Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 145 (2018) 488–494

Procedia Computer Science

www.elsevier.com/locate/procedia

Postproceedings of the 9th Annual International Conference on Biologically Inspired Cognitive Architectures, BICA 2018 (Ninth Annual Meeting of the BICA Society)

Estimation of the influence of spiking neural network parameters on classification accuracy using a genetic algorithm

Aleksandr Sboev[a,b], Alexey Serenko[a], Roman Rybka[a], Danila Vlasov[a,b], Andrey Filchenkov[c]

[a]National Research Centre "Kurchatov Institute", Moscow, Russia
[b]MEPhI National Research Nuclear University, Moscow, Russia
[c]ITMO University, Saint Petersburg, Russia

Abstract

Spiking neural network learning is closely connected to the plasticity mechanism chosen. The aim of this work is to assess, for a network of Leaky Integrate-and-Fire neurons with Spike-Timing-Dependent Plasticity, the influence of STDP constants and input encoding parameters on the accuracy of solving the simple model task of Fisher's Iris classification. Parameter adjustment is performed with a genetic algorithm (GA) on base of NeuroEvolution of Augmenting Topologies (NEAT). The results of the GA with different fitness functions are compared to randomly generating the network parameters. The use of GA is shown to be able to persistently select the network parameters that achieve the highest accuracy for the configuration considered. The network is shown to be more sensitive to its input encoding parameters (encoding rates), than to plasticity constants.

© 2018 The Authors. Published by Elsevier B.V.
This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0/)
Peer-review under responsibility of the scientific committee of the 9th Annual International Conference on Biologically Inspired Cognitive Architectures.

Keywords: synaptic plasticity; spiking neural network; genetic algorithm; computational neuroscience

1. Introduction

The recent years have seen an increasing interest in biologically motivated spiking neural networks applied to clustering and classification tasks. In particular, researchers paid attention to networks with Spike-Timing-Dependent Plasticity (STDP) model [1, 2, 3]. However, choosing STDP constants, along with other network components remains to be a research question. In many cases [1, 2], the parameter choice is empirical, based on expert knowledge of the model developers, or it is borrowed from biological data [3]. However, parameters chosen so may not be optimal for the practical task considered.

E-mail address: Sboev_AG@nrcki.ru

1877-0509 © 2018 The Authors. Published by Elsevier B.V.
This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0/)
Peer-review under responsibility of the scientific committee of the 9th Annual International Conference on Biologically Inspired Cognitive Architectures.
10.1016/j.procs.2018.11.111

The purpose of this paper is to study, to what extent does the classification accuracy depend on how the network parameters are chosen. We use a model task of classification of the well-known Fisher's Iris. The neural network configuration and the training mechanism (see Section 2) are based on the effect of stabilising the neuron average output frequency by STDP [4]. For the parameter selection, we employ the genetic algorithm, for which in Section 3 we present several fitness functions that are based on the algorithm used for decoding the network output by the neurons' mean spiking rates.

2. Network configuration

The network consists of three neurons corresponding to three classes in the Fisher's Iris dataset, they are not connected to each other. On the training stage, each neuron receives instances belonging to one class. On the testing stage, a neuron is expected to distinguish instances of the class it has been trained on (further referred to as "own" class for the neuron) from the other classes.

We use Leaky Integrate-and-Fire as a neuron model:

$$\frac{dV}{dt} = \frac{-(V(t) - V_{\text{rest}})}{\tau_{\text{m}}} + \frac{I_{\text{syn}}(t)}{C_{\text{m}}},$$

and as soon as $V \geq V_{\text{th}} = -54$ mV, $V \leftarrow V_{\text{rest}} = -70$ mV, and the neuron is unable to change its potential during the refractory period $\tau_{\text{ref}} = 3$ ms. $\tau_{\text{m}} = 10$ ms. C_{m} is chosen to be an adjustable parameter. The postsynaptic current is of exponential form, in which a spike at time t^i_{sp} from i-th synapse adds $w(t_{\text{sp}})\frac{q_{\text{syn}}}{\tau_{\text{syn}}} e^{-\frac{t-t_{\text{sp}}}{\tau_j}} \theta(t-t_{\text{sp}})$ to $I_{\text{syn}}(t)$, where w is the weight of the synapse, $q_{\text{syn}} = 5$ fC is the total electric charge injected into the neuron by one incoming spike, $\tau_{\text{syn}} = 5$ ms, and θ is the Heaviside step function.

