Mapping Spatio-temporally Encoded Patterns by Reward-Modulated STDP in Spiking Neurons

Ibrahim Ozturk and David M. Halliday

***Abstract*—In this paper, a simple structure of two-layer feed-forward spiking neural network (SNN) is developed which is trained by reward-modulated Spike Timing Dependent Plasticity (STDP). Neurons based on leaky integrate-and-fire (LIF) neuron model are trained to associate input temporal sequences with a desired output spike pattern, both consisting of multiple spikes. A biologically plausible Reward-Modulated STDP learning rule is used so that the network can efficiently converge optimal spike generation. The relative timing of pre- and postsynaptic firings can only modify synaptic weights once the reward has occurred. The history of Hebbian events are stored in the synaptic eligibility traces. STDP process are applied to all synapses with different delays. We experimentally demonstrate a benchmark with spatio-temporally encoded spike pairs. Results demonstrate successful transformations with high accuracy and quick convergence during learning cycles. Therefore, the proposed SNN architecture with modulated STDP can learn how to map temporally encoded spike trains based on Poisson processes in a stable manner.**

I. INTRODUCTION

Animals are receiving information of recognizing food or danger during interaction with the environment. Then they are taking proper actions after processing incoming signals. Mimicking this mechanism of natural neuronal processing into computational point of view is the paradigm known as artificial neural networks (ANNs). Spiking Neural Networks (SNNs) as the third generation of neural network models inspired by features found in biology are able to encode and process the information using timing of individual spikes arriving at a neuron [1].

SNNs are bio-inspired, adaptive, computationally powerful structures compared with conventional artificial neural network [2]. They process information encoded in the timing between neuron firings. As biological neurons communicate by receiving and transmitting pulses known as "spikes" to indicate its short and transient nature then SNNs also carry information across synapses between other neurons in the network.

Research supported by the Ministry of Education, Turkey and University of York, UK.

I. Ozturk is with the Intelligent Systems Research Group, Department of Electronics, The University of York, Heslington, YORK, YO10 5DD, UK, on leave from the Faculty of Engineering, Osmaniye Korkut Ata University, Osmaniye, Turkey (Email: io517@york.ac.uk)

D. M. Halliday is with the Intelligent Systems Research Group, Department of Electronics, The University of York, York YO10 5DD UK (e-mail: david.halliday@york.ac.uk)

The complexity and the computational power of artificial SNNs are restricted compared with the abilities of biological neural systems [3]. However, developing efficient learning techniques and recognizing information codes in SNNs are still open problems from a computational point of view. Therefore, various characteristics of SNNs such as learning algorithms and theoretical models could be considered to understand the ability of SNNs.

Rate coding and temporal coding are two widely used information encoding schemes in SNNs. In rate coding the number of spikes within the encoding window is considered regardless of their temporal pattern, while for the temporal coding the precise timings of spikes are considered [4]. From a biological perspective, precise spike timing of individual spikes is common for neuronal information processing [5]. Precise timing of spikes allow neurons to carry more information than random spike timings with the paradigm of rate-coding schemes; therefore, temporal synchrony might play an important role in neural computation. For the optimal efficiency of information computation, temporal synchrony paradigm has been implemented here using a simple Poisson process for generating input and output spike patterns.

The temporal order of presynaptic and postsynaptic firing times can determine synaptic potentiation (Long Term Potentiation-LTP) vs. depression (Long Term Depression-LTD). A presynaptic spike arriving before a postsynaptic spike leads to synaptic potentiation, arrival after the postsynaptic spike activity is causing depression of the connection [5]. This temporal relationship between pre- and postsynaptic activity is a paradigm derived from neurobiological experiments known as Spike Time Dependent Plasticity (STDP) which is experimentally observed in hippocampal neurons [6]. Here we use STDP with time-dependent weight change as a biologically plausible mechanism in order to map input spike patterns into output spiking activity.

It is clear that multiple-spike timings in a pair of spike trains is more suitable to empirically derived rules and is more biologically realistic. However, only a few learning rules to teach a SNN have been proposed which can tackle precise input-output mappings of spike patterns with multiple pulses on each train instead of single spikes in spatio-temporal patterns. ReSuMe [3], is the supervised learning of desired spike trains in response to input patterns. It uses a

combination of STDP plasticity and anti-STDP plasticity in a single synapse. Although, this technique is ideal in classifying input patterns by spatio temporally encoded patterns, it is not reliable and stable enough for information processing in the nervous system [7]. SpikeProp is a gradient-descent learning algorithm similar to backpropogation for rate coding. The euclidean distance between target and actual spiking activity determines the synaptic weights to minimize the error. However, applications of SpikeProp are mainly based on the timing of a single output spike [8], [9]. Although [8] claims that it could be applicable for multiple spikes, there is limited evidence. Also, [7] explained that SpikeProp failed during their preliminary experiments for multiple output spikes.

In behavioural learning paradigms, strength of a behaviour is modified by receiving reward or punishment as reinforcements. Reinforcement learning (RL) inspired by behaviorist psychology is algorithmic approach to reward learning in a machine learning discipline [10]. Recently, several plasticity experiments, including STDP, demonstrated that neuromodulators, particularly dopamine related to novelty and reward prediction, have global mechanisms for synaptic modification [11], [12]. In order to consolidate changes of synaptic strength in response to pre- and postsynaptic neuronal activity such another signal could be used. as reward [13]. Inspired by these observations, two levels of plasticities using STDP and reward have been hypothesised as Reward-modulated STDP in several theoretical and practical studies [14], [15], [16], [17], [18], [19], and [20]. Similar to them, experiments in this paper are implemented using Reward-modulated STDP mechanism.

