Evolutionary Feature Selection for Spiking Neural Network Pattern Classifiers

Michal Valko, Nuno Cavalheiro, Marco Castelani

Evolutionary Feature Selection for Spiking Neural Network Pattern Classifiers

Michal Valko
Department of Artificial Intelligence,
Institute of Applied Informatics,
Faculty of Mathematics, Physics
and Informatics, Comenius University,
842 48 Bratislava,
Slovakia
Email: michal@cs.pitt.edu

Nuno C. Marques
CENTRIA,
Departmento de Informatica,
Faculdade de Ciencias e Tecnologia,
University Nova de Lisboa,
Quinta da Torre, 2829-516 Caparica,
Portugal
Email: nmm@di.fct.unl.pt

Marco Castellani
CENTRIA,
Faculdade de Ciencias e Tecnologia,
University Nova de Lisboa,
Quinta da Torre, 2829-516 Caparica,
Portugal
Email: mcas@fct.unl.pt

***Abstract*— This paper presents an application of the biologically realistic JASTAP neural network model to classification tasks. The JASTAP neural network model is presented as an alternative to the basic multi–layer perceptron model. An evolutionary procedure previously applied to the simultaneous solution of feature selection and neural network training on standard multi–layer perceptrons is extended with JASTAP model. Preliminary results on IRIS standard data set give evidence that this extension allows the use of smaller neural networks that can handle noisier data without any degradation in classification accuracy.**

Index Terms—**Artificial Intelligence, Neural Networks, Genetic Algorithms, Spiking Neuron Models, JASTAP**

I. Introduction

Classification of patterns is important to many data mining processes. Among possible classifiers, artificial neural network classifiers have proven to be one of the most robust classification systems. Their capability to deal with noisy input patterns, to handle both noisy and continuous value data did prove to be an essential tool for classification and prediction [1].

This paper aims to present an alternative to the basic perceptron model [2] by using the neural realistic JASTAP model [3]. JASTAP model implements biologically plausible neural networks in a well parameterized model that can be statistically related [4]. Since the usual simplifications for other spiking models [5] are not done a priori, we can keep all the flexibility known in biological neurons. On the same time, since parameters can also be predefined in advance (giving rise to more simple models) JASTAP proves as a very powerful new way of representing both model knowledge and experience. This way, only needed features can be activated in the model. A first demonstrative study on the use of this new kind of artificial neural network for classification tasks is presented on IRIS dataset.

Standard backpropagation learning [2] on JASTAP neural networks (and on simpler spiking models in general) is still an open issue (mainly due to the different weights that need to be adjusted and to cyclic connections). So, a more generic and powerful learning method is used: we have incorporated in JASTAP the evolutionary learning model of FeaSANNT [6]. FeaSANNT is an evolutionary procedure for simultaneous solution of the two combinatory search problems of feature selection and parameter learning in artificial neural network classifier systems. FeaSANNT was already successfully applied to feature selection on traditional artificial neural network pattern classifiers [6]. In this paper we extend previous work by applying the JASTAP model to the same type of classification problems.

An additional motivation for this work relies on the need to encode recurrence and context in neural networks. The simple MLP (and the related popular backpropagation learning algorithm [2]), can not handle cyclic connections. Indeed, the well known usage of MLPs as universal approximators can only be achieved by means of enlarging the hidden layer neurons as needed (e.x. Kürková's *Universal Approximators* article in [7]). As a result, in problems where context or time representation is needed, repetitive iterations can not be used. Spiking neurons in general and JASTAP model in particular may present a solution for this problem.

The remaining of this paper is organized as follows. Section II presents the JASTAP spiking neural model. Then the learning procedure used is presented in section III. The changes made to adapt this procedure and some advantages and properties specific for JASTAP networks are described in section IV. Experimental results are presented in section V. Finally, the contributions of this work are presented in section VI.

II. JASTAP

A. JASTAP As a Spiking Neuron Model

JASTAP[1] model aims to simulate some biologically realistic functions of neural networks [3]. JASTAP belongs to the family of spiking models. Formal spiking neuron models in general work with temporal coding with mostly biologically relevant action potentials. Spiking models can also perform complex non–linear classification [8]. This can be achieved

[1] pronounced as Yastap

with less units (spiking neurons) than with classical rate–coded networks (please see section V-C).

In this section we briefly present JASTAP model [3] and the parameters used for easier adaptation of this model to classification tasks. More details on the JASTAP model can be found in [3].

