A Self-Organizing Map Model for Sequence Classification

Otávio A. S. Carpinteiro and Harry G. Barrow
School of Cognitive and Computing Sciences
University of Sussex
Falmer, East Sussex, BN1 9QH, England
e-mail: otavioc,harryb@cogs.sussex.ac.uk
phone: (01273) 606755 ext. 8524

July 1996

CSRP 424

Abstract

A novel neural model made up of two self-organizing map nets — one on top of the other — is introduced and analysed experimentally. The model makes an effective use of context information, and that enables it to perform sequence classification and discrimination efficiently. It was successfully applied to a set of contrived sequences, and also to a real sequence — the third voice of the sixteenth four-part fugue in G minor of the Well-Tempered Clavier (vol. I) of J. S. Bach. The model has application in cognitive domains which demand classifying either a set of sequences of vectors in time or sub-sequences into a unique and large sequence of vectors in time.

1 Introduction

Several researchers have extended the self-organizing feature map model (Kohonen, 1989) to classify sequential information. The problem involves either classifying a set of sequences of vectors in time or recognizing sub-sequences inside a large and unique sequence. The most recent approaches are described below.

In windowed data approach, as in Kangas' (1994) model, the input vectors are concatenated in a fixed-size window which slides in time. The size of the memory to the past inputs is limited to the size of the window. The approach has well-known deficiencies (Elman, 1990). The most serious is that the model becomes computationally expensive as wider windows are required.

In time integral approach[1], as in Chappell and Taylor's (1993) model, the activation of a unit is a combination of its current input and its former outputs decayed in time. Despite its biological plausibility, the approach suffers from loss of context. Sequences that slightly differ in their initial elements (e.g., abccc, baccc, and bbccc) would probably have the same classification.

Another recent approach is James and Miikkulainen's (1995) model. When a vector in the sequence is input, the output unit in the map which wins the competition for that vector is disabled for further competition, and its activation decays in time to indicate which vector in the sequence it is representing. Each winning output unit represents just a single vector in the input sequence, and so, the representation for the whole input sequence is given by a sequence of winning output units in the map. This distributed representation is more complicated because it has to take in consideration not only the winning units but also their activations. For example, input sequences with the same elements but in different order (e.g., abcd and dcba) will have the same winning units, and so, one has to look at their activations to verify which input sequence the map is representing. Another disadvantage is that the model is unable to recognize sub-sequences inside a large and unique input sequence.

The model introduced here follows the time integral approach. As Chappell and Taylor's model, it is also biologically more plausible. Yet, the time integrator is applied to the input units as opposed to the output units as

[1] Also known as leaky integral approach.

in their model. In the following sections, we shall describe the model, and present two experiments. In the first one, the model is applied to a small scale problem in order to analyse its behaviour. In the second, it is applied to a large scale real example.

2 The model

The model is shown in figure 1. It is made up of two self-organizing maps (SOMs) — one on top of the other. The idea of having neural nets placed in such a hierarchical way is not original. Gjerdingen (1991) has proposed it before, although as far as we know, he has not explored it fully. Our approach differs from his in that we explore the idea using SOM nets, whereas he proposed it for ART2 (Carpenter & Grossberg, 1987) nets.

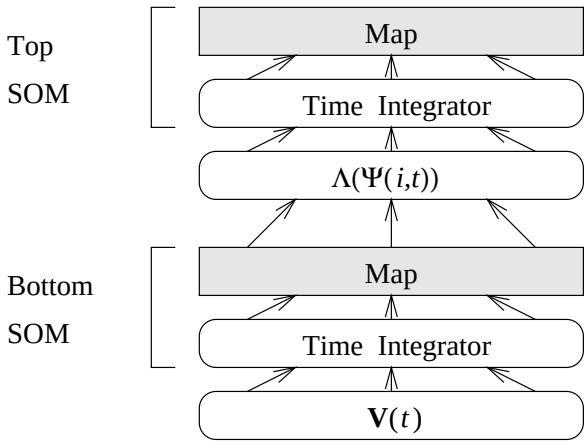

Figure 1: The model

The input to the model is a sequence in time of m-dimensional vectors, $\mathbf{S_1} = \mathbf{V}(1), \mathbf{V}(2), \ldots, \mathbf{V}(t), \ldots, \mathbf{V}(z)$, where the components of each vector are non-negative real values. The sequence is presented to the input layer of the bottom SOM, one vector at a time. The input layer has m units, one for each component of the input vector $\mathbf{V}(t)$, and a time integrator. The activation $\mathbf{X}(t)$ of the units in the input layer is given by