Our motivation for the current study is that the precise times of individual spikes might be fundamental for efficient computation in capturing certain temporal dynamics. This paradigm has also been demonstrated in various Neurobiological research [6]. Therefore, the goal of current spiking neural network is to modify the vector of synaptic weights w to achieve desired Poisson spike patterns at the training neuron's output $S_d^{out}(t)$ in response to the given n input Poisson sequences $S_1^{in}(t)$, $S_2^{in}(t)$, ..., $S_n^{in}(t)$. The reward is provided by criticizing the actual and desired output neuron activities at the end of each learning cycle, which is applied into synaptic connections subject to an STDP learning process.

In this work, the reward-maximising STDP rule has been implemented [18]. Hebbian plasticity is modulated by the reward signal analogous to dopamine modulation. A transient memory of synaptic events is stored at each synapse as synaptic eligibility traces. Once the reward occurs, synaptic weights are updated. Only delayed rewards have been taken into account because of biological plausibility.
The novel aspect of this study is using multiple-spike timings not only in each input pairs also in desired output train to achieve biologically more realistic scheme. Also, a dopaminergic inspired learning rule combined with STDP mechanism

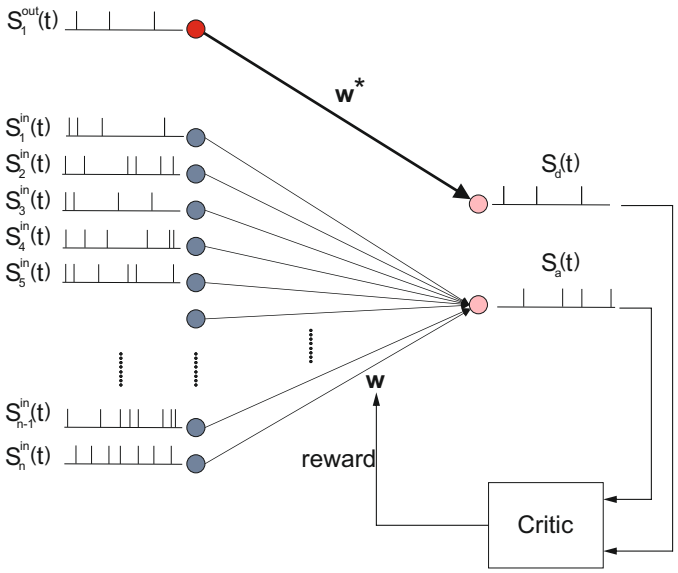

Fig. 1. **The network structure for reinforcement learning of spiking neural network.** See text for a detailed explanation of the figure and notation used.

is shown both analytically and through computer experiments with rapid convergence in less than 30 learning cycles. The development of reward-modulated STDP with its learning speed helps generic learning tasks where a neuron is supposed to respond to input spike patterns with specific output spikes.

II. METHODS AND EXPERIMENTS

A. Network Architecture

One of the common network topologies in biological systems is a feed-forward structure where the data flow from input layer to output neurons without recurrent connections. In this paper, fully connected two layered feed-forward spiking neural network architecture without hidden layer is proposed. The network structure is inspired by the actor-critic architecture [21] illustrated in Figure 1 with artificially depicted input-output neuron spiking activity over a typical simulation presentation $T_{pe} = 120ms$ in Table II.

Our implementation of the actor-critic spiking neural network contain $k = 10$ input neurons in the input layer. Each neuron in this layer is fed with different spike train pattern, $S_1^{in}(t)$, $S_2^{in}(t)$, ..., $S_k^{in}(t)$, constructed by external Poisson mechanism which is outlined in the *Encoding* section. There is also bias input neuron layer as a reference with a single neuron in order to force comparator neuron's desired spike pattern generation. In the output layer, there are two neurons, one is generating actual outputs $S_a^{out}(t)$, the other is producing desired outputs $S_d^{out}(t)$. Furthermore, a critic unit which criticizes actual cell activity and desired target cell output in order to generate appropriate reward modulation of weights between input-output layers.

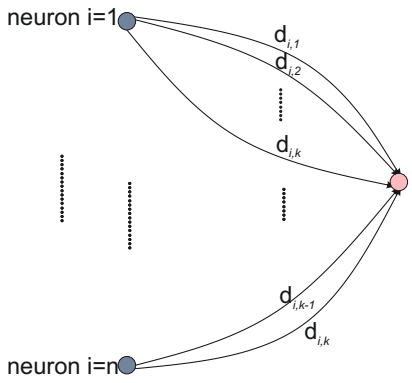

Fig. 2. **Delay mechanism.** i:index for input neuron, k is total delay connection number per input-output neuron connection, n is the total number of input layer neuron. with non-negative delays $d_{i,j} \in R^{+}$.

Each individual connection from an input layer neuron i to an output layer neuron j has a fixed number of m synaptic terminals, where each terminal serves as a sub-connection that is associated with a delay $s_{i,t}$ inspired from [22]. In the proposed network architecture, $s_{i,t}$ with $t \in [1, k]$ indicates the sub terminal where i is the preysnaptic neuron index, t is the sub-terminal index as it is drawn in Figure2. The number of synaptic connections is $m = 10$ with non-programmable delays between $1 - 10ms$ as $s_{i,t} = t(ms)$. However,the bias input neuron $n_{i=0}$ has single connection without a delay as $s_i = 0ms$. An exponential increase of input neurons and the increasing dimensionality of the spike trains can be avoided with this mechanism.