B. Model Description

A JASTAP model is an artificial NN, which consists of JASTAP neurons as the basic elements. A neuron is described with:

- SET OF SYNAPSES As it is usual in any neural network model, an JASTAP neuron is interconnected with its environment by one or more synaptic inputs and a single output (axon). The output can be connected with one or more neuron synapses in the network.
  For each synapse we consider:
  - INPUT — can be internal (connected with axon of other neuron) or external (from the outer environment).
  - SHAPE OF POSTSYNAPTIC POTENTIAL (PSP) PROTOTYPE — Biological neuron waveform evoked by a spike arriving at a synapse is described in JASTAP model by

$$\mathrm{PSP}(t) = k \cdot \left(1 - \mathrm{e}^{-\frac{t}{t_1}}\right)^2 \cdot \mathrm{e}^{-\frac{2t}{t_2}} \qquad (1)$$

    The waveform inter alia (controlled by parameters t_1 and t_2) emulates whether the synapse is located on a soma or on a dendritic tree. t_1 and t_2 can vary from synapse to synapse. They determine the potential decay. For example, [9] mentions $t_1 = 0.3\,\mathrm{ms}$ and $t_2 = 2.7\,\mathrm{ms}$. As the neuron carries out the time–and–space summation of the input potentials and due to time discretisation during simulation, a more moderate decay can cause more sophisticated information processing. Therefore, we used t_1 up to $5\,\mathrm{ms}$ and t_2 up to $15\,\mathrm{ms}$ (please see figure 1).
  - LATENCY — the time delay of the synaptic transmission and the axonal conduction. In other words, synapse is silent for the latency time before it starts to translate the input impulse into an action potential.
  - SYNAPTIC WEIGHT (SW) — a value from $\langle -1, 1\rangle$ that represents the strength of the synaptic input. An excitatory PSP belongs to a synapse with a positive SW and an inhibitory PSP to a synapse with a negative one.
  - PLASTIC CHANGES — Depending of the synapse type, SW can be influenced by some mechanisms, for instance Hebbian learning or heterosynaptic presynaptic mechanism. In future this can be used to enable online learning. In this work this feature of the model is not used, because the learning during information processing phase is not our goal.
- INSTANTANEOUS MEMBRANE POTENTIAL (MP) — a quantity within the $\langle -1, 1\rangle$ range, determined as the sum of PSPs limited by the non–linear function (figure 2)

$$\mathrm{MP}(t) = \frac{2}{\pi} \cdot \mathrm{atan}\left(\sum\nolimits_{\mathrm{synapses}} \mathrm{PSP}(t)\right) \qquad (2)$$

- THRESHOLD — θ, a value from $\langle 0, 1\rangle$ — that determines the limit for firing.
- SPIKE FREQUENCY — the spike frequency is restricted by the absolute refractory period. This is managed by setting minimum $I_{\min}$ and maximum $I_{\max}$ inter–spike interval for the firing pattern. The standard value we use is $I_{\min} = 1\,\mathrm{ms}$ for the lowest and $I_{\max} = 10\,\mathrm{ms}$ for the highest value. The actual inter–spike interval I_a is determined as:

$$I_a = I_{\max} - (I_{\max} - I_{\min}) \cdot \frac{2}{\pi} \cdot \mathrm{atan}\left(\frac{\mathrm{MP} - \theta}{1 - \mathrm{MP}}\right) \qquad (3)$$

  The spike frequency condition does not allow the neuron to fire sooner, even if the MP exceeds the threshold.

C. Tasks Where JASTAP Has Already Succeeded

JASTAP was primarily designed to model real information flow in human brain. Until now, most experiments with JASTAP intend to prove this behavior. On one hand, [10] shows that the JASTAP networks can handle (even noised) information coded in temporal patterns. On the other hand, [11] shows its ability to recognize and distinguish different spike rates. This is particularly interesting since the standard MLP weights can be seen as the mean spiking rate concept.

