ECE 523 Final Project

David Schwartz
Department of electrical and computer engineering
University of Arizona
Tucson, AZ
dmschwar@email.arizona.edu

Zhengzhong Liang
Department of electrical and computer engineering
University of Arizona
Tucson, AZ
zhengzhongliang@email.arizona.edu

Abstract

We proposed for this machine learning project the development of a classifier of a time series of discrete events implemented as a spiking neural network. Here, we discuss the development and implementation of dynamics of spiking neurons which learn to classify patterns of spike times presented to their afferent connections, closelly following the abstract model described in [1]. We present and discuss the results of numerical simulations in which we test the accuracy of various spiking neural classifiers, and compare to more pedestrian schemes. Our conclusions produce more questions than answers, and so we propose further investigation of phenomena studied here, as well as an exploration of extensions of this model that may lead to a better informed understanding of learning and information processing in spiking neural networks.

1 Introduction

We proposed for this machine learning project the development of a classifier of a time series of discrete events implemented as a spiking neural network. Here, we discuss the development and implementation of dynamics of spiking neurons which learn to classify patterns of spike times presented to their afferent connections, closelly following the abstract model described in [1]. We first demontrate that a network consisting of dynamically similar neurons, whose excitatory synapses are subject to classical spike timing dependent plasticity (STDP) as observed to occur in spiking neural networks (collected from hippocampal slices) in vitro [2], acheives poor classification performance compared to that of more traditional classification methods (namely linear regression and a feed-forward multi-layer perceptron). In order to improve stability of learning and classification performance, we quantify a performance improvement obtained from a modification to this first formulation of STDP, in which we model a homeostatic drive towards equilibrium as normalization of synaptic strengths. We explore theoretically the qualitative effects of this alteration and confirm the predicted behavior with numerical simulations.
This paper is organized as follows: In section 2, we precisely map out the theoretical framework supporting our model, experiments, and results. The following sections present the promised observations and discuss these results. In section 4, we propose a more complete investigation of phenomena observed in this study, as well as an exploration of extensions of this model that may lead to a better informed understanding of learning and information processing in spiking neural networks.

ECE 523: Machine learning - final project, University of Arizona, 2017, Tucson, Arizona.

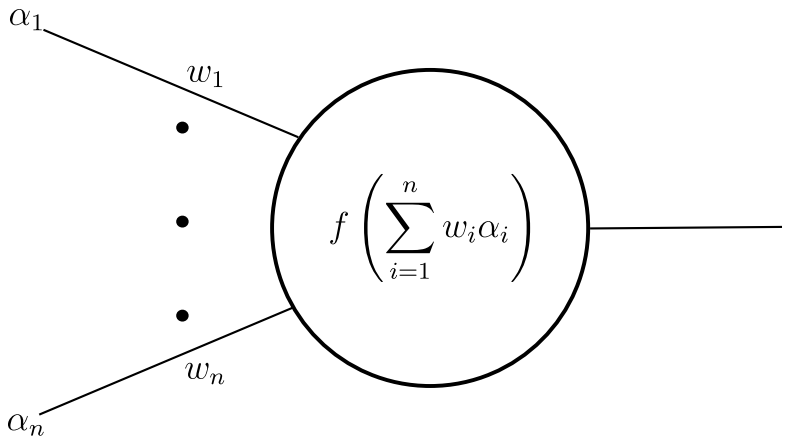

Figure 1: Shown here is an artificial neuron, with activation function, f. w_i represents the strength of this neuron's i^{th} input

2 Theoretical framework

2.1 Traditional time series classification

As a first benchmark for classification performance, we implement softmax regression in the tensor flow framework. This naive approach is capable of acheiving acceptable performance when training data sets are sufficiently large, and the data to classify are linearly separable (i.e. there exists some hyperplane that separates the data).

2.2 Artificial neural networks

It has been demonstrated that an appropriately structured network of artificial neurons, such as those shown in 1, whose computation accumulates the sum of inputs α_i, weighted by the corresponding connection strength, $_i$, and passed through a (usually nonlinear) activation function, f, can develop a set of weights such that the output layer computes a bayes optimal discriminant function to classify data pressented at the input layer [3]. That is to say, the output layer approximates posterior probabilities of the classes on which it is trained. One may then read the output layer and select the class corresponding to the most confident neuron. Any MLP will consist of an input layer, a collection of hidden layers, and an output layer. The input layer will consist of a single neuron for each feature of the data to classfy. The output layer will have a single neuron for each class. In general, hidden layers may be connected in any manner, with any number of neurons. We demonstrated in homework 3 that with an appropriate learning algorithm coupled to a properly weighted regularizer, as long as the training data-set is sufficiently large, such an MLP can acheive excellent performance (i.e. accuracy from 5-fold cross validation exceeds 0.99 on 10-ary classification of MNIST digits).As a benchmark for classification performance, we consider a feed-forward MLP with a single hidden layer, implemented in tensor flow. Our artificial neurons are equipped with rectified linear activation functions. This MLP is trained using tensorflow's Adam optimizer, to which we've wed an L2 regularizer.