In this model, the actual output neuron is trained to respond with spatio-temporally encoded input spike train sets. The reward is calculated based on the timing difference between action potentials of the actual neuron and desired neuron. The synaptic connection w^* between input layer and desired output neuron is fixed and initially set to maximum weight value as $w^* = w_{max}$ in order to force desired neuron to generate spikes at each target time. Other weights are initialised uniformly with range $[w_{init_{min}}, w_{init_{max}}]$ in Table II. They have been chosen quite small so there will not be any spike generation until the learning perform sufficiently or adjustment of homoeostatic firing rate. Synaptic connections are allowed to contain a mix of both positive and negative weight values. Negative connection weights between the two neurons behave like inhibitory synapse. Therefore, the initial weight range has been set as %20 (inhibitory) and %80(excitatory) analogous to observations in mammalian brains.

B. Neuron Model and Synaptic Model

In order to reconstruct the predefined target spike pattern precisely on the networks output over training, Leaky Integrate-and-Fire (LIF) neuron modelling has been used as one of the most common simplified spiking neural models [1].

TABLE I
PARAMETERS FOR THE NEURON MODELS AND SYNAPSES

Parameter	Value	Unit
$Vrest$	-60	mV
$Vreset$	-55	mV
Vth	-65	mV
τ_m	10	ms
R	10	MΩ
C_m	1	nF
$t_{refract}$	0	ms

$$\frac{dV}{dt} = -\frac{1}{\tau_m}(V_m - V_{rest}) + \frac{1}{C_m}I \tag{1}$$

where V_m represents the membrane voltage, C_m is membrane capacitance, V_{rest} is resting potential, and I is the input current to the neuron. $\tau_m = R * C_m$ is the time constant during integration. Immediately after reaching a defined voltage, V_{th} the fixed voltage threshold, the potential is reset to V_{reset} as the reset potential. This process emits a spike. Also if there no input, the membrane potential decays to the resting voltage $V_m = V_{rest}$. Parameters are summarized in Table I.

Input layer neurons are dummy LIF neurons which transfer the input from presynaptic to postsynaptic spikes. Whenever those neurons have any spikes, they are directly emitting spike without integration. This is achieved by increasing V_m by $V_{th} - V_{reset}$ when a spike receives.

C. Spike Train

The activity of neuron i could be described by a sequence of times at which the neuron has emitted a spike,

$$t_i = \{t_i^{(1)}, t_i^{(2)}, t_i^{(3)}, ..., t_i^{(L)}\} \tag{2}$$

where all $t_i^{'} > 0$ and L is the number of spikes. This abstract representation of a spike sequence is referred to as a spike train. Those impulses could also be written by a sum of Dirac delta functions,

$$S_i(t) = \sum_{l=1}^{L} \delta(t - t_i^{(l)}) \tag{3}$$

with spike times t_i by using the Dirac-delta form as

$$\delta(t) = \begin{cases} 1, & \text{if} \quad t = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

D. Plasticity rules

The standard rule for STDP is used which adjusts each connection strength in response to the time difference between the spikes of pre- and postsynaptic neurons. Integral kernels as τ_{pre} and τ_{post} defines the shape of the STDP process [1]. It is modelled by

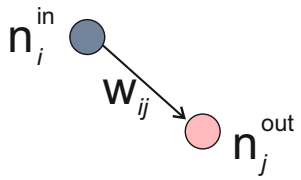

Fig. 3. **Simplified pre- and postsynaptic connection.** The graph could visualize the used notation in the paper. Single synapse s_{ij} with its strength w_{ij} between input neuron with n_i and output neuron n_j. i is the presynaptic neuron index and j is the postsynaptic index.

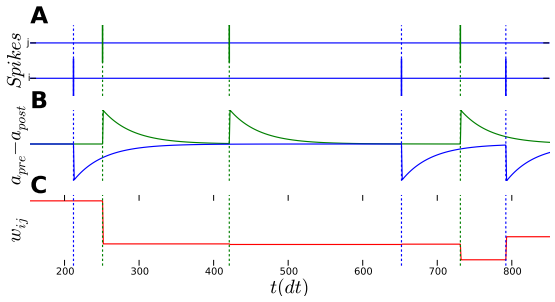

Fig. 4. **Illustration of the STDP process over simulation time steps** dt**.** A) The presynaptic firing times as the top row at t_j^f, green, and the postsynaptic spike train as the bottom row at t_i^f, blue. B) LTP in blue and LTD in green subwindows for excitatory connections can bee seen with a_{pre} and a_{post} trajectories based on presynaptic and postsynaptic spikes from (A). The synaptic change ΔW (Y-axis) determines the amount of the weight modifications. C) First weight modification as sharp decrease because of the postsynaptic spike generation shortly before the presynaptic spike. Second change scenario is slight weight decrease because of longer time difference between pre-post spiking times. Increasing weight changes is induced by the presynaptic spike followed by the postsynaptic spike nearly at the end of the presentation.

$$W(\Delta t) = \begin{cases} +A_{pre}e^{-\Delta t/\tau_{pre}}, & \text{if} \quad \Delta t \geq 0, \\ -A_{post}e^{+\Delta t/\tau_{post}}, & \text{if} \quad \Delta t < 0. \end{cases} \tag{5}$$

where $\Delta t = t_{post}^f - t_{pre}^f$ is the difference between the pre- and postsynaptic neuron firing times. τ_{pre} and $\tau_{post} > 0$ are the decay constants of the STDP learning kernel for LTP and LTD. Also A_{pre} and $A_{post} > 0$ are the starting amplitudes of decays once presynaptic and postsynaptic events happened, respectively. For the current plasticity modelling, all determined parameters has been summarized in Table II.