$$\mathbf{X}(t) = \mathbf{V}(t) + \delta_1 \mathbf{X}(t-1) \tag{1}$$

where $\delta_1 \in (0,1)$ is the decay rate. The winning unit i^* in the map[2] is the unit which has the smallest distance $\Psi(i^*,t)$. For each output unit i, the distance $\Psi(i,t)$ between the input vector $\mathbf{X}(t)$ and the unit's weight vector $\mathbf{W}_i$ is given by

$$\Psi(i,t) = \sum_{j=1}^{m} [x_j(t) - w_{ij}(t)]^2 \tag{2}$$

Each output unit i in the neighbourhood N^* of the winning unit i^* has its weight $\mathbf{W}_i$ updated by

$$w_{ij}(t+1) = w_{ij}(t) + \alpha \Upsilon(i)[x_j(t) - w_{ij}(t)] \tag{3}$$

where $\alpha \in (0,1)$ is the learning rate. $\Upsilon(i)$ is the *neighbourhood interaction function* (Lo & Bavarian, 1991), a gaussian type function, and is given by

$$\Upsilon(i) = \kappa_1 + \kappa_2 e^{-\frac{\kappa_3 [\Phi(i,i^*)]^2}{2\sigma^2}} \tag{4}$$

where κ_1, κ_2, and κ_3 are constants which confer the shape to the function. We have set κ_1, κ_2, and κ_3 to be 0.1, 0.7, and 10 in our experiments. σ is the radius of the neighbourhood N^*, and $\Phi(i,i^*)$ is the distance in the map between the unit i and the winning unit i^*. The distance $\Phi(i',i'')$ between any two units i' and i'' in the map is calculated according to the maximum norm,

$$\Phi(i',i'') = max\left\{\left|l' - l''\right|, \left|c' - c''\right|\right\} \tag{5}$$

where (l',c') and (l'',c'') are the coordinates of the units i' and i'' respectively in the map.

The neighbourhood interaction function has proved to be useful, indeed. It provokes two main effects. First, it speeds up the training of the network

[2] Also known as array, grid, or output layer.

by reducing the number of epochs required. Second, it improves the quality of the map by enforcing its topological order (Lo et al., 1991). In rough terms, the neighbourhood interaction function avoids the existence of *local winning units*. The values of the distances $\Psi(i,t)$ increase orderly as the values of the distances $\Phi(i,i^*)$ increase.

The input to the top SOM is determined by the distances $\Psi(i,t)$ of the n units in the map of the bottom SOM. The input is thus a sequence in time of n-dimensional vectors, $\mathbf{S_2} = \Lambda(\Psi(i,1)), \Lambda(\Psi(i,2)), \ldots, \Lambda(\Psi(i,t)), \ldots, \Lambda(\Psi(i,z))$, where $\Lambda(\Psi(i,t))$ is a n-dimensional *transfer function* on a n-dimensional space domain. We have used two different kinds of transfer function in our experiments. Λ can be defined as a gaussian type function as

$$\Lambda(\Psi(i,t)) = e^{-\frac{\kappa[\Psi(i,t)]^2}{\rho^2}} \tag{6}$$

where κ is a constant, and ρ is the radius of the gaussian. The advantage of using such a function is that the contributions to the input of the top SOM depend entirely upon the distances Ψ of the units i, no matter how close or far away they be from the winning unit i^*. The disadvantage is that the accuracy of the input delivered to the top SOM relies heavily upon the quality of the classifications in the map of the bottom SOM. For instance, it is difficult to find a suitable radius ρ to the gaussian when, for each vector $\mathbf{V}(t)$, the distances of the winning units $\Psi(i^*,t)$ vary in a wide range. Alternatively, Λ may be defined as

$$\Lambda(\Psi(i,t)) = \begin{cases} 1 - \kappa\Phi(i,i^*) & \text{if } i \in N^* \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where κ is a constant, and N^* is a neighbourhood of the winning unit. The main advantage is that it is simple to use this type of function. Yet, depending on the radius of the neighbourhood chosen, either pertinent information can be discarded or non-pertinent information be included. We have not considered the situation in which information given by units far away from the neighbourhood N^* is lost, for, as said before, the neighbourhood interaction function avoided the existence of local winning units.

