Wide Learning

Using an Ensemble of Biologically-Plausible Spiking Neural Networks for Unsupervised Parallel Classification of Spatio-Temporal Patterns

Katarzyna Kozdon
Department of Computer Science
University College London
London, United Kingdom
k.kozdon@cs.ucl.ac.uk

Peter Bentley
Department of Computer Science
University College London
London, United Kingdom
p.bentley@cs.ucl.ac.uk

***Abstract*— Spiking neural networks have been previously used to perform tasks such as object recognition without supervision. One of the concerns relating to the spiking neural networks is their speed of operation and the number of iterations necessary to train and use the network. Here, we propose a biologically plausible model of a spiking neural network which is used in multiple, separately trained copies to process subsets of data in parallel. This ensemble of networks is tested by applying it to the task of unsupervised classification of spatio-temporal patterns. Results show that despite different starting weights and independent training, the networks produce highly similar spiking patterns in response to the same class of inputs, enabling classification with fast training time.**

Keywords—spiking neural network; spike timing dependent plasticity; pattern detection; parallel computing; ensemble network

I. Introduction

Spiking neural networks (SNNs) are the third generation of neural networks (NNs) [1]. SNNs were inspired by information processing in biological neurons: the predominantly binary action potentials (APs) which propagate the signal across the synapses, and the analogue membrane potentials of individual neurons, which conveys the information about the quantity and timing of incoming APs. The APs can be described by a function with a spike-like appearance, where incoming signals accumulate until the membrane potential reaches the firing threshold leading to a rapid and brief further increase of the membrane potential, followed by a decrease below the resting potential and the subsequent restoration of the resting membrane potential.

Two standard information encoding paradigms for SNNs are rate coding and temporal coding. In the former, information is encoded as the number of APs within a certain time window, whereas the latter relies on the exact timing of each AP. Recent biological findings suggest that the brain uses the temporal spiking rule [2], [3].

The SNNs are more computationally expensive than traditional NNs, but aspire to mimic the activity of biological neurons more closely than traditional NNs and to take advantage of the information contained in the temporal encoding of the signals. However, while SNNs mimic some aspects of information processing in the brain, they fail to mimic the parallelisation of information processing. One of the key reasons for this is the nature of the hardware commonly used in computing [4]. While there is much research focused on tackling this issue in hardware [5], [6], in this paper we propose a software-based model of an ensemble of unsupervised SNN for parallel, distributed processing of spatio-temporal data.

Section II describes biological inspiration of our model, similar existing models and their limitations. Section III contains the technical description of our model, and section IV provides the experimental set-up used. Results are described in section V, followed by conclusions in section VI.

II. Background

The most common way of updating the weights in SNNs is the widely-understood spike-timing-dependent plasticity (STDP). STDP algorithms were inspired by the Hebbian learning in the brain, commonly referred to as “fire together, wire together” [7]–[9]. According to this hypothesis, the timing of APs in a pair of neurons determines the strengthening and weakening of the synapse between these neurons. According to the experimental data [8], if the presynaptic action potential took place up to 20 ms before the postsynaptic one, the connection between the neurons strengthens and this process is referred to as long term potentiation (LTP). Conversely, if the postsynaptic neuron fired up to 20 ms before the presynaptic one, the connection weakens (long term depression, LTD). It has been reported that the strength of the connection between a pair of neurons that fire together increases by 20-38% [9].

Slightly different rules for updating synaptic strengths were observed when inhibitory neurons are involved. No LTD and LTP were observed if the postsynaptic neuron was inhibitory [8]. However, when the presynaptic neuron was inhibitory, the connection between a pair of neurons firing within 10 ms from each other strengthened irrespectively of the order in which they fired [10].

SNNs with STDP have previously been used to classify static images [11]–[13] and movement [14]–[16].

It has been reported that increasing the width of the network increases the performance in convolutional networks [17]. In case of unsupervised SNNs, wide modular voter ensembles where each of the networks is shown identical input and contributes towards the ensemble's output were reported to outperform individual classifiers [18].

III. Neuron model with spike-timing-dependent plastic synapses

In this section, we describe our neuronal model, the differences between excitatory and inhibitory neurons in our model, and STDP algorithm used to update the weights of the synapses. Our neural network was written in C.

The biological inspiration for our model comes from the brain's association cortex, which, amongst others, merges information from the primary sensory cortical areas receiving unimodal inputs such as vision and hearing, to enable tasks to be performed such as object recognition and naming [19].

A. Integrate and fire neuron model

We use the integrate and fire neuron (IF) model [20], which has been previously used to develop computationally-inexpensive models of the cortex [21], [22].

The membrane potential of each neuron is calculated as

$$v = v + (I * \mathrm{R}) \tag{1}$$

where v is the membrane potential (mV), I is injected current (mA), R is membrane resistance constant (10 mΩ).

B. Signal processing

The resting membrane potential is defined as -70 mV. When inputs accumulate and the firing threshold of -55 mV is reached, the membrane potential is reset to -75 mV. Additionally, all downstream targets of the neuron have their input current updated according to

$$I = I + w * \mathrm{d} * S \tag{2}$$

where I is the input current (mA), w is synaptic weight, d is a constant defining the size of a single input which, after preliminary experiments, is set to 0.006. S describes if the input came from an excitatory or inhibitory synapse and is equal to 1 or -1 respectively, thus defining if the change in current – and subsequently the membrane potential - is positive or negative. In our model, we specifically define if a neuron is inhibitory or excitatory. This is in contrast to the commonly used backpropagation learning algorithm [23] which leads to the development of neurons containing both inhibitory and excitatory synapses, which is biologically implausible. In the nervous system, once the fate of a biological neuron is established during embryogenesis, it remains either excitatory or inhibitory, and it shifts the membrane potential of all its target neurons in the same direction, albeit with different strength [24].

