Probabilistic Neuromorphic Computing and Communications

Osvaldo Simeone

King's College London

Joint KIT-TU/e Workshop 2021

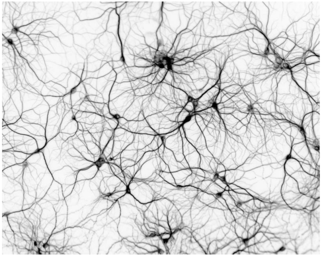
Overview

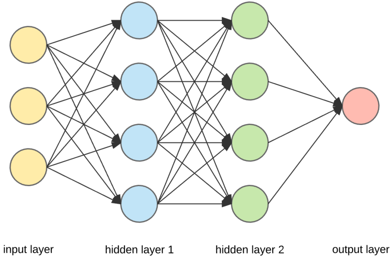
- Motivation
- Models
- Learning for probabilistic SNNs
- Bayesian learning for SNNs

Motivation

Neuromorphic Computing

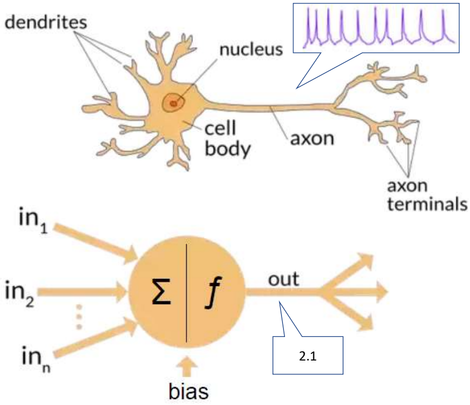
- Artificial neural networks (ANN) borrow from biological brains parallelism and high connectivity among similar computing units (neurons)...





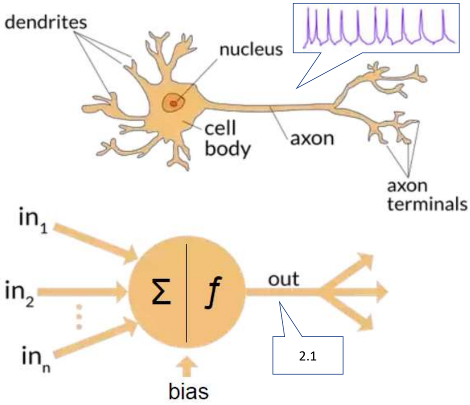
Neuromorphic Computing

- Neurons in ANNs abstract away the dynamic, sparse, event-driven, operation of biological neurons...
- What can be gained by developing machines that rely on more accurate neuronal models?



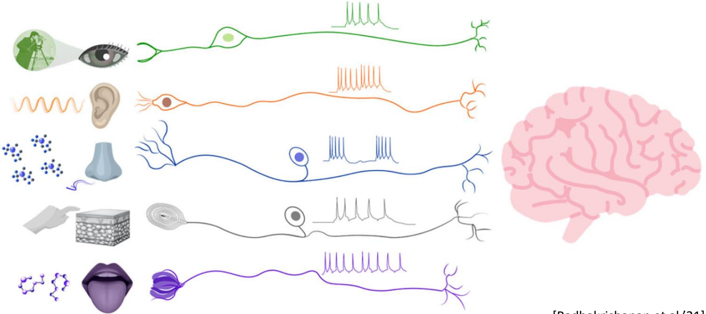
Neuromorphic Computing

- Neurons in ANNs abstract away the dynamic, sparse, event-driven, operation of biological neurons...
- What can be gained by developing machines that rely on more accurate neuronal models?



Neuromorphic Computing

- The question extends to sensing: Can it be useful to mimic more closely biological sensors such as retinas or cochleas?



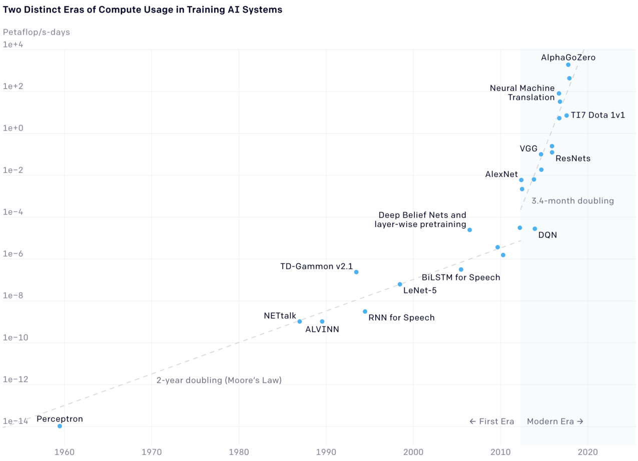
[Radhakrishanan et al '21]

Neuromorphic Computing

- The idea originates in the 90s with the work of Carver Mead

Machine Learning Today

- Computing power required for training of ANN-based models has seen a 300,000 times increase in 6 years.



[https://openai.com/blog/ai-and-compute/]

Neuromorphic Computing: Potential Applications

- Inference and learning on mobile or embedded devices with limited energy and memory resources [Welling '18]

mobile personal assistants

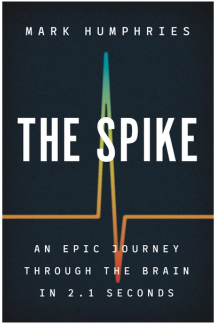
medical and health wearables

IoT mobile or embedded devices

neural prosthetics

Neuromorphic Computing and Spiking Neural Networks

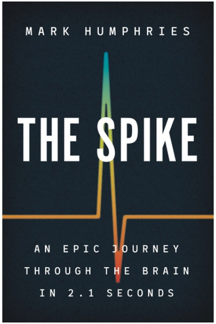
- Current neuromorphic computing platforms and algorithms implement Spiking Neural Networks (SNNs).
- SNNs replace static neurons with spiking, dynamic, neuronal models.
- Spikes enable high-capacity time encoding ("accurate"), low-delay signalling ("fast"), and low-SNR communications ("far").



