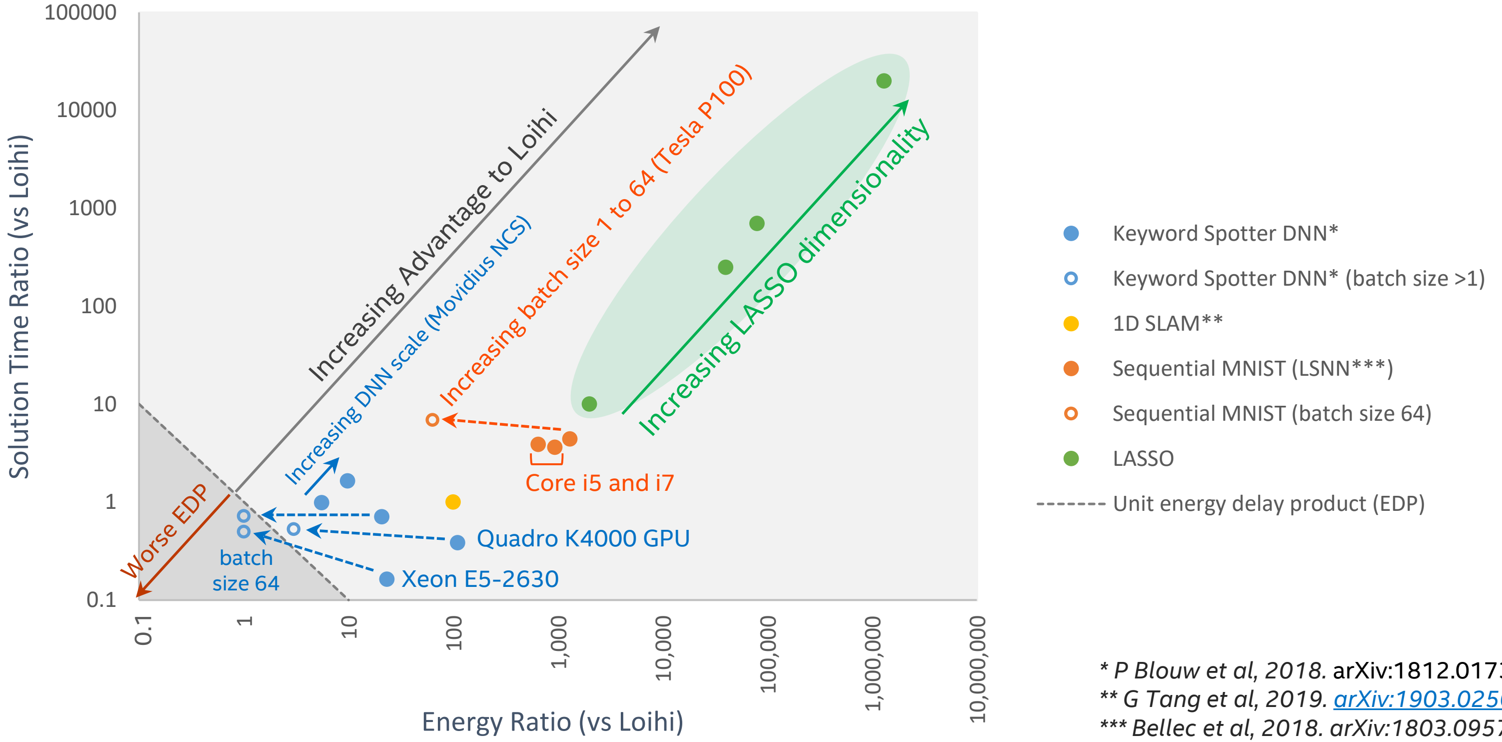
Compelling computational efficiency





Performance results are based on testing as of December 2018 and may not reflect all publicly available security updates. No product can be absolutely secure.

Loihi Benchmarking Summary



* *P Blouw et al, 2018.* arXiv:1812.01739
** *G Tang et al, 2019.* *arXiv:1903.02504*
*** *Bellec et al, 2018. arXiv:1803.09574*

Perspectives on Spikes

Findings from our research

1) Sparse communication in time optimizes energy efficiency (**bits/J vs bits/s**)
2) Spikes efficiently compute many **rate-based models**
3) Spikes provide efficient and natural **processing of temporal data**
4) Spikes support **event-based algorithms** that have nothing to do with rates
5) Spikes efficiently implement **phasor neural networks**

In all examples studied so far, benefits vs conventional architectures
increase with increasing problem scale