Although, various shapes of STDP windows have been proposed in recent literature [23], [24], [25], [26], we will use one of the most common learning window for the STDP process as illustrated in Fig 4.

E. Spike Encoding

For input neurons, a spike train $S^{in}(t)$ consists of a homogeneous Poisson spike train with constant spike probability $r_{in} = 0.4Hz$ and minimum Inter-Spike Interval $ISI = 10ms$ is created as in Table II. $S^{in}(t)$ is divided into n spike trains as $S_1^{in}(t)$, $S_2^{in}(t)$, ..., $S_n^{in}(t)$ with each spike train duration with $T_p = 100ms$. n indicates the number of neuron in the input layer. Each extracted spike train $S_1^{in}(t)$, $S_2^{in}(t)$, ..., $S_k^{in}(t)$ from $S^{in}(t)$ will feed each neuron in the input layer over each episode.

For output neuron, a spike train $S_d^{out}(t)$ consists of a homogeneous Poisson spike train with constant spike probability $r_{out} = 0.06Hz$ and minimum Inter-Spike Interval $ISI = 10ms$ is created as in Table II. The same extraction for input layer doesn't need to be applied to $S_d^{out}(t)$ because the network contains a single training output neuron as its readout. The only restriction for the output spike extraction is that those have no spiking activity in the first $2 * \tau_m = 20ms$ over $T_p = 100ms$ have been selected for simulations.

F. Spike Train Distance

One of the important metrics in order to compute dissimilarity between spike trains is the van Rossum Distance (vRD) method which is a dimensionless distance [27]. It takes into account additional and missing spikes. The distance measure maps the two spike trains as actual spike trian with S_a and desired spike train with S_d onto a profile $S_a, S_d \rightarrow R(t)$ with $0 \leq R(t) \leq 1$. The overall spike distance value can be calculated by integration as $D_R = \int R(t)dt$.

In this spike distance, the discrete spike trains s_a and s_d as in Eq 3 are transformed into continuous function

$$f(t) = e^{-t/\tau_{vRD}}H(t) \tag{6}$$

where τ_{vRD} is the time decay constant and $f(t)$ is the convolution of each spike t_i by an exponential kernel with the Heaviside step function as $H(t)$,

$$H(t) = \begin{cases} 1, & \text{if} \quad t \geq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

over the discrete convolution as

$$(f * S)(t) = \sum_{s:0 \leq t-s < T_{pe}} f(s)S(t-s), \quad 0 < t < T_{pe} \tag{8}$$

where T_{pe} is the fixed time of single presentation. Then the van Rossum distance D_R between s_a and s_d can be calculated as

$$D_R(S_a, S_d) = \sum_{0 \leq t < T_{pe}} \Big((f * S_a)(t) - (f * S_d)(t)\Big)^2 \tag{9}$$

Parameter selection : The decay constant τ_{vRD} parametrizes the metric and should be selected in order to be sensitive to temporal difference between spikes. Otherwise it will be sensitive to rate relationship of spike trains. Taking into account the ISI of input and desired output trains, we use τ_{vRD} = 10ms, in Table II.

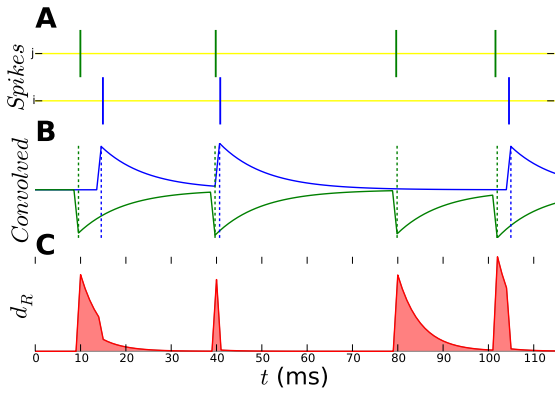

Fig. 5. **Illustration of van Rossum Distance measure.** A) Two example spike train have been illustrated. Top one (shown as green coloured with spike train index 2) is desired spike activity and the following train (shown as blue coloured with spike train index 1) is the actual spikes over a period. B) Definition of the van Rossum Distance : Above two spike trains (actual train flipped) are convolved with Heaviside function with time constant t_c. C) The van Rossum difference has been measured by the squared of differed spike trains. Then, the spike train distance is determined by the bounded area under the curve by the integration.

In order to remove the dependence of the vRD on the number of desired spikes, we used a normalized version of the measure D_R which has been calculated by

$$D_R^{norm} = D_R / D_R^d \tag{10}$$

where normalized vRD, current vRD and vRD of desired spike train, are respectively, D_R^{norm}, D_R and D_R^d. The measure is illustrated in Figure 5.

G. Reward Mechanism

The normalized measure D_R^{norm} between the actual and desired spike trains for the output neuron as a function of time t in the current presentation is mapped into a reward signal r [14] as

$$r = exp(-\alpha D_R^{norm}) \tag{11}$$

where $D_R^{norm} \in [0, \infty]$ and $r \in (0, 1]$. Also, reward mapping factor α is setted into 3 in order to ignore distances $D_R^{norm} > 1$ as in Table II.

$$r = \begin{cases} 0, & \text{if no output spike,} \\ 1, & \text{if } D_R^{norm} = 0, \\ exp(-\alpha D_R^{norm}), & \text{otherwise.} \end{cases} \tag{12}$$

$r = 1$ indicates a perfect match between actual and desired spike train. We explicitly set $r = 0$ if there is not any output spike generated in order to avoid network stagnation.