2.3 Spiking neurons

$$\begin{cases} \frac{\partial v}{\partial t} & = \frac{(V_{\text{rest}} - v + E)}{\tau_m} \\ v & = V_{\text{reset}}, \text{ if } v \geq V_t \end{cases} \tag{1}$$

We construct a spiking neural classifier from leaky integrate and fire (LIF) units. This neural model coarsely mimics properties of realistic neurons while maintaining reasonable computational complexity. We prescribe the dynamics of this model as governed by equation 1. Here, v is membrane

potential, V_{rest} is the resting potential of neuron. E is the post-synaptic potential evoked by a pre-synaptic spike. That is to say, E is the increase in membrane potential produced by an input spike. τ_m is the membran potential time constant. V_t is the neuron's spiking threshold. When v exceeds V_t, the neuron emits a spike, and its membrane potential resets to V_{reset}. A neuron's membrane potential settles to V at equilibrium (for example, when it receives no pre-synaptic spikes).

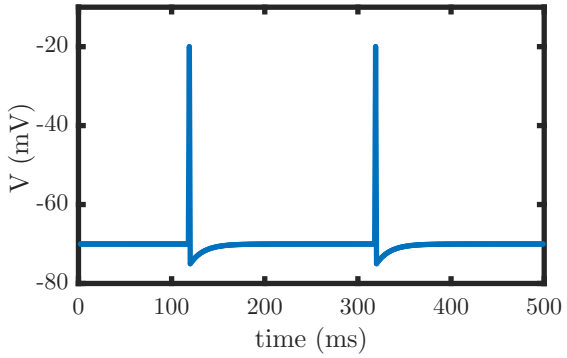

Figure 2: Depicted above is a trace of membrane potential (shown as a function of time) of a typical spiking neuron engaging in two spiking events

Important properties of membrane potential dynamics emerge from equation 1. First, observe that a neuron's spiking is driven by its membrane potential. Primarily, this can be stimulated by increasing E, an effect induced by reception of input spikes. Additionally, note that choice of the parameter, τ_m, determines a neuron's excitability. If τ_m is large, then the neuron tends to be reluctant to vary its membrane potential. Conversely, when τ_m is too small, even very small perturbations in E can produce spikes. Figure 2 demonstrates the desired spiking behavior emerges from our presribed neuronal dynamics.

2.4 Learning and STDP

$$\begin{cases} E_j & = E_j + \alpha \sum_{i=1}^{I} w_{i,j} s_i, \quad \text{if a pre-synaptic spike is received} \\ \frac{\partial E_j}{\partial t} & = -\frac{E_j}{\tau_n}, \qquad \text{otherwise} \end{cases} \tag{2}$$

Equation 2 describes our dynamical model of synaptic transmission (i.e. the effect on post-synaptic potential induced by incoming spikes). i (resp. j) indexes pre(resp. post)-synaptic neurons. E_j is the increase in post-synaptic potential evoked by the spike in question. I is the number of pre-synaptic neurons. $w_{i,j}$ is the strength of the connection from neuron i to neuron j. s_i is an indicator function taking the value 1 when the pre-synaptic neuron spikes. α, a constant in our model, is a unitless quantity, included to incorporate the effect of synaptic resistance/conductance. If a post-synaptic spike occurs in the absence of presynaptic spikes, E decays exponentially.

2.5 Classical STDP

$$\Delta w = \begin{cases} \text{A}_{\text{pre}} \cdot \exp(-\frac{t_{\text{post}} - t_{\text{pre}}}{\tau_s}), & t_{\text{post}} > t_{\text{pre}} \\ \text{A}_{\text{post}} \cdot \exp(-\frac{t_{\text{pre}} - t_{\text{post}}}{\tau_s}), & t_{\text{post}} < t_{\text{pre}} \end{cases} \tag{3}$$

Equation 3 shows a classical STDP weight update rule, demonstrated in [2] to govern variation of synaptic weights as a function of relative spike times in vitro. t_{pre} is the time of the most recent pre-synaptic spike, and t_{post}, the time of the most recent post-synaptic spike. A_{pre} and A_{post} determine learning rate. We choose $\text{A}_{\text{pre}} > 0$ and $\text{A}_{\text{post}} < 0$ so that $w_{i,j}$ strengthens (and $w_{j,i}$ weakens) when

neuron j spikes after neuron i. Notably, the change in synaptic strength is maximized when the time beween pre and post synaptic spikes is minimized.