[Gerstner '14]

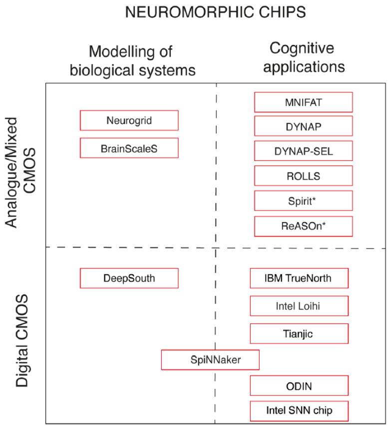
Neuromorphic Computing and Spiking Neural Networks

- Current neuromorphic computing platforms and algorithms implement Spiking Neural Networks (SNNs).
- SNNs replace static neurons with spiking, dynamic, neuronal models.
- Spikes enable high-capacity time encoding ("accurate"), low-delay signalling ("fast"), and low-SNR communications ("far").



[Gerstner '14]

Spiking Neural Networks



* Implemented with memristors

PROPERTIES:

- In-memory computing
- Fine-grained parallelism
- Learning in hardware
- Event based and asynchronous communication
- Reduced precision
- Spike-based processing
- Adaptability
- Leveraging noise and stochasticity

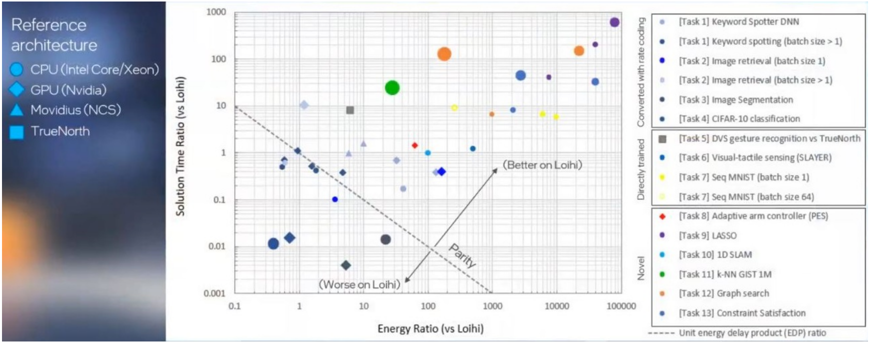
⋮

- Brain-inspired

[Mehonic and Kenyon '21]

Spiking Neural Networks

- Orders of magnitude gains in latency and energy have already been shown when selecting suitable workloads.



[INRC '21]

Applications

- Application properties necessary for realizing gains on neuromorphic architectures [INRC '21]:
  - **Streaming input data**, e.g. audio, video, or any signals changing on microsecond-to-second time scales
  - A need for **fast pattern matching, search, and optimization**
  - A need for **adaptation, fine-tuning, or associative learning** in response to arriving information
  - A need for **low latency responses**, e.g. as in closed-loop control applications (batching and vectorization may be unacceptable)
  - **Power constrained**
  - **Relatively small problem scales or else cost insensitive** due to use of a compute/memory-integrated architecture, which makes it costly to scale to large workloads (requiring more processing chips)

Applications

- Application properties necessary for realizing gains on neuromorphic architectures [INRC '21]:
  - **Streaming input data**, e.g. audio, video, or any signals changing on microsecond-to-second time scales
  - A need for **fast pattern matching, search, and optimization**
  - A need for **adaptation, fine-tuning, or associative learning** in response to arriving information
  - A need for **low latency responses**, e.g. as in closed-loop control applications (batching and vectorization may be unacceptable)
  - **Power constrained**
  - **Relatively small problem scales or else cost insensitive** due to use of a compute/memory-integrated architecture, which makes it costly to scale to large workloads (requiring more processing chips)

Applications

- Application properties necessary for realizing gains on neuromorphic architectures [INRC '21]:
  - **Streaming input data**, e.g. audio, video, or any signals changing on microsecond-to-second time scales
  - A need for **fast pattern matching, search, and optimization**
  - A need for **adaptation, fine-tuning, or associative learning** in response to arriving information
  - A need for **low latency responses**, e.g. as in closed-loop control applications (batching and vectorization may be unacceptable)
  - **Power constrained**
  - **Relatively small problem scales or else cost insensitive** due to use of a compute/memory-integrated architecture, which makes it costly to scale to large workloads (requiring more processing chips)

Applications

- Application properties necessary for realizing gains on neuromorphic architectures [INRC '21]:
  - **Streaming input data**, e.g. audio, video, or any signals changing on microsecond-to-second time scales
  - A need for **fast pattern matching, search, and optimization**
  - A need for **adaptation, fine-tuning, or associative learning** in response to arriving information
  - A need for **low latency responses**, e.g. as in closed-loop control applications (batching and vectorization may be unacceptable)
  - **Power constrained**
  - **Relatively small problem scales or else cost insensitive** due to use of a compute/memory-integrated architecture, which makes it costly to scale to large workloads (requiring more processing chips)

Applications

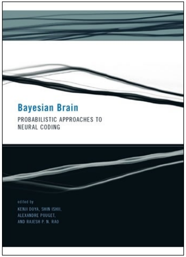
- Application properties necessary for realizing gains on neuromorphic architectures [INRC '21]:
  - **Streaming input data**, e.g. audio, video, or any signals changing on microsecond-to-second time scales
  - A need for **fast pattern matching, search, and optimization**
  - A need for **adaptation, fine-tuning, or associative learning** in response to arriving information
  - A need for **low latency responses**, e.g. as in closed-loop control applications (batching and vectorization may be unacceptable)
  - **Power constrained**
  - **Relatively small problem scales or else cost insensitive** due to use of a compute/memory-integrated architecture, which makes it costly to scale to large workloads (requiring more processing chips)

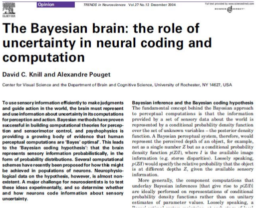
Applications

- Application properties necessary for realizing gains on neuromorphic architectures [INRC '21]:
  - **Streaming input data**, e.g. audio, video, or any signals changing on microsecond-to-second time scales
  - A need for **fast pattern matching, search, and optimization**
  - A need for **adaptation, fine-tuning, or associative learning** in response to arriving information
  - A need for **low latency responses**, e.g. as in closed-loop control applications (batching and vectorization may be unacceptable)
  - **Power constrained**
  - **Relatively small problem scales or else cost insensitive** due to use of a compute/memory-integrated architecture, which makes it costly to scale to large workloads (requiring more processing chips)

Probabilistic Models and Cognition

- Current SNN implementations are deterministic (implementing frequentist learning).
- But probabilistic modelling and Bayesian reasoning are central to dominant theories of cognition...
- which give a central role to the modelling of uncertainty.