After running preliminary experiments, we determined the optimal percentage of inhibitory neurons to be 15%, which is consistent with the 15-20% reported in the cortex [25], [26].

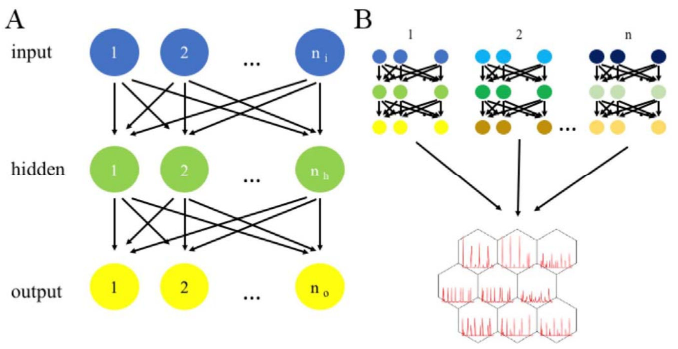

Fig. 1. Network's architecture A) Each network in the ensemble shares the same architecture and is composed of three fully connected layers. B) The networks are used in combination as an ensemble of individual networks processing separate problems in parallel. The activity of the networks is interpreted using a self organising map.

After all neurons in one layer have been iterated over and the input currents of their target neurons in the subsequent layer are updated, the same process is repeated in the next layer.

C. Spike-timing dependent plasticity

The STDP rules have been defined separately for excitatory and inhibitory neurons, in accordance with the published biological observations [8], [10] .

When a neuron and its target fire within the same iteration, the synaptic weight is doubled, and if the target neuron fires one iteration after the presynaptic one, their synapse is strengthened by 10%. Conversely, if the postsynaptic neuron fires but presynaptic does not, the synaptic weight between them is decreased by 70%.

However, for pairs of neurons where the presynaptic neuron is inhibitory and postsynaptic excitatory, if the target neuron fires during the same iteration, an iteration before or after the inhibitory neuron, the connection between them is strengthened by 50%.

Synaptic weights are initiated with random values between 0 and 2, and are clamped between 0 and 4.

D. Network architecture

We developed a three layer, feed-forward network (Fig. 1A). The network is fully connected, although it is possible for the synaptic weights to decrease to 0 during training thus effectively silencing some of the synapses.

E. Using self organizing maps to interpret network activity

The activity of the networks is interpreted using self organising maps (SOMs) (Fig. 1B). Unlike classical ensembles in which the activity of their components is averaged in some form [27], each of our networks is given its own separate task to perform thus creating a "wide" neural network where the width is created by the multiple networks.

The output of the network is interpreted using SOMs using the Kohonen package for R [28] at two levels of abstraction: as the sum of all APs at a given time point (collective) and as binary

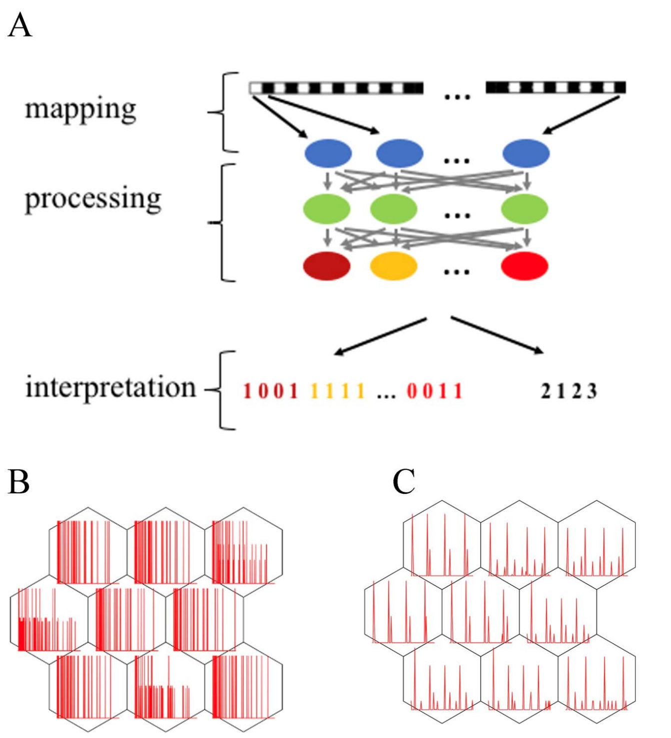

Fig. 2. A) Vectorised inputs (seen in Fig. 3) are mapped onto the input layer. Data is processed by the network and then interpreted based on either the activity of each output neuron (individual vector, left arrow), where vectors indicating the timing of APs for each neuron are concatenated, or based on the overall activity of the network (collective vector, right arrow), where vectors for all individual output neurons are added together. Following this, SOMs are used to cluster spiking patterns of the networks in response to different stimuli. B) An example of codebook vectors for individual neurons C) and corresponding vectors for the network.

vectors tracking each output neuron separately throughout the time (individual) (Fig. 2).

