Combining STDP and Reward-Modulated STDP in Deep Convolutional Spiking Neural Networks for Digit Recognition

Milad Mozafari[1], Mohammad Ganjtabesh[1,*], Abbas Nowzari-Dalini[1], Simon J. Thorpe[2], and Timothée Masquelier[2]

[1] *Department of Computer Science, School of Mathematics, Statistics, and Computer Science, University of Tehran, Tehran, Iran*

[2] *CerCo UMR 5549, CNRS – Université Toulouse 3, France*

Abstract

The primate visual system has inspired the development of deep artificial neural networks, which have revolutionized the computer vision domain. Yet these networks are much less energy-efficient than their biological counterparts, and they are typically trained with backpropagation, which is extremely data-hungry. To address these limitations, we used a deep convolutional spiking neural network (DCSNN) and a latency-coding scheme. We trained it using a combination of spike-timing-dependent plasticity (STDP) for the lowest layers and reward-modulated STDP (R-STDP) for the highest ones. In short, with R-STDP a correct (resp. incorrect) decision leads to STDP (resp. anti-STDP). This approach led to an accuracy of 97.2% on MNIST, without requiring an external classifier. In addition, we demonstrated that R-STDP extracts features that are diagnostic for the task at hand, and discards the other ones, whereas STDP extracts any feature that repeats. Finally, our approach is biologically plausible, hardware friendly, and energy-efficient.

Keywords: Spiking Neural Networks, Deep Architecture, Digit Recognition, STDP, Reward-Modulated STDP, Latency-Coding

1 Introduction

In recent years, deep convolutional neural networks (DCNNs) have revolutionized machine vision and can now outperform human vision in many object recognition tasks with natural images [1]. Despite their outstanding levels of performance, the search for brain inspired computational models continues and is attracting more and more researchers from around the world. Pursuing this line of research, a large number of models with enhanced bio-plausibility based on spiking neural networks (SNNs) have emerged. However, SNNs are not yet competitive with DCNNs in terms of recognition accuracy. If DCNNs work well, what is the reason for this increased interest in neurobiological inspiration and use of SNNs?

To begin with, energy consumption is of great importance. Thanks to the to millions of years of optimisation by evolution, the human brain consumes about 20 Watts [2] – roughly the power con-

*Corresponding author.
Email addresses:
milad.mozafari@ut.ac.ir (MM),
mgtabesh@ut.ac.ir (MG)
nowzari@ut.ac.ir (AND)
simon.thorpe@cnrs.fr (SJT)
timothee.masquelier@cnrs.fr (TM).

sumption of an average laptop. Although we are far from understanding the secrets of this remarkable efficiency, the use of spike-based processing has already helped neuromorphic researchers to design energy-efficient microchips [3, 4].

Furthermore, employing embedded and real-time systems for artificial intelligence (AI) is important with the advent of small and portable computing devices. In recent years, specialized real-time chips for DCNNs have been released that are capable of fast simulation of pre-trained networks. However, online on-chip training of DCNNs with exact error backpropagation, due to the high-precision and time-consuming operations is not yet practical. This problem has made researchers modifying error backpropagation algorithm and making it hardware-friendly [5]. Conversely, computation and communication with spikes can be extremely fast and applicable in low precision platforms [6]. Additionally, biologically inspired learning rules such as spike-timing-dependent plasticity (STDP) [7, 8] can be hardware-friendly, but also appropriate for online on-chip training [9].

Due to the aforementioned reasons, together with the powerful SNN's spatio-temporal activity domain, researchers have tried many different methods to make SNNs work on visual tasks. The use of hierarchically structured neural networks is a common approach, yet configuring other parameters such as the number of layers, neuron models, information encoding, and learning rules is the subject of much debate. There are shallow [10, 11] and deep [12, 13] SNNs with various types of connectivity structures, such as recurrent [14], convolutional [10, 15, 16], and fully connected [17]. Information encoding is the other aspect of this debate, where rate-based coding [18, 19, 20], and temporal coding [10, 11, 17, 21] are two of the main options. Different learning techniques are also applied to SNNs, from backpropagation [15, 22, 23, 24], tempotron [11, 25], and other supervised techniques [19, 20, 21, 26, 27], to unsupervised STDP and its variants [17, 28].

Regarding the pursuit of brain inspiration, STDP-based SNNs are the most biologically plausible ones. Using STDP, the network can successfully extract frequently occurring visual features. However, an unsupervised learning rule alone is not sufficient for decision-making, where external classifiers such as support vector machines (SVMs) and radial basis functions (RBFs), or supervised variants of STDP, are usually required. There are several STDP-based SNNs that have been applied to MNIST dataset for digit recognition. For example, Brader et. al. [29] with 96.5%, Querlioz et. al. [30] with 93.5%, and Diehl and Cook [17] with 95% of recognition accuracy are the successful models with shallow structure in digit recognition. With deep structure, Beyer et. al. [31] achieved a not so good performance of 91.6%, however, Kheradpisheh et. al. [32] trained a deep convolutional SNN (DCSNN) and increased the accuracy to 98.4%.

Researchers have started exploring the potential of using reinforcement learning (RL) in SNNs and DCNNs [33, 34, 35, 36, 37, 38, 39, 40]. By RL, the learner is encouraged to repeat rewarding behaviors and avoid those leading to punishments [41]. Using supervised learning, the network learns at most what the supervisor knows, while with RL, it is able to explore the environment and learn novel skills (unknown to any supervisor) that increase the fitness and reward acquisition [42].

We previously developed a shallow SNN with a single trainable layer [43], where the plasticity was governed by reward-modulated STDP (R-STDP). R-STDP is a reinforcement learning rule inspired by the roles of neuromodulators such as Dopamine (DA) and Acetylcholine (ACh) in modulation of STDP [44, 45]. Our network made decisions about category of objects solely based on the earliest spike in the last layer without using any external classifier. Together with rank-order encoding and at most one spike per neuron, our network was biologically plausible, fast, energy-efficient, and hardware-friendly with acceptable performance on natural images. However, its shallow structure made it inappropriate for large and complex datasets with high degrees of variations.