STDP is additive:

$$\Delta w = \begin{cases} -\alpha\lambda \cdot \exp\left(-\dfrac{t_{\text{pre}} - t_{\text{post}}}{\tau_-}\right), & \text{if } t_{\text{pre}} - t_{\text{post}} > 0; \\ \lambda \cdot \exp\left(-\dfrac{t_{\text{post}} - t_{\text{pre}}}{\tau_+}\right), & \text{if } t_{\text{pre}} - t_{\text{post}} < 0; \end{cases}$$

where $\lambda = 0.01$ is the learning rate, and α and $\tau_{+/-}$ are adjustable parameters. The STDP spike pairing scheme is restricted symmetric [5]: only pairs of consecutive presynaptic and postsynaptic spikes with no other spikes between are accounted in the weight change rule. The ability of these neuron and plasticity models to be used for a classification task was shown previously [6].

The learning method is based on the mean spiking rate stabilising effect [4]: STDP of the considered type leads to synaptic weights with which the neuron spikes with certain mean rate. This rate does not depend on the input spike rates (in a sufficiently broad range), and depends only on the neuron and STDP parameters. So, with rate-encoded input data, this certain mean output rate may indicate that the neuron is receiving an instance of the class it was trained on.

Therefore, after learning, an instance from the testing set is assigned to class i if the neuron mean output rate in response to this instances is close to the output rates in response to instances from class i training set (that is, closer than the output rates of that neuron in response to instances from any other class). If several neurons recognise an input instance to belong to their own classes, we account one neuron, the one that better distinguishes the training set (that is, has the biggest difference in mean rates in response to different classes).

The input data, Fisher's Iris (3 classes of 50 4-dimensional vectors, the first class being linearly distinguishable from the two other), are normalised so that the Eucledian norm of each vector equals 1. Then, each

component x of a vector corresponds to 2-s-long Poisson spike sequence with mean rate $\nu_{\text{low}} + x \cdot \nu_{\text{high}}$ (generated by presenting an incoming spike with the probability of this rate divided by the simulation timestep 0.1 ms). Here ν_{low} and ν_{high} are adjustable parameters. Moreover, each vector component is encoded by a bunch of several input synapses receiving such mean spike rate. Because of this, in spite of the fact that under additive STDP the synaptic weights form a bimodal distribution [7] (tend either to 0 or to their maximum value of 1), the average weight over the synapses of a bunch can have intermediate values between 0 and 1.

3. The genetic algorithm

We use NeuroEvolution of Augmented Topologies (NEAT) [8] as a genetic algorithm in the implementation by the MultiNEAT project [9]. There, a candidate solution (or individual) is a set of network parameters. A population of individuals is divided into species, and individuals compete mostly within their species. The algorithm comprises the following steps:

Fitness function normalisation.. The fitness function of each individual is normalised by the number of individuals in its species to penalise overpopulated species. Furthermore, individuals from young species (species created less than 5 generations ago) get their fitness multiplied by 1.1 in order to encourage mutations that may eventually be useful but reduce fitness in short term. Accordingly, the fitness of species older than 30 generations is multiplied by 0.5. If the total fitness of a species does not improve during 50 generations, it is multiplied by 10^{-7}, which is supposed to inevitably make the species extinct in the next generation.

Mating.. 1% best performing individuals enter the next generation unchanged. 25% best individuals produce offspring, with probability of 0.3 it happens by asexual reproduction (just copying), otherwise — by mating with each other, in which case the sibling's parameters are an average of the parents' ones. Inter-species mating occurs with probability of 10^{-4}. During mating, a mutation occurs with probability of 0.25. Each adjustable parameter value has its own probability `mutation_prob` to be involved in the mutation; being mutated, a parameter is chosen anew from a uniform distribution (with minimum and maximum boundaries as in Table 1) with probability of 0.25%, otherwise changed by a uniform random value from the range `±mut_power`.

Speciation.. Then, all individuals are rearranged into species. An individual is assigned its species by calculating for each species the distance between a random representative of the species (from the previous generation) and the individual. If the distance to some species does not exceed `CompatibilityThreshold`, the individual is assigned to the species without further checking any other species. `CompatibilityThreshold` changes dynamically so that there always be 5 to 10 species. If an individual is farther than `CompatibilityThreshold` from all species, a new species is created for it. If after the rearrangement a species loses all its individuals, it becomes extinct.

We consider 4 fitness functions.