Opinion — TRENDS in Neurosciences Vol.27 No.12 December 2004

The Bayesian brain: the role of uncertainty in neural coding and computation

David C. Knill and Alexandre Pouget

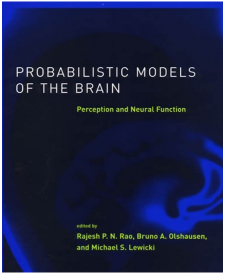
Center for Visual Science and the Department of Brain and Cognitive Science, University of Rochester, NY 14627, USA

To use sensory information efficiently to make judgments and guide action in the world, the brain must represent and use information about uncertainty in its computations for perception and action. Bayesian methods have proven successful in building computational theories for perception and sensorimotor control, and psychophysics is providing a growing body of evidence that human perceptual computations are 'Bayes' optimal'. This leads to the 'Bayesian coding hypothesis': that the brain represents sensory information probabilistically, in the form of probability distributions. Several computational schemes have recently been proposed for how this might be achieved in populations of neurons. Neurophysiological data on the hypothesis, however, is almost non-existent. A major challenge for neuroscientists is to test these ideas experimentally, and so determine whether and how neurons code information about sensory uncertainty.

Bayesian inference and the Bayesian coding hypothesis

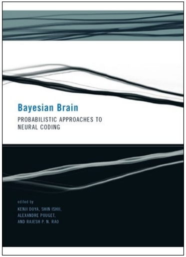
The fundamental concept behind the Bayesian approach to perceptual computations is that the information provided by a set of sensory data about the world is represented by a conditional probability density function over the set of unknown variables – the posterior density function. A Bayesian perceptual system, therefore, would represent the perceived depth of an object, for example, not as a single number Z but as a conditional probability density function $p(Z/I)$, where I is the available image information (e.g. stereo disparities). Loosely speaking, $p(Z/I)$ would specify the relative probability that the object is at different depths Z, given the available sensory information.

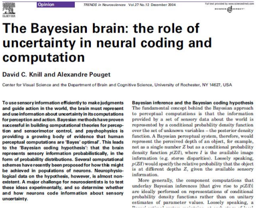
More generally, the component computations that underlay Bayesian inferences [that give rise to $p(Z/I)$] are ideally performed on representations of conditional probability density functions rather than on unitary estimates of parameter values. Loosely speaking, a



Probabilistic Models and Cognition

- Current SNN implementations are deterministic (implementing frequentist learning).
- But probabilistic modelling and Bayesian reasoning are central to dominant theories of cognition...
- which give a central role to the modelling of uncertainty.




Opinion TRENDS in Neurosciences Vol.27 No.12 December 2004

Full text provided by www.sciencedirect.com

The Bayesian brain: the role of uncertainty in neural coding and computation

David C. Knill and Alexandre Pouget

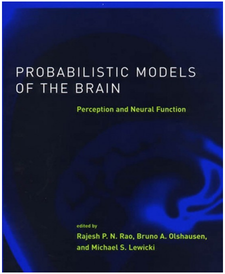
Center for Visual Science and the Department of Brain and Cognitive Science, University of Rochester, NY 14627, USA

To use sensory information efficiently to make judgments and guide action in the world, the brain must represent and use information about uncertainty in its computations for perception and action. Bayesian methods have proven successful in building computational theories for perception and sensorimotor control, and psychophysics is providing a growing body of evidence that human perceptual computations are 'Bayes' optimal'. This leads to the 'Bayesian coding hypothesis': that the brain represents sensory information probabilistically, in the form of probability distributions. Several computational schemes have recently been proposed for how this might be achieved in populations of neurons. Neurophysiological data on the hypothesis, however, is almost non-existent. A major challenge for neuroscientists is to test these ideas experimentally, and so determine whether and how neurons code information about sensory uncertainty.

Bayesian inference and the Bayesian coding hypothesis

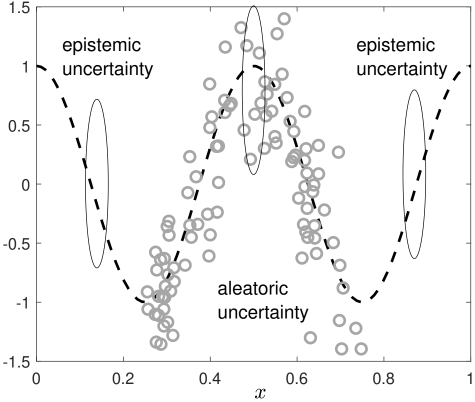
The fundamental concept behind the Bayesian approach to perceptual computations is that the information provided by a set of sensory data about the world is represented by a conditional probability density function over the set of unknown variables – the posterior density function. A Bayesian perceptual system, therefore, would represent the perceived depth of an object, for example, not as a single number Z but as a conditional probability density function $p(Z/I)$, where I is the available image information (e.g. stereo disparities). Loosely speaking, $p(Z/I)$ would specify the relative probability that the object is at different depths Z, given the available sensory information.

More generally, the component computations that underlay Bayesian inferences [that give rise to $p(Z/I)$] are ideally performed on representations of conditional probability density functions rather than on unitary estimates of parameter values. Loosely speaking, a



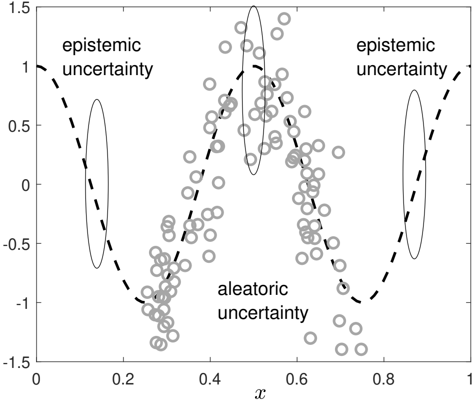
Aleatoric and Epistemic Uncertainty

- There are two types of uncertainty
  - Aleatoric uncertainty, caused by inherent randomness in the data generation mechanism
  - Epistemic uncertainty, caused by lack of data



Aleatoric and Epistemic Uncertainty

- There are two types of uncertainty
  - Aleatoric uncertainty, caused by inherent randomness in the data generation mechanism
  - Epistemic uncertainty, caused by lack of data



Probabilistic Spiking Neural Network Models

- Two complementary frameworks for the design of learning algorithms for SNNs:

  1) Probabilistic spiking neuron models:
     - spikes are generated according to a "stochastic threshold" mechanism
     - can account for aleatoric uncertainty in data generation processes
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the use of principled information-theoretic learning criteria

  2) Probabilistic synaptic models:
     - synaptic weights are randomly generated before inference
     - can account for epistemic uncertainty due to limited availability of data
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the application of Bayesian learning

Probabilistic Spiking Neural Network Models

- Two complementary frameworks for the design of learning algorithms for SNNs:

  1) Probabilistic spiking neuron models:
     - spikes are generated according to a "stochastic threshold" mechanism
     - can account for aleatoric uncertainty in data generation processes
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the use of principled information-theoretic learning criteria

  2) Probabilistic synaptic models:
     - synaptic weights are randomly generated before inference
     - can account for epistemic uncertainty due to limited availability of data
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the application of Bayesian learning

Probabilistic Spiking Neural Network Models

- Two complementary frameworks for the design of learning algorithms for SNNs:

  1) Probabilistic spiking neuron models:
     - spikes are generated according to a "stochastic threshold" mechanism
     - can account for aleatoric uncertainty in data generation processes
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the use of principled information-theoretic learning criteria

  2) Probabilistic synaptic models:
     - synaptic weights are randomly generated before inference
     - can account for epistemic uncertainty due to limited availability of data
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the application of Bayesian learning

Probabilistic Spiking Neural Network Models

- Two complementary frameworks for the design of learning algorithms for SNNs:

  1) Probabilistic spiking neuron models:
     - spikes are generated according to a “stochastic threshold” mechanism
     - can account for aleatoric uncertainty in data generation processes
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the use of principled information-theoretic learning criteria

  2) Probabilistic synaptic models:
     - synaptic weights are randomly generated before inference
     - can account for epistemic uncertainty due to limited availability of data
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the application of Bayesian learning

Probabilistic Spiking Neural Network Models

- Two complementary frameworks for the design of learning algorithms for SNNs:

  1) Probabilistic spiking neuron models:
     - spikes are generated according to a "stochastic threshold" mechanism
     - can account for aleatoric uncertainty in data generation processes
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the use of principled information-theoretic learning criteria

  2) Probabilistic synaptic models:
     - synaptic weights are randomly generated before inference
     - can account for epistemic uncertainty due to limited availability of data
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the application of Bayesian learning

Probabilistic Spiking Neural Network Models

- Two complementary frameworks for the design of learning algorithms for SNNs:

  1) Probabilistic spiking neuron models:
     - spikes are generated according to a “stochastic threshold” mechanism
     - can account for aleatoric uncertainty in data generation processes
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the use of principled information-theoretic learning criteria

  2) Probabilistic synaptic models:
     - synaptic weights are randomly generated before inference
     - can account for epistemic uncertainty due to limited availability of data
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the application of Bayesian learning

Probabilistic Spiking Neural Network Models

- Two complementary frameworks for the design of learning algorithms for SNNs:

  1) Probabilistic spiking neuron models:
     - spikes are generated according to a “stochastic threshold” mechanism
     - can account for aleatoric uncertainty in data generation processes
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the use of principled information-theoretic learning criteria

  2) Probabilistic synaptic models:
     - synaptic weights are randomly generated before inference
     - can account for epistemic uncertainty due to limited availability of data
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the application of Bayesian learning

Probabilistic Spiking Neural Network Models

- Two complementary frameworks for the design of learning algorithms for SNNs:

  1) Probabilistic spiking neuron models:
     - spikes are generated according to a "stochastic threshold" mechanism
     - can account for aleatoric uncertainty in data generation processes
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the use of principled information-theoretic learning criteria

  2) Probabilistic synaptic models:
     - synaptic weights are randomly generated before inference
     - can account for epistemic uncertainty due to limited availability of data
     - can also be useful to model hardware imperfections at the level of neurons
     - enable the application of Bayesian learning

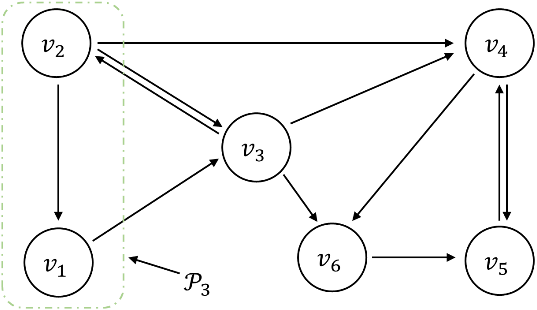
Overview

- Motivation
- Models
- Learning for probabilistic SNNs
- Bayesian learning for SNNs

Models

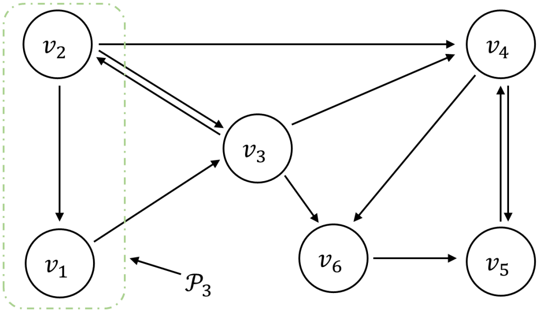
Models

- An SNN is a network $\mathcal{V}$ of spiking neurons.
- Its operation is defined by:
  - connectivity graph
  - neuron model



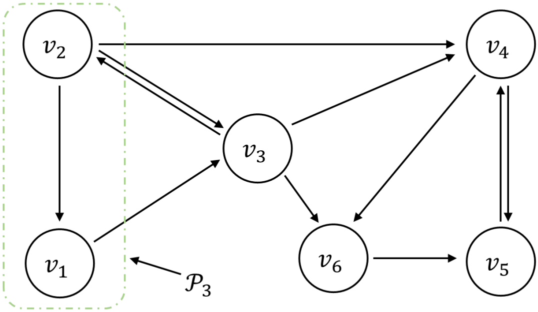
Connectivity Graph

- Arbitrary directed, possibly cyclic, graph with directed links representing synaptic connections.
- "Parent", or pre-synaptic, neuron affects causally spiking behavior of "child", or post-synaptic, neuron.
- Enables recurrent connectivity (directed loops).



Connectivity Graph

- Arbitrary directed, possibly cyclic, graph with directed links representing synaptic connections.
- "Parent", or pre-synaptic, neuron affects causally spiking behavior of "child", or post-synaptic, neuron.
- Enables recurrent connectivity (directed loops).