In this research, we designed a 3-layer DCSNN with a structure adopted from [32], mainly for digit recognition. The proposed network does not need any external classifier and uses a neuron-based

decision-making layer trained with R-STDP. First, the input image is convolved with difference of Gaussian (DoG) filters of different scales. Then, by an intensity-to-latency encoding [46], a spike wave is generated and propagated to the next layer. After passing through multiple convolutional and pooling layers with neurons that are allowed to fire at most once, the spike wave reaches the last layer, where there are decision-making neurons that are pre-assigned to each digit. For each input image, the neuron in the last layer with earliest spike time or maximum potential indicates the decision of the network.

We evaluated our DCSNN on the MNIST dataset for digit recognition. First, we applied R-STDP only to the last trainable layer and STDP to the first two, achieving 97.2% of recognition performance. Next, we investigated if applying R-STDP to the penultimate trainable layer is helpful. We found that in case of having frequent distractors in the input and limitations on the computational resources, using R-STDP instead of STDP in the penultimate layer is beneficial.

The rest of this paper is organized as follows: A precise description of the components of the proposed network, synaptic plasticity, and decision-making is provided in Section 2. Then, in Section 3 and 4, the performance of the network in digit recognition and the results of applying R-STDP in multiple layers are presented. Finally, in Section 5, the proposed network is discussed from different points of view and the possible future works are highlighted.

2 Methods

The focus of this paper is to train and tune a DCSNN by a combination of biologically plausible learning rules, i.e. STDP and R-STDP, for digit recognition. To this end, we modify the network proposed in [32] by dropping its external classifier (SVM) and adding a neuron-based decision-making layer, as well as using R-STDP for synaptic plasticity. This part of the paper explains the structure of the network and its components.

2.1 Overall Structure

The proposed network has six layers, that are three convolutional layers ($S1, S2, S3$), each followed by a pooling layer ($C1, C2, C3$). To convert MNIST images into spike waves, they are filtered by DoG kernels and encoded into spike times by the intensity-to-latency scheme. Plasticity of the afferents of convolutional layers is done by either STDP or R-STDP learning rules. The ultimate layer ($C3$) is a global pooling in which neurons are pre-assigned to each digit. These neurons are indicators for the network's decision. Figure 1 plots the outline of the proposed network.

Details of the input spike generation, functionality of each layer, learning rules, and decision-making process are given in the rest of this section.

2.2 Input Spike Waves

On each input image, On- and Off-center DoG filters of three different scales are applied with zero padding. Window sizes are set to 3×3, 7×7, and 13×13, where their standard deviations (σ_1, σ_2) are $(3/9, 6/9)$, $(7/9, 14/9)$, and $(13/9, 26/9)$, respectively. We keep the ratio of 2 between two sigmas in each scale. These values are chosen practically based on the statistics of input images.

In the resulting feature maps, all the values below 50 are ignored and the remaining values are descendingly sorted, denoting the order of spike propagation.

In order to speed up running time of our parallel implementation of the network, we equally distribute the ordered input spikes into a fixed number of bins. All the spikes in each bin are propagated simultaneously.

2.3 Convolutional Layers

Each convolutional layer (S-layer) in the proposed network contains several 2-dimensional grids of Integrate-and-Fire (IF) neurons, called feature maps. Each neuron in a S-layer has a fixed 3-dimensional input window of afferents with equal width and height, and a depth equal to the number of feature maps in the previous layer. The firing

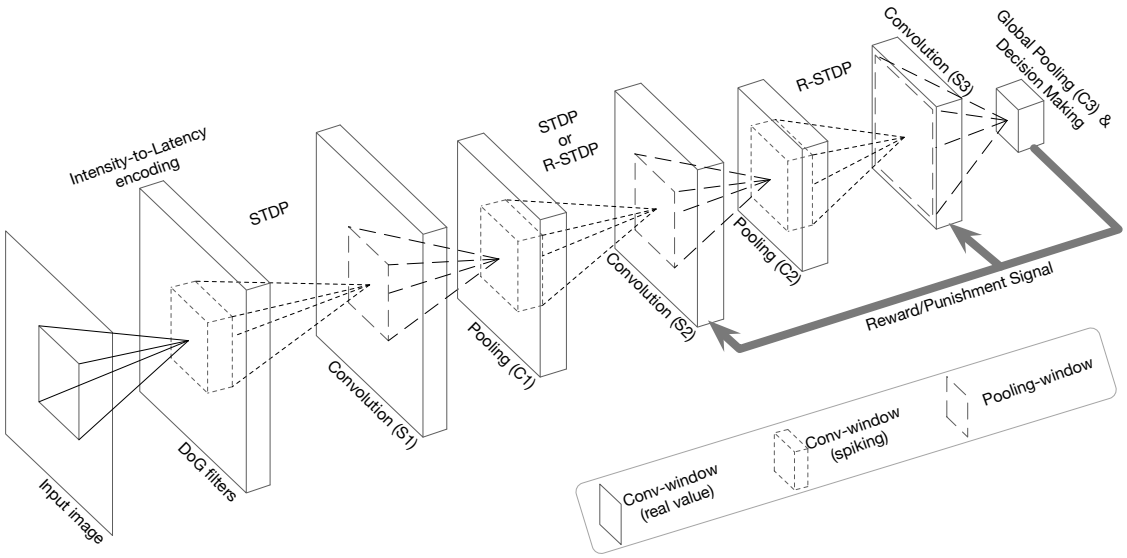

Figure 1: Overall structure of the proposed DCSNN for digit recognition. The input image is convolved by six DoG filters (on- and off-center in three different scales) and converted into spike latencies based on intensity-to-latency encoding scheme. Generated spikes are processed through three spiking convolution-then-pooling layers ($S1-C1$,$S2-C2$, and $S3-C3$). All of the spiking convolution layers are trainable, employing either STDP or R-STDP learning rules. The network makes its decision in $C3$, where neurons are pre-assigned to digits, and the decision is the digit assigned to the neuron with either the maximum internal potential or the earliest spike time. Regarding the decision, a reward/punishment (reinforcement) signal will be generated globally, which is received by R-STDP-enabled layers.

threshold is also set to be the same across all the neurons in a single layer.