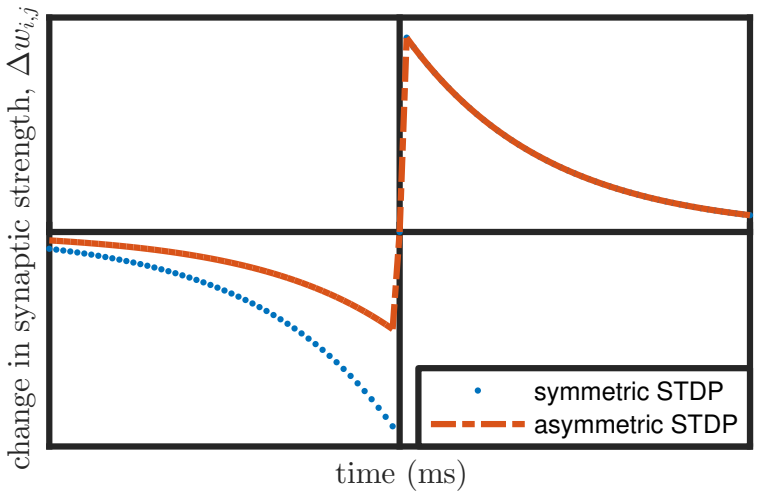

Figure 3: Typical STDP windows (i.e. graphical depictions of $\Delta w_{i,j}$); a symmetric window, in which $\tau_{\text{pre}} = \tau_{\text{post}}$ and an asymmetric window, in which $\tau_{\text{pre}} \neq \tau_{\text{post}}$

Figure 3 graphically depicts this change in synaptic efficacy as a function of the time between the relevant pre-synaptic and post-synaptic spikes. In the symmetric case, $\tau_{\text{pre}} = \tau_{\text{post}}$, which produces potentiation of $w_{i,j}$, equal in magnitude to depression experienced by $w_{j,i}$, assuming both connections exist. Any asymmetric window weights potentiation and depression unequally. Our implentation of classical STDP assumes a symmetric window.

2.6 Competitive STDP

$$\begin{cases} \Delta w_{i,j} & = \text{A}_{\text{pre}} \cdot \exp(\frac{t_i - t_j}{\tau_s}) \\ \Delta w_{i,k} & = -w_{i,k} \cdot \frac{\Delta w_{i,j}}{w_{\text{cons}} - w_{i,j}} \end{cases}$$

We extend classical STDP with a variant of competition at the outputs, such that when the synapse, $w_{i,j}$ is strengthend, the other connections from neuron i to neuron k are weakened. We implement this competitivity via normalization of output synapses to the summed strengths of outputs of each input neuron at each synpatic update operation. To accomplish this, we impose a constraint w_{cons}, which bounds from above strength of connections departing a neuron. Traditionally, as discussed in [4], synaptic competition is considered as implemented by normalization of weights. Our implementation of competitive spike time based learning differs from examples discussed in [4, 5] in that theirs implement competition as normalization of synpatic strengths to the summed strengths of common inputs (i.e. they consider competition among synapses projecting to a common post-synaptic neuron) while we consider competition among synapses originating from a common pre-synaptic neuron. Our approach follows from an intuitionistic argument: A neuron projecting synapses is burdened by physics with a a strict upper bound on the energy it may expend on communicating a spike to its post-synaptic neighbors. Additionally, physics limits a neuron's neurotransmitter[1] budget. It follows that if the neuron is driven invest more energy in a particular channel, it must divest of others.
Our competitive learning rule has 3 important features. Firstly, it imposes an upper bound on the sum of efficacies of synapses departing a neuron. Secondly, this learning rule allows this sum to increase slowly and more stably. And finally, the learning rule ramps up competitivity (i.e. increases the impact of this normalization) as strength approaches a hypothetical maximum, w_{cons}. For precise

[1]Neurotransmitters are molecules released at a synapse, and communicated to dendrites of post-synaptic neurons via diffusion across a gap[6].

choices of this and other parameters, refer to section 5.2. While our implementation of classical STDP assumes a symmetric window, our competitive spiking classifier assumes an assymetric window that is qualitatively similar to the one shown in figure 3.
Consider equation 4 and assume that neuron i emits a spike shortly before j. In response, synapse $w_{i,j}$ is strengthened, and all other synapses $w_{i,k}$ are weakened. We know that

$$\sum_{k=1,k\neq j}^{K} \Delta w_{i,k} = \sum_{k=1,k\neq j}^{K} (-\frac{w_{i,k}\Delta w_{i,j}}{w_{\text{cons}} - w_{i,j}}), \tag{4}$$

where K is the number of synapses projected by the neuron in question. Now, if the sum of outgoing synaptic efficacies, $\Psi = w_{i,j} + \sum_{k=1,k\neq j}^{K} w_{i,k}$ hits w_{cons}, we have

$$w_{\text{cons}} = w_{i,j} + \sum_{k=1,k\neq j}^{K} w_{i,k} \tag{5}$$

Combining equation 4 and 5 we have:

$$\sum_{k=1,k\neq j}^{K} \Delta w_{i,k} = \sum_{k=1,k\neq j}^{K} (-\frac{w_{i,k}\Delta w_{i,j}}{\sum_{k=1,k\neq j}^{K} w_{i,k}}) \tag{6}$$

$$= -\Delta w_{i,j} \tag{7}$$

Equation 7 reveals that when Ψ reaches w_{cons}, competition should impose depression in magnitude equal to that of the potentiation induced by the pair of spikes of neurons i and j. This should drive the network towards equilibrium and prevents epilleptic destabilization that results from run-away potentiation.
If Ψ remains much smaller than w_{cons}, we have

$$w_{\text{cons}} = B + w_{i,j} + \sum_{k=1,k\neq j}^{K} w_{i,k}, \tag{8}$$

where B is a variable defining competitivity equal to the difference between w_{cons} and the quantity of synaptic efficacy already invested after the potentiation induced by the most recent pair of spikes. Thus, $\frac{1}{B}$ can be thought of as a measure of competitivity. Combining equation 8 and 4 we have

$$\sum_{k=1,k\neq j}^{K} \Delta w_{i,k} = \sum_{k=1,k\neq j}^{K} (-\frac{w_{i,k}\Delta w_{i,j}}{B + \sum_{k=1,k\neq j}^{K} w_{i,k}}) \tag{9}$$

$$= -\Delta w_{i,j}(\frac{\sum_{k=1,k\neq j}^{K} w_{i,k}}{B + \sum_{k=1,k\neq j}^{K} w_{i,k}}) \tag{10}$$

When the neuron has lots of room in its budget of energy and neurotransmitter resources, B is relatively large, thus the decrease, $\Delta w_{i,k}$ is small. In this case, competition is mild. On the other hand, when the Ψ nears w_{cons}, B is very close to zero. Equation 10 demonstrates that in this case, total synaptic depression (i.e. depression summed over all outgoing synapses, $w_{i,k}$ where $k \neq i$) is equal to potentiation, $w_{i,j}$.

2.7 Spiking classifiers

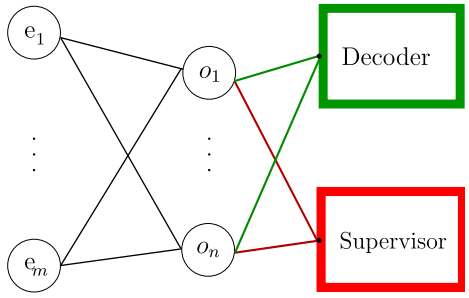

Figure 4: Architecture of the proposed spiking neural classifier

Following the example of [1] we implement a network of leaky integrate and fire (LIF) neurons with plastic synapses with the dynamics described in 2.3 to construct a bilayer feedforward spiking neural network. The structure of this network is illustrated in figure 4. The input layer consists of edge detection units, e_i. These project to the output layer, which consists of tempotrons, synapses subject to the plasticity dynamics described in the previous section. This output layer is trained in a supervised learning paradigm by a teaching signal realized as the induction of a pre-determined number of spikes induced in the target tempotron. This learning process, described in algorithm 1, stimulates the edge detectors with visual information encoded in spike times, and near the end of these time series, induces spikes in the target tempotron. Weights are updated during each learning iteration by evolving states of each component of the collective dynamical system formed by the network for a duration long enough to allow the network to settle to an equilibrium state at the conclusion of each evolution of network activity. This numerical computation of the trajectory of the dynamic network activity is the essential functionality performed upon executing the 'evolve' function called in line 3 of algorithm 1.

Algorithm 1 Tempotron learning

Input: A list of time series, X_{train}; A list of targets, Y_{train}; A training threshold, ξ
Output: A trained network of tempotrons, T

1: **for** time series, $X_t \in X_{\text{train}}$, $Y_t \in Y_{\text{train}}$ **do**
2: **while** $s_{\text{target}} < \xi$ **do**
3: $T \leftarrow \text{evolve}(T, X_t, Y_t)$
4: $s_{\text{target}} \leftarrow$ number of spikes emmitted by target tempotron
5: **end while**
6: **end for**

Algorithm 2 implements a readout scheme or decoder, which converts spike counts, recorded from the tempotrons during stimulation by the encoded test pattern in question. We use a naive majority voting process in that the classification, $\hat{Y}$ corresponds to the class on which the most confident (i.e. most active) neuron was positively trained.