Correctness of pattern classification (precision) is scored with the same equation for both levels of abstraction as follows

$$\text{score} = \sum_{k=0}^{nn} \frac{i}{total} * \frac{i\_total}{total} \tag{3}$$

where *nn* is the number of nodes in the SOM, *i* is the sum of the instances of the most common pattern in the node, *i_total* is the total number of patterns assigned to the node, and total is the total number of patterns in the whole SOM.

IV. Experiments

In order to assess the effectiveness of this "wide learning" approach, a series of experiments were performed. Several input datasets with different spatio-temporal properties, and different levels of noise were provided to the networks.

Table 1 lists all parameter values used during the experiments. In all experiments, we used an ensemble of ten separately trained networks. The SOMs used to interpret their activity were divided into 9 nodes; learning parameters were kept constant throughout the study. Two types of input with four subtypes each were used. Ten separate trials were performed for each experiment.

TABLE I. Parameter values

Neural networks		
Percent of inhibitory neurons	15 %	
Number of input neurons (n_i)	500	
Number of hidden neurons (n_h)	500	
Number of output neurons (n_o)	10	
Number of networks in the ensemble (n)	10	
Injected current, input layer	15	
Discharge size (when stripes used as input)	0.006 mA	
Discharge size (when shapes used as input)	0.06 mA	
Resting potential	-70 mV	
Reset potential	75 mV	
Firing threshold	55 mV	
Resistance (R)	10 mΩ	
Iterations per training round	5	
Iterations per testing round	100	
SOM		
Number of nodes (nn)	9	
	Collective score	Individual score
Training iterations	200	400
alpha	0.05, 0.01	0.5, 0.005

A. Experiment 1 - Classification of spatio-temporal patterns

The objective of the first experiment was to test if the output of the network had discrete properties when a different subtype of a stimuli was presented, if the activity patterns of individual ensemble member networks would cluster by the input or by the network, and how the performance of the networks changed during training.

Ten networks, each initialised with different random synaptic weights, were trained with moving stripes as an input. Stripes were designed to mimic visual neuroscience experiments in which rats watch a screen with moving stripes [23] (Fig. 3). Regular horizontal and vertical stripes moved up, down, left or right with the speed of one row or column per iteration (where each iteration was equivalent to a 1 ms time step). The order in which the stripes were presented during training was random and different for each network. The width of the black stripes was one and of the white stripes two pixels. The "screen" was a 32 pixel-wide bitmap.

The stripe matrices were vectorised and first 500 pixels were mapped onto the 500 input layer neurons, with pixel one mapped to neuron one etc. (Because of the regularity of the input, the same pattern is represented by the first 500 pixels as in the remaining 524, so it is not necessary to provide the full 1024 pixels.) “Black” pixels always caused the corresponding neurons to fire and “white” pixels did not change the neurons’ state; the values of the current injected into the neurons were 15 and 0 mA respectively.

Stripes moving in one of the four directions (Fig. 3A) were presented in random order and each pattern subtype was presented in a random order to each network for 5 iterations. The training outcome was tested after 100, 200 and 400 patterns. During testing phase, each of the separately trained networks was shown each of the stripe subtypes for 100 iterations. Inputs shown to each network were identical. The networks’ activity was then interpreted at the individual and collective level by using SOMs to cluster spiking patterns obtained from the networks in response to different stimuli. The SOMs’ training parameters were 200 iterations and alpha = 0.05, 0.01 for the analysis of the collective activity of the output layer, and 400 iterations and alpha = 0.5, 0.005 for individual activity.

B. *Experiment 2 - Clasification of novel inputs*

The second experiment tested the ability of the networks to generalise, by using stripes of previously unseen width - the width of black, white or both stripes was changed and ranged between one and five pixels. Throughout this paper we use the notation s*x* b*y*, where *x* is the stripe width and *y* is the background stripe width. The only previously seen set up s1 b2 was treated as the baseline response. The experiments were run as previously.

C. *Experiment 3 - Classification of noisy inputs*

In order to test how the networks perform in response to noisy data, in the third experiment we replaced 0, 1, 5, 10, 25, 50, 75, 90 and 100% percent of randomly chosen inputs with random values between 0 and 15 mA. While the level of noise was kept constant throughout the test, the exact pattern of noise was individually generated for each network and each frame. The networks were shown each pattern for 100 iterations and then assessed on their ability to correctly cluster patterns containing noise as well as how closely this clustering resembled clustering of noise-free data.

D. *Experiment 4 – Classification of incomplete inputs*

In the fourth experiment, the ability of networks to recognise patterns when some of the data is missing was tested by presenting standard stripe patterns while silencing 0, 1, 3, 5, 10, 25, 50, 75 and 90 % of input neurons. Higher percentage of missing data was not tested because our model does not exhibit spontaneous activity and the output layer is silent when few or no inputs are present. The percentage of missing data was kept constant throughout each test but the exact pattern of silencing was randomly generated for each network and each frame.