In each time step, the internal potential of each IF neuron is increased by the incoming spikes within its input window using the magnitude of the corresponding synaptic weights. These neurons have no leakage and if a neuron reaches the firing threshold, it will emit a single spike, after which remains silent until the next input image is fed to the network. For each new input, the internal potential of all the neurons are reset to zero. Note that weight-sharing mechanism is applied for all neurons in a single feature map.

2.4 Pooling Layers

Pooling layers (C-layers) in our network are employed for introducing position invariance and reducing information redundancy. Each C-layer has the same number of feature maps as its previous S-layer and there is a one-to-one association between maps of the two layers.

There are two types of pooling layers in our network: spike-based and potential-based. Both types have a 2-dimensional input window and a particular stride. Each neuron in the spike- and potential-based C-layers, indicates the earliest spike time and the maximum potential of the neurons within its input window, respectively.

2.5 Decision-Making and Reinforcement Signal

As mentioned before, neurons in the final C-layer ($C3$) perform global pooling on their corresponding $S3$ grids. These neurons are labeled with input categories such that each $C3$ neuron is assigned to single category, but each category may be assigned to multiple neurons. Providing the labeled neurons in $C3$, the decision of the network for each input is the label of the neuron with the earliest spike time or maximum internal potential in case of using spike- or potential-based pooling layer, respectively.

When the decision of the network is indicated, it will be compared with the original label of the input image. If they match (mismatch), a reward (punishment) signal will be generated globally. This signal will be received by layers that employ R-STDP rule for synaptic plasticity (see the next section).

2.6 Plasticity

We initially set the synaptic weights with random values drawn from a normal distribution with mean 0.8 and standard deviation 0.02 [32]. Both STDP and R-STDP learning rules are used to train S-layers in our network. We define a general formula that can be employed for both rules as follows:

$$\delta_{ij} = \begin{cases} \alpha\phi_r a_r^+ + \beta\phi_p a_p^- & \text{if } t_j - t_i \leq 0, \\ \alpha\phi_r a_r^- + \beta\phi_p a_p^+ & \text{if } t_j - t_i > 0, \\ & \text{or neuron } j \text{ never fires} \end{cases} \quad (1)$$

$$\Delta w_{ij} = \delta_{ij}(w_{ij})(1 - w_{ij}), \quad (2)$$

where i and j refer to the post- and pre-synaptic neurons, respectively, Δw_{ij} is the amount of weight change for the synapse connecting the two neurons, and a_r^+, a_r^-, a_p^+, and a_p^- scale the magnitude of weight change. Furthermore, to specify the direction of weight change, we set $a_r^+, a_p^+ > 0$ and $a_r^-, a_p^- < 0$. Our learning rule only needs the sign of the spike time difference, not the exact value, and uses an infinite time window. The other parameters, say ϕ_r, ϕ_p, α, and β, are employed as controlling factors which will be set differently for each learning rule.

To apply STDP on a S-layer, the controlling factors are $\phi_r = 1$, $\phi_p = 0$, $\alpha = 1$, and $\beta = 0$. For R-STDP, the values of α and β depends on the reinforcement signal generated by the decision-making layer. If a "reward" signal is generated, then $\alpha = 1$ and $\beta = 0$, whereas if a "punishment" signal is generated, then $\alpha = 0$ and $\beta = 1$. The reinforcement signal can be "neutral" as well, for which $\alpha = 0$ and $\beta = 0$. The neutral signal is useful when we have unlabeled data. Similar to our previous work, we apply adjustment factors by $\phi_r = \frac{N_{miss}}{N}$ and $\phi_p = \frac{N_{hit}}{N}$, where N_{miss} and N_{hit} denote the number of samples that are classified correctly and incorrectly over the last batch of N input samples, respectively. Note that plasticity is done for each image, while ϕ_r and ϕ_p are updated after each batch of samples.

To do plasticity in each trainable layer, we employ a k-winner-take-all mechanism, by which only k neurons are eligible to do plasticity. These neurons cannot be from a same feature map because of the weight sharing strategy through the entire map. Choosing the k winners are first based on the earliest spike times, and then the higher internal potentials. During the training process, when a winner is indicated, a $r \times r$ inhibition window, centering the winner neuron, will be applied to all the feature maps, preventing them to be selected as the next winners for the current input image.

Additionally, we multiply the term $w_{ij}(1 - w_{ij})$ by δ_{ij} which not only keeps weights between the range $[0, 1]$, but also stabilizes the weight changes as the weights converge.

3 Task 1: Solving 10-class Digit Recognition

In this task, we train our proposed DCSNN on all of the 60000 images of handwritten digits from the MNIST dataset. Then the recognition performance of the network is tested over the 10000 unseen samples provided by the author of the dataset.

3.1 Network Configuration

The input layer consists of six feature maps corresponding to the six DoG filters (three scales with both on- and off-center polarities), each contains the spike latencies. After that, there are three S-layers, each followed by a C-layer with specific parameters that are summarized in Tables 1–3.

The first two S-layers have specific thresholds, however the last one has an "infinite" threshold. This means that $S3$ neurons never fire for an input image. Instead, they accumulate all the incoming spikes as their internal potential. When there are no more spikes to accumulate, a manual spike time, greater that the previous spike times, will be set for all of the $S3$ neurons.

$C1$ and $C2$ neurons perform spike-based local pooling, whereas $C3$ neurons apply potential-based global pooling which is consistent with the behavior of $S3$ neurons. Accordingly, the proposed network makes decisions based on the maximum potential among $C3$ neurons. In other words, the label associated to the $C3$ neuron with the maximum po-

tential will be selected as the network's decision for each input image.

It is worth mentioning that in this task, we drop the weight stabilizer term from the plasticity equation for layer $S3$. Instead, we manually limit the weights between 0.2 and 0.8. This modification is done based on our experimental results, where we found that it is better to continue synaptic plasticity even in late iterations.