The Research Frontier

Advancing from Compelling Algorithms to Viable Applications

- Inference and learning of sparse feature representations
- Video and speech recognition
- Event-based camera processing
- Chemosensing
- Robotics
- Adaptive dynamic control
- Anomaly detection for security and industrial monitoring
- Optimization: Constraint Satisfaction, QUBO, Convex optimization
- Autonomy: SLAM, planning, closed-loop behavior

Low Energy | Low Latency | Adaptive | Batch Size = 1 | High Cost

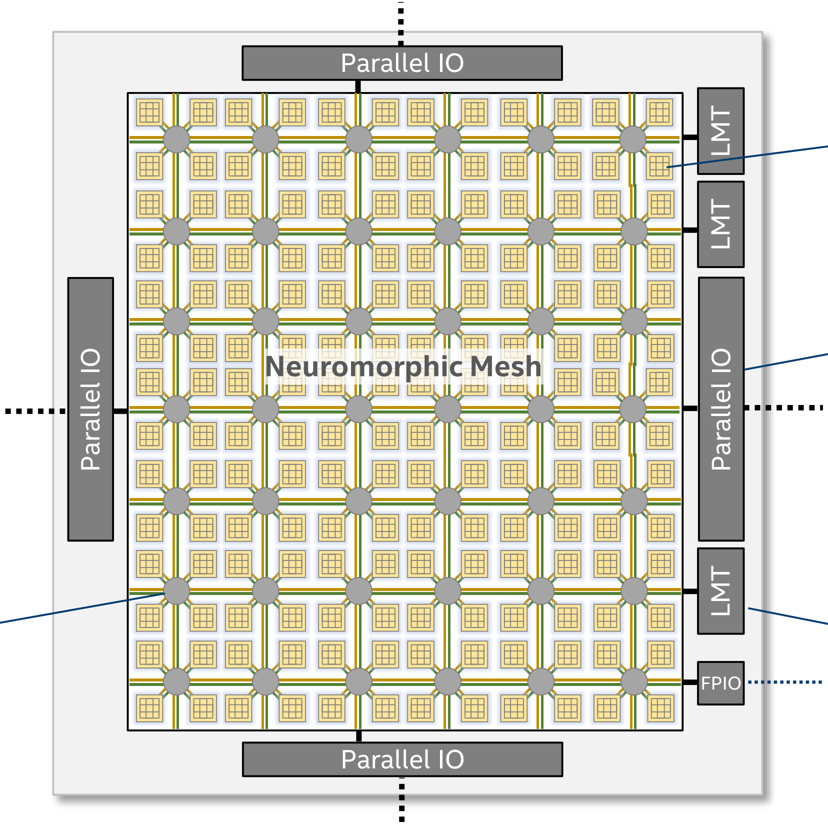
LOIHI ARCHITECTURE OVERVIEW

Neuromorphic Computing Lab | Intel Labs

Nengo Summer School 2019

Chip Architecture

Technology:	14nm
Die Area:	60 mm^2
Core area:	0.41 mm^2
NmC cores:	128 cores
x86 cores:	3 LMT cores
Max # neurons:	128K neurons
Max # synapses:	128M synapses
Transistors:	2.07 billion



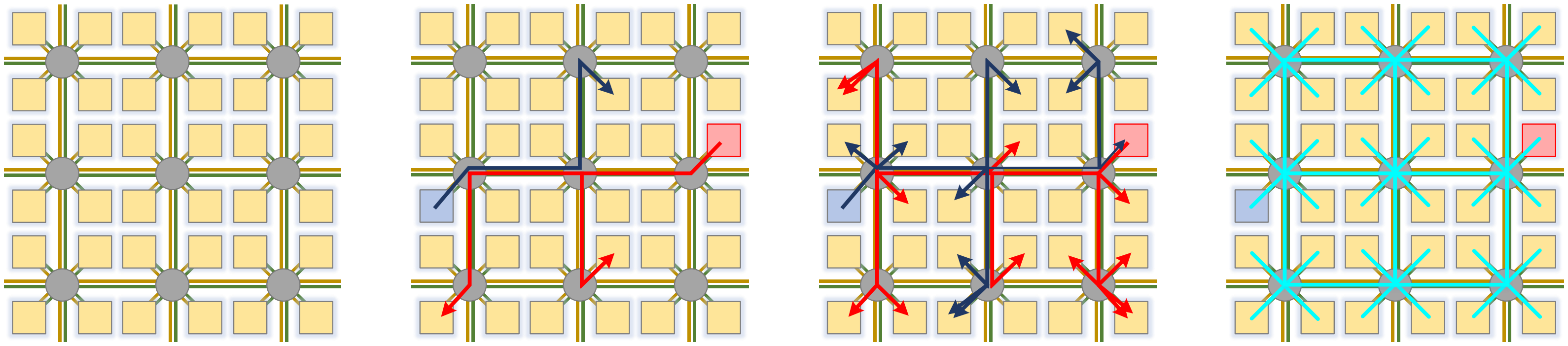
Neuromorphic core
- LIF neuron model
- Programmable learning
- 128 KB synaptic memory
- Up to 1,024 neurons
- Asynchronous design