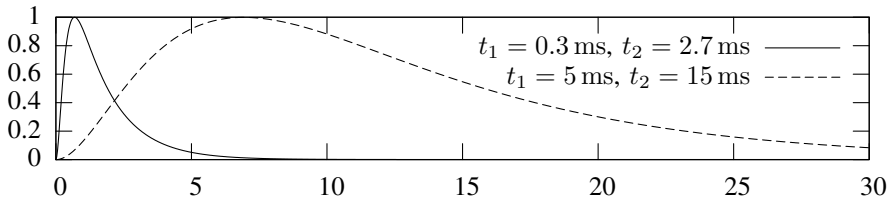
JASTAP model also presents several capabilities that are related with the Gamma distribution [4]. Gamma distribution was chosen for experiments because of biological evidence of its plausibility [12]. The results showed that the JASTAP models are able to make decisions about features (mean rate, coefficient of variation) of Gamma distribution even with very few neuron units. Moreover [4] explored evolved networks and described at low–level how the actual decisions are made.

III. FEASANNT

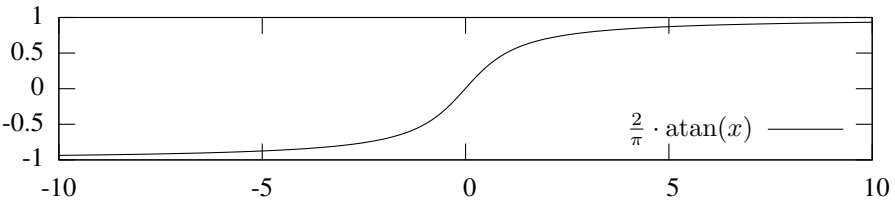
A. What Is FeaSANNT?

One of the major problems in the use of ANNs is to train the frequently large set of parameters (that is, the connection weights). Because most of ANN weight training procedures are based on gradient descent of the error surface, they are prone to sub-optimal convergence to local minima. Due to its large number of parameters, JASTAP model is particularly sensible to this problem. Also, since standard backpropagation learning [2] is still an open issue in spiking neural networks and in JASTAP neural networks in particular, we have applied a more powerful learning model for this problem.

Global search techniques such as evolutionary algorithms (EAs) or simulated annealing are known to produce more robust results in difficult search spaces such as the one presented by JASTAP model. FeaSANNT is an evolutionary procedure for simultaneous solution of the two combinatorial search problems of feature selection and parameter learning for artificial neural network classifier systems [6]. Because FeaSANNT incorporates the use of an embedded approach,



1: Postsynaptic potential (PSP) with different waveform inter alia.



2: Limiting non–linear function

both feature selection and NN parameter learning can be achieved at (relatively) reduced computational costs.

B. Feature Selection in an Embedded Approach

Feature selection can be regarded as a search problem in the discrete space of the subsets of data attributes. Unfortunately, due to the often large set of attributes and their interactions, selecting the optimal feature vector is usually difficult and time consuming. Once again, the search space is noisy, complex, nondifferentiable, multimodal and deceptive and the results are strongly related to the type of classifier system used. However, as pattern classification is based on the information carried by the feature vector, feature selection has a crucial impact on the classifier accuracy, learning capabilities and size. The problem of defining the feature vector stems from the uncertainty about which set of attributes best characterises the data patterns. Finding the optimal one requires removing unnecessary, conflicting, overlapping and redundant features in order to maximise the classifier accuracy, compactness and learning capabilities. Due to the often large number of attributes, searching exhaustively for the best subset is impractical.

Most of current feature selection algorithms either select variables by ranking them according to statistical, geometrical or information theoretical measures or directly evaluate subsets of features by running the induction algorithm and taking the classification accuracy as fitness measure. While the former approach is faster but liable to unforeseen failure as it does not take into account the classifier system, the latter usually implies extensive trial and error guided by hill climbing of the optimisation landscape [13]. Moreover, due to the locality of the search method, algorithms following the second approach are likely to be led astray by noise or to get trapped into secondary peaks of performance.

Two main approaches are customary for selection of the input vector of features, the filter approach and the wrapper approach [13]. The former pre-processes (filters) the initial set of attributes based on desirable properties such as orthogonality or information content while the latter evaluates each candidate feature vector on the learning results of the classifier. Feature extraction routines embedded in the learning algorithm (embedded approach, [13] were also reported even though their application is so far less common. While the filter approach is prone to unexpected failures since it completely ignores the induction algorithm, the wrapper approach involves a far more severe computational effort that may hinder adequate exploration of the search space. Moreover, since the learning results are often dependent on the initialisation of the ANN weights, the latter approach must take into account possibly inaccurate evaluations of the candidate solutions.

FeaSANNT implements an embedded approach in an evolutionary feature selection paradigm. The search is guided by using GAs for learning how to mask inputs. Including the selection of the input features into the learning algorithm of the classifier avoids the computational overheads of repeating the whole ANN training procedure for every evaluation of an individual.