Algorithm 2 Tempotron decoding

Input: Vector of spike counts emitted by each tempotron in the network, $\mathbf{n}$
Output: Classification, $\hat{Y}$

1:

$$\hat{Y} = \underset{i \in \{1,\ldots,C\}}{\text{argmax}}\ n_i$$

2.8 Encoding

Any natural (i.e. biologically implemented) spiking neural classifier - and especially one receptive to visual information - should take advantage of the efficient coding employed by the mammalian brian. For example, humans typically have ≈ 4.6 million cone cells and ≈ 92 million rod cells, for a total of ≈ 96.6 million photoreceptors in each eye [7]. The output of the human eye typically has between 0.71 and 1.54 million retinal ganglion cells, though this is highly variable across eyes surveyed [8]. This means that there is an encoding process that reduces the dimensionality of the visual data by between 8 and 9 orders of magnitude, before any neurons located in the brain perceive the visual signal. Visual information flows from retinal ganglion cells to V1, the mammalian primary visual cortex. V1 preprocesses the visual information for higher layers of processing by performing edge detection (and probably several other computations) [9, 10].

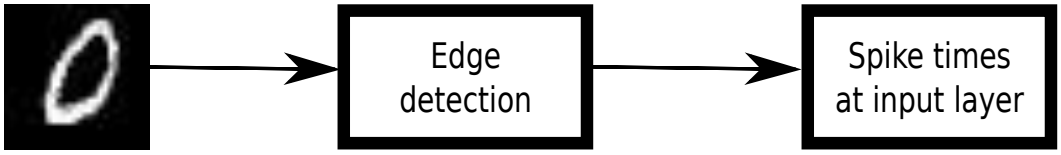

Figure 5: A block diagram of the encoding model that processes MNIST images to produce patterns of spike times

Luckily, no tractable classification problem and data-set would have 96 million degrees of freedom. As a result, we do not implement the lossy dimensionality reduction encoding performed by the retinal ganglion cells. Instead, we assume that the MNIST data-set, which consists of 784 pixel images, has already provided us with a lossily compressed image. In order to emulate the image preprocessing performed by V1, we preprocess each MNIST image using scikit-image's implementation of the canny edge detector. This produces a vector of 784 boolean values, each of which indicates the presence of an edge at the corresponding pixel. In order to encode these edge detections as time series, as this is the domain in which the tempotron learns and classifies, we map bijectively map the pixel indices both to discrete instances in a time series, and neurons in the input layer. The presence of an edge at pixel i is encoded as a spike emmitted by neuron i at $i$ms.

2.9 Performance testing methods and data

To benchmark the performance of our spiking neural classifiers, we train and test on encoded time series sourced from MNIST digit images using the encoding process described in section 2.8. Each MNIST image is first encoded as a series of input layer spikes. By partitioning this dataset, we implement k-fold cross validation, in which we train partition the dataset into k folds, train on (i.e. fit the classifier to) $k-1$ folds and test the classfication performance on the remaining fold. To demonstrate the powerful generalizability of the tempotron classifiers considered here, we also measure classifier performance using reverse k-fold cross validation (in which we train on one fold, and test on the remaining $k-1$ folds).

3 Results

3.1 Training results

Figure 6 shows mean accuracy of classifcation after learning for the corresponding number of iterations. Accuracy is averaged across five mutually disjoint data sets, each of which is disjoint from the set of potential training data. Training data sets consisted of one instance of each class for each supervised training iteration. The training signal here is implemented as a single spike induced in the target tempotron. Error bars indicate standard error of the mean (SEM) - i.e. ratio of standard deviation to $\sqrt{N_{\text{test}}}$, where N_{test} is the number of samples over which we test. This demonstrates that accuracy improves dramatically for the first few training iterations, after which, it appears to noisily osillate near the maximum accuracy acheived.

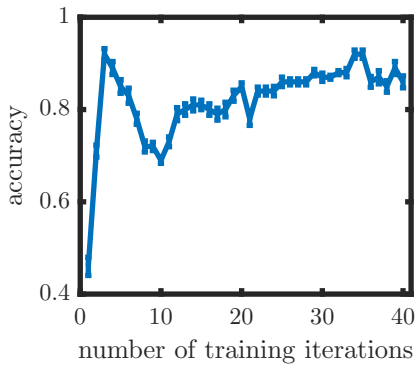

Figure 6: Accuracy of classification (with a competitive network) vs. number of learning iterations

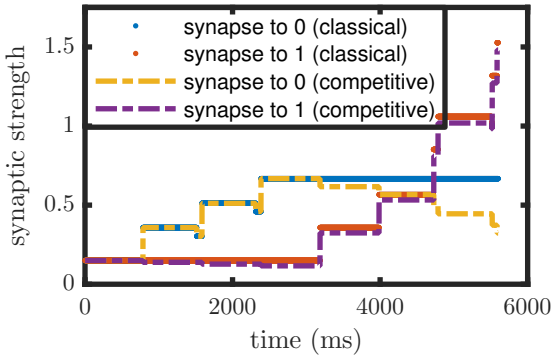

Figure 7: Variation in synaptic efficacy resulting from competitive STDP

Figure 7 shows the distribution of weights after a competitive network is trained on 8 samples chosen from a randomly allocated training set (chosen from the MNIST digit images and encoded as described in section 2.8). Each pixel's color represents the relative efficacy (normalized to largest strength observed) of the connection from the corresponding edge detector to the indicated tempotron, according to the colorbar provided. The left (resp. right) panel shows strengths of afferent connections of tempotron 0 (resp. 1). We have arranged the indices so as to preserve the spatial organization of the edge detectors in our encoding model. Qualitatively, these demonstrate that STDP associates edge detectors with the target tempotron, and depresses connections between target tempotrons and edge detectors that exclusively respond to non-target stimuli. To see this, note the similarity of the heat maps to the corresponding digit.