Parallel off-chip interfaces
- Two-phase asynchronous
- Single-ended signaling
- 100-200 MB/s BW

Low-overhead NoC fabric
- 8x16-core 2D mesh
- Scalable to 1000's cores
- Dimension order routed
- Two physical fabrics
- 8 GB/s per hop

Embedded x86 processors
- Efficient spike-based communication with neuromorphic cores
- Data encoding/decoding
- Network configuration
- Synchronous design

Mesh Operation: Fine-Grained Synchronization



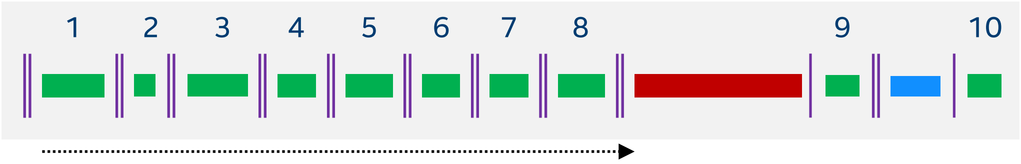
Time step T begins.

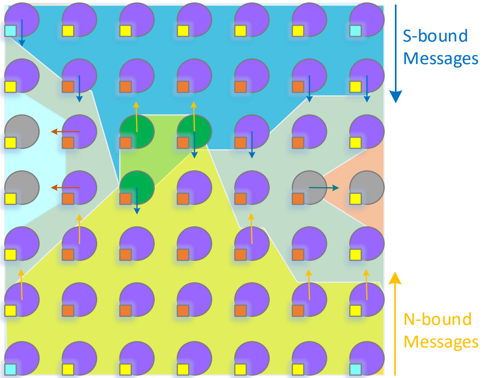
Cores update dynamic neuron state and evaluate firing thresholds

Above-threshold neurons send spike messages to fanout cores

(Two neuron firings shown.)

All neurons that fire in time T route their spike messages to all destination cores.





Neuromorphic Core Architecture

All synaptic connections pooled
128KB shared memory

Sparse, *dense*, and *hierarchical*
Synaptic mapping representations

Synaptic delays

Synaptic eligibility traces

Flexible 3-tuple synaptic variables
(1-9b weight, 0-6b delay, 0-8b tag)

Graded “reward spikes”

Flexible synaptic plasticity with microcode-programmable rules

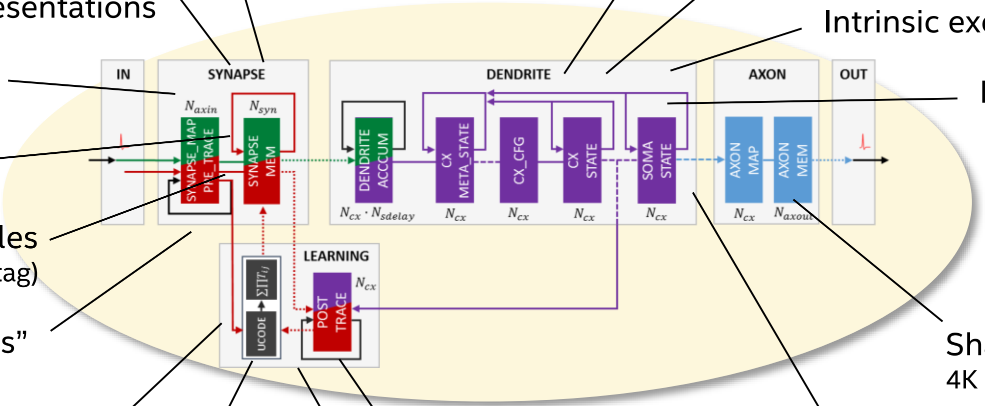
Sum-of-products rule semantics

Discrete time LIF neuron model (CUBA)