The sequence $\mathbf{S_2}$ is then presented to the input layer of the top SOM, one vector at a time. The input layer has n units, one for each component

of the input vector $\Lambda(\Psi(i,t))$, and a time integrator. The activation $\mathbf{X}(t)$ of the units in the input layer is given by

$$\mathbf{X}(t) = \Lambda(\Psi(i,t)) + \delta_2 \mathbf{X}(t-1) \tag{8}$$

where $\delta_2 \in (0,1)$ is the decay rate.

The dynamics of the top SOM is identical to that of the bottom SOM. In all experiments, we have first trained the bottom SOM, and then, the top SOM, for the sake of efficiency. As the model presented has two SOMs, it will be referenced as *model II* in the rest of the paper. To be better evaluated, the performance of the model II is compared to that of the *model I*. Model I has only one SOM. As the bottom SOM of the model II, model I also has m input units, a time integrator applied on them, and the same dynamics. Model I can be classified as a single SOM which follows the time integral approach. So, as pointed out before, model I suffers from loss of context.

3 First experiment

The first experiment was on mapping a set of sequences. The input data consisted of a set of sixty six-bit binary sequences (e.g., 011101). The sequences were generated randomly. The sequence 001011 was chosen as a reference. It is named *referential sequence* ($\mathbf{S_r}$). The referential sequence has the largest number of similar sequences in the set, that means, sequences which differ slightly from the referential sequence in the order and values of the bits.

The experiment aimed at verifying how accurate the classification of the referential sequence yielded by models I and II was. In other words, we verified the number of sequences in the set which were misclassified by models I and II as the referential sequence.

The two SOMs of model II and the SOM of model I were trained in two phases — coarse-mapping and fine-tuning. The initial learning rate was set to 0.5, and the size of the neighbourhood was set to the size of the map in the coarse-mapping phase. Both the learning rate and the radius of the neighbourhood were reduced linearly to the values 0.01 and 1 respectively. In the fine-tuning phase, the learning rate was kept constant in 0.01, and

the radius in 1. The coarse-mapping phase took 20%, and the fine-tuning phase took 80% of the total number of epochs. The initial weights were given randomly, in the range between 0 and 0.1, to all SOMs.

Different decay rates were tried. In the bottom SOM of model II, they ranged from 0.4 to 0.7, and in the top SOM, from 0.7 to 0.95. In the model I, the decay rate ranged from 0.7 to 0.95. The input layer of the model I and of the bottom SOM of model II held two units. Bits 0's of the sequences were represented as (1,0), whereas bits 1's were represented as (0,1).

Model I was tested with three different map sizes, 9×9, 15×15, and 21×21, trained in 400, 700, and 1000 epochs respectively. In model II, the map sizes were set to 6×6 (trained in 250 epochs) and 9×9 (trained in 400 epochs) to the bottom and top SOM respectively. The transfer function Λ was given by equation 7, with $N^* = \{i^*\}$.

The best results of models I and II are displayed in the tables 1 and 2 respectively. A sequence $\mathbf{S_a}$ is said to have the same classification as that of the referential sequence $\mathbf{S_r}$ if the distance $\Phi(i_a^*, i_r^*) < 2$, where i_a^* and i_r^* are the (last) winning units of $\mathbf{S_a}$ and $\mathbf{S_r}$.