"By instances": maximising for each instance x_i of the training set the difference between the "own" neuron's output rate and other neurons' rates in response to the instance.
The performance of recognising an input instance x_i, which belongs to a class C, is
$\text{Fitness}(x_i) = \sum_{\text{over neurons } j \in \{1;2;3\} \setminus C} \left| y_j(x_i) - y_C(x_i) \right|$, where $y_j(x_i)$ is the output rate of neuron j in response to x_i after learning.
The performance of recognising the whole set is the sum of $\text{Fitness}(x_i)$ over all x_i.

"With minimising spread in own class": maximising for each instance x_i of the training set the difference between the output rate of the "own" neuron in response to the instance and the output rate of other neurons in response to their own classes (averaged over all the instances of the classes), along with minimising the difference in the output rate of the "own" neuron in response to the instance and the averaged output rate of the "own" neuron in response to its own class.

Table 1. Spiking network parameters to be adjusted with the genetic algorithm

Parameter	Fixed	Genetic adjustment parameters			
		min	max	mutation_prob	mut_power
STDP constants					
$\alpha = \frac{A_-}{A_+}$	1.035	0.3	3	0.1	0.001
τ_+, ms	20	0	100	0.1	1
τ_-, ms	20	0	100	0.1	1
Encoding rates and neuron membrane capacity					
encoding rate ν_{low}, Hz	4	0	20	0.1	1
encoding rate ν_{high}, Hz	50	20	300	0.1	10
membrane capacity C_m, pF	2	1	4	0.1	0.001

$\text{Fitness}(x_i) = \sum_{\text{over neurons } j\in\{1;2;3\}\setminus C} \left|y_j(x_i) - y_C(C)\right| - \left|y_C(x_i) - y_C(C)\right|$,
where $y_C(X_j) = \frac{\sum_{x_i\in X_j} y_C(x_i)}{\sum_{x_i\in X_j}}$ is the output rate of the neuron C in response to class j, averaged over all the instances from class j.

"By classes": maximising for each neuron the difference between its averaged output rate in response to its own class and its averaged output rate in response to other classes. The performance of recognising class C is
$\text{Fitness}(C) = \sum_{\text{over classes } j\in\{1;2;3\}\setminus C} \left|y_C(X_j) - y_C(X_C)\right|$, The performance of recognising the whole set is the sum over classes $\sum_{j\in\{1;2;3\}} \text{Fitness}(j)$.

"By accuracy": F1-macro on the training set.

4. Experiments and results

Experiments are conducted with three sets of adjustable parameters from Table 1: STDP constants, input encoding parameters (encoding rates) and neuron membrane capacity, and all of these. The "fixed" column in the table denotes the value the parameter gets if is not adjusted. With all parameters taken from that column, accuracy is 35 to 50% (over cross-validation folds), which shows these parameter values to be far from optimal.

For every parameter set, we try the four fitness functions enlisted in Section 3.

The spiking network learning involves cross-validation over 5 folds: the Iris dataset is split into 5 parts of 30 instances (10 instances from each of the classes), four parts of which are used for training and the fifth one for testing; then splitting is repeated 5 times so that each of the parts is once used for testing. Splitting into parts is performed with random reshuffling for each individual. The fitness of the individual is the average over folds of the fitness functions calculated for output rates in each of the folds.

With the number of individuals in the population being 100, the fitness function of the best fit individual reaches its steady value already on the second generation, around which it then fluctuates with populations. This is observed for all the four fitness functions and all the three adjustable parameter sets. The classification accuracy thus reaches its highest achievable value, which is the F1-score range of 86 to 97% over cross-validation folds. The best F1-scores over all generations, their mean values and fold ranges for different combinations of fitness functions and adjustable parameter sets are presented in Table 2. Further increase in accuracy by means of the genetic algorithm is impossible, due to the fluctuation of the best fit individual's F1-score from generation to generation is comparable to its fluctuation from run to run with fixed network parameters and fixed training and testing sets, which is up to 10%.

To compare, the work [10] that employed a genetic algorithm to adjust synaptic delays of a neuron with STDP, in case of rate encoding of the input data achieved the accuracy of 81% on the training set and 74% on the testing set (no information on cross-validation was present).