Multi-compartment dendritic trees
up to 1K compartments

Intrinsic excitability homeostasis

Random noise sources



Shared output routing table
4K axon routes

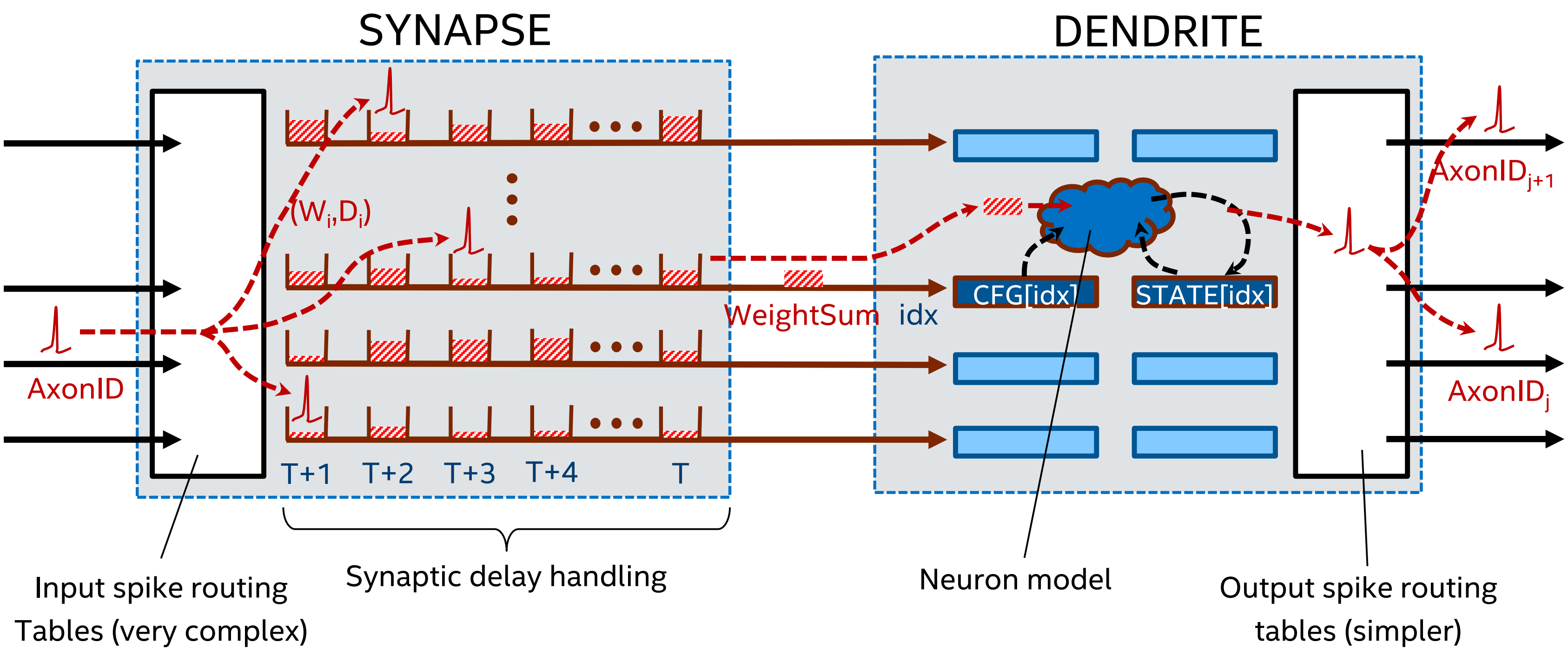
Axon delays
Refractory delays (+ random)

Filtered spike train traces

Plasticity rules target any synaptic variable

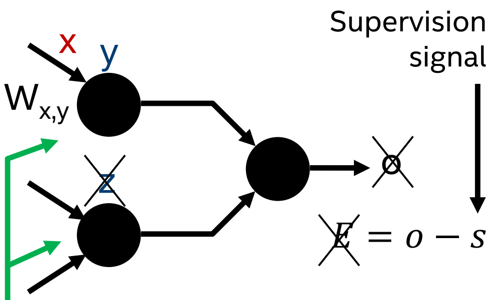
Basic Core Operation (Non-Learning)

(Time multiplexing illustrated unrolled in space)