Table 1: Results for model I (first experiment)

Map Size	Decay Rate	No. Miscl.
9×9	0.7	9
15×15	0.7/0.9	5
21×21	0.9	2

Table 2: Results for model II (first experiment)

Decay Rate Bottom SOM	Decay Rate Top SOM	No. Miscl.
0.4	0.7	1
0.5	0.75/0.8	1
0.6	0.8	1

As expected, model I suffers from loss of context and misclassifies several sequences. It is difficult for the model to distinguish variations in the first

bits of a sequence because the contribution of these first bits to the classification of the sequence is very low. For instance, let $\mathbf{S_a}$ = 100000 and $\mathbf{S_b}$ = 010000 be two sequences. Considering a decay rate of 0.8, the activations of the two input units would be 3.362 and 0.328 after the entrance of the last bit of $\mathbf{S_a}$. The activations would be 3.280 and 0.410 for $\mathbf{S_b}$. The differences in the activations between $\mathbf{S_a}$ and $\mathbf{S_b}$ are not relevant, and probably the sequences would be classified as identical by model I.

The problem with model I is that the SOM sees just bits in its input. Yet, its performance would be much improved if the input not only represented bits, but also the context where they appeared. Different input units would then be activated depending upon the order that the bits were input. For example, considering a representation that includes three bits at most, $\mathbf{S_a}$ and $\mathbf{S_b}$ would be represented by table 3. As the representation makes a clear distinction between the beginnings of $\mathbf{S_a}$ and $\mathbf{S_b}$, it helps model I to distinguish between the two sequences as well.

Table 3: Context representation for two binary sequences

Seq.	time: 1	time: 2	time: 3	time: 4	time: 5	time: 6
$\mathbf{S_a}$	(1)	(10)	(100)	(000)	(000)	(000)
$\mathbf{S_b}$	(0)	(01)	(010)	(100)	(000)	(000)

The idea of encoding context in the representation to distinguish variations in sequences is not original. Wickelphones (Wickelgren, 1969) and Wickelfeatures (Rumelhart & McClelland, 1988) are examples of such a representation. Model II also makes use of the representation, and that is the reason why its performance is much superior than that of model I. The top SOM of model II sees bits and over all, context in its input. As opposed to Wickelphones and Wickelfeatures, the representations in the input layer of the top SOM are not handmade beforehand, but instead, they are built up by the bottom SOM. The advantage of this approach is twofold. First, one does not need to worry about encoding context once the bottom SOM is in charge of making an internal representation of context in its map. Second, only the representations required by the application will be built up by the bottom SOM reducing thus, the necessary number of units in the input layer of the top SOM.

The size of context is the size of memory of past inputs, that means,

the maximum number of past input bits that the bottom SOM may recognize. The size of context is directly dependent of the decay rate in the bottom SOM. If the decay rate is very large, the size of the map ought to be increased to recognize properly the large number of different contexts in the representations in the input layer. However, in most cases, a large memory of the past inputs is not necessary. If the decay rate is very small, the existence of the bottom SOM is unnecessary because it merely maps a contextless input of single bits in its map.

We might illustrate these ideas in an example. The sequence $\mathbf{S_a} = 001111$ is presented to the input layer of the bottom SOM. A correspondent sequence $\mathbf{S_a^*}$ of winning units will be activated in the map. A second sequence $\mathbf{S_b} = 101111$ which differs from the first only in the first bit is now presented. A correspondent sequence $\mathbf{S_b^*}$ of winning units is also activated in the map. By comparing the distances Φ (equation 5) between the winning units in $\mathbf{S_a^*}$ and $\mathbf{S_b^*}$, we may trace the memory size for past inputs of the SOM.

Figure 2 displays the distances when using different decay rates in the SOM. Using a decay rate of 0.6, we verify that after entering with the third bit, the SOM is still capable of distinguishing the difference in the first bit between $\mathbf{S_a}$ and $\mathbf{S_b}$. The size of the memory is then three bits. The memory size is two bits for decay rates of 0.4 and 0.5. We have presented sequences of up to three bits to the bottom SOM with decay rate of 0.6 to confirm that the SOM is able to classify them separately. The map is shown in figure 3.