This paper reports on the extension of FeaSANNT model to handle JASTAP Spiking Neural Networks. FeaSANNT's embedded approach was found as particularly useful when a genetic algorithm was used for evolution as well as for learning. This way we were able to use the global nature of the evolutionary search to avoid being trapped by sub–optimal peaks of performance while learning both the best set of features, neuron threshold, synapse weights and synapse latencies.

C. What has been already done?

Previous experiments with FeaSANNT [6], shown that the simultaneous evolution of the input vector and the ANN

weights allows significant saving of computational resources. FeaSANNT algorithm was applied on the popular multi–layer perceptron classifier. Also several experiments were preformed on six real–world numerical data sets that gave accurate and robust learning results. Significant reduction of the initial set of input features was achieved in most of the benchmark problems considered. Examination of the evolution curves revealed selection of the optimal feature vector takes place at an initial stage of the search. FeaSANNT seemed also to compare well with other classification algorithms in the literature. However, the proposed algorithm entails lower computational costs due to the embedded feature selection strategy.

IV. Using JASTAP NN model for Pattern Classification

A. Expected Benefits

Expected benefits of spiking models, like JASTAP for pattern recognition (together with feature selection), include:

a) Smaller network structures: As stated previously, when compared to MLP, JASTAP neurons should require a smaller number of units. There is already some evidence that the decrease in number of neuron units can be seen even in the simple XOR problem [8] when using Spike Response Model [14].

b) Noise filtering: In [10] authors point out that JASTAP is able to extract information with *background spiking noise*. In section V it will shown how much noise JASTAP model can bear when applied to the IRIS dataset. We will be especially interested in processing data with Gamma noise, as this kind of noise that is biologically relevant [12].

c) Results amenable to analysis: A smaller number of units and detailed processing (temporal coding instead of rate coding) give us a chance to take an insight to internal operations at the level of synapses. Simply speaking, we will try to *decode* what is the evolved network actually doing.

B. Encoding Variables into Spike–Time Patterns

Encoding processed data into time patterns is still an open issue even for Neurophysiology. For example, it is not clear if the temporal [15] or rate coding representations play a crucial role in information processing [16].

We found a first mention to this issue in [8]. Authors used *receptive fields* to transform real–world data into patterns. Although this approach can be neurologically plausible, for our (FeaSANNT) purposes it was decided to use a *repetitive* encoding when the same inter–spike interval representing the data value is repeated as an input to input neuron. When the mask element is applied to a particular input, that neuron just receives an empty input train.

Every input is associated with one input neuron as an external synapse. Spike potentials for input neurons are precalculated from input data in such a way that the inter–spike lengths are linearly dependent on the data value (in the range of 5–15 ms). If no noise is taken into account interspike intervals corresponding to one input train are the same. When we test the noise handling, each interspike interval receives a value chosen from the Gamma distribution. This kind of noise is considered as more biologically relevant (rather than applying noise to a particular value) because it indicates cumulative noise during information processing over several neurons (fig. 3).

Decoding to output can be simple. The following method is used: Some neurons in the network are designated as the *output* neurons. When any of the output neurons fire, they are taken as a *hot–spot* network decision. The meaning of the decision is task–dependent. Thereafter the simulation in progress can be stopped, because the subsequent activity has no impact on the final decision.

C. Designing Fitness Function for JASTAP Classification

During the tests preformed the selection of fitness function was a crucial step when evolving (recurrent) JASTAP networks. The fitness function was crucial not only for the number of iterations for evolution (i.e. the time the network took to learn), but also in effectivity of learning.

In the first place, the simple successful ratio based fitness functions gave very poor results. A problem with these functions is that they cannot distinguish between networks with no response from those which are able to fire (even if not correctly). Furthermore local minima are common in these cases. This is the consequence of the emergence of the individuals responding to all inputs in the same way. To avoid such problems it was created a fitness function that favors the following criteria:

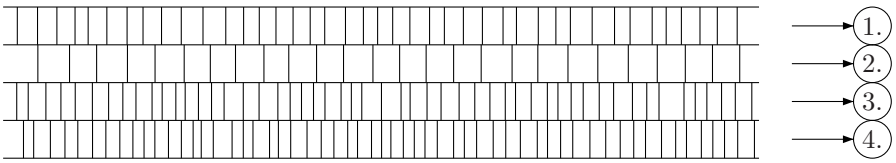
- *early* or non–silent *responses* on input pattern can be classified as incorrect.
- *heterogeneousness*: A network responding (correctly) to many patterns from various classes is better that than a network that can correctly respond to only one. (With same overall ratio).
- *selectivity*: we should evaluate with higher fitness networks responding mostly correctly to one class, even if responding randomly (or not responding) to the other classes. Indeed these networks exhibit selectivity to some class are better that networks which respond to all patterns in the same way (this is often case, when a population is initialized with random weights).
- For datasets with many classes, individuals responding in several classes should be evaluated with higher fitness.

A linear combination of the several (sub–)fitness criteria stated above was used for the final fitness function. First, by favoring small differences in correct responses:

$$c_1 / \Big(1 + \varepsilon - \sum_{i \in \text{classes}} (\text{correct–ratio}[i])\Big)$$

Second, heterogeneousness and selectivity is maintained by emphasizing the minimal fitness, i. e. the fitness based on minimal (with respect to classes) correct responses:

$$c_2 / \Big(1 + \varepsilon - \min_{i \in \text{classes}} (\text{correct–ratio}[i])\Big)$$



3: Example of encoding data to temporal code: first iris–setosa training example (i.e. [5.1, 3.5, 1.4, 0.2] scaled to [7.2, 11.3, 5.7, 5.4]). Each row represents an input to 1 of 4 input neurons. Data are scaled to 5–15 ms and repeated over time period of 300 ms. Every interspike–interval is noised with ± 1 ms of Gamma noise.

Finally, to handle and favor ability to respond in multiclassed datasets, the function gives little credit when the network is able to respond correctly in any pair of classes.

$$c_3 \cdot \sum_{i \neq j \in \text{classes}} \min(\text{correct–ratio}[i], \text{correct–ratio}[j])$$

Experiments were carried out with the values $[c_1, c_2, c_3] = [1, 30, 7]$.

V. Results and Comparisons

A. IRIS Data Set

In order to evaluate results the JASTAP in FeaSANNT framework was tested on IRIS UCI ML [17] well-known benchmark problem. Inputs were converted into temporal code as described in section IV-B.

The Iris data set was firstly presented in a statistical study for determining the classification of three sets of flowers. The classification set has three distinct classes of plants and four numeric features (corresponding to the petal and sepal width and length). A sample of 150 examples were collected (50 for each class). Only one class is linearly separable from the remaining ones, because some data points on the other two classes are intersecting each other (however most of data points are fairly distinct). From statistical evidence only the measures of petal width and length are enough for classification.

In FeaSANNT algorithm only two features were selected with an average accuracy of 94.7. By using the four features with an MLP with backpropagation learning 96.2% of accuracy were achieved.

B. Set–Up

1) Chromosomes: Structure is not evolved so the individual consists of chromosome defining values subject to evolution and the binary mask for feature selection. Gray binary coding was used for the chromosome network values. The evolution of neuron thresholds and synapse weights and latencies is crucial for network function. Following results of [4], JASTAP is not evolved for PSP's t_1 and t_2 parameters $I_{\min}$ and $I_{\max}$ defining bounding firing periods. Table II displays used JASTAP setup.

2) Evolutionary Algorithm: Selection is made in an elitism–like manner: Recombined individuals are evaluated with the same examples as their parents and then the best part (a half) of all chromosomes (parents and offspring together) is taken as the next population.

feature	min value	max value
synapse weight	−1	1
synapse latency	0 ms	40 ms
synapse waveform t_1 value	5 ms	
synapse waveform t_2 value	15 ms	
neuron threshold	0	1
minimal neuron firing period	1 ms	
maximal neuron firing period	10 ms	
maximal network response time	300 ms	

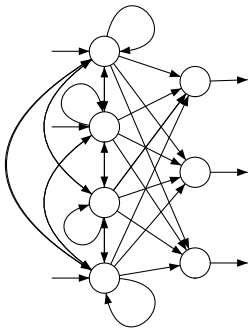
II: Parameter boundaries used for evolution

feature	value
trials	1
generations	100–500
population size	100
values crossover rate	0.7
values mutation rate	0.05
mask mutation rate	0.05