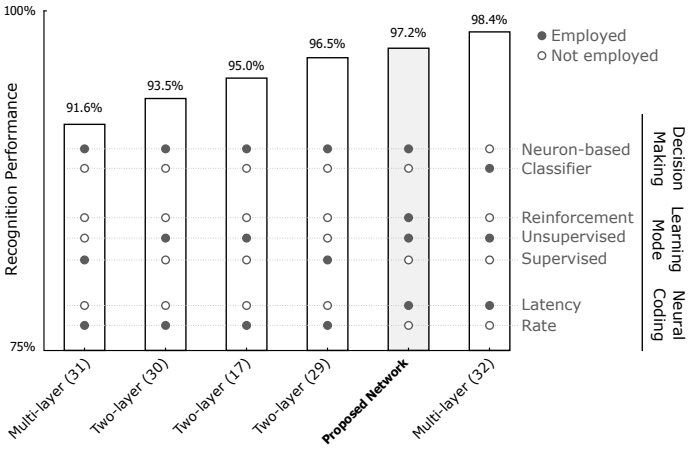

Figure 2: Recognition performance of several available SNNs trained by STDP-based learning rules on the MNIST dataset, as well as the proposed network.

3.2 Training and Evaluation

We trained the network in a layer-by-layer manner. Layers $S1$ and $S2$ were trained by STDP for 10^5 and 2×10^5 iterations, respectively. In each iteration, an image was fed to the network and the learning rates were multiplied by 2 after each 500 iterations. We kept doing multiplication while a_r^+ was less than 0.15.

Training of $S3$ was governed by R-STDP employing the reinforcement signal produced by layer $C3$. We trained $S3$ for 4×10^7 iterations in total and evaluated the recognition performance after each 6×10^4 iterations. Our network achieved 97.2% of performance, which is higher than most of the previous STDP-based SNNs evaluated on the MNIST dataset (see Table 2). It is worth mentioning that our network ranked as the second best while it does not use any external classifier. It not only increased the biological plausibility, but also the hardware friendliness of the network [32, 43].

We also examined applying R-STDP to the penultimate S-layer ($S2$) as well, however it did not improve the performance in this particular task. By analyzing the influence of R-STDP on $S2$, we hypothesized that it can help extracting target-specific diagnostic features, if there are frequent distractors among the input images and the available computational resources are limited. Our next task is designed to examine this hypothesis.

4 Task 2: R-STDP Can Discard Non-Diagnostic Features

The goal of this task is to distinguish two specific digits while other digits are presented as distractors. This task is conducted over a subset of MNIST images employing a smaller network, which allows us to analyze the advantages of applying R-STDP to more than one layer. This part of the paper is started by introducing a handmade problem and a simple two-layer network to solve it, for the sake of a better illustration. Then, we show the same outcome using images of the MNIST dataset.

4.1 Handmade Problem

We prepare a set of artificial images of size 9×3, each contains two oriented bars in its left and right most 3×3 regions. As shown in Figure 3a , we make all of the possible combinations of bars from four different orientations. We characterize three classes of target inputs by pair-wise combinations of three differently oriented bars (Figure 3b), leaving the other inputs as distractors. All of the inputs are fed to the network in a random order.

We design a minimal two-layer spiking network that distinguishes the three target classes, if and only if it learns the three diagnostic oriented lines

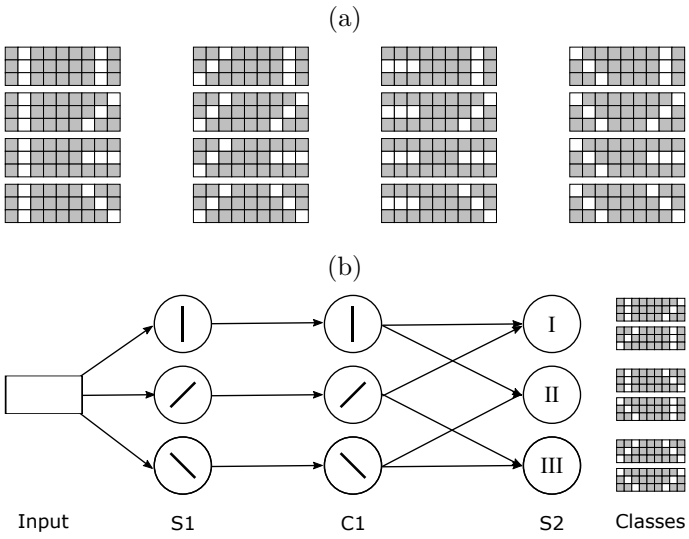

Figure 3: (a) Input set for the handmade problem. It contains all combinations of bars in four different oriented. For each input, spikes are simultaneously propagated from the white tiles. A 3×3 gap is placed between the two oriented bars in order to prevent neurons seeing parts of the two bars at the same time. (b) A simple network with two trainable layers ($S1$, and $S2$) that solves a 3-class recognition task with layer $S2$ as the decision making layer. Each class is the position invariant combination of two different bars. Classes are defined in a way that the horizontal bar carries no distinguishing information (distractor), while others are vital for final recognition (targets). Since there are only three feature maps in $S1$, the task is fully solvable if and only if the network learns all the three targets in this layer.

by the neurons in the penultimate S-layer, i.e. $S1$ (see Figure 3b). According to the target classes, the horizontal bar carries no diagnostic information and learning it is a waste of neuronal resources. To this end, we put three feature maps in layer $S1$ whose neurons have 3×3 input windows. After a spike-based global pooling layer, we put three neurons that have $1 \times 1 \times 3$ input windows.

We examined both STDP and R-STDP learning rules to train $S1$, while the last layer was always trained by R-STDP in order to make decisions. Note that when we used R-STDP on two layers, plasticity of both layers was postponed until arrival of the reinforcement signal, thus they were trained simultaneously. Moreover, the reinforcement signal for non-target images was "neutral" regardless of the decision of the network.

According to the results, the network with STDP-trained $S1$ failed most of the times to extract the correct orientations. In contrast, training $S1$ with R-STDP enabled the network to solve the problem without any failure. The obtained results agree with the nature of STDP, which extracts frequent features regardless of the outcome. Since the probability of the appearance of each oriented bar is the same, the chance of extracting all the diagnostic bars is 25% for STDP (the probability of discarding the non-diagnostic feature among the four is 1/4).