A Temporal-Difference (TD) technique tracks discounted reward as it is defined in Reinforcement Learning [10]. However, an adapted version of TD rule will be applied in order to track average reward as proposed in [14].

$$\delta_R(n) = r_n^{curr} - r_n^{avg} \tag{13}$$

δ_R is the difference between the current reward received r_n^{curr} and the expected reward r_n^{avg} averaged over previous trials is used to improve current policy on synaptic strengths. Maximizing the average reward per presentation will maximize the total future reward in order to achieve an optimal policy. The moving average of reward over presentations is calculated as,

$$r_n^{avg} = \gamma r_{n-1}^{avg}(t) + (1 - \gamma) r_n^{curr} \tag{14}$$

where n indicates the presentation number. Here, all future rewards are discounted by discount factor, γ where $0 < \gamma < 1$. It is selected near to highest value to express sooner rewards have higher utility than older rewards as $\gamma = \frac{9}{10}$ and $(1-\gamma) = \frac{1}{10}$ as in [14], [10]. This selection for γ also helps the learning algorithm converge.

H. Learning Rule

The learning task is to map the timings of input and target output spike patterns precisely. Main learning phenomena relied on here is the framework of modulated STDP, which modulates the outcome of STDP by a neuromodulator. Dopamine (DA) is the neuromodulator often associated with a reward signal. Synaptic eligibility traces based on theoretical considerations store a temporary memory of the past Hebbian coincidence events until a delayed reward is received. Inspired from the essence of DA modulation of STDP in [16],

$$c_{ij}(t) = STDP(\Delta t)\delta(t - t_{pre/post}) \tag{15}$$

where $\delta(t)$ is the Dirac delta function that step-increases the eligibility c_{ij}. Firing times of pre- and postsynaptic neurons, occurring at times $t_{pre/post}$ respectively, change c_{ij} by the amount of $STDP(\Delta t)$ with Δt = t_{post} - t_{pre}.

Moving average of eligibility is used by filtering with an exponential function called synaptic eligibility trace $C_{ij}(T_{pe})$ of the synapse from neuron j to i is defined by

$$\tau_c \frac{dC_{ij}(t)}{dt} = c_{ij}(t) - C_{ij}(t) \tag{16}$$

where τ_c decay time constant of the eligibility trace. τ_c determines the maximal interval between the pre-post coincidences and the reward signal. τ_c is the same as the period of input pattern in our simulations.

The actual synaptic modification also requires the presence of a neuromodulatory signal which was derived previously as δ_R in Eq 13. Therefore, the multiplication of eligibility trace with with reward defines the amplification formula of STDP weights as

$$\Delta w_{ij}(n) = \eta \delta_R(n) C_{ij}(T_{pe}) \tag{17}$$

where $\Delta w_{ij}(n)$ is the weight change, α is the learning rate, $C_{ij}(T_{pe})$ is the eligibility trace over the n^{th} presentation with time $t = T_{pe}$. This learning paradigm was applied into all synapses between input neurons and actual output neuron after each presentation. The modification Δw_{ij} has been applied to

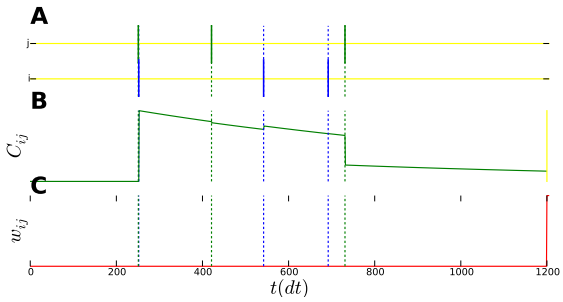

Fig. 6. **Eligibility Trace and Weight Update during a Single Learning Cycle.** A) The presynaptic firing times as the top row at t_j^f with green coloured and the postsynaptic spike train as the bottom row at t_i^f with blue coloured. B) The eligibility trace c_{ij} with green coloured keeps past Hebbian events from presynaptic and postsynaptic spikes in (A). At the end of the learning cycle $T_{pe} = 120ms$, the current reward is receiving which is drawn in a yellow colour. C) Weight change has been only applied once the reward recieved at the end of the presentation.

a weight w according to an additive update rule as $w_{ij}(n+1) \leftarrow w_{ij}(n) + \Delta w_{ij}(n)$ [28]. Instead of feeding the network by the reward at every time moment, we apply it only at the end of each learning cycle as depicted in Fig 6.

In order to avoid either silencing or extreme network activity by frequent firings of neurons, all synaptic weights are bounded with a lower boundary w_{min} and an upper boundary w_{max}.

I. Homeostatic firing rate and Scaling rules

Network must maintain its stable functionality in the face of synaptic strength changes during training. This stabilization of neuronal functions is handled by homeostatic plasticity mechanisms. One such mechanism is synaptic activity-dependent scaling [29] which allows neurons to regulate their overall firing rate without touching the stability of the trained weight distribution. Simplified version of this synaptic scaling [30] is applied into the current network as

$$\frac{dw_{ij}(t)}{dt} = \beta w_{ij}(t)[N_{des} - N_{act}] \tag{18}$$

where w_{ij}indicated the synaptic weight from pre-synaptic neuron *i* to post-synaptic neuron *j* as in Fig3, β is constant determining the strength of the synaptic scaling. Post-synaptic activity as generated spike number from the desired neuron and actual neuron are N_{des}, N_{act}, respectively.

Activity-dependent scaling of synaptic weights preserve stability of the network by preventing lower and over-activation levels from all synapses during training runs.