III: Parameter settings of evolutionary algorithm

C. Results

1) Iris data set: This experiment on IRIS data set aimed at showing the potential of JASTAP due to the smaller number of neuron units needed for good classification. Therefore the hidden layer was totally dropped leaving only 7 neurons in structure (4 for inputs and 3 for outputs). In addition we interconnected neurons in input layer to enable mutual information exchange and recurrence. The results are shown in table IV. BPrule stands for classical BP algorithm [2]. ANNT and FeaSANNT refer to the EA algorithm either disabled or enabled feature selection module [6]. Present results are in



4: Iris network

name	source	size	features	classes	training set
Iris	UCI ML	150	4	3	80 % – random

I: Data sets

iris	BPrule	ANNT	FeaSANNT	FeaSTAP
hidden layer size	2	2	2	no
inputs	4	4	1.5	3
accuracy	96.2 %	95.3 %	94.7 %	100 %
iterations	700	300	400	300

IV: Iris data set

Noised Iris	no noise	$1\% \approx 0.1\,\mathrm{ms}$	$10\,\% \approx 1\,\mathrm{ms}$
hidden layer size	no	no	no
inputs	3	3	3
accuracy	100 %	100 %	100 %
iterations	200	200	300

V: Noised Iris data set

FeaSTAP[2] column.

2) Handling the noise: JASTAP model was designed to be as detailed as it is needed to simulate biorealistic functions in reasonable way. With this in mind, we have conducted experiments to test the noise handling during classification (please see subsection IV-B). To keep on being biologically inspired we have used the noise generated from Gamma distribution. The noise value in cortical spike times is about 1–2 ms [5] what is about 10 % due to our encoding. Such a noise can be generated from $\mathscr{G}(\alpha = 25, \beta = 0.8)$ Smaller amounts of noise (at the 1 % level) were also tested.

Gamma distribution generators GS* and GKM1 from [18] were used. These generators use a combination of the acceptance–rejection, composition and inverse transform methods.

This results show that noise levels until the 10 % level don't reduce the classification quality. However the problem becomes more difficult to learn on the 10 % noise level. By increasing the noise level beyond this limits classifier accuracy starts being compromised. It should be noted that noise was applied to each inter–spike interval independently (even in the same input train).

VI. Conclusions

This paper main goal is to show that the basic perceptron model can be replaced by the biologically realistic JASTAP neural network model for classification tasks. In order to do so, an evolutionary procedure for simultaneous solution of feature selection and for JASTAP neural network parameter adjusting was used.

Preliminary experimental results show that not only JASTAP seems to be able to deal with the same learning problems with smaller neural networks, but also JASTAP unique features enable the insertion of artificially generated noise into the learning set without degrading learning performance. Indeed, results on the Iris standard data set [17] use smaller neural networks without compromising accuracy. Also, noise was artificial inserted into training data without any degradation in classification accuracy. However, it was noticed that the feature selection was not so significant as in original FeaSANNT work. The JASTAP network seems to use the spike potentials even from irrelevant inputs just to improve precision. This is probably the reason why 100% precision is achieved on the IRIS data-set. Since this data set is known not to be fully separable, this high value is probably due to a statistical abnormality either in sample data or in the evaluation procedure. Anyway, care should be taken on future experiments to avoid opportunistic overlearning strategies.

Although not discussed in the paper, the behavior of the evolved networks for IRIS classification was also studied. Indeed, it was observed that the networks do the classification by emphasizing differences in the features correlating with the classes *petal length* and *width*.

A major problem with JASTAP learning model was the computational time it took to learn. This paper is presenting a learning model for a difficult problem: there are several parameters to tune, time is directly simulated and we are performing feature selection. As a result the learning of more complicated networks or problems causes very slow learning (i.e. the number of iterations per unit of time decreases). In future work we hope to overcome some of these problems by improving the time simulation and by restricting more the parameters that should be learned. Running FeaSANNT procedure in parallel should help to reduce learning times.

Also in future work, we hope to use JASTAP neural networks to represent logic models as neural networks. Recent work presents the so called neuro–symbolic networks [19]. These systems provide simple ways to represent predicate logic programs as MLP neural networks [20]. Unfortunately, providing simple and adjustable structures for encoding previous knowledge or algorithms in a neural network is a difficult task (e.x. Siegelmann's *Neural Automata and Analog Computational Complexity* article in [7]). Probably a more compact and modular JASTAP neural network would provide a good solution for this problem. In such a neural network, the architecture and some set of weights and parameters will be fixed for encoding background information. For that the relations of the JASTAP model with Gamma distribution [4] should also be studied.