E. *Experiment 5 – Classification of merged patterns*

As real world data, such as audio data, often is a result of interference between multiple patterns, in the fifth experiment we combined inputs into pairs: one standard full-strength input was superimposed over another standard input of varying strength (Fig. 3B). Superimposed input pairs used were vertical stripes moving left and horizontal stripes moving upwards, and

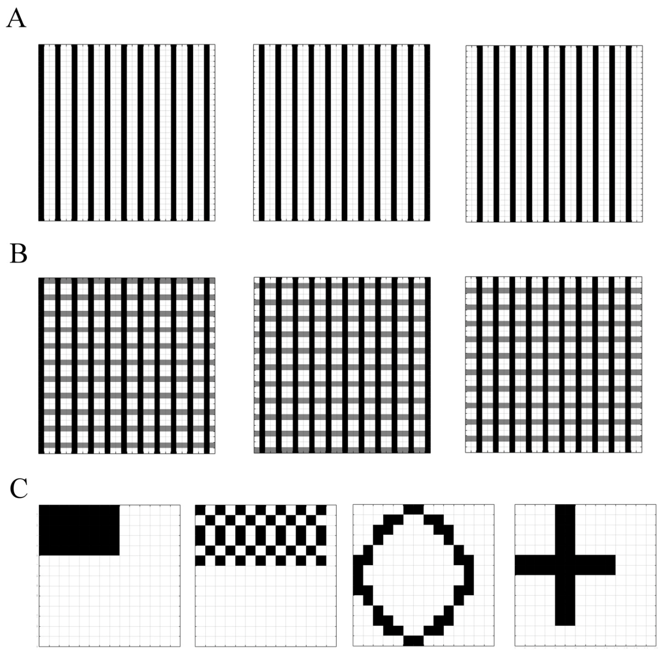

Fig. 3. Types of data used. We mapped moving visual patterns onto the neurons in the input layer. The first pattern type was vertical stripes moving left or right and horizontal stripes moving up or down. Modified versions of these inputs were used in later tests. A) An example of vertical stripes moving left during three consecutive iterations. Each of the stripe patterns was treated as a different input class, and created on a 32 pixel-wide bitmap. B) An example of two overlapping inputs of different strengths: vertical stripes moving left and horizontal stripes moving downwards. C) Rectangle, grid, ellipse and cross were used as a second type of input. Each of the shapes was treated as a different input class, positioned randomly in a 25x20 bitmap space, and moved downwards.

vertical stripes moving right and horizontal stripes moving downwards. The strength of one of the inputs was fixed at 100%, while the strength of the second was set to 0, 1, 3, 5, 10, 25, 50, 75, 90 and 100%. The strength of the second input would be fixed at 100% and the strength of the first one changed.

The networks were then tested to determine if they could recognise the dominating pattern, either of the patterns as well as how this performance correlates with the strength of the second overlapped input.

F. *Experiment 6 – Classification of complex patterns*

Finally, experiment 6 tested the ability of the networks to process spatially non-uniform data with a higher degree of pattern variability. For this task, we designed inputs using four geometric shapes: squares, grids, ellipses and crosses (Fig. 3C). Each of the shapes was composed of 40 black pixels in order to keep the number of stimulated input neurons constant.

Each input frame was a bitmap 25 by 20 pixels, and contained one of the patterns presented in triplicate. The shapes had a random starting position, wrapped around the edges and moved upwards at the speed of one row per iteration.

The input matrices were vectorised and mapped onto the input layer neurons, with pixel one mapped to neuron one etc. The networks were trained with 500 patterns presented in random order for 5 iterations each. All shapes were moving in the same direction (downwards). Following training, each of the four shapes was presented to each of the ten networks. As during

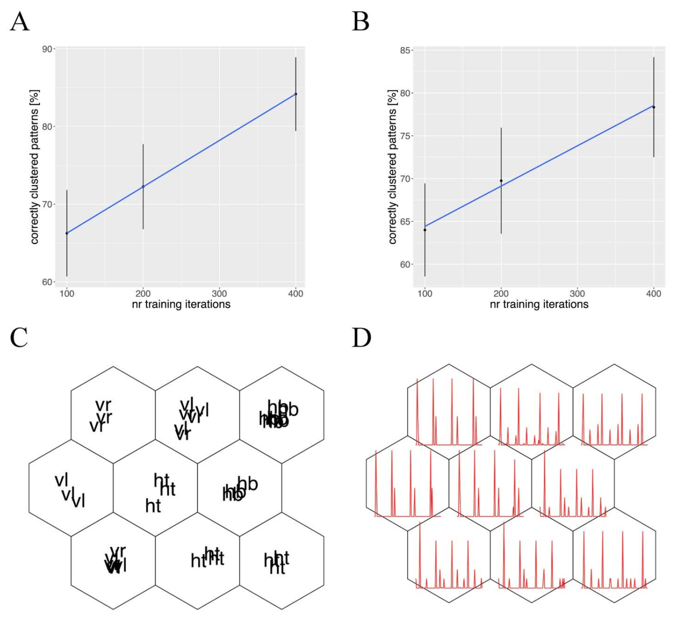

Fig. 4. Results for Experiment 1: Improvement of pattern clustering. Performance of the ensemble improved linearly with training at the collective and, more slowly, the individual level. A) input clustering based on the collective activity of the output layer B) and on the activity of the individual output neurons. C) An example of pattern separation by the network (hb – horizontal bottom, ht – horizontal top, vr – vertical right, vl – vertical left) D) The corresponding codebook vectors.

training, they shared the same direction of movement but had different starting locations thus a different image was presented to each network even when the shape was the same. The ensemble was then scored on their ability to cluster shapes.

V. Results

A. Classification of spatio-temporal patterns

The results for Experiment 1 showed that the performance of the ensemble improved linearly with training and after 400 rounds of training, it was able to correctly cluster 84.42% (SD = 4.7%) of patterns on average (Fig. 4A). Distinctive firing patterns in response to different stimuli were also observed at the level of individual output neurons (Fig. 4B), and after 400 rounds of training 78.4% (SD = 5.9%) of inputs were clustered correctly. In both cases, the input pattern and not the network was the decisive factor for unsupervised clustering - separately trained networks produced similar firing pattern in response to the same stimuli.

