Neuromorphic Algorithm Assessment

The purpose of this assessment is to determine (a) if the proposed neuromorphic algorithm is sufficiently well defined with properties that match well to a neuromorphic hardware implementation, (b) the general usefulness of the proposed algorithm, (c) if current Loihi silicon and software can support the algorithm, (d) how the algorithm will be assessed relative to state-of-the-art alternatives.

Algorithm definition and requirements

|  |  |  |
| --- | --- | --- |
| WHAT | What computational problem does the proposed algorithm solve? | *Ideally, this is a clear mathematical objective.*  |
|  |
| What learning paradigms are involved, if any? | *E.g. Supervised (online or offline?), self-supervised, unsupervised, reinforcement-based, associative, gradient-based adaptation, continual, etc.* |
|  |
| What are the algorithm’s essential “neuromorphic” properties? | *E.g. Temporal neuron models, binary activations, sparse spike/event-based communication, sparse connectivity, recurrence, E/I balance, parameter plasticity, structural plasticity, fully local synaptic & neural information processing, distributed data representations.* |
|  |
| Are data input/output interfaces and encodings well defined? | *Does the algorithm operate on conventional data types, or spiking/event-based data types? Does it process time series data streams with temporal structure (e.g. video), or isolated, uncorrelated samples (e.g. images)?* |
|  |
| How far towards a deployable neuromorphic solution will the proposed algorithmic research be taken? | *Is the aim to develop software that executes the algorithm on Loihi to process real-world data? Or is the goal a simulation-based demonstration believed to be compatible with Loihi HW? Something in between?* |
|  |
| WHY | How broadly applicable is the proposed algorithm? | *What real-world capabilities, applications, and technologies could this algorithm be used in?* |
|  |
| Characterize the difficulty of the problem to be solved. | *E.g. NP-complete/hard, existing state-of-the-art DNN network scale, typical Energy-Delay-Product application constraints, typical CPU runtime, etc.*  |
|  |
| What value does a neuromorphic solution promise? | *Does the algorithm primarily improve the energy, speed, or data efficiency of existing algorithms? Or is there no known conventional solution?* |
|  |
| Is the algorithm modular and composable? | *Can the algorithm be integrated into a larger application where the whole is greater than the sum of its parts?*  |
|  |
| REQUIREMENTS | How mature is the proposed algorithmic approach? | *Are the key algorithmic or implementation ingredients understood or is it part of this proposal to develop such an understanding? What are the key open questions and risks? Do conventional ANN implementations exist?* |
|  |
| At what problem scale is the algorithm expected to be demonstrated? | *What scale will be demonstrated relative to the scale demanded by useful and impactful real-world applications? (In terms of physical metrics like I/O dimensionality, #parameters, #neurons, dataset size, stored patterns/classes, etc.)*  |
|  |
| What requirements does the algorithm impose on Lava SW infrastructure or other algorithms? | *What infrastructure is expected to exist (or when will it need to exist)? Are there critical requirements on input/output bandwidth?* |
|  |
| What can be said about the specific features and numeric precision required of the neuromorphic hardware? | *Note: Loihi’s synaptic variables provide up to 1B, while neural variables offer 1B, 2B or 3B of (un)signed integer precision.* |
|  |
| Does the algorithm require on-chip synaptic plasticity? | *If so, are the learning rules supported by Loihi’s micro-code programmable learning engine, if known?*  |
|  |
| Does the algorithm currently depend on features not supported by Loihi 2? | *(If known) Examples: division or transcendental functions applied to neuron/synaptic state changes, non-local weight normalization or transport.* |
|  |

Proposed approach and evaluation methodology

|  |  |  |
| --- | --- | --- |
| HOW | What methodology is being followed to develop this algorithm? | *Is the proposed algorithm from the category of mathematically derived algorithms or directly inspired by neuroscience modeling?* |
|  |
| What software tools will be used to develop the algorithm? | *E.g. Lava, Brian, Nengo, Matlab, directly coded Python/C++, TensorFlow, PyTorch, SLAYER, SpyTorch, Fugu, etc.* |
|  |
| How is the algorithm or initial neural network configured, parameterized, or trained? | *Does it rely on pre-training with back-propagation, manual parameter tuning, evolutionary methods or is the network configuration computed analytically?* |
|  |
| Does the algorithm involve continual online learning? If so, how? | *Is new knowledge absorbed into existing resources or reliant on allocating new memory resources over time? How is forgetting mitigated? Does learning rely on assumptions of IID input data?* |
|  |
| What neuron model(s) will be used? | *E.g. LIF, ALIF, CUBA, COBA, ReLU, Resonate-and-fire, Izhikevich, GLM, etc.* |
|  |
| How is data coded in the network? | *E.g. spike-based or integer-valued events, temporal coding, rate coding, population coding, mixtures thereof.* |
|  |
| COMPETITION | How will the performance of the algorithm be evaluated? | *What are critical performance metrics? Will standardized benchmarks, datasets, simulation environments, etc. be used? If so, please list them. If not, how will performance be quantified in a replicatable manner?* |
|  |
| What competitive state-of-the-art *conventional* and *neuromorphic* solutions exist today, if any, and what HW platforms do they run on? | *Which of the current solutions are state-of-the-art with respect to the metrics/benchmarks above? Does the proposer have access to a working reference implementation of any of these competitive algorithms?* |
|  |
| What are the limitations or major pain points of current solutions? | *In what way are current solutions inadequate? Are they failing to meet real-world application needs in terms of key metrics?* |
|  |

Note: A neuromorphic algorithm solves a new, specific and well-defined computational problem by exploiting features of the underlying neuromorphic hardware system following established mathematical principles in machine learning, optimization, etc. or might be inspired by neuroscientific, bio-inspired modeling. As an algorithmic project, there might still be open theoretical questions or how this algorithm can be supported by neuromorphic hardware, yet the value for overcoming those risks should be clearly spelled out and significant. If the theoretical open questions are too broad, the research best belongs in the Theory vector (RV1).

In contrast, applications mostly build on top of previously coded algorithms that each have little execution risk. Application projects draw their novelty and impact from the composition of algorithms deployed and the value of real-world problem solved.