The decay rate of the bottom SOM also affects the outcomes of the top SOM, for the input of the later depends upon the classifications made by the former. We might verify this effect in the experiment. Despite the identical number of misclassifications (table 2), the misclassified sequence changes when varying the decay rate of the bottom SOM. Reducing the decay rate makes the memory size shorter. So, by reducing the decay rate, we would expect that the differences between the misclassified and the referential sequences would move more and more to the first bits. The experiment has indeed shown this behaviour. For decay rates of 0.4, 0.5, and 0.6, the misclassified sequences were 101011, 001001, and 001010 respectively.

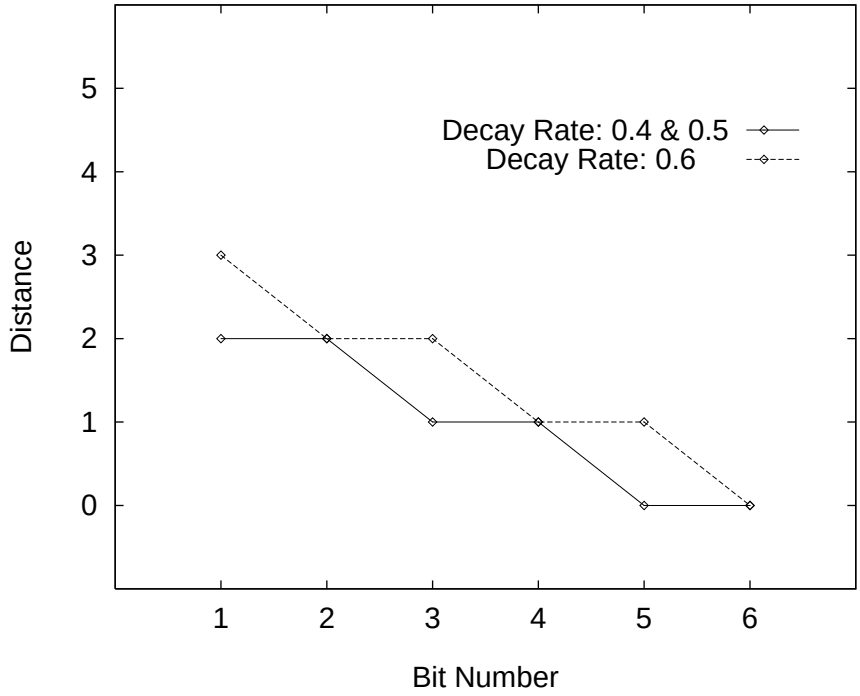

Figure 2: Distances between the winning units of two binary sequences

4 Second experiment

The second experiment was on recognizing sub-sequences into a large and unique input sequence. The input data consisted of a sequence of musical intervals that corresponded to the third voice of the four-part fugue in G minor of Bach (Bach, 1989). The *theme* of the fugue (figure 4), a *referential sub-sequence*, was divided into two parts — *theme I* and *theme II*. Several perfect and modified instances of theme I and II occur in the third voice of the fugue.

The experiment pursued two aims. First, to verify whether models I and II recognize all instances of theme I and II in the third voice of the fugue. Second, to verify whether any other sub-sequence, which was not an instance, was not misclassified as theme I or II.