B. Clasification of novel inputs

The results for the second experiment show how the networks can generalise when presented with unseen versions of the input – inputs with modified widths of the stripes (Fig. 5). In most cases, the percentage of correctly classified patterns decreased when compared to the seen inputs. However, it did improve in case of widely distributed stripes (stripe width = 1, background stripe with = 5; s1b5) and reached 93.75% (SD = 6%) and 89.25% (SD = 5.53%) for collective and individual clustering respectively. In all cases, activity of the individual output neurons allowed better pattern classification.

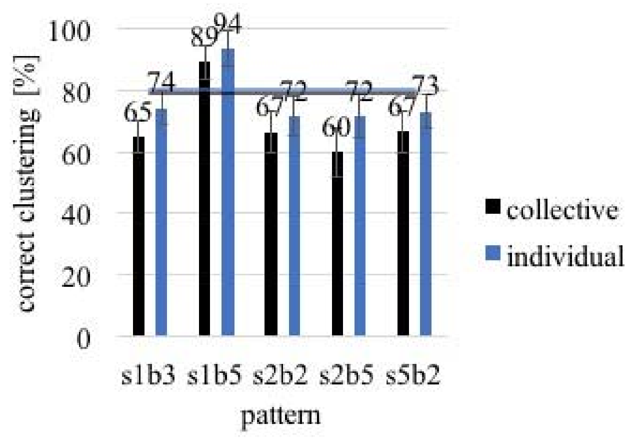

Fig. 5. Results for Experiment 2: Classification of unseen inputs. Modified input patterns were used in order to test if the networks can generalise to classify unseen patterns. Columns are marked s*x* b*y*, where *x* is the stripe width, *y* is the background stripe width. The horizontal black and blue lines (which nearly overlap) indicate the baseline performance in response to standard, seen inputs (s1b2) at the collective and individual levels respectively. In all but one (s1b5) cases, clustering was less correct than in case of previously seen patterns. In all cases, better performance was achieved using individual output neuron activity.

C. Classification of noisy inputs

The results for the third experiment show how the stability of the networks' predictions is affected by the addition of varying amounts of random noise to the standard inputs. The performance of the ensemble decreased non-linearly. On average, for 1% of noise, collective pattern clustering was 72.75% (SD = 3.62%, decrease by 8.58%) correct, and individual 80.25% (SD = 4.16%, decrease by 1.4%) correct, see Fig. 6A and 6B, and Table II. At 50% noise level, collective pattern clustering was 52.75% (SD = 3.22%, decrease by 28.9%) correct, and individual 57.5% (SD = 3.73%, decrease by 23.33%) correct. In comparison to collective activity of the output layer, activity of individual output neurons produced better results at all level of noise except 90%.

The similarity between the SOMs (i.e. to which node and with which other patterns each activity pattern was assigned; linked to what are the perceived relationships between the patterns and indicating possible disruptions of those relationships) of the noisy and noise-free inputs decreased sharply and with 10% noise plateaued around 20% similarity (Fig. 6C), which is close to 25% identity expected by chance.

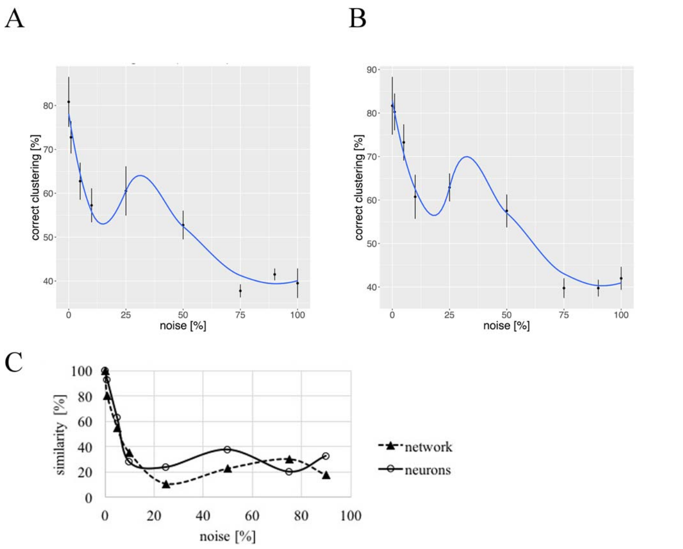

Fig. 6. Results for Experiment 3: Resistance to random noise. Inputs with a different percent of random noise were presented to the networks. Clustering based on A) collective and B) individual spiking patterns of the output neurons. C) Identity with firing patterns in response to noise-free signal. Performance of the ensemble decreased in a non-linear fashion in response to increasing levels of noise.

D. *Classification of incomplete inputs*

The results for Experiment 4 show the resistance of the networks to processing corrupted data (with randomly removed inputs). The networks performance initially sharply drops but then plateaus in the middle section. When 10% of the data removed, collective pattern clustering was an average of 61.25% (SD = 2.43%, decrease by 8.25%) correct, and individual 71.75% (SD = 4.57%, decrease by 20%) correct, see Fig. 7A and 7B, and Table II. When 50% of the data was removed, collective pattern clustering was 60% (SD = 2.64%, decrease by 21.25%) correct, and individual 60.75% (SD = 4.57%, decrease by 19.25%) correct.