J. Coincidence factor

The coincidence factor Γ as described by [31] between two spike trains determines the rate of similarity/dissimilarity between spike trains. The bound limits of this correlation measure is [-1, 1] where 1 is exactly the same, 0 is not correlated, and -1 is perfectly anti-correlated. It is formulated as

$$\Gamma = \frac{N_{coinc} - E\left(N_{coinc}\right)}{\frac{1}{2}\left(N_{des} + N_{act}\right) - E\left(N_{coinc}\right)} \tag{19}$$

where N_{des} are the number of spikes in the desired train as a reference, N_{act} is the number of spikes in the actual output as a comparing train, N_{coinc} is the number of coincident spikes within a time window $\Delta, E\left(N_{coinc}\right) = 2r_{out}\Delta N_{ref}$ is the expected number of coincident spikes generated by a homogeneous Poisson process with its rate r_{out} in Table II.

TABLE II
MODEL PARAMETERS USED FOR THE COMPUTER SIMULATIONS

Parameter Type	Parameter Names	Values
Plasticity Window	A_{pre}, A_{post}	0.005, 0.005
	τ_{pre}, τ_{post}	10ms, 10ms
Scaling Factor	β	0.001
Weight limits	$[w_{min}, w_{max}]$	[-3, +3]
	$[w_{init_{min}}, w_{init_{max}}]$	[-0.02, 0.08]
Input-Output Firing Rate	r_{in}, r_{out}	0.4Hz, 0.06Hz
Spike Train Length	T_p	100ms
Presentation Time	T_{pe}	120ms
Run Time Resolution	dt	0.1ms
Input Layer Neuron Number	n	20
Synaptic Terminal Number	m	10
Eligibility Decay	τ_c	10ms
vRD Decay	τ_{RD}	10ms
Learning Rate	η	500
Reward Mapping Factor	α	3
Minimum Inter-Spike Interval	ISI	10ms

III. RESULTS

To simulate spiking neural networks in software, Brian package [32] based on Python language has been used. Brian simulation tool allows to define flexible models by writing readable textual descriptions based on mathematical expressions [33]. The Python code for all simulations including neurons, synapses and network models based on Brian architecture is also available on the authors web page www.ozturkibrahim.com.

The magnitude and direction of synaptic update is determined by the relative timing of pre- and postsynaptic firings as STDP. However, one of the common problems for Supervised Hebbian paradigms: Synaptic weights continue to adjust their parameters although the desired spike timings have been achieved by applying global modulator over the reward signal. This issue has been eliminated and stable solutions are experimentally demonstrated.

All simulations are carried out using a network of two-layer with 20 input neurons, 1 bias input neuron, 1 desired output neuron and 1 training output neuron. For each simulation, both input data as a set of $n = 20$ different spike trains and

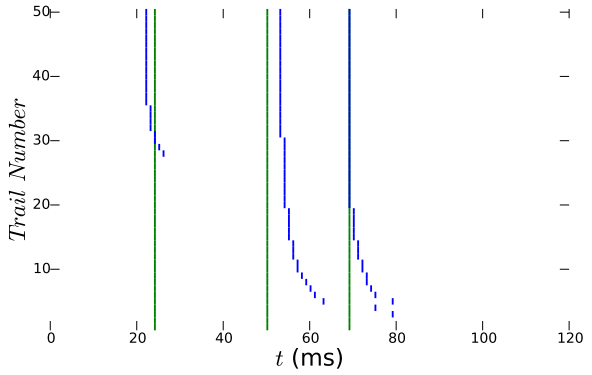

Fig. 7. **Reconstruction of the transformation from input patterns to output spike timings.** The current network is trained to map a spatio temporally encoded input spike trains into another spatio temporally encoded output pattern. t_{out}^{act} with blue colour converges to t_{out}^{des} with green colour.

the desired spike pattern for output have been independently prepared with the duration of each $T_p = 100ms$. Each run has different random weight initialization. Each presentation as learning cycle runs with $T_{pe} = 120ms$ with simulation time resolution $dt = 0.1ms$.

A desired output signal $S_d^{out}(t)$ and the convergence of the actual output $S_a^{out}(t)$ over training cycles are depicted in Fig 7. In this simulation, the desired pattern has 3 spikes. Early trials had less spiking activity or no activity in the network output. Once the training continues the number of generated spikes at the output of network becoming closer to desired activity. After less than 30 episodes the network produce 3 desired spikes within 3 ms of target. With this configuration, it could be clearly seen that the network reliably learned the given pattern within 25-35 cycles with $\eta = 500$ as in Table II.

The learning trajectory for Reward Modulated STDP framework is shown in Figure 7 over a single simulation. It demonstrates that actual and desired spike trains eventually converge to the desired spike train by decreasing the mismatch of firing times between them. Furthermore, the stability preservation of the network by keeping frequencies of neuronal activity in a range could be seen over the entire simulation.

