Annales UMCS Informatica AI 5 (2006) 121-131

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI

http://www.annales.umcs.lublin.pl/

Sigma-if neural network as the use of selective attention technique in classification and knowledge discovery problems solving

Maciej Huk*
Faculty of Computer Science and Management, Wrocław University of Technology, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland

Abstract

The article presents the most important properties of Sigma-if neuron and neural network, which use a selective attention technique to solve classification problems. Abilities of Sigma-if neuron to perform active aggregation of input signals and to solve linearly inseparable problems are discussed. Variety of conducted experiments, during which Sigma-if network was compared with multilayer perceptron, are also presented. These experiments show benefits from using Sigma-if network instead of MLP, both in classification problems solving and in knowledge discovery from data.

1. Introduction

Selective attention technique make it possible for living organisms more or less knowingly reduce the number of processed incoming signals, which in the face of limited computational abilities of their brains is usually one of the most important conditions of proper behaviour in a very complex environment. From the computational point of view, selective attention property can allow not only for effective reduction of processed information but also to exempt the potential user from necessity of deciding what information is essential to obtain the final solution of a problem – these decisions are made automatically for a new portion of data. According to the John K. Tsotsos's definition: "Attention is a set of strategies that attempts to reduce the computational cost of the search processes inherent in stimuli perception" [1].

2. Sigma-if neuron

The idea of Sigma-if neuron arose from extending the simple perceptron model [2] with the selective attention technique to solve chosen linearly inseparable problems [3,4]. The Sigma-if neuron, in contrast to typical perceptron, aggregates input signals for the given data vector not in one but in

* *E-mail address*: maciej.huk@pwr.wroc.pl

series of given K steps according to the corresponding state graph. It is input connections are divided into K distinguish subsets during the training process. After that, when the neuron's aggregation function value is computed in every *k*-th step of this process the subset of input signals X_k is taken from the environment and processed to determine the current value of the partial activation level φ_k of the neuron. The process continues till the value of φ_k exceeds a given aggregation threshold φ^*. When that condition is met, signals which were not analyzed are ignored, and φ_k is considered as an input value for neuron's activation function *F*. Owing to that, the signal level information coding and even the use of non local activation function (e.g. sigmoidal), do not deteriorate neuron's selective attention abilities. This distinguishes Sigma-if neuron from the previously studied aggregate and fire neurons [5]. The sample scheme of a state graph for a Sigma-if neuron is presented in Figure 1.

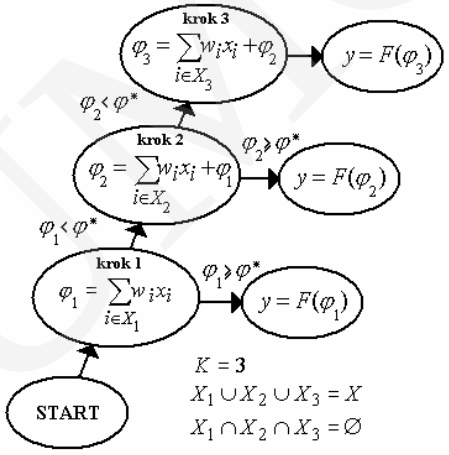

Fig. 1. The sample process of a three-step input signals' aggregation in the Sigma-if neuron

The described model assumes that the state graph used during the signals aggregation is always a simple directed path of nonterminal nodes corresponding to the neural activation accumulation procedure. This is why there is only a need for additional memory to store information about particular division of neuron inputs into *K* enumerated subsets. So in the general case, Sigma-if neuron besides the weights vector *w,* has the additional connections grouping vector θ with one nominal valued coefficient for each neuron input connection.

While there is no simple and effective method for independent searching of weights and grouping vector, the proposed solution assumes, that θ vector coefficients are, in fact, direct functions of weight vector. Weights are established by the well known error backpropagation algorithm, but every ω training epoch actual grouping vector is computed. This is application of a selfconsistency idea widely used in physics. The grouping vector computation

procedure used simply divides input connections into a given number of groups, according to the "the greater connection weight, the smaller connection group number is" principle. Such search problem reduction leads to very interesting results, and to practical elimination of additional parameters vector. But in further analysis of the Sigma-if model it is still very helpful to use the grouping vector concept.

The main property of the described model is an active aggregation of input signals of a given data vector. Namely, the aggregation function of the neuron allows it to determine, which input signals should be taken from the environment and in what order they will be processed. In contrast to it, other models of the neurons considered in literature (except for some modified high level neuron) take and process all the available input signals – this can be treated as passive signals aggregation. Active aggergation allows not only for prioritised information processing but also makes it possible to identify subsets of decision attributes the neuron uses to evaluate its output value. This can facilitate both explaining the way the neuron makes up the decision and a process of knowledge extraction from data – it allows creating lists of attributes significance for conducted classification tasks.