Similarity between the SOMs of the complete and incomplete inputs decreased sharply and when 10% of the data was removed, it plateaued around 35% similarity (Fig. 7C), which was a better result than when processing noisy inputs.

E. *Classification of superimposed inputs*

The results for experiment 5 show that the networks performed better than in the presence of the corresponding amount of noise (30%) (Fig. 8 A, B, and Table II). Moreover, when either of the superimposed patterns was accepted as the correct answer, the networks' performance improved as the strength of the second pattern increased, and reached 97.5% (SD = 3.54%) correct clustering when the second input's strength was 75%, and 100% (SD = 0%) correct clustering when the strength was 100% (Fig. 8C). The inputs were highly clustered also at the level of the activity of individual output neurons (Fig. 8D) with 97.5% (SD = 2.58%) and 100% (SD = 0%) patterns correctly clustered when the second pattern's strength was 75% and 100% respectively.

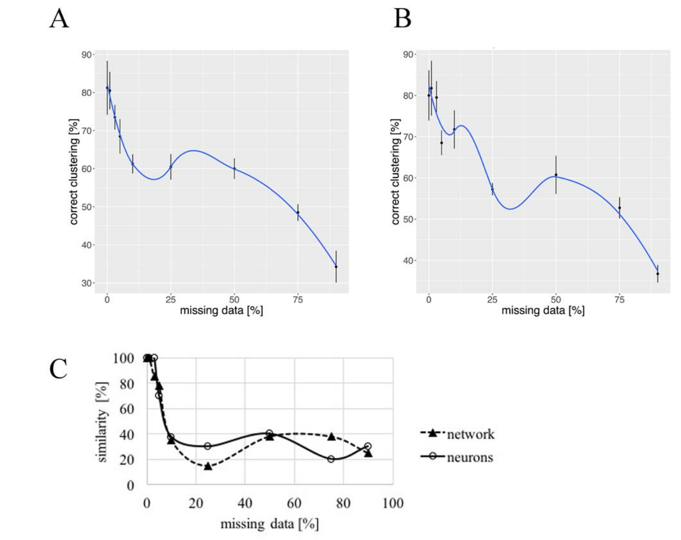

Fig. 7. Results for Experiment 4: Classification of incomplete information. Inputs with different percent of data missing were presented to the networks. Clustering based on A) collective and B) individual spiking patterns of the output neurons. C) Identity with firing patterns in response to 100% of information present. The performance of the ensemble decreased in a non-linear fashion when data was removed.

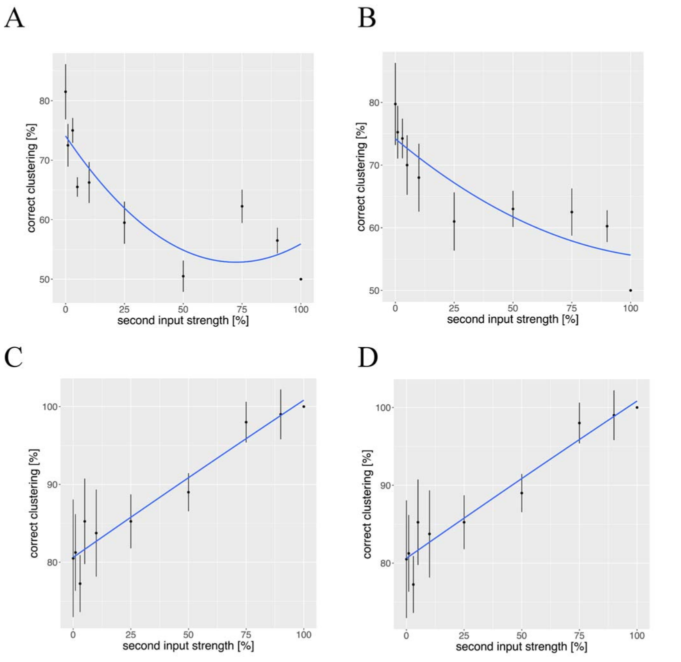

Fig. 8. Results for Experiment 5: Pattern classification of superimposed inputs. One standard pattern and one additional superimposed pattern of varying strength were simultaneously presented to the networks. A) clustering based on the collective activity of the output layer B) and on the activity of the individual output neurons. Additionally, clustering was assessed with either of the two presented patterns accepted as the correct answer C) clustering based on the collective activity of the output layer D) and on the activity of the individual output neurons. The ability of the ensemble to identify the dominant pattern decreased as the strength of the second superimposed pattern increased. Simultaneously, the ensemble became better at identifying the input as one of the superimposed patterns and sharply distinguishing it from the patterns that were not superimposed.

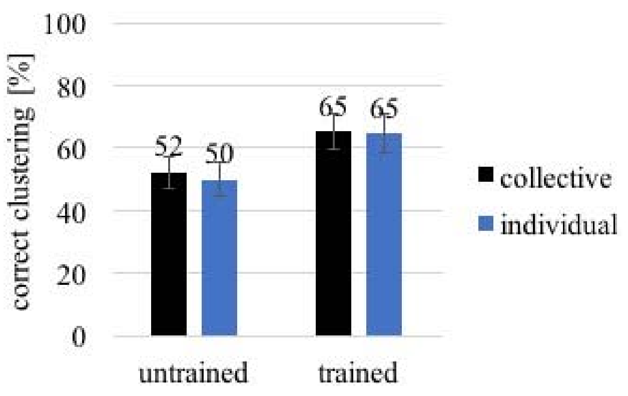

Fig. 9. Results for Experiment 6: Classification of complex patterns. One out of four geometric shapes was presented to either untrained or trained networks. Correctness of pattern classification was assessed at the level of network (collective) and individual neuron activities. Untrained networks performed better than expected by chance. The performance of the networks was further improved with modest training.