The training of the two SOMs of model II and the SOM of model I

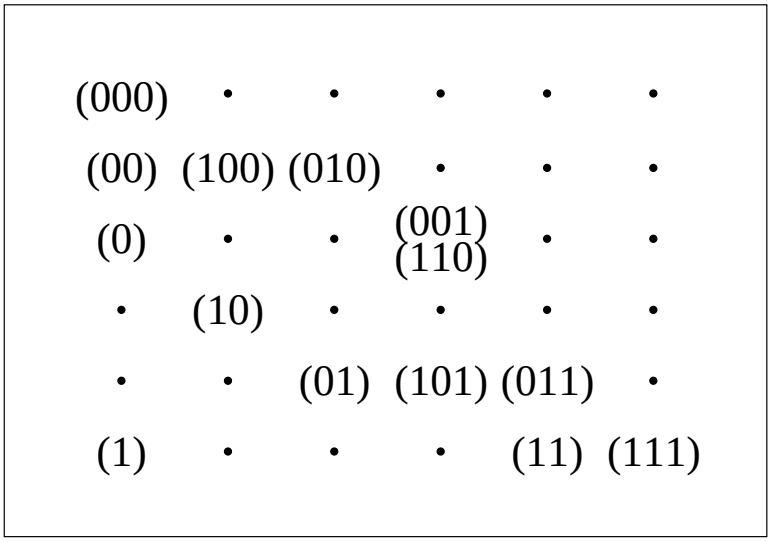

Figure 3: The map for three-bit binary sequences

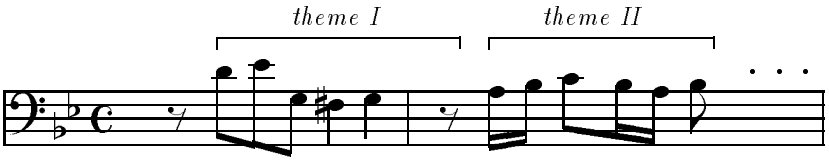

Figure 4: Theme of the fugue in G minor

was identical to that of the first experiment. They were trained in two phases — coarse-mapping and fine-tuning — with the same initial and final learning rates and sizes of the neighbourhood used in the first experiment. Again, in the coarse-mapping phase, the learning rate and the radius of the neighbourhood were reduced linearly whereas in the fine-tuning phase, they were kept constant. The coarse-mapping phase took 20%, and the fine-tuning phase took 80% of the total number of epochs. The initial weights were given randomly, in the range between 0 and 0.1, to all SOMs.

As in the first experiment, we have tested different values for the decay rate. In the bottom SOM of model II, they varied from 0.4 to 0.6, and in

the top SOM, from 0.7 to 0.9. In model I, the decay rate ranged from 0.7 to 0.9. We present here only the results using decay rates of 0.5 and 0.85 respectively for the bottom and top SOM of model II, and 0.85 for the SOM of model I.

The SOM of model I was tested with map size of 18×18, and was trained in 850 epochs. In model II, the map sizes were set to 15×15 (trained in 700 epochs) and 18×18 (trained in 850 epochs) to the bottom and top SOMs respectively. The transfer function Λ was a gaussian type function, given by equation 6, with κ and ρ set to 10 and 0.05.

The input layer of the model I and of the bottom SOM of model II held fifteen units, one for each musical interval ranging from an octave down to an octave up. The activation of these units was dependent of the rhythm as well. We chose the small figure in the fugue to be the *time interval (TI)*. Thus, all other figures are multiple of TI. For example, if TI is an eighth note, a quarter note lasts two TIs, a half note lasts four TIs, and so on. We also defined a counter, *time interval counter (TIC)*. It is the unit in which the fugue is measured.

The fugue in G minor has 544 TICs, and TI is a sixteenth note. At each TIC, either there is a rest, or a note is onset, or a note is sustained. If there is a rest, none of the input units receives activation. If a note is onset, as it makes up an interval, the unit corresponding to that interval receives an activation of 1.0. If it is sustained, the unit which corresponds to the interval receives 0.5.

A sub-sequence $\mathbf{S_a}$ is said to have the same classification as that of the theme $\mathbf{S_t}$ if the distance $\Phi(i_a^*, i_t^*) < 2$, where i_a^* and i_t^* are the (last) winning units of $\mathbf{S_a}$ and $\mathbf{S_t}$. In fact, if $\mathbf{S_a}$ is also an instance of the theme, when $\mathbf{S_a}$ and $\mathbf{S_t}$ have the same classification, not only i_a^* and i_t^* are adjacent but the winning units of $\mathbf{S_a}$ converge TIC by TIC to the winning units of the theme $\mathbf{S_t}$. The error of the instance $\mathbf{S_a}$ is given then by calculating the sum of the distances between each winning unit of $\mathbf{S_a}$ and its corresponding in $\mathbf{S_t}$. The mean error is given by the sum of the errors of each instance divided by the number of instances.

Tables 4 and 5 display the classifications and misclassifications of both models. One may verify in the table 4 that the mean errors of model I are lower than those of model II. The reason is that, as expected, model II takes into a better account the past context. As the previous context varies

from instance to instance, model II takes more time to discard the previous context of the instance to converge to the theme. One may also verify that both models fail in recognizing three instances of theme II. In all of these cases however, theme II either was preceded by a modified theme I or was not preceded by theme I at all. Once more, the different previous context was responsible for that failure.