We implemented a similar task with the MNIST dataset. In each instance of the task, two digits are selected as the targets and the other digits marked as distractors. The goal is to distinguish the two target digits using a small number of features, while the network receives images of all digits.

4.2 Network Configuration

The number of layers and their arrangement is the same as the network in Task 1, however there are several differences. First, MNIST images are fed to the network without application of DoG filters. Second, we put fewer feature maps with different input window sizes and thresholds in all of the S-layers. Third, $S3$ neurons have specific thresholds, that allow them to fire, thus $C3$ performs a spike-based global pooling operation. In addition, for each category, there is only one neuron in $C3$. The values of parameters are summarized in Tables 1–3.

4.3 Training and Evaluation

We performed the task for all pairs of digits ($\binom{10}{2} = 45$ pairs), and different number of feature maps in $S2$, ranging from 2 to 20. For each pair, the network was trained over 10^4 samples from the training set and tested over 100 testing samples of each target digit (200 testing samples in total). $S1$ was separately trained by STDP, while on $S2$ both STDP and R-STDP were examined. Plasticity in $S3$ always governed by R-STDP.

For each particular number of feature maps in $S2$, we calculated the average recognition performance of the network for both cases of using STDP and R-STDP over all of the digit pairs. As shown in Figure 4, the results of the experiments clearly support our claim. As we have fewer neuronal resources, using R-STDP is more beneficial than STDP. However, by increasing the number of fea-

Table 1: Values of the S-layer parameters used in each digit recognition task.

	Layer	Number of Feature Maps	Input Window (Width, Height, Depth)	Threshold
Task 1	$S1$	30	$(5, 5, 6)$	15
	$S2$	250	$(3, 3, 30)$	10
	$S3$	200	$(5, 5, 250)$	∞
Task 2	$S1$	10	$(5, 5, 1)$	5
	$S2$	$x \in \{2, 4, ..., 20\} \cup \{30, 40\}$	$(15, 15, 10)$	40
	$S3$	2	$(1, 1, x)$	0.9

Table 2: Values of the C-layer parameters used in each digit recognition task.

	Layer	Input Window (Width, Height)	Stride	Type
Task 1	$C1$	$(2, 2)$	2	Spike-Based
	$C2$	$(3, 3)$	3	Spike-Based
	$C3$	$(5, 5)$	0	Potential-Based
Task 2	$C1$	$(2, 2)$	2	Spike-Based
	$C2$	$(14, 14)$	0	Spike-Based
	$C3$	$(1, 1)$	0	Spike-Based

Table 3: Values of parameters for the synaptic plasticity used in each digit recognition task.

	Layer	a_r^+	a_r^-	a_p^+	a_p^-	k	r
Task 1	$S1$	0.004	-0.003	0	0	5	3
	$S2$	0.004	-0.003	0	0	8	2
	$S3$	0.004	-0.003	0.0005	-0.004	1	0
Task 2	$S1$	0.004	-0.003	0	0	1	0
	$S2$	0.04	-0.03	0.005	0.04	1	0
	$S3$	0.004	-0.003	0.0005	-0.004	1	0

ture maps in $S2$, STDP also has the chance to extract diagnostic features and fills the performance gap. According to Figure 5, R-STDP helps $S2$ neurons to extract target-aware features. In contrast, using STDP, $S2$ neurons are blind to the targets and extract the same features for all the pairs (the same random seed is used for all of the tasks).

We acknowledge that the network could achieve higher accuracies if its parameters are tuned for each pair of digits. Besides, in order to get the results in a feasible time for large number of simulations, we had to limit the number of iterations and use faster learning rates which again, degrade the performance.

5 Discussion

By the emerge of DCNNs, AI is now able to solve sophisticated problems such as visual object recognition, in which humans used to be the masters [1]. Most of the early DCNNs employ supervised learning techniques to be trained for a specific task. However, supervised learning limits the learner to the information provided by the supervisor. Recently, researchers have found that rein-

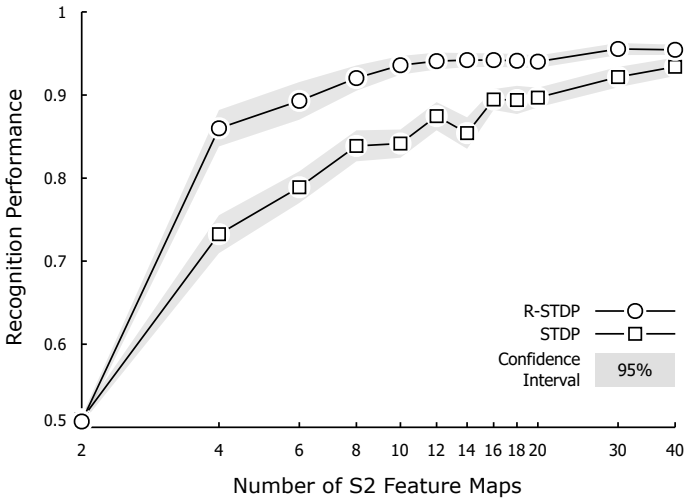

Figure 4: Comparison of recognition performance when STDP or R-STDP is applied to $S2$ in Task 2. Superiority of R-STDP over STDP in finding diagnostic features (and discarding distractors) is clearly shown when there are small number of feature maps in $S2$. As the number of feature maps increases, STDP has more chance to extract features for target digits, therefore approaches the performance of applying R-STDP. With respect to the number of feature maps, R-STDP shows a monotonic behavior in increasing the performance and confidence level, however, STDP is not as robust as R-STDP, specially in terms of confidence level.

forcement learning would be the new game changer and started to combine DCNNs with reinforcement learning techniques, also known as deep reinforcement learning. This combination has introduced networks that are able to achieve super-human skills by finding solutions that have never been tested by a human before [39, 40, 42].