Convergence speed and error trajectory can be seen over 4 different simulations in Figure 8. Learning curves for various simulations are very similar, four are demonstrated here. The figure is clipped after 50 learning cycles because the convergence behaviour and error measure does not change after that. The depicted number of trial number as 50 is sufficiently enough to learn input spike patterns in a stable manner. The trend of averaged reward, applied during the learning, has smoother convergence compared to current reward which is not directly applied. The level of reward signal is converging its maximum value of 0.85 during learning cycles. The distance between desired and actual activity are converging lower

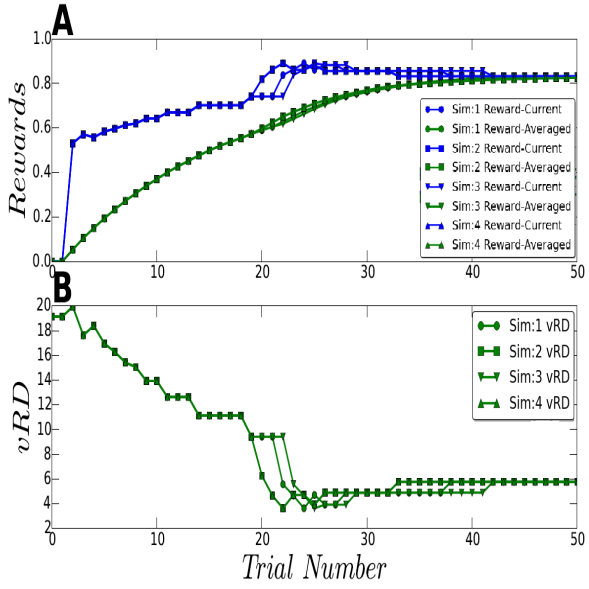

Fig. 8. **The trajectory of current and averaged reward versus vRD(van Rossum Distance) during 4 randomly selected simulations.** A)A snapshot of averaged rewards with running average of current rewards. B) The evolution of mismatch between the desired and the actual output signal, $D_R(S_a, S_d)$ based on van Rossum Distance. For each simulation, the input/output firing patterns are different.

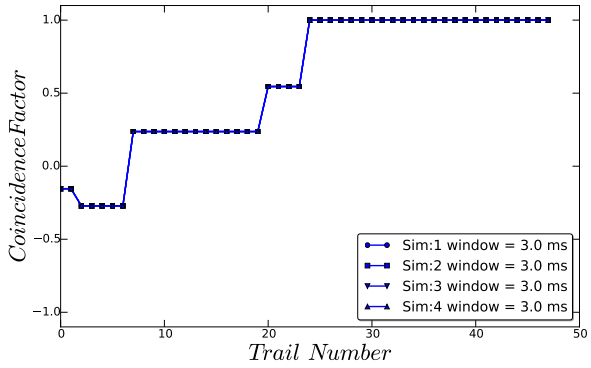

Fig. 9. **Illustration of Coincidence Factor.** Coincidence factor for four different simulations have quite similar behaviour with windowing $\Delta = 3ms$. See text for a detailed explanation of the figure.

values. Reward has inverse relation with van Rossum distance as it is depicted in Figure 8.

To characterise learning performance, coincidence factor has been used. The coincidence factor over 4 different simulations is illustrated in Figure 9. For each presentation, desired spike train is compared with generated actual spike pattern with the value of windowing 3ms. This indicates that all individual spikes generated within the $[t_d^f, t_d^f + 3]ms$ of the desired spike times t_d^f are accepted as correctly fired at t_d^f. After 25 learning cycles, Γ achieves its maximum value as 1 which shows desired and trained activity perfectly correlated and entire spike pattern with the 3 ms window is generated precisely.

IV. Conclusion And Future Work

One of main aspects in this work is becoming closer to biologically plausible approaches with reinforcement learning techniques. We derived a learning framework for feed-forward spiking neural networks by Reward modulated STDP. Reward modulated STDP is used which relies on both Hebbian plasticity modulated by reward and synaptic eligibility traces as transient memory of past Hebbian events in each individual synapses. The reward signal for each episode is derived from the between the outputs of the actual neuron and desired spiking times.

In this study, we proposed a two-layer SNN with multiple delays between the neurons. Using delayed synapses requires less spiking neurons and less set of input patterns than direct connection without delays. Experiment results demonstrated its learning capability and performance with high accuracy for temporal sequences of spikes. Furthermore, it can been seen that such reward modifications of the STDP do indeed significantly speed up convergence compared with other gradient descent techniques such as ReSuMe. The introduced learning mechanism is able to learn to map input patterns into output pattern with multiple timings in less than 50 learning cycles which compares favourably with ReSuMe which needs around 1000 training steps [7].

Future work will investigate learning of multiple pairs of input-output spike patterns, which could be performed in different learning tasks such as classification or real life task of navigation problem. Those could give chances to mapping from the patterns of neural activity to events and actions from real tasks. Furthermore, a non-linear problem such as the XOR problem with taking into account hidden units in multi-layered feed-forward architecture, and navigation task in a maze will be performed.

References

[1] W. Gerstner and W. M. Kistler, *Spiking Neuron Models : Single Neurons, Populations, Plasticity*, 1st ed. Cambridge: Cambridge University Press, 2002.

[2] W. Maass, "Networks of spiking neurons: The third generation of neural network models," *NeuralNetworks*, vol. 10, no. 9, pp. 1659–1671, 1996.

[3] A. Kasinski and F. Ponulak, "Comparison of supervised learning methods for spike time coding in spiking neural networks," *Int. J. Appl. Math. Comput. Sci*, vol. 16, no. 1, pp. 101–113, 2006.

[4] S. Panzeri, N. Brunel, N. K. Logothetis, and C. Kayser, "Sensory neural codes using multiplexed temporal scales," *Trends Neurosci.*, vol. 33, no. 3, pp. 111–120, Mar 2010.

[5] S. M. Bohte, "The evidence for neural information processing with precise spike-times: A survey," *Natural Computing*, vol. 3, no. 2, pp. 195–206, 2004.

[6] G. Q. Bi and M. M. Poo, "Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type," *J. Neurosci.*, vol. 18, no. 24, pp. 10 464–10 472, Dec 1998.