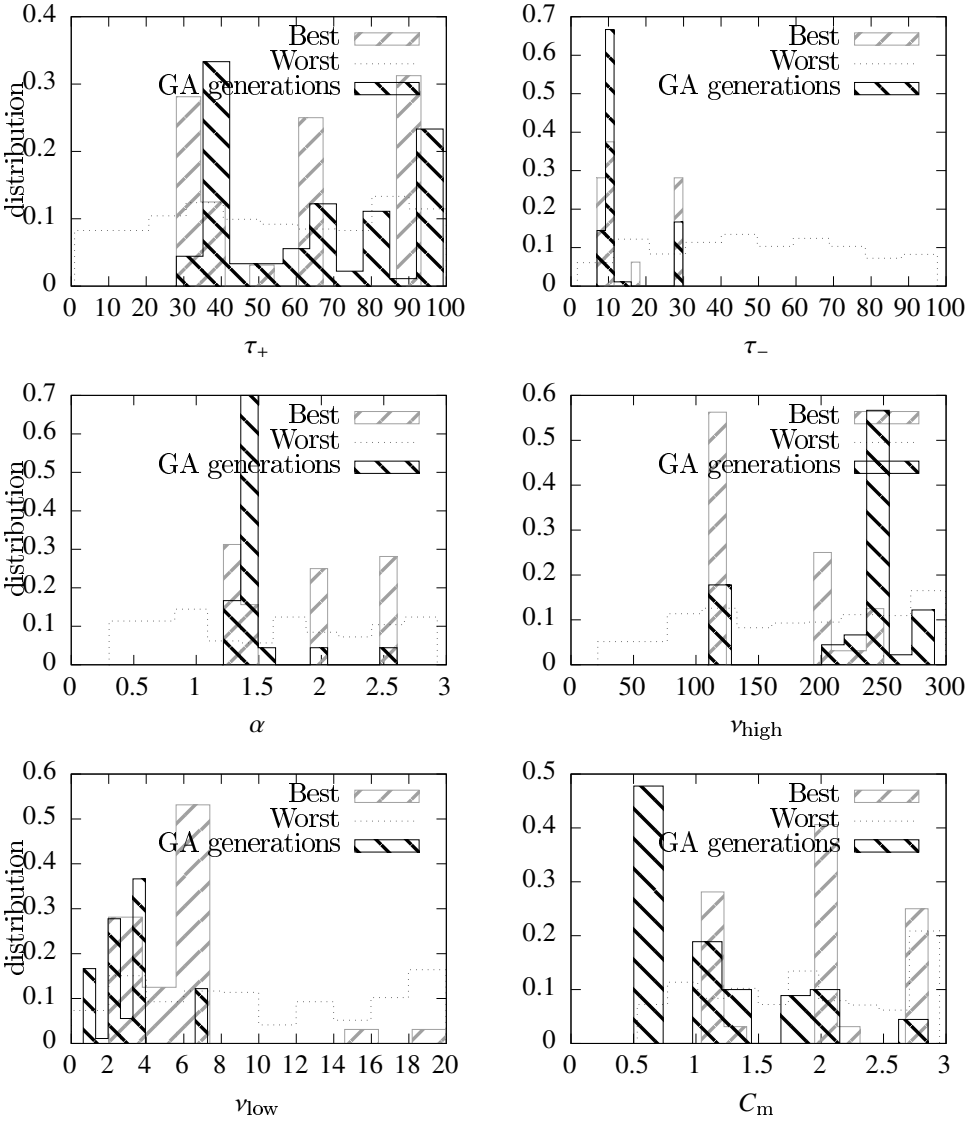

Fig. 1. The distributions of different adjustable parameters among 1 000 individuals with randomly generated parameters ("Best" are individuals with F1-score above 85%, and "Worst" are the others), and among the generation's best fit individuals for all generations during the run of the genetic algorithm ("GA generations")

Generating all of the network adjustable parameters randomly from a uniform distribution without the genetic algorithm caused the (fold-average) F1-score to be above 85% for 10 parameter sets out of 1 000 generated, that is about 1%. The parameters of such sets are denoted as "Best" in Fig. 1. The genetic algorithm with 100 individuals in a population reaches such accuracy in 2 generations, and with 20 individuals in a population — in 5 generations. The parameters obtained this way, composed of the population's best fit individuals from each generation (of the experiments adjusting all the parameters at once, but uniting the results for all fitness functions) are denoted as "Genetic algorithm generations" in the Figure, and generally coincide with the ones found randomly. The broad range of the distributions, with several peaks, indicate that there exist several sets of parameters that provide high accuracy. The distribution of ν_{high} explains the low accuracy in case only STDP constants are adjusted, encoding rates and the neuron membrane capacity C_{m} staying fixed (the three first rows in Table 2): the "fixed" value of ν_{high} happened to be far

Table 2. F1-scores of the best individuals of all genetic algorithm generations, %