Learning with Synaptic Plasticity

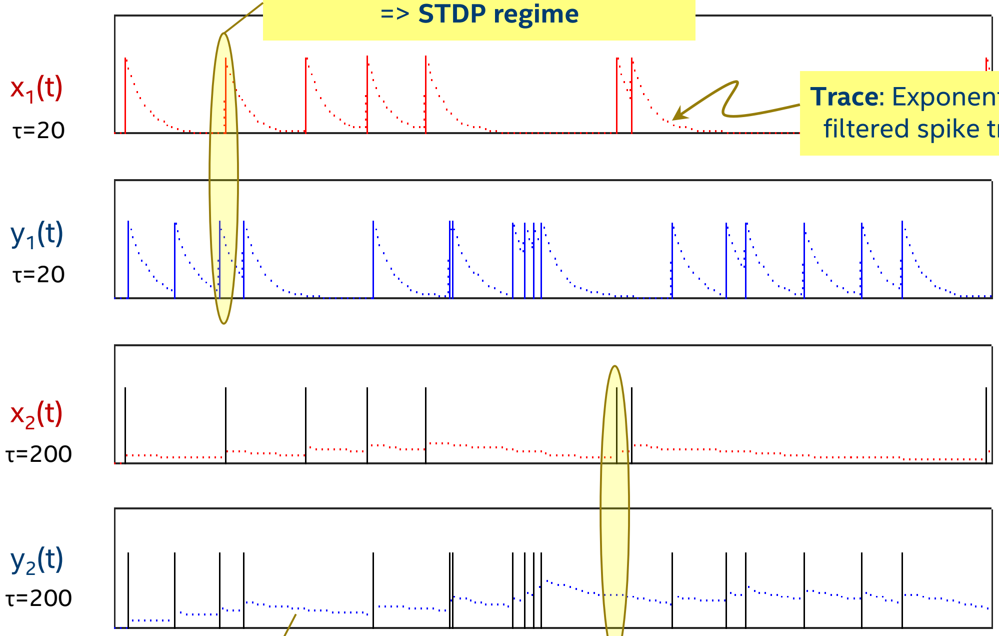
- **Local learning rules** – essential property for efficient scalability
- Rules derived by **optimizing an emergent statistical objective**
- Plasticity on **wide range of time scales** for
  - ✓ Immediate supervised (labelled) learning
  - ✓ Unsupervised self-organization
  - ✓ Working memory
  - ✓ Reinforcement-based delayed feedback



Learning rules for weight $W_{x,y}$ may *only* access presynaptic state x and postsynaptic state y

However *reward spikes* may be used to distribute graded reward/punishment values to a particular set of axon fanouts

Trace-Based Programmable Learning

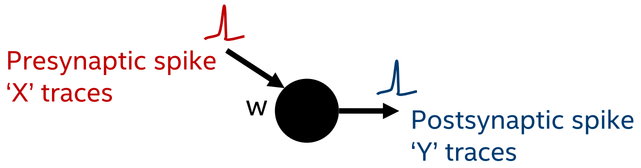


Short time scale trace correlations
=> **STDP regime**

Trace: Exponentially filtered spike train

Traces are **low precision** (7-9b) and may decay **stochastically** for implementation efficiency

Long time scale traces respond to correlations in activity rates



Weight, Delay, and Tag learning rules programmed as **sum-of-product equations**

$$w' = w + \sum_{i=1}^{N_P} S_i \prod_{j=1}^{n_i} (V_{i,j} + C_{i,j})$$

<u>Synaptic Variables</u>
Wgt, Delay, Tag
(variable precision)

<u>Variable Dependencies</u>
X_0, Y_0, X_1, Y_1, X_2, Y_2,
Wgt, Delay, Tag, etc.

Learning Rule Examples

Pairwise STDP:

$$W(t+1) = W(t) - A_{-}x_0(t)y_1(t) + A_{+}x_1(t)y_0(t)$$

Triplet STDP with heterosynaptic decay:

$$W(t+1) = W(t) - A_{-}x_0(t)y_1(t) + A_{+}x_1(t)y_0(t)y_2(t) - B \cdot W(t) \cdot y_3(t)$$

Delay STDP:

$$D(t+1) = D(t) - A_{-}x_0(t)(127 - y_1(t)) + A_{+}(127 - x_1(t))y_0(t)$$

Two-variable Learning Rule Examples

Distal Reward with Synaptic Tags:

$$T(t+1) = T(t) - A_{-}x_0(t)y_1(t) + A_{+}x_1(t)y_0(t) - B \cdot T(t)$$

$$W(t+1) = W(t) + C \cdot r_1(t) \cdot T(t)$$

STDP with dynamic weight consolidation:

$$W(t+1) = W(t) - A_{-}x_0(t)y_1(t) + A_{+}x_1(t)y_0(t)y_2(t) - B_1(W-T)y_3(t)y_0(t)$$

$$T(t+1) = T(t) + \frac{1}{\tau_{cons}}(W-T) - B_2T(w_\theta - T)(w_{max} - T)$$

intel®