The way input signals are aggregated by Sigma-if neuron can also be considered from another point of view. In particular, a very interesting issue is an evolution of its decision space in each case of data vector processing. During that, the Sigma-if neuron repeatedly try to dissect the decision space with different hypersurfaces of increasing number of dimensions, which depends on the number of elements in the subset of input signals analyzed in a given trial. The maximum number of attempts is determined by a number of input subsets and the final dimension, and what's more – shape of decision space is not known until neuron activation. This property significantly improve classification ability of a one neuron. Sigma-if neuron – unlike the classic perceptron – can solve many of linearly inseparable problems.

As an example let us consider a behavior of a single Sigma-if neuron during solving a classification problem given by an expression:

$$D(x_1,x_2)=\begin{cases}0:(x_1,x_2)\in\{(0,0),(0.8,0.8)\}\\1:(x_1,x_2)\in\{(0,1),(1,0)\}\end{cases} \tag{1}$$

This problem can not be solved by a single perceptron. Figure 2 shows subsequent stages of an evolution of two inputs Sigma-if neuron decision space, during solving it by stepped signals aggregation. To analyze that process, let $K = 2$ be the number of steps of the iterative aggregation, so the two neuronal inputs are divided into two distinguish subsets. For the given aggregation threshold $\varphi^* = 0.4$ and the grouping vector $\theta = [0,1]$, a suitable setting of neuron's weights $w = [1.3, 2.5]$ can divide the linearly inseparable problem into two linearly separable problems. In the first step of classification, since θ_1 is less

than θ_2, only dimension connected to the input signal x_1 is considered. Adequately the weight w_1 and the aggregation threshold φ^* can define the point c. The vertical line crossing point c divides the single dimensional decision space. In this case

$$c = \frac{\varphi^*}{w_1}. \tag{2}$$

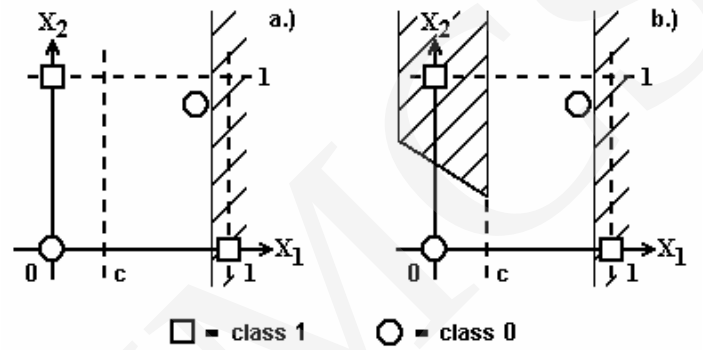

Fig. 2. Subsequent stages of an evolution of decision space for single Sigma-if neuron with two neuronal inputs during signals aggregation, a) division into two problems and solving of a linear problem, b) connection of decision spaces of the first and second stages

The second input x_2 is considered only if the partial activation level of the neuron φ_1 is less then φ^*. Figure 2.b illustrates the final shape of the decision space for a given problem. But when analyzing the above example, it is necessary to remember that in the case of considered Sigma-if neuron influence for the decision space partitioning has additionally bipolar, sigmoidal threshold activation function and connected with it the necessity of discretization of the neuron's output values. It can be also seen that this technique can not solve the XOR problem.

3. Sigma-if neural networks – experiments

The described neuron model can be successfully applied to build synchronous multilayer neural networks, that possess selective attention abilities. In the conducted research, Sigma-if neural networks were built by extending their special case – fully connected multilayer perceptron network (MLP). Suitable modifications were limited to swapping hidden layer perceptrons for Sigma-if neurons. In all experiments neural networks with one hidden layer were considered. The standard input signals coding was used, and the response of the network were computed in the *winner-takes-all* manner.

During the research various UCI Machine Learning Repository test problems were used. The particular choice was made not only for the sake of structure (the number of attributes or classes) or the information content of data sets, but also

for their popularity among researchers (to allow direct comparison of results). The results obtained for the Sigma-if networks were compared with those for MLP and other machine learning methods (e.g. CART, C4.5, ID3). The experiments can be divided into three groups: verification of classification abilities of Sigma-if neural network, comparison and analysis of knowledge extracted from a Sigma-if network and MLP, and investigating the possibilities of interpretation of internal structure of a Sigma-if network.