[7] A. Gruning and I. Sporea, "Supervised learning of logical operations in layered spiking neural networks with spike train encoding," *Neural Processing Letters*, vol. 36, no. 2, pp. 117–134, Oct 2012.

[8] O. Booij and H. Nguyen, "A gradient descent rule for spiking neurons emitting multiple spikes," *Information Processing Letters*, vol. 95, no. 6, pp. 552–558, Sep 2005.

[9] B. Schrauwen and J. V. Campenhout, "Extending spikeprop," *Proceedings of the International Joint Conference on Neural Networks*, vol. 1, pp. 471–476, Jul 2004.

[10] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 1st ed. Cambridge MA: MIT Press, 1998.

[11] V. Pawlak and J. N. Kerr, "Dopamine receptor activation is required for corticostriatal spike-timing-dependent plasticity," *J. Neurosci.*, vol. 28, no. 10, pp. 2435–2446, Mar 2008.

[12] J. R. Wickens, "Synaptic plasticity in the basal ganglia," *Behav Brain Res.*, vol. 199, no. 1, pp. 119–128, Apr 2009.

[13] C. H. Bailey, M. Giustetto, Y. Y. Huang, R. D. Hawkins, and E. R. Kandel, "Is heterosynaptic modulation essential for stabilizing Hebbian plasticity and memory?" *Nat. Rev. Neurosci.*, vol. 1, no. 1, pp. 11–20, Oct 2000.

[14] M. A. Farries and A. L. Fairhall, "Reinforcement learning with modulated spike timing dependent synaptic plasticity," *J. Neurophysiol.*, vol. 98, no. 6, pp. 3648–3665, Dec 2007.

[15] R. V. Florian, "Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity," *Neural Comput*, vol. 19, no. 6, pp. 1468–1502, Jun 2007.

[16] E. M. Izhikevich, "Solving the distal reward problem through linkage of STDP and dopamine signaling," *Cereb. Cortex*, vol. 17, no. 10, pp. 2443–2452, Oct 2007.

[17] R. Legenstein, D. Pecevski, and W. Maass, "A learning theory for reward-modulated spike-timing-dependent plasticity with application to biofeedback," *PLoS Comput. Biol.*, vol. 4, no. 10, p. e1000180, Oct 2008.

[18] N. Fremaux, H. Sprekeler, and W. Gerstner, "Functional requirements for reward-modulated spike-timing-dependent plasticity," *The Journal of Neuroscience*, vol. 30(40), pp. 13 326–13 337, 2010.

[19] ——, "Reinforcement learning using a continuous time actor-critic framework with spiking neurons," *PLoS Comput. Biol.*, vol. 9, no. 4, p. e1003024, Apr 2013.

[20] N. Fremaux and W. Gerstner, "Neuromodulated Spike-Timing-Dependent Plasticity, and Theory of Three-Factor Learning Rules," *Front Neural Circuits*, vol. 9, p. 85, 2015.

[21] W. Potjans, A. Morrison, and M. Diesmann, "A spiking neural network model of an actor-critic learning agent," *Neural Computation*, vol. 21, no. 2, pp. 301–339, Feb 2009.

[22] S. M. Bohte, E. M. Bohte, H. L. Poutr, and J. N. Kok, "Unsupervised clustering with spiking neurons by sparse temporal coding and multi-layer rbf networks," *IEEE Trans. Neural Netw*, vol. 13, pp. 426–435, 2002.

[23] S. Song, K. Miller, and L. Abbott, "Competitive hebbian learning through spike-timing-dependent synaptic plasticity," *Nat Neurosci*, vol. 3, No. 9, pp. 919–926, 2000.

[24] W. Senn, H. Markram, and M. Tsodyks, "An algorithm for modifying neurotransmitter release probability based on pre- and postsynaptic spike timing," *Neural Comput*, vol. 13, no. 1, pp. 35–67, Jan 2001.

[25] E. M. Izhikevich, J. A. Gally, and G. M. Edelman, "Spike-timing dynamics of neuronal groups," *Cereb. Cortex*, vol. 14, no. 8, pp. 933–944, Aug 2004.

[26] W. M. Kistler, "Spike-timing dependent synaptic plasticity: a phenomenological framework," *Biol Cybern*, vol. 87, no. 5-6, pp. 416–427, Dec 2002.

[27] M. C. W. van Rossum, "A novel spike distance," *Neural Computation 13*, vol. 20, pp. 751–763, Nov 2001.

[28] A. Morrison, M. Diesmann, and W. Gerstner, "Phenomenological models of synaptic plasticity based on spike timing," *Biol Cybern*, vol. 98, no. 6, pp. 459–478, Jun 2008.

[29] G. G. Turrigiano, "Homeostatic plasticity in neuronal networks: the more things change, the more they stay the same," *Trends Neurosci.*, vol. 22, no. 5, pp. 221–227, May 1999.

[30] M. C. van Rossum, G. Q. Bi, and G. G. Turrigiano, "Stable Hebbian learning from spike timing-dependent plasticity," *J. Neurosci.*, vol. 20, no. 23, pp. 8812–8821, Dec 2000.

[31] W. M. Kistler, W. Gerstner, and J. L. v. Hemmen, "Reduction of the hodgkin-huxley equations to a single-variable threshold model," *Neural Computation*, vol. 9, pp. 1015–1045, 1997.

[32] D. Goodman and R. Brette, "Brian: a simulator for spiking neural networks in python," *Front Neuroinform*, vol. 2, p. 5, 2008.

[33] M. Stimberg, D. F. Goodman, V. Benichoux, and R. Brette, "Equation-oriented specification of neural models for simulations," *Front Neuroinform*, vol. 8, p. 6, 2014.