Despite the revolutionary performance of DCNNs in machine vision, SNNs have always been under improvement to solve complex vision tasks. SNNs are not as perfect as DCNNs, however, they possess a great spatio-temporal capacity by which they are able to fill the performance gap or even bypass DCNNs. Additionally, SNNs have been shown to be energy efficient and hardware friendly, which makes them suitable to be deployed at AI hardwares [3, 4].

STDP has been successfully applied to SNNs for digit recognition [17], however, because of its unsupervised nature, a supervised readout is required to achieve high accuracies [32]. Apart from the fact that supervised readouts (e.g. SVMs) are not biologically supported, they are not appropriate for hardware implementation.

In this paper, we presented a DCSNN for digit recognition which is trained by a combination of STDP and R-STDP learning rules. R-STDP is a biologically plausible reinforcement learning rule [44], appropriate for hardware implementation as well. This combination is inspired by the brain where the synaptic plasticity is mostly unsupervised, taking into account the local activity of pre- and post-synaptic neurons, but it can be modulated by the release of neuromodulators, reflecting the feedbacks received from the surrounding environment [7, 8, 44, 45].

The proposed network extracts regularities in the input images by STDP in early layers and uses R-STDP to learn rewarding decision (correct label) in the last layer, using the information provided by early layers. Using R-STDP in the last convolution layer of the proposed network enabled us to use a neuron-based decision making layer instead of complex external classifiers. In this network, information is encoded in the earliest spike time of neurons. The input image is convolved with multiple DoG filters and converted into spike latencies by the intensity-to-latency encoding scheme. Spike propagation as well as plasticity are done in a layer-wise manner with an exception that all the R-STDP-enabled layers are trained simultaneously. When the spike wave reaches the decision making layer, the label of the neuron with either the maximum potential or the earliest spike time will be selected as the network's decision. Comparing the decision to the original label of the input, a reward/punishment (reinforcement) signal is globally generated which modulates the plasticity in trainable layers that employ R-STDP learning rule.

First, we applied our network to the whole MNIST dataset employing R-STDP only on the last trainable layer. Our network achieved 97.2% of recognition accuracy which is higher than most of the previously proposed STDP-based SNNs. We then investigated if applying R-STDP in more than one layer is helpful, by allowing the penultimate trainable layer to use R-STDP. Reviewing the results of a handful of different experiments reveled that if the input set is polluted with frequent distractors, applying R-STDP in more than one layer

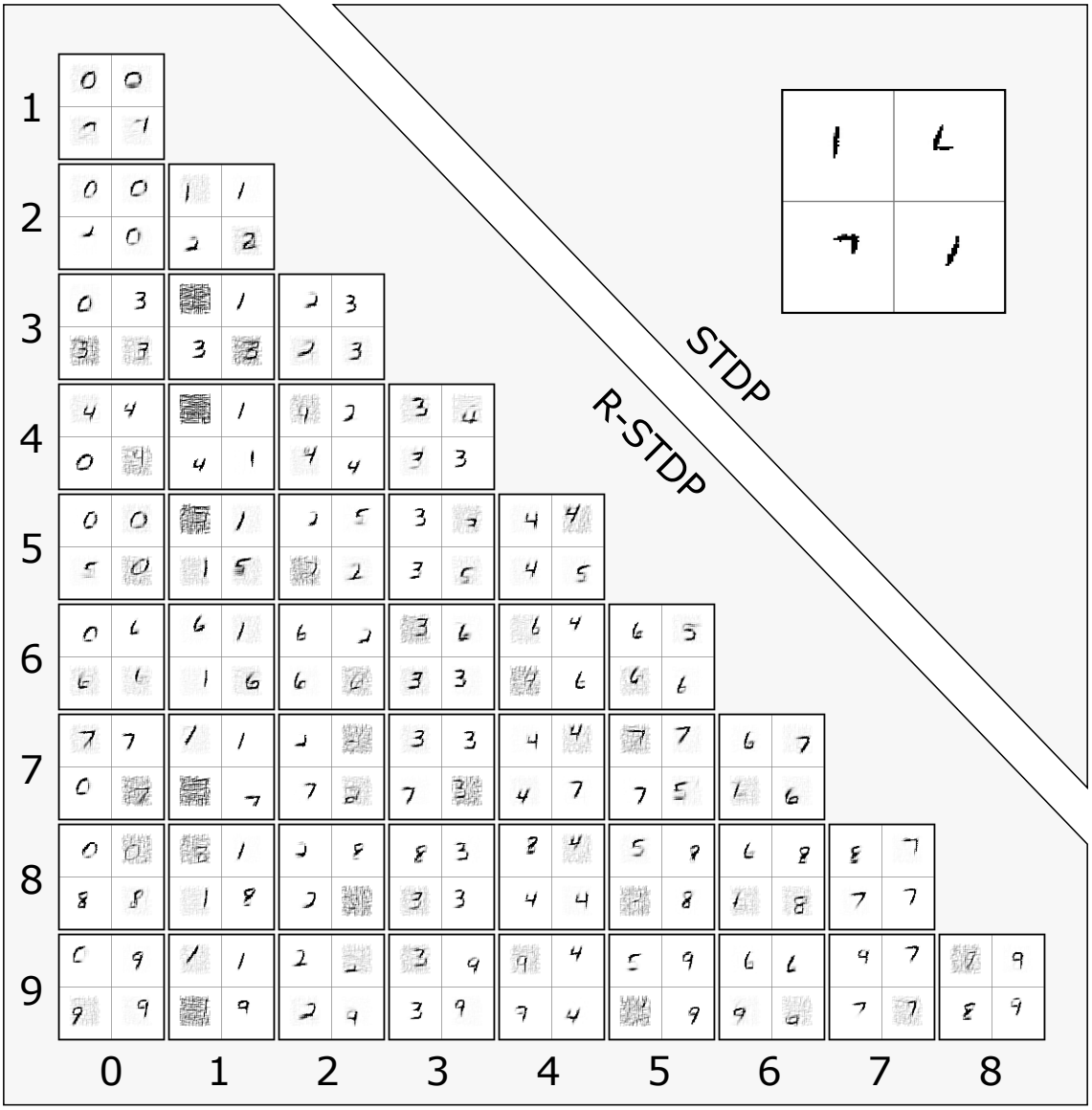

Figure 5: Reconstruction of features extracted by $S2$ neurons in Task 2, when the number of feature maps is limited by four. The lower part of the figure illustrates features that are extracted by applying R-STDP to $S2$ afferents, solving each of the pair-wise digit recognition tasks (corresponding to row and column indices). The existence of non-trained or semi-trained feature maps (dark or gray squares) is due to the fact that other features that are representative enough, win the plasticity competition for all images, leaving no chance for others to learn or converge. In the upper part, the only four features obtained by applying STDP to $S2$ are shown.

helps avoiding distractors and extracting diagnostic features using lower computational resources than a STDP-enabled layer.