Neuron Models

- Several spiking neuron models exists.
- They model biological neurons to various degrees of detail:
  - Integrate-and-fire (IF)
  - Leaky integrate-and-fire (LIF)
  - Spike response model (SRM)
  - Resonate-and-Fire
  - ...
- They can be deterministic or probabilistic.

(Deterministic) Spike Response Model

- SRM is a standard deterministic neural model.
- Internal state of a neuron i at time t is represented by membrane potential $u_{i,t}$
- Output of neuron i at time t:

$$s_{i,t} = \Theta\big(u_{i,t} - \vartheta\big) \in \{0, 1\}.$$

  - $\Theta(\cdot)$: Heaviside step function
  - $u_{i,t}$: membrane potential of neuron i at time t
  - ϑ: a fixed threshold
  - A spike $s_{i,t} = 1$ is emitted when the membrane potential $u_{i,t}$ crosses a fixed threshold ϑ, after which the membrane potential is reset

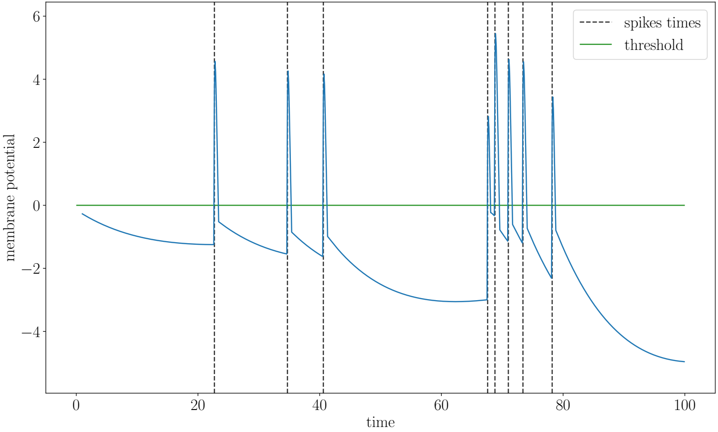
(Deterministic) Spike Response Model

- SRM is a standard deterministic neural model.
- Internal state of a neuron i at time t is represented by membrane potential $u_{i,t}$
- Output of neuron i at time t:

$$s_{i,t} = \Theta\big(u_{i,t} - \vartheta\big) \in \{0, 1\}.$$

  - $\Theta(\cdot)$: Heaviside step function
  - $u_{i,t}$: membrane potential of neuron i at time t
  - ϑ: a fixed threshold
  - A spike $s_{i,t} = 1$ is emitted when the membrane potential $u_{i,t}$ crosses a fixed threshold ϑ, after which the membrane potential is reset

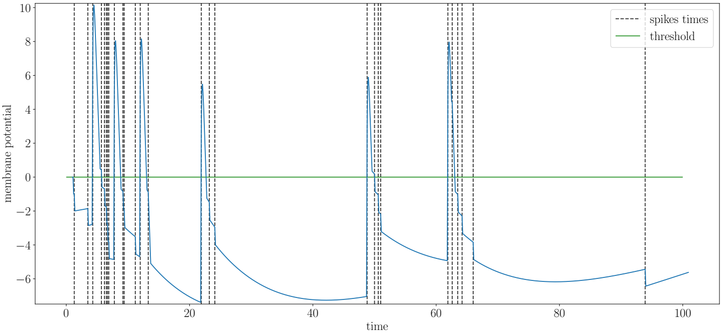
(Deterministic) Spike Response Model



(Probabilistic) SRM with Stochastic Threshold

- SRM with stochastic threshold implement probabilistic spiking neuron models
- Neuron i at time t spikes with probability increasing with the membrane potential $u_{i,t}$

$$s_{i,t} \sim p(s_{i,t} = 1 | u_{i,t}) = \sigma(u_{i,t} - \vartheta) = \frac{1}{1 + \exp\big(-(u_{i,t} - \vartheta)\big)}$$



(Probabilistic) SRM with Stochastic Weights

- SRM with stochastic weights model probabilistic synaptic models
- They follow the standard SRM, with either deterministic or probabilistic neurons, with one caveat:
  - before the presentation of an input, model parameters are generated from a distribution $q(\theta)$, i.e.,

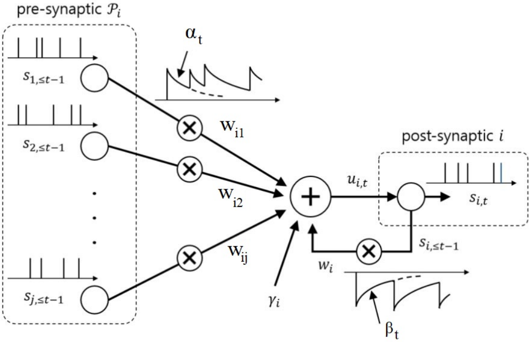
$$\theta \sim q(\theta)$$

Membrane Potential

- For all models, we have:

$$u_{i,t} = \underbrace{\sum_{j \in \mathcal{P}_i} w_{ij} \left(\alpha_t * s_{j,t}\right)}_{\text{pre-synaptic}} + \underbrace{\left(\beta_t * s_{i,t}\right)}_{\text{post-synaptic}} + \gamma_i$$

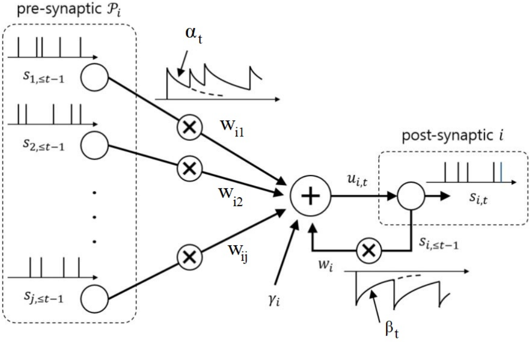
- The contribution of pre-synaptic neurons depends on the synaptic filter α_t with learnable synaptic weights w_{ij}.



Membrane Potential

$$u_{i,t} = \underbrace{\sum_{j \in \mathcal{P}_i} w_{ij} \left(\alpha_t * s_{j,t} \right)}_{\text{pre-synaptic}} + \underbrace{\left(\beta_t * s_{i,t} \right)}_{\text{post-synaptic}} + \gamma_i$$

- The post-synaptic contribution of the neuron depends on the feedback filter β_t.