Parameters to adjust	Fitness functions (as defined in Section 3)	Training			Testing		
		avg	min	max	avg	min	max
STDP constants	"By instances"	75	61	88	78	69	83
	"With min. spread in own class"	74	72	77	79	72	86
	"By classes"	75	65	81	78	69	83
	"By accuracy"	80	73	89	78	65	90
Encoding rates and C_m	"By instances"	86	82	90	88	76	97
	"With min. spread in own class"	88	84	92	89	73	97
	"By classes"	87	86	89	89	87	93
	"By accuracy"	90	87	93	90	86	93
STDP, encoding rates and C_m	"By instances"	89	84	94	86	83	90
	"With min. spread in own class"	88	85	91	88	79	97
	"By classes"	86	81	90	85	75	93
	"By accuracy"	90	84	94	90	86	97

from its optimal range. On the contrary, adjusting encoding rates and C_m (the three centre rows in the table) can compensate for suboptimal STDP constants, providing the same accuracy as when adjusting all the parameters (the three last rows in the table). The latter might mean that the learning scheme is more sensitive to its input encoding parameters — encoding rates than to STDP constants.

5. Conclusion

Adjustment of the network parameters, namely STDP constants and encoding rates, allows to achieve the accuracy of Fisher's Iris dataset classification in the range of 86 to 97% (over cross-validation folds). This accuracy persists if adjusting both STDP constants and encoding rates, or adjusting only encoding rates leaving the STDP parameters suboptimal, but not if adjusting only STDP constants. This shows the network to be robust to its STDP constants but more sensitive to its input encoding parameters.

The proposed method for adjusting spiking network parameters with the help of a genetic algorithm can be applied to more complex network models, which is a direction of further work.

Acknowledgements

Developing topology, encoding and decoding techniques, studying the output rate stabilisation effect which the learning algorithm used in this work is based on, have been conducted under the support of the Russian Science Foundation grant 17-71-20111 "Study and justification of mechanisms for spiking neural networks learning based on synaptic plasticity in order to create biologically inspired nonlinear information models capable of solving practical tasks". The development of the fitness functions for the genetic algorithm, and the implementation of the interface connecting the genetic algorithm to the NEST [11] spiking network simulator have been supported by the Russian Foundation for Basic Research (RFBR) grant 17-37-50097 'мол-нр' and have been carried out using computing resources of the federal collective usage center Complex for Simulation and Data Processing for Mega-science Facilities at NRC "Kurchatov Institute", http://ckp.nrcki.ru/.

References

[1] R. Legenstein, C. Naeger, W. Maass, What can a neuron learn with Spike-Timing-Dependent Plasticity, Neural Computation 17 (2005) 2337–2382.

[2] P. U. Diehl, M. Cook, Unsupervised learning of digit recognition using Spike-Timing-Dependent plasticity, Frontiers in Computational Neuroscience.

[3] T. Masquelier, R. Guyonneau, S. J. Thorpe, Spike Timing Dependent Plasticity finds the start of repeating patterns in continuous spike trains, PloS one 3 (1) (2008) e1377.
[4] A. Sboev, R. Rybka, A. Serenko, On the effect of stabilizing mean firing rate of a neuron due to STDP, Procedia Computer Science 119 (2017) 166–173.
[5] A. Morrison, M. Diesmann, W. Gerstner, Phenomenological models of synaptic plasticity based on spike timing, Biological Cybernetics 98 (2008) 459–478.
[6] A. Sboev, R. Rybka, A. Serenko, D. Vlasov, N. Kudryashov, V. Demin, To the role of the choice of the neuron model in spiking network learning on base of Spike-Timing-Dependent Plasticity, Procedia Computer Science 123 (2018) 432–439.
[7] S. Song, K. D. Miller, L. F. Abbott, Competitive Hebbian learning through spike-timing-dependent synaptic plasticity, Nature neuroscience 3 (9) (2000) 919–926.
[8] K. O. Stanley, R. Miikkulainen, Evolving neural networks through augmenting topologies, Evolutionary computation 10 (2) (2002) 99–127.
[9] K. Stanley, R. Miikkulainen, MultiNEAT – a portable software library for performing neuroevolution.
URL http://multineat.com/index.html
[10] S. Johnston, G. Prasad, L. Maguire, T. McGinnity, A hybrid learning algorithm fusing STDP with GA based explicit delay learning for spiking neurons, in: 2006 3rd International IEEE Conference Intelligent Systems, 2006. doi:10.1109/IS.2006.348493.
[11] Susanne Kunkel et al., NEST 2.12.0 (Mar. 2017). doi:10.5281/zenodo.259534.
URL https://doi.org/10.5281/zenodo.259534