In order to improve the performance of the proposed network, we tested various configurations and values for parameters. Here, we survey our findings regarding the results of these tests:

Deciding based on the maximum potential in Task 1. We tried decision making based on the earliest spike in the first step, but the results were not competing. By exploring the misclassified samples, we found that the digits with common partial features are the most problematic ones. For instance, digits 1 and 7 share a skewed vertical line (1 is mostly written skewed in MNIST). Lets say a neuron has learned to generate the earliest spike for digit 1. Obviously, it will generate the earliest spike for many samples of digit 7 as well, resulting in high rate of false alarms. Thus, we decided to omit the threshold and consider the maximum potential as the decision indicator. Since the network has to wait until the last spike, this approach increased the computation time, however successfully relaxed the aforementioned problem.

R-STDP is only applied to the last layer in Task 1. As we showed in Task 2 and our previous work [43], R-STDP can be better than STDP when computational resources are limited. In Task 1, the goal was to increase the performance of the network to an acceptable and competing level. Since MNIST is a dataset with high range of variations for each digit, we had to put enough feature maps in each layer to cover them. As a consequence, the large number of feature maps and the fact that there were no distractors in the inputs, STDP worked well in extracting useful intermediate features.

Learning in the ultimate layer is slower than the penultimate one in Task 2. In Task 2, $S2$ and $S3$ were under simultaneous training. If $S3$ modifies synaptic weights as fast as $S2$, each change in $S2$ may change a previously good behavior of $S3$ into a bad one, which then ruin $S2$, and this cycle may continue forever, destroying stability of the network.

The proposed network inherits bio-plausability, energy efficiency and hardware friendliness from its ancestors [32, 43], which now is able to solve harder and complex recognition tasks with its deeper structure. We believe that our DCSNN can be further improved from multiple aspects.

As mentioned before, potential-based decision making degrades the computational efficiency. One solution to overcome false alarms because of common partial features while having spike-based decision making in is to engage inhibitory neurons. Inhibitory neurons can suppress a neuron if it receives something more than the preferred stimulus (e.g. the horizontal line of digit 7, inhibits neurons responsible for digit 1, thus producing no false alarm). Another aspect would be performance improvement by designing a population-based decision making layer. For instance, one can consider a group of earliest spikes instead of one and applies major voting, which may affect the training strategy as well. Another direction for future work would be investigation of the layer-wise application of R-STDP for intermediate layers where the reward/punishment signal can be generated based on the other of activity measures such as sparsity, information content, and diversity.

Acknowledgment

This research received funding from Iran National Science Foundation: INSF (No. 96005286) and the European Research Council under the European Union's 7th Framework Program (FP/20072013)/ERC Grant Agreement no. 323711 (M4 project).

References

[1] W. Rawat, Z. Wang, Deep convolutional neural networks for image classification: A comprehensive review, Neural computation 29 (9) (2017) 2352–2449.

[2] J. W. Mink, R. J. Blumenschine, D. B. Adams, Ratio of central nervous system to body metabolism in vertebrates: its constancy and functional basis, American Journal of Physiology-Regulatory, Integrative and Comparative Physiology 241 (3) (1981) R203–R212.

[3] S. Furber, Large-scale neuromorphic computing systems, Journal of neural engineering 13 (5) (2016) 051001.

[4] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain, et al., Loihi: A neuromorphic manycore processor with on-chip learning, IEEE Micro 38 (1) (2018) 82–99.

[5] E. O. Neftci, C. Augustine, S. Paul, G. Detorakis, Event-driven random back-propagation: Enabling neuromorphic deep learning machines, Frontiers in neuroscience 11 (2017) 324.

[6] B. Rueckauer, Y. Hu, I.-A. Lungu, M. Pfeiffer, S.-C. Liu, Conversion of continuous-valued deep networks to efficient event-driven networks for image classification, Frontiers in neuroscience 11 (2017) 682.

[7] W. Gerstner, R. Kempter, J. L. van Hemmen, H. Wagner, A neuronal learning rule for sub-millisecond temporal coding, Nature 383 (6595) (1996) 76.

[8] G.-q. Bi, M.-m. Poo, Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type, Journal of Neuroscience 18 (24) (1998) 10464–10472.

[9] A. Yousefzadeh, T. Masquelier, T. Serrano-Gotarredona, B. Linares-Barranco, Hardware implementation of convolutional stdp for on-line visual feature learning, in: Circuits and Systems (ISCAS), 2017 IEEE International Symposium on, IEEE, 2017, pp. 1–4.

[10] T. Masquelier, S. J. Thorpe, Unsupervised learning of visual features through spike timing dependent plasticity, PLoS Computational Biology 3 (2) (2007) e31.

[11] Q. Yu, H. Tang, K. C. Tan, H. Li, Rapid feedforward computation by temporal encoding and learning with spiking neurons, IEEE transactions on neural networks and learning systems 24 (10) (2013) 1539–1552.

[12] J. H. Lee, T. Delbruck, M. Pfeiffer, Training deep spiking neural networks using backpropagation, Frontiers in Neuroscience 10.

[13] P. O'Connor, M. Welling, Deep spiking networks, arXiv preprint arXiv:1602.08323.

[14] J. Thiele, P. U. Diehl, M. Cook, A wake-sleep algorithm for recurrent, spiking neural networks, arXiv preprint arXiv:1703.06290.