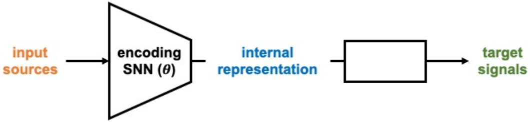


Learning for Probabilistic SNNs

Information-Theoretic Learning

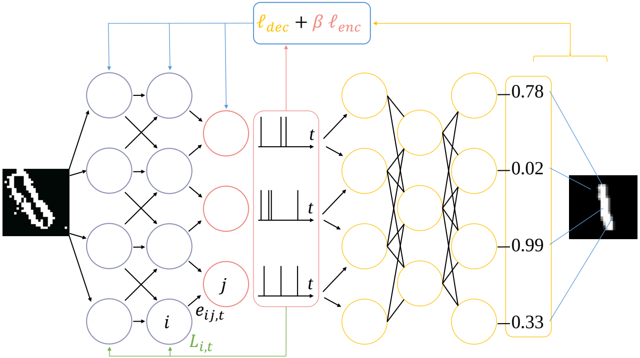
- A general form of the learning criterion for (probabilistic) SRM SNNs with stochastic threshold is given by the Information Bottleneck (IB) problem $\max_\theta \mathcal{L}_{\text{IB}}(\theta)$, with

$$\mathcal{L}_{\text{IB}}(\theta) = MI(\text{target}; \text{repr}|\theta) - \beta \cdot MI(\text{input}; \text{repr}|\theta)$$



  - aims at learning a representation, defined by the output of spiking neurons, that is maximally informative about target signals,
  - while being maximally compressive about input sources
  - If $\beta = 0$, supervised learning (maximum likelihood, ML)

Information-Theoretic Learning



- One can also mix SNNs and ANNs

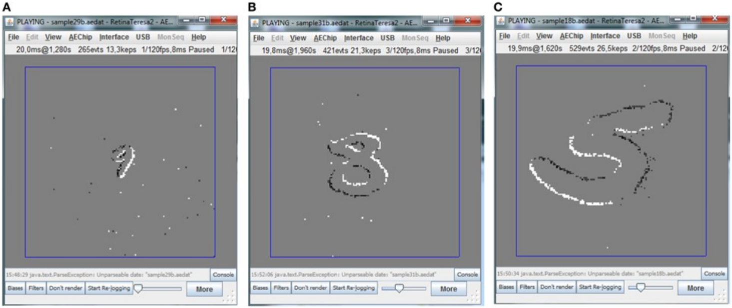
Information-Theoretic Learning

- Gradient-based optimization yields local learning rules with global feedback (no backprop):

$$\theta_{j,i} \leftarrow \theta_{j,i} - \eta \cdot \Big(\big(\text{error}\big) \cdot \big(\text{post-synaptic error}_i\big) \cdot \big(\text{pre-synaptic trace}_j\big) \Big)$$

Application 1

- Neuromorphic dataset obtained by filming moving MNIST digits displayed on a screen with a neuromorphic camera.



[Serrano-Gotarredona and Linares-Barranco, 2015]

Application 1

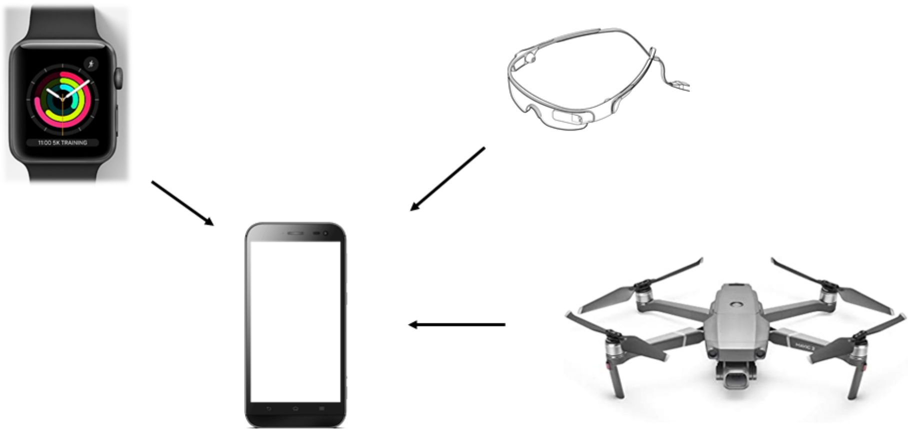
- DECOLLE: surrogate gradient method with local pseudo-targets [Kaiser et al. '21].
  - convolutional or layered architectures
- SRM with stochastic threshold
  - fully connected architecture
- Rate decoding
- SNNs are equipped with $N = N_H + N_V$ neurons, where the number N_V of output neurons is equal to the number of classes.

Application 1

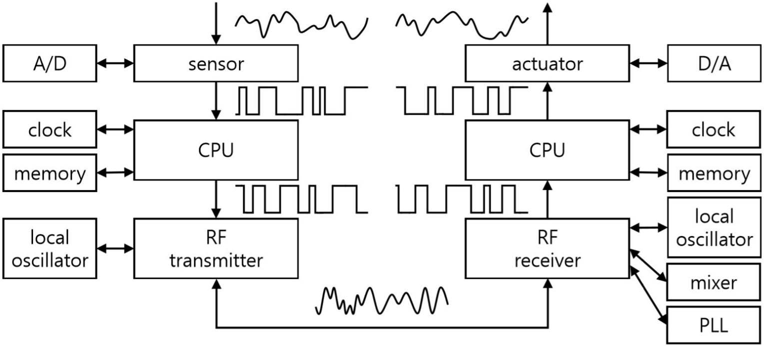
MODEL	PERIOD	N_H	INPUT	ACC.
DECOLLE	1 MS	39,232	PER-SIGN	99.4%
	1 MS	19,616	BINARY	98.9%
	1 MS	512	PER-SIGN	86.8%
	10 MS	256	PER-SIGN	73.8%
	25 MS	256	PER-SIGN	65.8%
GLM-SNN	25 MS	512	PER-SIGN	83.50%
	25 MS	512	BINARY	80.80%
	25 MS	256	PER-SIGN	82.80%
	25 MS	256	BINARY	79.3%

- SRM with stochastic threshold, also known as generalized linear model (GLM), are more robust to coarser sampling rates smaller topologies.
- Probabilistic models are better suited to capture aleatoric uncertainty.