3.1. Verification of classification abilities of a Sigma-if network

Firstly, various experiments were conducted to establish an optimal number of neurons in a hidden layer of an MLP, according to its generalization error. For such optimally trained MLP networks their basic properties (e.g. generalization, time of information processing) were investigated. The obtained results are averaged outcomes of ten independent 10-fold cross validations, and are presented in Table 1.

Table 1. Properties of MLP networks with optimal the number of neurons in the hidden layer

Problem	Network inputs	Network outputs	Neurons in a hidden layer	Acuuracy for training data [%]	Accuracy for test data [%]	Processing time [μs]	Training epochs
Anneal	109	6	10	99.75	99.57	40.2	400
Br.cancer w.	9	2	35	98.85	98.3	40.8	930
Crx	47	2	20	98.9	85.1	39.9	2200
HeartC	28	5	10	98.4	89.4	17.2	990
Hypoth	55	2	3	97.35	97.55	12.3	910
Iris	4	3	4	98.6	99.3	7.1	470
Monk1	17	2	4	100	100	7.2	12.1
Monk2	17	2	3	100	100	6	12
Monk3	17	2	2	98.92	98.917	4.6	320
Mushroom	125	2	2	100	100	9.45	3.3
Sonar	60	2	30	99.8	90.5	69.5	420
Soybean	134	19	20	99	96.5	106	940
Votes	48	2	2	99.53	97.7	6.3	480
Wine	13	3	10	92.9	95.3	14.9	630

The next step was to train Sigma-if networks, of the same architecture as in the case of MLP networks. The values of parameters characteristic of proposed model were as follows: the number of input groups K – from 1 (equivalent of MLP) to the doubled value of network inputs for a given problem, the aggregation threshold φ^* – from 0.1 to 1.2 and the grouping vector actualization interval ω – from 1 to 100 epochs.

For comparison with MLP networks, Sigma-if neural networks with the suboptimal *K* value were chosen. For a given problem the suboptimal value of *K* was that for which Sigma-if network it generalized better and faster than MLP. The results of this comparison are presented in Table 2.

Table 2. Percentage changes of Sigma if properties with the optimal number of subsets *K* in comparison to the MLP network

Problem	Neurons in hidden layer (nh)	K	Accuracy on training data [%]	Accuracy on test data [%]	Error on training data [%]	Error on test data [%]	Proc. time [%]	Training epochs [%]	Training time [%]
Br.canc. w.	35	7	-0.86	0.2	74	-11.8	-8.3	-44	-49
Sonar	30	31	-0.1	5.19	50	-49.5	-36.7	47	-7
Soybean	20	11	-1.01	1.04	100	-28.6	-47.2	-22	-59
Crx	20	3	-5.76	2.12	518	-12.1	-23.8	-48	-60
Wine	10	14	2.15	4.09	-28.2	-83	-16.1	-24	-36
HeartC	10	16	-5.59	3.02	344	-25.5	-20.4	-41	-53
Anneal	10	7	-0.03	0.03	12	-7	-49	34	-32
Monk1	4	3	0	0	0	0	-9.7	280	243
Iris	4	2	-0.41	-0.1	28.6	14.3	-1.4	19	18
Hypoth	3	5	-0.07	0.11	2.6	-4.5	-26	3,4	-24
Monk2	3	1	0	0	0	0	-1.7	2,5	0,8
Votes	2	7	-1.74	-0.2	368	8.7	-20.6	4,8	-17
Monk3	2	1	0	0	0	-0.2	8.7	0	9
Mushroom	2	1	0	0	0	0	14.3	3	18
Mean for nh≥10			-1,6	2,2	153	-31.1	-28.8	-14	-42.3
Mean for nh<10			-0,32	-0,027	57	2.6	-5.2	44.7	35.4

As can be seen, for the majority of problems (86%) the Sigma-if model was able to obtain results not worse than those of MLP network. However, what is more important, for all considered problems, for which MLP networks demanded 10 or more neurons in a hidden layer, using the Sigma-if neural network make it possible to obtain models generalizing better up to 5% (average 2.2%) and faster up to 49% (average 29%) than their MLP equivalent. The difference is better seen when classification errors are compared: the Sigma-if network makes up to 83% fewer errors on the testing sets (average 31%).

It is also worth to mentioning that in the case of the Sigma-if network a time needed for training the network demanded a fewer number of epochs (average 14%). As a result the time for training the Sigma-if network was 42% shorter. Only for the network with the number of neurons in the hidden layer fewer than 5, using Sigma-if neurons instead of perceptrons has no justification (an exception is the *Hypothyroid* problem demanding network with a large number of connections due to a large number of inputs). The Sigma-if model for very

little networks needs longer time to train (average 35%) and is conductive to the increasing classification error (average 2.6%).