[15] Y. Cao, Y. Chen, D. Khosla, Spiking deep convolutional neural networks for energy-efficient object recognition, International Journal of Computer Vision 113 (1) (2015) 54–66.

[16] A. Tavanaei, A. S. Maida, Bio-inspired spiking convolutional neural network using layer-wise sparse coding and STDP learning, arXiv preprint arXiv:1611.03000.

[17] P. U. Diehl, M. Cook, Unsupervised learning of digit recognition using spike-timing-dependent plasticity, Frontiers in Computational Neuroscience 9 (2015) 99.

[18] P. Merolla, J. Arthur, F. Akopyan, N. Imam, R. Manohar, D. S. Modha, A digital neurosynaptic core using embedded crossbar memory with 45pJ per spike in 45nm, in: Custom Integrated Circuits Conference (CICC), 2011 IEEE, IEEE, 2011, pp. 1–4.

[19] S. Hussain, S.-C. Liu, A. Basu, Improved margin multiclass classification using dendritic neurons with morphological learning, in: Circuits and Systems (ISCAS), 2014 IEEE International Symposium on, IEEE, 2014, pp. 2640–2643.

[20] P. O'Connor, D. Neil, S.-C. Liu, T. Delbruck, M. Pfeiffer, Real-time classification and sensor fusion with a spiking deep belief network, Frontiers in Neuroscience 7 (2013) 178.

[21] H. Mostafa, Supervised learning based on temporal coding in spiking neural networks, IEEE transactions on neural networks and learning systems.

[22] P. U. Diehl, D. Neil, J. Binas, M. Cook, S.-C. Liu, M. Pfeiffer, Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing, in: Neural Networks (IJCNN), 2015 International Joint Conference on, IEEE, 2015, pp. 1–8.

[23] Y. Wu, L. Deng, G. Li, J. Zhu, L. Shi, Spatio-temporal backpropagation for training high-performance spiking neural networks, arXiv preprint arXiv:1706.02609.

[24] T. Liu, Z. Liu, F. Lin, Y. Jin, G. Quan, W. Wen, Mt-spike: A multilayer time-based spiking neuromorphic architecture with temporal error backpropagation, in: 2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2017, pp. 450–457. doi:10.1109/ICCAD.2017.8203812.

[25] B. Zhao, R. Ding, S. Chen, B. Linares-Barranco, H. Tang, Feedforward categorization on AER motion events using cortex-like features in a spiking neural network, IEEE Transactions on Neural Networks and Learning Systems 26 (9) (2015) 1963–1978.

[26] F. Ponulak, A. Kasiński, Supervised learning in spiking neural networks with ReSuMe: sequence learning, classification, and spike shifting, Neural Computation 22 (2) (2010) 467–510.

[27] E. Neftci, S. Das, B. Pedroni, K. Kreutz-Delgado, G. Cauwenberghs, Event-driven contrastive divergence for spiking neuromorphic systems., Frontiers in Neuroscience 7 (2012) 272–272.

[28] A. Tavanaei, T. Masquelier, A. S. Maida, Acquisition of visual features through probabilistic spike-timing-dependent plasticity, in: International Joint Conference on Neural Networks (IJCNN), IEEE, 2016, pp. 307–314.

[29] J. M. Brader, W. Senn, S. Fusi, Learning real-world stimuli in a neural network with spike-driven synaptic dynamics, Neural Computation 19 (11) (2007) 2881–2912.

[30] D. Querlioz, O. Bichler, P. Dollfus, C. Gamrat, Immunity to device variations in a spiking neural network with memristive nanodevices, IEEE Transactions on Nanotechnology 12 (3) (2013) 288–295.

[31] M. Beyeler, N. D. Dutt, J. L. Krichmar, Categorization and decision-making in a neurobiologically plausible spiking network using a STDP-like learning rule, Neural Networks 48 (2013) 109–124.

[32] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, T. Masquelier, Stdp-based spiking deep convolutional neural networks for object recognition, Neural Networks 99 (2018) 56 – 67.

[33] P. Dayan, B. W. Balleine, Reward, motivation, and reinforcement learning, Neuron 36 (2) (2002) 285–298.

[34] N. D. Daw, K. Doya, The computational neurobiology of learning and reward, Current Opinion in Neurobiology 16 (2) (2006) 199–204.

[35] Y. Niv, Reinforcement learning in the brain, Journal of Mathematical Psychology 53 (3) (2009) 139–154.

[36] D. Lee, H. Seo, M. W. Jung, Neural basis of reinforcement learning and decision making, Annual Review of Neuroscience 35 (2012) 287–308.

[37] E. E. Steinberg, R. Keiflin, J. R. Boivin, I. B. Witten, K. Deisseroth, P. H. Janak, A causal link between prediction errors, dopamine neurons and learning, Nature Neuroscience 16 (7) (2013) 966–973.

[38] W. Schultz, Neuronal reward and decision signals: from theories to data, Physiological Reviews 95 (3) (2015) 853–951.

[39] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.

[40] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., Mastering the game of Go with deep neural networks and tree search, Nature 529 (7587) (2016) 484–489.

[41] R. S. Sutton, A. G. Barto, Introduction to reinforcement learning, Vol. 135, MIT Press Cambridge, 1998.

[42] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al., Mastering the game of go without human knowledge, Nature 550 (7676) (2017) 354.

[43] M. Mozafari, S. R. Kheradpisheh, T. Masquelier, A. Nowzari-Dalini, M. Ganjtabesh, First-spike based visual categorization using reward-modulated stdp, arXiv preprint arXiv:1705.09132.

[44] N. Frémaux, W. Gerstner, Neuromodulated spike-timing-dependent plasticity, and theory of three-factor learning rules, Frontiers in neural circuits 9 (2016) 85.

[45] Z. Brzosko, S. Zannone, W. Schultz, C. Clopath, O. Paulsen, Sequential neuromodulation of hebbian plasticity offers mechanism for effective reward-based navigation, eLife 6.

[46] J. Gautrais, S. Thorpe, Rate coding versus temporal order coding: a theoretical approach, Biosystems 48 (1-3) (1998) 57–65.