Application 2: Remote Sensing and Inference



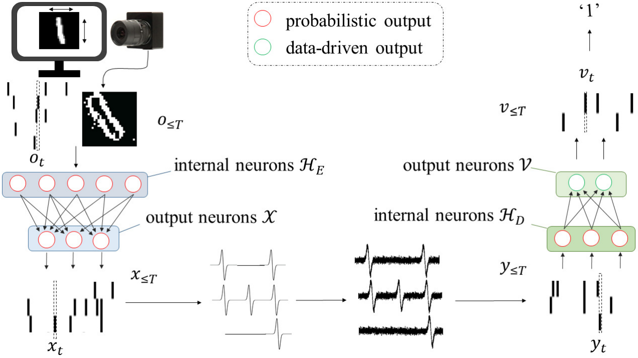
Conventional Solution



- Digital sensing, computing, and communications:
  - High energy consumption for always-on operation
  - Latency caused by frame-based transmission

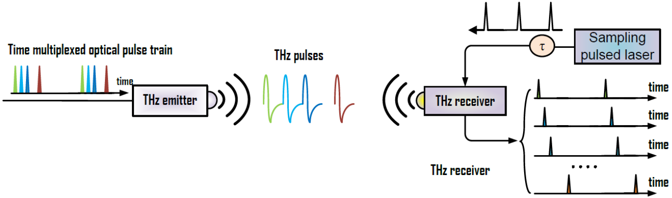
Neuromorphic Joint Source-Channel Coding (NeuroJSCC)

- NeuroJSCC replaces:
  - digital sensing with neuromorphic sensing
  - digital processors with neuromorphic processors
  - digital communications with impulse radio
- Low energy consumption and low latency.



Neuromorphic Joint Source-Channel Coding (NeuroJSCC)

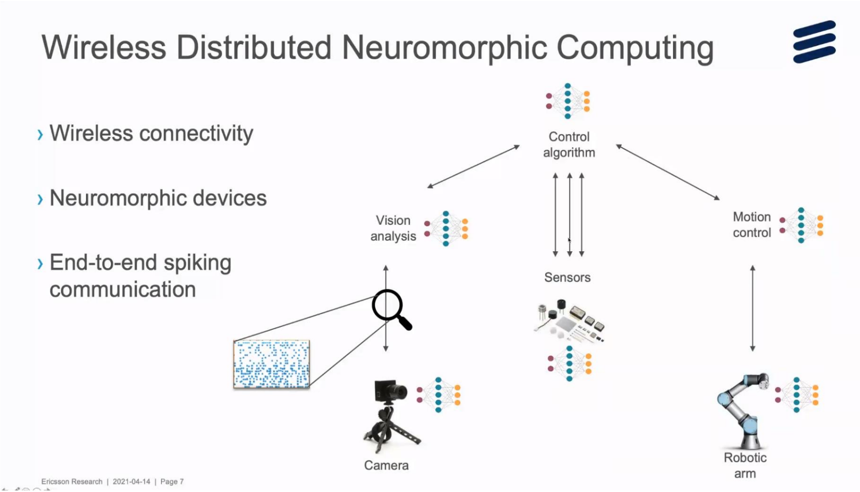
- Impulse radio communicates with baseband pulses of very short duration.
- Candidate for beyond-5G systems in the Terahertz range
- Used for extremely low-power transmission in the IEEE 802.15.4z standard



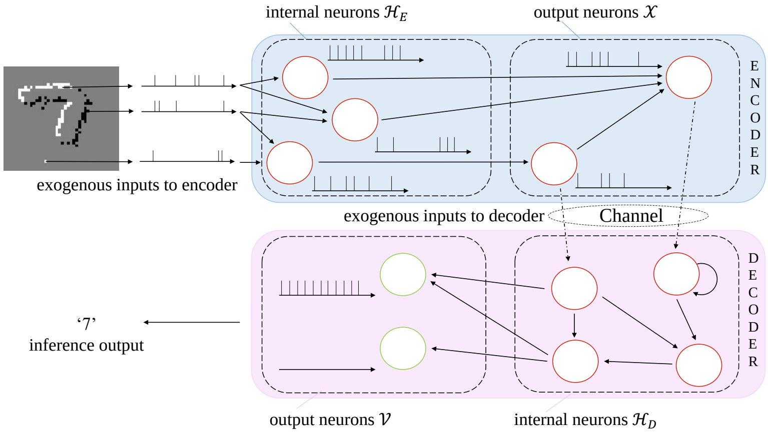
[Yu et al '15]

Neuromorphic Joint Source-Channel Coding (NeuroJSCC)

- Recently, the idea is being investigated in the industry too...



Encoding and Decoding SNNs

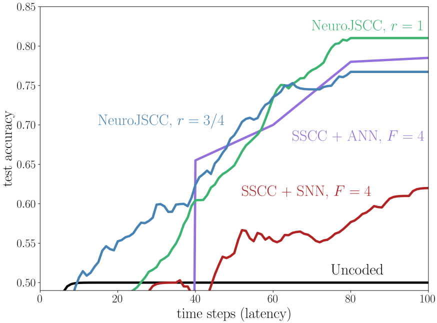


Results

- AWGN transmission on parallel channels
- Benchmarks:
  - Uncoded transmission: On-Off Shift Keying + hard demodulation + SNN classifier trained on noisy signals (i.e., remove encoding SNN)
  - Frame-based separate source-channel coding (SSCC): State of the art VQ-VAE [Van den Oord, 2017] with compression rate 2 + LDPC encoding (rate 1/2) + hard demodulation + LDPC decoding + VQ-VAE decompression + ANN/ SNN classifier.

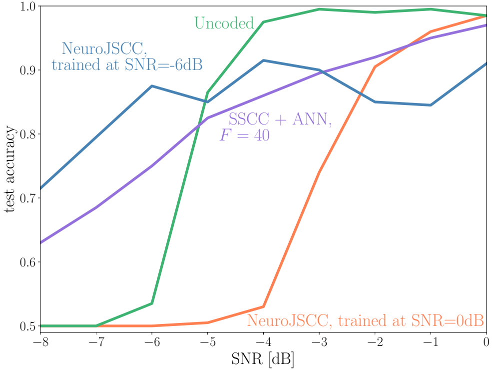
Results

- SNR $= -8$ dB (average per-symbol signal power over noise)



- NeuroJSCC and Uncoded have zero latency, while SSCC has to form and process frames.
- NeuroJSCC exhibits a graceful trade-off between the number of processed samples and the classification performance.