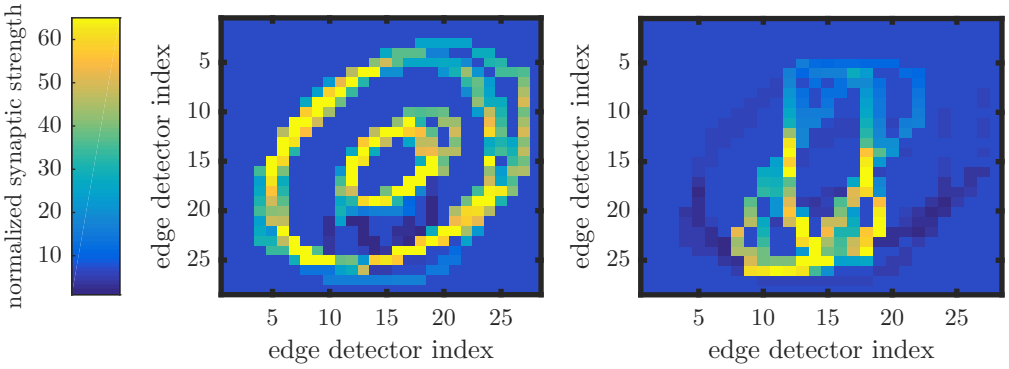

Figure 8: Distributions of synaptic strength from edge detection units to tempotrons

Figure 8 shows change in synaptic weight over time. The blue (resp. orange) curve corresponds to the connection from edge detector 712 to tempotron 0 (resp. 1). In first 3 epochs shown here, $[0, 2400]$ms, the network is trained on an instance of the encoded image of the number 0. During this training session both synapses undergo modification in response to the teaching signal (i.e. spikes induced in tempotron 0): $W_{712,0}$ is potentiated (i.e. strengthened), while $W_{712,1}$ is depressed (weakened), albeit quite slowly. This slow depression is the essential enhancing effect of competitive learning, and its combination with a Hebbian learning process, STDP. In [11], it is suggested that homeostatic (i.e. regulatory and equilibrating) processes are necessary in order to balance the instability imposed by the rapid potentiation that occurs. Without such regulation, the network may become overconnected and epilleptic in response to further stimulation (e.g. during additional training or classification). In the remainder of the session, the network is trained on an encoded image of the number 1. Here too, we see the desired effect: $W_{712,1}$ is potentiated, while $W_{712,0}$ is slowly depressed. In contrast, classical STDP exhibits qualitatively similar potentiation and no depression.

3.2 Performance results

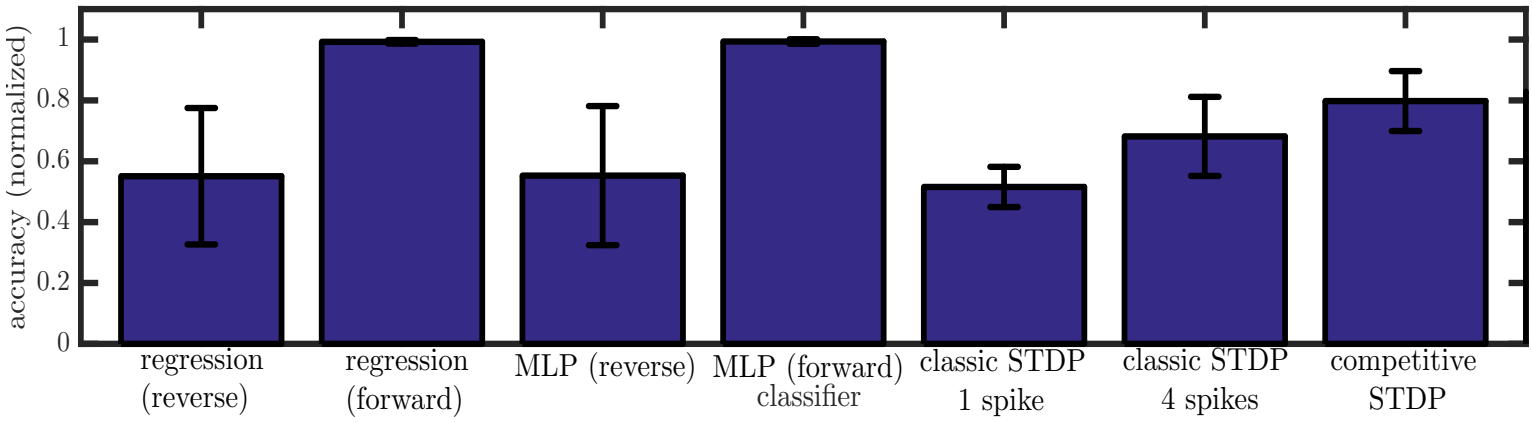

Figure 9: Accuracy of classification compared across all relevant classifiers considered