F. Classification of complex patterns.

The results for experiment 6 show that untrained networks correctly classified 52.25% (SD = 3.43%) and 50% (SD = 2.64%) patterns at the network and neuron level respectively, which is higher than 25% expected in case of random classification. After a short training (500 patterns shown for 5 iterations each), 65.25% (SD = 4.63%) and 64.75% (SD = 3.63%) of patterns were correctly classified at the collective and individual level respectively. Given that each network in the ensemble were exposed to a different random view of the input shapes, this classification performance is considered good.

TABLE II. SUMMARY OF SELECTED RESULTS

Experiment 3 Classification of noisy inputs										
Noise level [%]	***0***	***1***	***5***	***10***	***25***	***50***	***75***	***90***	***100***	
Collective score	80.8	72.8	62.8	57.3	60.5	52.8	37.8	41.5	39.5	
Individual score	81.7	80.2	73.3	60.8	62.9	57.5	39.8	39.8	42.0	
Experiment 4 Classification of incomplete inputs										
Missing inputs [%]	***0***	***1***	***3***	***5***	***10***	***25***	***50***	***75***	***90***	
Collective score	81.3	80.5	*73.5*	*68.5*	61.3	60.5	60.0	48.5	34.3	
Individual score	80.0	81.8	79.5	68.5	71.8	57.3	60.8	52.8	36.8	
Experiment 5 Classification of superimposed inputs										
Second input strength [%]	***0***	***1***	***3***	***5***	***10***	***25***	***50***	***75***	***90***	***100***
Collective score	81.5	72.5	75.0	65.5	66.3	59.5	50.5	62.3	56.5	50.0
Individual score	79.8	75.3	74.3	70.0	68.0	61.0	63.0	62.5	60.3	50.0

VI. CONCLUSIONS

The goal of this work was to develop a biologically inspired model of a network – a spiking network with activity-dependent plasticity – that has a potential to overcome the traditional limitation of biologically-realistic networks, namely low speed and high computational cost in comparison to non-spiking networks.

In this study, we used two types of simple spatio-temporal patterns as inputs: moving stripes and geometric shapes, each with four subclasses. An ensemble of ten independently trained, unsupervised networks was used to process subtasks in parallel, and their activity was interpreted using SOMs.

We demonstrated that the networks can be used to process subsets of data independently of each other, and produce similar spiking patterns in response to the same class of input. In the current form, the ensemble has a capacity to generalise and a limited capacity to process noisy, incomplete data, which has been previously observed in some SNN [29], [30]. However, in our experiments we used random noise and complete silencing of a proportion of input neurons, which are extreme cases of data corruption as there is no link between the corrupted values and the original data.

Even before training, the networks exhibited some ability to classify inputs because of the use of SOMs. During training, unlike classical networks using backpropagation, spiking networks using STDP adjusted their weights in response network's activity but did not optimise for the seen inputs in the classical sense – the activity of the trained network was an emergent property rather than the results of weights adjusting to minimise an error. These properties are of vital importance for understanding information processing and optimisation in the biological neural networks. When it comes to applied research, we anticipate that the suitability of the networks to perform user-defined tasks could be improved with supervision [11], [31] and reinforcement [32].

In this proof of concept the networks were small and could easily be used on a personal computer. One hundred iterations of training took on average 801631.3 clock ticks (SD = 7811.5, n = 20), whereas one hundred iterations of testing 425568.8 (SD = 7502.4, n = 20). However, their ability to train and process data independently of each other offers the possibility of easy parallelisation and scalability of data processing. We anticipate that in the future, such "wide learning" networks could be used to identify patterns in massive data sets that change in time based on the shared temporal and quantitative properties of the data. They could also be used to examine how novel patterns relate to known ones suggesting potential applications in fraud detection or bioinformatics.

VII. References

[1] W. Maass, "Networks of spiking neurons: The third generation of neural network models," *Neural Networks*, vol. 10, no. 9, pp. 1659–1671, 1997.

[2] W. Bair and C. Koch, "Temporal precision of spike trains in extrastriate cortex of the behaving macaque monkey.," *Neural Comput.*, vol. 8, no. 6, pp. 1185–202, Aug. 1996.

[3] P. Reinagel and R. C. Reid, "Temporal Coding of Visual Information in the Thalamus," *J. N*, vol. 20, no. 14, pp. 5392–5400, 2000.

[4] J. Backus and John, "Can programming be liberated from the von Neumann style?: a functional style and its algebra of programs," *Commun. ACM*, vol. 21, no. 8, pp. 613–641, Aug. 1978.

[5] E. Painkras, L. A. Plana, J. Garside, S. Temple, F. Galluppi, C. Patterson, D. R. Lester, A. D. Brown, and S. B. Furber, "SpiNNaker: A 1-W 18-Core System-on-Chip for Massively-Parallel Neural Network Simulation," *IEEE J. Solid-State Circuits*, vol. 48, no. 8, pp. 1943–1953, Aug. 2013.

[6] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura, B. Brezzo, I. Vo, S. K. Esser, R. Appuswamy, B. Taba, A. Amir, M. D. Flickner, W. P. Risk, R. Manohar, and D. S. Modha, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science (80-.).*, vol. 345, no. 6197, 2014.