Results



- NeuroJSCC maintains a test accuracy of 80%, even at an SNR level as low as -8 dB.

Bayesian Learning for SNNs

Bayesian Learning for SNNs

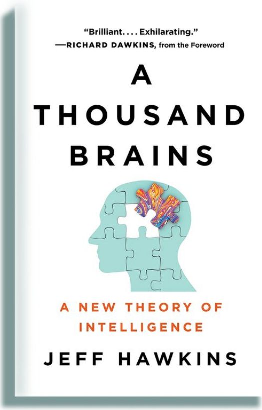
- SRM with stochastic threshold introduces randomness in the spiking mechanism...
- ... which allows to better capture aleatoric uncertainty.
- Bayesian models introduce randomness at the level of weights...
- which captures epistemic uncertainty due to limited data...
- and enables the combination of models specialized to different parts of the problem space.

Bayesian Learning for SNNs

- SRM with stochastic threshold introduces randomness in the spiking mechanism...
- ... which allows to better capture aleatoric uncertainty.
- Bayesian models introduce randomness at the level of weights...
- which captures epistemic uncertainty due to limited data...
- and enables the combination of models specialized to different parts of the problem space.

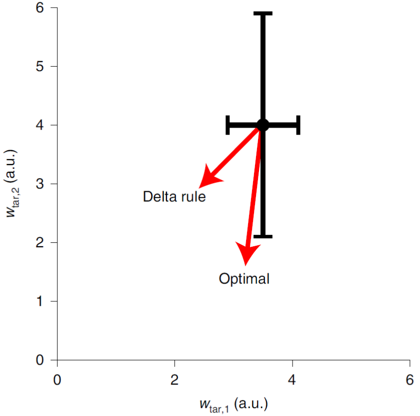
Bayesian Learning for SNNs

- Bayesian learning optimizes a distribution $q(\theta)$ over the SNN weights.
- During inference, weights are randomly sampled from $q(\theta)$, and the final prediction may be averaged over multiple models.



Bayesian Learning for SNNs

- During training, accounting for "error bars" in the model parameter space can improve accuracy by guiding the update process.



[Aitchinson, 2021]

Bayesian Learning for SNNs

- The problem amounts to minimizing the free energy: $\min_{q(\theta)} \mathcal{F}(\theta)$, with

$$\mathcal{F}(\theta) := \underbrace{E_{q(\theta)}\Big[\text{log-loss of the SNN } (\theta)\Big]}_{\text{fitting the training data}} + \rho \cdot \underbrace{\mathrm{KL}\Big[q(\theta)||\text{prior}(\theta)\Big]}_{\text{regularizing penalty}}$$

  - The KL term accounts for epistemic uncertainty due to the presence of limited data
  - ρ: temperature parameter

Bayesian Learning for SNNs

- Optimization via stochastic natural gradient descent results in an update that follows again a three-factor rule:

$$\theta^{\mathrm{r}}_{j,i} \leftarrow (1-\eta\rho)\cdot\theta^{\mathrm{r}}_{j,i} - \eta\cdot\Big(\big(\mathrm{error}_{j,i}\big)\cdot \big(\text{post-synaptic sensitivity}_i\big)\cdot\big(\text{pre-synaptic trace}_j\big) - \rho\cdot\theta^{\mathrm{r}}_0\Big)$$

- Unlike the frequentist rules seen above, the error term is specific to each synapse and it grows with the uncertainty concerning the corresponding weights (natural gradient).

Bayesian Learning for SNNs

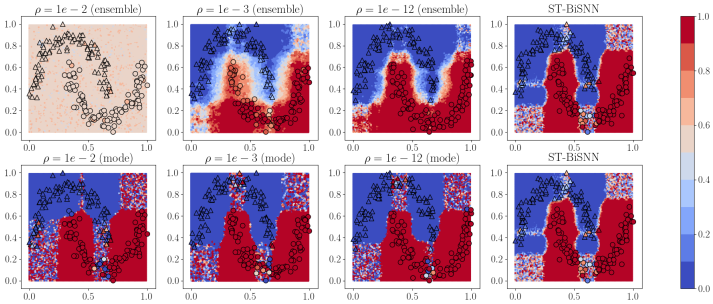
- Optimization via stochastic natural gradient descent results in an update that follows again a three-factor rule:

$$\theta^{\mathrm{r}}_{j,i} \leftarrow (1-\eta\rho)\cdot\theta^{\mathrm{r}}_{j,i} - \eta\cdot\Big(\big(\mathrm{error}_{j,i}\big)\cdot \big(\text{post-synaptic sensitivity}_i\big)\cdot\big(\text{pre-synaptic trace}_j\big) - \rho\cdot\theta^{\mathrm{r}}_0\Big)$$

- Unlike the frequentist rules seen above, the error term is specific to each synapse and it grows with the uncertainty concerning the corresponding weights (natural gradient).

Bayesian Learning for SNNs

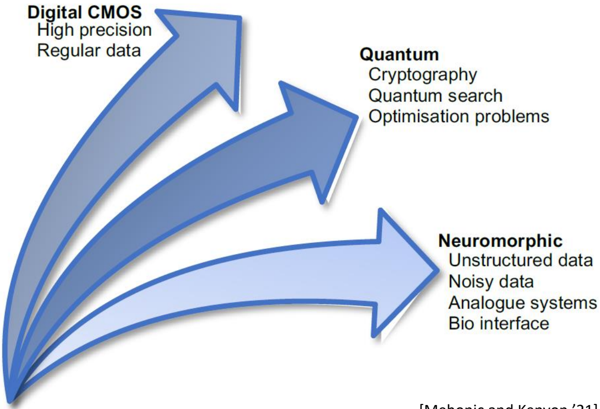
- Bayesian learning captures epistemic uncertainty, while maintaining competitive performance as compared to SNNs with full-precision weights.



Conclusions

- Neuromorphic computing aims at harnessing the efficiency of biological brains by using a more realistic abstraction for the neurons via SNNs.
- Potential energy and latency gains when implemented on specialized hardware
- Training for deterministic and probabilistic SNN models can be done via different, but related, three factor training rules (offline or on-chip).
- Applications of neuromorphic computing to communication systems may be found for battery-powered remote inference and learning applications.

Conclusions



[Mehonic and Kenyon '21]

Acknowledgements

This work has been supported by Intel via the Intel Neuromorphic Research Community (INRC) and by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 725731)