Figure 9 shows accuracy of various classifiers, averaged (for each classifier) across each iteration of either 10-fold cross validation or reverse 10-fold cross validation. Aggregate training and testing data sets are disjoint. Both forwared and reverse cross validation is computed over a dataset compiled from 100 instances of each class, chosen pseudo-randomly and programatically before any experimenation from the complete MNIST set of digit images. This pre-selection process was implemented to ensure reproducibility of our experiments. Error bars indicate SEM. We compare performance of our spiking classifier, subject to classical or competitive STDP, with those of soft-max linear regression and a MLP. All accuracies of spiking classifiers were obtained via reverse 10-fold cross validation. Unsurprisingly, a network equipped with classical STDP experiences a performance improvement when the teaching signal is enhanced by number of spikes in the training signal is quadrupled. This results from the increase in abundance of plasticity inducing spike pairs. In either case, competitive STDP tends to outperform classical STDP. Surprisingly, competitive STDP dramatically outperforms linear regression and the MLP in the reverse cross validation experiments, while classical STDP with an enhanced teaching signal achieves only a modest improvement over these more traditional techniques. With the present formulation of the learning and decoding processes, even competitive STDP does not improve its performance with additional learning iterations after the obvious phase transition in accuracy (as a function of training iteration count, shown in figure 6) that occurs very early in training. Thus performances of spiking classifiers in forward cross validation experiments do not differ significantly from those in reverse cross validation experiments. For brevity, we do not show those data. A more detailed description of implementations of systems discussed here is available in section 5.1.

4 Discussion

We observe that classification performance improves dramatically with training iterations early in the learning process, after which point, the accuracy oscillates near the maximum acheived. The early rapid improvement in accuracy (which is not observed to occur on the same time scale in soft-max regressors or MLPs) is a highly desired trait in any machine learning system - especially those adapted to operate online. The later noisy oscillations may be an effect produced by training on atypical representations of the target class. That the tempotrons are able to rapidly develop accuracy exceeding 80% indicates optimistically, that improvements to learning arising from optimization of the teaching signal and STDP windows (i.e. dynamics underlying $\Delta w_{i,j}$) should close this gap. One such improvement may arise from a new training procedure, which requires that training continue until the target tempotron spikes more confidently than any other tempotron (i.e. the decoder would declare the target tempotron's class in response to the recent epoch of stimulation). Following the arguments discussed in [12], which demonstrate an equivalence between learning with stochatic gradient descent on error of the readout (i.e. decoded classification) and STDP, when learning ensures this constraint, we hypothesize that this augmentation should improve classification performance. However applying this stronger requirement to the learning process will result in an increase in required number of training epochs. One interesting direction for further research involves following the approach of [13] to develop a more general information theoretic characterization of learning and computation in spiking neural classifiers.

The first glaring limitations arise when considering the chosen encoding model. Further investigation may explore how natural neural codes may arise from self organization of a preprocessing layer placed between stimulus information and the tempotrons. The authors would be remiss to omit admission that conclusions drawn from results presented here are further limited in the sense that this work considers only binary classification problems. We hypothesize that results shown here extend to multi-class problems, with any caveats imposed by changes in network stability and fundamental limits of information processing [13]. We therefore propose further investigation of this hypothesis, as well as other questions, whose answers may allow us to develop a more complete understanding of spike time based learning, with the goal of classifying perceived patterns of spike times.

It is apparent that a fixed learning window, defining $\Delta w_{i,j}$, fails to induce plasticity outside of its relevant time-scale. We propose to rectify this issue by studying how an adaptive window, which varies in response to changes in the apparent statistics of pre-synaptic spike times, may enable efficient and quickly generalizing learning of patterns at widely varying time scales. This engineering approach to developing a stable network whose synaptic modifications are governed by a time varying window was explored in metal oxide memristor networks in [14]. However, there is sparse general theoretical exploration of the impact of adaptive STDP windows on a network's stability and information processing power. Less studied still are interactions between each of these and various synaptic depression mechanisms (which may be modeled as operating across a broad range of timescales), such as those explored in [11], in which such homeostatic mechanisms are shown to be necessary - but not necessarily sufficient - for stable learning in a spiking neural network. Additionally, we only explore competition among output synapses. While intuitively, such a scheme is reasonable, a complete investigation will characterize impacts of dendritic competition. Note that this need not be implemented at dendrites in a physical model, but could be thought of as occuring between a set of synapses projecting to common neuron. We speculate that in combination with the newly proposed learning paradigm, an adaptive STDP window may yield a stable, quickly genearlizing spiking neural classifier.