Table 4: Classifications of model I and II (second experiment)

Theme	No. Instances	Model	No. Failures	Mean Error
I	4	I	1	36.25
		II	0	63.00
II	6	I	3	24.67
		II	3	25.50

Table 5: Misclassifications of model I and II (second experiment)

Theme	Model	No. Minor Miscl.	No. Major Miscl.
I	I	4	1
	II	0	0
II	I	3	4
	II	2	0

The performance of model II is better appreciated in the results displayed in table 5. The fugue is made up mostly by contiguous intervals (e.g., seconds and thirds up and down) in different orders and rhythms. It is worthwhile to observe the fact that any sub-sequence which has either some intervals of the theme in any order or rhythm, or all the intervals of the theme but in a different order or rhythm is not an instance of the theme, and so, must not be classified as such. The number of occurrences of this kind of sub-sequences in the fugue is high, and so is the probability that any neural model has of making misclassifications. Model II however, had only two cases of minor misclassification[3]. Model I suffered from loss of context, and seriously misclassified five sub-sequences which contained intervals which

[3]We consider a case of minor misclassification when the model keeps on classifying as theme the next few TICs which follow the theme.

were also present in the theme.

5 Conclusion

A novel neural model made up of two self-organizing map networks — one on top of the other — is presented. It has application in domains which demand classifying either a set of sequences of vectors in time or sub-sequences into a unique and large sequence of vectors in time. The model makes an effective use of context information, and that enables it to perform sequence classification and discrimination efficiently. Despite the good results, it is still open to further research. In principle, the model could have any number of self-organizing map nets — the more nets, the more similar and longer the sequences of vectors in time which could be recognized.

Acknowledgements

This research was fully supported by CAPES, Brazil.

Reference

Bach, J. S. (1989). *Das Wohltemperierte Klavier*. Vol. 1. BWV 846–869. Bärenreiter Kassel, Basel, Germany.

Carpenter, G. A., & Grossberg, S. (1987). ART2: self-organization of stable category recognition codes for analog input patterns. *Applied Optics*, *26*(23), 4919–4930.

Chappell, G. J., & Taylor, J. G. (1993). The temporal Kohonen map. *Neural Networks*, *6*, 441–445.

Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, *14*, 179–211.

Gjerdingen, R. O. (1991). Using connectionist models to explore complex musical patterns. In Todd, P. M., & Loy, D. G. (Eds.), *Music and Connectionism*, pp. 138–149. The MIT Press, Cambridge, MA.

James, D. L., & Miikkulainen, R. (1995). SARDNET: a self-organizing feature map for sequences. In Tesauro, G., Touretzky, D. S., & Leen, T. K. (Eds.), *Proceedings of the Advances in Neural Information Processing Systems*, Vol. 7. Morgan Kaufmann.

Kangas, J. (1994). *On the Analysis of Pattern Sequences by Self-Organizing Maps*. Ph.D. thesis, Laboratory of Computer and Information Science, Helsinki University of Technology, Rakentajanaukio 2 C, SF-02150, Finland.

Kohonen, T. (1989). *Self-Organization and Associative Memory* (Third edition). Springer-Verlag, Berlin.

Lo, Z., & Bavarian, B. (1991). Improved rate of convergence in Kohonen neural network. In *Proceedings of the International Joint Conference on Neural Networks*, Vol. 2, pp. 201–206.

Lo, Z., Fujita, M., & Bavarian, B. (1991). Analysis of neighborhood interaction in Kohonen neural networks. In *Proceedings of the Fifth International Parallel Processing Symposium*, pp. 246–249.

Rumelhart, D. E., & McClelland, J. L. (1988). On learning the past tenses of english verbs. In McClelland, J. L., Rumelhart, D. E., & the PDP Research Group (Eds.), *Parallel Distributed Processing*, Vol. 2, chap. 18, pp. 216–271. The MIT Press, Cambridge, MA.

Wickelgren, W. A. (1969). Context-sensitive coding, associative memory, and serial order in (speech) behavior. *Psychological Review*, *76*, 1–15.