3.2. Analyzing knowledge extracted from the Sigma-if network

Other experiments aimed at comparison of knowledge derived from MLP's and Sigma-if networks, so the decision trees extracted from both trained networks were compared. To build trees, the Trepan algorithm was used, which presently is one of the best algorithms for knowledge discovery from neural networks. This algorithm is also one of the sparse groups of algorithms, which treat a neural network as a black box necessary in the case of the Sigma-if network. The algorithm was used to extract trees or M of N type with the maximum 40 number of nodes.

Table 3. Average sizes and classification performances on testing data sets of decision trees obtained from the Sigma-if networks in comparison to their equivalents obtained from the MLP networks

Nr	Problem name	Average size of tree extracted from Sigma-if network [%]		Average trees classification accuracy on test data [%]		Average networks classification accuracy [%]
		Number of nodes	Number of leaves	Tree source: MLP	Treee source: Sigma-if	
1	Anneal	97.9	98.4	98.7	98.6	99.5
2	BrCancerW	112	110	96.4	96.7	97.8
3	Crx	84.7	86.3	84.9	85.5	86.2
4	HeartC	88.9	90.9	88.5	91.3	90.75
5	Iris	50	60	94.4	94.3	95.2
6	Sonar	96.2	97.6	87.2	92.4	92.8
7	Soybean	97.2	97.3	95.3	95.7	97.1
8	Votes	71.5	73.4	94.1	98.5	97.6
9	Wine	72.1	78.1	91.95	96.96	97.3

As it is shown in Table 3, for the most analyzed problems, decision trees have better characteristics if they are obtained from the Sigma-if network. They have not worse accuracy but have fewer number of nodes and leaves, which makes them easier to understand and interpret by man. Additionally, in the case of four problems (*HeartC*, *Sonar*, *Votes*, *Wine*) such decision trees have not only a simpler structure but also generalization abilities better than in the case of trees extracted from the MLP networks.

As an example of effectiveness of knowledge discovery from the Sigma-if network let us consider one of the trees (M of N type) obtained during the analysis of the *Wine* problem, shown in Figure 3 in the form of pseudocode. This tree is built of 2 nodes and 3 leaves which makes it possible to classify the test

set with 93% accuracy with the use of only 30% of attributes. This result is better than those obtained by known, popular algorithms CART and C4.5tree.

```
if 2 of {Malic_acid > 1,245, Color_intensity > 3,975} then
    if 1 of {Hue ≤ 0,815, Flavanoids ≤ 1,4} then class 3
    else class 1
else class 2
```

Figure 3. The sample decision tree extracted from the Sigma-if neural network for the *Wine* problem

3.3. Investigating the internal structure of the Sigma-if network

Another issue considered during the analysis of the Sigma-if network properties was activity of neural connections of the previously trained Sigma-if networks during processing new data vectors from test data set. An active connection is a connection used to establish neuron's activity level for a particular data vector.

During experiments different quantities describing various network activities were taken into account – e.g. averaged percentage activity of network's inputs (ANI) or averaged percentage activity of hidden layer connections (AHC). Table 4 presents the results obtained for the *Sonar* problem.

Table 4. Average percentage connections activity of the Sigma-if network for the *Sonar* problem, according to the value of activation threshold (number of hidden neurons: 30, $K = 60$; AIN – the percent of active inputs)

Aggregation threshold	Network properties			AHC [%]			ANI [%]			AIN [%]
	acc_test [%]	processing time [μs]	training epochs	avg	min	max	avg	min	max	
0.1	90.4	32.1	820	1.97	0.048	5.6	42.2	37.8	56.6	64.9
0.2	92	33.7	670	2.9	0.39	7.6	52.7	39.5	85.1	91.1
0.4	94.2	39.3	689	6.5	2.2	16	76.3	47.3	100	100
0.6	95.7	48.4	662	13.1	6.8	23.7	93	63.3	100	100
0.8	96.2	63.7	573	22	13.8	33.4	98.5	85.2	100	100
1	95	83.7	507	32.9	23.6	45	99.9	98.2	100	100
1.2	94.8	106.7	481	44.5	35.6	56.6	100	99.8	100	100

It can be noticed that the number of active connections between the input and hidden layer neurons (AHC) influences significantly the time of processing information by the network and the average number of active input connections needed for classification vectors from the data test. What is more, if maximum ANI is small, some of network inputs (AIN) can be not used for any of data vectors from the testing data set. It is a manifestation of Sigma-if network's

naturally appearing mechanism of attribute selection. One can also see how aggregation threshold value influences Sigma-if network properties.