Acknowledgments

We would like to thank Dr. Koyluoglu for support and guidance in developing a time series prediction system implemented in a spiking neural network, which inspired the study of the spiking neural classifiers studied here.

References

[1] Robert Gütig and Haim Sompolinsky. The tempotron: a neuron that learns spike timing–based decisions. *Nature neuroscience*, 9(3):420–428, 2006.

[2] Yang Dan and Mu-ming Poo. Spike timing-dependent plasticity of neural circuits. *Neuron*, 44(1):23–30, 2004.

[3] Dennis W Ruck, Steven K Rogers, Matthew Kabrisky, Mark E Oxley, and Bruce W Suter. The multilayer perceptron as an approximation to a bayes optimal discriminant function. *IEEE Transactions on Neural Networks*, 1(4):296–298, 1990.

[4] Wulfram Gerstner and Werner M Kistler. Mathematical formulations of hebbian learning. *Biological cybernetics*, 87(5-6):404–415, 2002.

[5] Sen Song, Kenneth D Miller, and Larry F Abbott. Competitive hebbian learning through spike-timing-dependent synaptic plasticity. *Nature neuroscience*, 3(9):919–926, 2000.

[6] Peter Dayan and LF Abbott. Theoretical neuroscience: computational and mathematical modeling of neural systems. *Journal of Cognitive Neuroscience*, 15(1):154–155, 2003.

[7] Christine A Curcio, Kenneth R Sloan, Robert E Kalina, and Anita E Hendrickson. Human photoreceptor topography. *Journal of comparative neurology*, 292(4):497–523, 1990.

[8] Andrew B Watson. A formula for human retinal ganglion cell receptive field density as a function of visual field location. *Journal of Vision*, 14(7):15–15, 2014.

[9] HB Barlow and DJ Tolhust. Why do you have edge detectors. In *Optical society of America Technical Digest*, volume 23, 1992.

[10] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106–154, 1962.

[11] Friedemann Zenke and Wulfram Gerstner. Hebbian plasticity requires compensatory processes on multiple timescales. *Phil. Trans. R. Soc. B*, 372(1715):20160259, 2017.

[12] Yoshua Bengio, Thomas Mesnard, Asja Fischer, Saizheng Zhang, and Yuhuai Wu. Stdp as presynaptic activity times rate of change of postsynaptic activity approximates backpropagation. *Neural Computation*, 2017.

[13] Ran Rubin, Rémi Monasson, and Haim Sompolinsky. Theory of spike timing-based neural classifiers. *Physical review letters*, 105(21):218102, 2010.

[14] Mirko Prezioso, F Merrikh Bayat, Brian Hoskins, K Likharev, and D Strukov. Self-adaptive spike-time-dependent plasticity of metal-oxide memristors. *Scientific reports*, 6, 2016.

5 Appendix

5.1 Implementations

Simulations of spiking neural networks subject to classical STDP are implemented and performed in the brian2 framework. Brian2 is a python module developed to allow for fast and straightforward development of simulations of spiking neural networks. Detailed installation instructions are available on the brian2 pythondocs page. The computational overhead associated with such a framework, as well as the difficulty in implementing causal, but counterintuitive constraints on plasticity, such as competition, motivated our development of a custom simulation environment specifically engineered in python with the goal of quickly and accurately simulating our spiking neural classifiers. This simulation environment, in which we produce all results related to competitive learning, is implemented in the directory 'competitiveSTDP', located at the root of the deliverable (a .zip archive) submitted for this assignment. Simulations of networks subject to classical STDP are implemented in the directory, 'classicalTempotron', located in the root of the deliverable. Classification with linear regression and a MLP are implemented in the directory 'ANNC' (artificial neural network classification), which is also located at the root of the deliverable. Please consult the readme files (each of which is named 'readme.txt') located in the directories 'ANNC', 'classicalTempotron', and 'competitiveSTDP' for more information.

5.2 Choices of parameters

In the classical STDP framework, we choose $V_{\text{reset}} = -75\text{mV}$, $V_{\text{rest}} = -70\text{mV}$, $V_t = -54\text{mV}$, $\tau_m = 5\text{ms}$, $\tau_n = 10\text{ms}$, $\tau_s = 50\text{ms}$, and $\alpha = 10\text{mv}$.
In the competitive framework, we choose $V_{\text{reset}} = -74\text{mV}$, $V_{\text{rest}} = -70\text{mV}$, $V_t = -55\text{mV}$, $\tau_m = 150\text{ms}$, $\tau_n = 40\text{ms}$, $\tau_s = 200\text{ms}$, and $\alpha = 10\text{mv}$.