[2]FeaSTAP means JASTAP in FeaSANNT

Several other features of JASTAP model can also be used with advantage for classification tasks. For example the ability for plastic changes (already studied in JASTAP) can be very interesting for online adaptation of the neural network classifier to small changes in the learned model. Because the JASTAP model works with temporal code, we argue that it can more easily discover statistical regularities over longer time in input pattern trains and encode them on the output. This could be useful in classification problems where context is an issue (e.g. [21], [22]).

REFERENCES

[1] T.M. Mitchell: "Machine Learning," McGraw-Hill – Higher Education, 1997.

[2] D.E. Rumelhart, G.E. Hinton, R.J. Williams: "Learning internal representations by error propagation," Parallel distributed processing: explorations in the microstructure of cognition, vol. 1: foundations, pp. 318–362, 1986.

[3] J. Janco, I. Stavrovsky, J. Pavlasek: "Modeling of neuronal functions: A neuronlike element with the graded response," Computers and Artificial Intelligence, vol. 13, pp. 603–620, 1994.

[4] M. Valko: "Evolving neural networks for statistical decision theory," Master's thesis, Comenius University, Bratislava, Slovakia, 2005.

[5] W. Maass, C.M. Bishop, eds.: "Pulsed Neural Networks," Volume 1. MIT Press, Cambridge, MA, USA, 1999.

[6] M. Castellani, N.C. Marques: "A technical report on the evolutionary feature selection for artificial neural network pattern classifiers," CENTRIA Technical Report, 2004.

[7] M. Arbib, ed.: "The Handbook of Brain Theory and Neural Networks,", 2nd. ed., MIT Press, 2003.

[8] S.M. Bohte, J.N. Kok, J.A.L. Poutré: "Error-backpropagation in temporally encoded networks of spiking neurons," Neurocomputing, vol. 48, pp. 17–37, 2002.

[9] S. Redman, B. Walmsley: "The time course of synaptic potentials evoked in cat spinal motoneurones at identified group ia synapses," J Physiol (Lond), vol. 343, pp. 117–138, 1983.

[10] J. Pavlasek, J. Jenca: "Temporal coding and recognition of uncued temporal patterns in neuronal spike trains: biologically plausible network of coincidence detectors and coordinated time delays," Biologia, Bratislava, vol. 56, pp. 591–604, 2001.

[11] J. Pavlasek, J. Jenca, R. Harman: "Rate coding: neurobiological network performing detection of the difference between mean spiking rates," Acta Neurobiol Exp (Wars), vol. 63, pp. 83–98, 2003.

[12] C. Koch: "Biophysics of Computation: Information Processing in Single Neurons," Computational Neuroscience, Oxford University Press. 1998.

[13] A. Blum, P. Langley: "Selection of Relevant Features and Examples in Machine Learning," Artificial Intelligence, vol. 97, pp. 245-271, 1997.

[14] W. Gerstner: "Time structure of the activity in neural network models," Phys. Rev. E, vol. 51, pp. 738–758, 1995.

[15] W. Singer: "Time as coding space?," Curt. Op. Neurobiol., vol. 9, pp. 189–194, 1999.

[16] L. Abbott, T.J. Sejnowski: "Neural codes and distributed representations: foundations of neural computation," MIT Press, Cambridge, MA, USA, 1999.

[17] C.B. S. Hettich, C. Merz: "UCI repository of machine learning databases," 1998.

[18] G.S. Fishman: "Monte–Carlo — concepts algorithms and applications," Springer-Verlag, New York, 1996.

[19] A. Garcez, D. Gabbay, S. Hölldobler, J. Taylor: "Journal of Applied Logic – Editorial," vol. 2, pp. 241–243, 2004.

[20] P. Hitzler, S. Hölldobler, A.K. Seda: "Logic programs and connectionist networks,", Journal of Applied Logic vol. 2, pp. 245–272, 2004.

[21] N.C. Marques, G.P. Lopes: "Tagging with small training corpora," Springer, Lecture Notes in Computer Science, vol. 2189, pp. 63-72, 2001.

[22] M. Castellani, N.C. Marques: "Automatic Detection of Meddies through Texture Analysis of Sea Surface Temperature Maps," EPIA'05-12th Portuguese Conference on Artificial Intelligence, Amilcar Cardoso, Gael Dias, Carlos Bento (eds.), Springer-Verlag, 2005.