[7] D. O. Hebb, "Organization of behavior. New York: Wiley," *J. Clin. Psychol.*, vol. 6, no. 3, pp. 307–307, Jul. 1950.

[8] 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. 10464–10472, 1988.

[9] Markram, H. Lubke, J. ; Frotscher, M. ; Sakmann, and Bert, "Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs," *Science (80-.).*, vol. 275, no. 5297, 1997.

[10] J. D'amour and R. Froemke, "Inhibitory and excitatory spike-timing-dependent plasticity in the auditory cortex," *Neuron*, 2015.

[11] J. Hu, H. Tang, K. C. Tan, H. Li, and L. Shi, "A Spike-Timing-Based Integrated Model for Pattern Recognition," *Neural Comput.*, vol. 25, no. 2, pp. 450–472, Feb. 2013.

[12] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier, "STDP-based spiking deep neural networks for object recognition."

[13] B. W. Mel, M. Tatsuno, P. U. Diehl, and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Front. Comput. Neurosci. | www.frontiersin.org 1*, vol. 9, 2015.

[14] P. U. Diehl and M. Cook, "Learning and Inferring Relations in Cortical Networks," 2016.

[15] A. Nere, U. Olcese, D. Balduzzi, and G. Tononi, "A Neuromorphic Architecture for Object Recognition and Motion Anticipation Using Burst-STDP," *PLoS One*, vol. 7, no. 5, 2012.

[16] Z. Yang, A. Murray, F. Wörgötter, K. Cameron, and V. Boonsobhak, "A Neuromorphic Depth-from-Motion Vision Model with STDP Adaptation."

[17] S. Zagoruyko and N. Komodakis, "Wide Residual Networks," May 2016.

[18] Y. Shim, A. Philippides, K. Staras, P. Husbands, N. Logothetis, and A. Tolias, "Unsupervised Learning in an Ensemble of Spiking Neural Networks Mediated by ITDP," *PLOS Comput. Biol.*, vol. 12, no. 10, p. e1005137, Oct. 2016.

[19] B. T. T. Yeo, F. M. Krienen, J. Sepulcre, M. R. Sabuncu, D. Lashkari, M. Hollinshead, J. L. Roffman, J. W. Smoller, L. Zöllei, J. R. Polimeni, B. Fischl, H. Liu, and R. L. Buckner, "The organization of the human cerebral cortex estimated by intrinsic functional connectivity.," *J. Neurophysiol.*, vol. 106, no. 3, pp. 1125–65, Sep. 2011.

[20] L. F. Abbott, "Lapicque's introduction of the integrate-and-fire model neuron (1907)."

[21] T. W. Troyer and K. D. Miller, "Physiological gain leads to high ISI variability in a simple model of a cortical regular spiking cell.," *Neural Comput.*, vol. 9, no. 5, pp. 971–83, Jul. 1997.

[22] M. N. Shadlen and W. T. Newsome, "The variable discharge of cortical neurons: implications for connectivity, computation, and information coding.," *J. Neurosci.*, vol. 18, no. 10, pp. 3870–96, May 1998.

[23] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

[24] C. Kelsom and W. Lu, "Development and specification of GABAergic cortical interneurons.," *Cell Biosci.*, vol. 3, no. 1, p. 19, Apr. 2013.

[25] S. Sahara, Y. Yanagawa, D. D. M. O'Leary, and C. F. Stevens, "The fraction of cortical GABAergic neurons is constant from near the start of cortical neurogenesis to adulthood.," *J. Neurosci.*, vol. 32, no. 14, pp. 4755–61, Apr. 2012.

[26] C. Beaulieu, Z. Kisvarday, P. Somogyi, M. Cynader, and A. Cowey, "Quantitative distribution of GABA-immunopositive and -immunonegative neurons and synapses in the monkey striate cortex (area 17).," *Cereb. Cortex*, vol. 2, no. 4, pp. 295–309.

[27] A. Krogh and J. Vedelsby, "Neural network ensembles, cross validation and active learning," *Proceedings of the 7th International Conference on Neural Information Processing Systems*. MIT Press, pp. 231–238, 1994.

[28] R. Wehrens, L. M. C. Buydens, R. Wehrens, and L. M. C. Buydens, "Self- and Super-organizing Maps in R: The kohonen Package," *J. Stat. Softw.*, vol. 21, no. i05, Oct. 2007.

[29] Q. Yu, R. Yan, H. Tang, K. C. Tan, and H. Li, "A Spiking Neural Network System for Robust Sequence Recognition," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 27, no. 3, pp. 621–635, Mar. 2016.

[30] T. Iakymchuk, A. Rosado-Muñoz, J. F. Guerrero-Martínez, M. Bataller-Mompeán, and J. V Francés-Víllora, "Simplified spiking neural network architecture and STDP learning algorithm applied to image classification," *EURASIP J. Image Video Process.*, no. 4, 2011.

[31] F. Ponulak and A. Kasiński, "Supervised learning in spiking neural networks with ReSuMe: sequence learning, classification, and spike shifting.," *Neural Comput.*, vol. 22, no. 2, pp. 467–510, 2010.

[32] R. V. Florian, "Reinforcement Learning Through Modulation of Spike-Timing-Dependent Synaptic Plasticity," *Neural Comput.*, vol. 19, no. 6, pp. 1468–1502, Jun. 2007.