Another use of interneural connections activity is knowledge discovery from data. The example of results for the problem *Wisconsin Breast Cancer* is shown in Table 5. It includes a list of averaged percentage usage of Sigma-if network's inputs during classification. This experiment was conducted for the Sigma-if networks with the optimal value of K and similar accuracy.

Table 5. Inputs average activity of 100 Sigma-if networks for the Wisconsin Breast Cancer problem. The average network test data classification error was 1.5%

Nr	Input data attribute name	Average network input activity [%]	Var.	Var. [%]
1	Bare_Nuclei	46.1	2.5	5.4
2	Uniformity_of_Cell_Shape	44.9	2.4	5.4
3	Clump_Thickness	44.2	2.6	5.8
4	Bland_Chromatin	41.2	0.9	2.3
5	Uniformity_of_Cell_Size	39.3	1.7	4.5
6	Normal_Nucleoli	37.7	3.1	8.3
7	Marginal_Adhesion	36.5	1.5	4.1
8	Single_Epithelial_Cell_Size	36.4	3.2	8.7
9	Mitoses	35.4	2.5	7.1

The above results present not only frequency of usage of particular information during classification, but what is more important, the tendency to favor some of inputs, which makes it possible to identify signals important for the network decision making process. Such network inputs activity analysis can be easily adapted to measure data attributes activity. It is important when data consist of nominal values, and multiple inputs activities can influence suitable attribute activity.

4. Comparison with other methods

For the above presentation completness, it must be shown how the Sigma-if neural network accomplishes classification tasks in comparison with other machine learning methods. Table 6 describes the average generalizations of chosen, most popular machine learning methods on test problems in question. It can be seen that for all problems proposed the solution gives better results (or sometimes similar in case of MLP) than those by other methods.

Table 6. Comparison of generalization of the chosen machine learning methods and the Sigma-if neural network

Nr	Problem name	IB1	CART	C45 tree	ID3	MLP	Sigma-if
1	Anneal	95,1	92,3	94,5	95,5	99,6	99,6
2	Br.Cancer W	96,3	94,4	94,7	94,3	98,3	98,5
3	Crx	81,3	85,5	83,5	80	85,1	86,9
4	HeartC	57	80,8	77,8	49,4	89,4	92,1
5	Iris	95,3	92	94	93,3	99,3	99,2
6	Sonar	86,5	70,7	73	74	90,5	95,2
7	Soybean	90,3	87,4	90,1	88,3	96,5	97,5
8	Votes	92,4	96,3	96,8	94,5	97,7	97,5
9	Wine	94,9	87,6	83,3	83,8	95,3	99,2

Conclusions

In this paper properties of Sigma-if neuron and advantages of using the Sigma-if neural network instead of the MLP network were considered. Single Sigma-if neuron with the sigmoid activation function due to its selective attention abilities can classify chosen linearly inseparable problems. But what is more important, the Sigma-if network for the medium size test problems has better generalization abilities and can process data in a shorter time than MLP.

Additional experiments show that the proposed solution can be also successfully used in applications where knowledge extraction from neural networks is important. The imposed structure of the Sigma-if network, allows storing additional procedural knowledge in neural connection weights at a cost of only one additional parameter φ^*. This has a positive influence on the results of knowledge extraction from neural networks. Decision trees obtained in this process have simpler structures and better generalization abilities when extracted from the Sigma-if networks.

Making use of the proposed selective attention technique not only reduces connections activity, but also introduces new possibility of networks decision processes analysis by their inputs activity interpretation. Simultaneous reduction of the average number of network inputs used during information processing can also reduce possible data acquisition costs.

References

[1] Tsotsos J.K., Culhane S., Wai W., Lai Y., Davis N., Nuflo F., *Modeling visual attention via selective tuning*, Artificial Intelligence, 78 (1995) 507.

[2] Duch W., Jankowski N., *Survey of neural transfer functions*, Neural Computing Surveys, 2 (1999) 163.

[3] Huk M., Kwaśnicka H., *The Concept and Properties of Sigma-if Neural Network*, In: Adaptive and Natural Computing Algorithms, Ribeiro B., Albrecht R.F., Dobnikar A., Pearson D.W., Steele N.C. (eds.), Springer Computer Science, (2005) 13.

[4] Huk M., *The Sigma-if neural network as a method of dynamic selection of decision subspaces for medical reasoning systems*, J. of Med. Inf. & Techn, 7 (2004) 65.
[5] Burkitt A.N., Clark G.M., *Analysis of integrate-and-fire neurons: synchronization of synaptic input and spike output,* Neural Comput., 11 (1999) 871.