Dynamic Proximity of Spatiotemporal Sequences

David Horn
School of Physics and Astronomy
Raymond and Beverly Sackler Faculty of Exact Sciences
Tel Aviv University, Tel Aviv 69978, Israel
horn@post.tau.ac.il
Gideon Dror
Department of Computer Science
The Academic College of Tel-Aviv-Yaffo
Tel Aviv 64044, Israel
gideon@mta.ac.il
Brigitte Quenet
Lab. d'Electronique
Ecole Superieure de Physique et Chimie Industrielles
Paris 75005, France
brigitte.quenet@espci.fr

***Abstract*— Recurrent networks can generate spatiotemporal neural sequences of very large cycles, having an apparent random behavior. Nonetheless a proximity measure between these sequences may be defined through comparison of the synaptic weight matrices that generate them. Following the Dynamic Neural Filter (DNF) formalism we demonstrate this concept by comparing teacher and student recurrent networks of binary neurons. We show that large sequences, providing a training set well exceeding the Cover limit, allow for good determination of the synaptic matrices. Alternatively, assuming the matrices to be known, very fast determination of the biases can be achieved. Thus a spatiotemporal sequence may be regarded as spatiotemporal encoding of the bias vector. We introduce a linear Support Vector Machine (SVM) variant of the DNF in order to specify an optimal weight matrix. This approach allows us to deal with noise. Spatiotemporal sequences generated by different DNFs with the same number of neurons may be compared by calculating correlations of the synaptic matrices of the reconstructed DNFs. Other types of spatiotemporal sequences need the introduction of hidden neurons, and/or the use of a kernel variant of the SVM approach. The latter is being defined as a Recurrent Support Vector Network (RSVN).**

I. Introduction

A recent paper [1] has introduced the concept of a Dynamic Neural Filter (DNF), a recurrent network projecting input space into spatiotemporal binary sequences. The one-step dynamics of a binary recurrent network of N neurons is defined by

$$n_i(t+1) = H(h_i(t+1)) = H(\sum_j w_{ij} n_j(t) + b_i) \qquad (1)$$

where n_i is the activity of neuron i whose binary values may be either 0 or 1. w_{ij} is the synaptic coupling matrix and b_i are the biases of the system [1] . H is the Heaviside step function taking the values 0 for negative arguments and 1 for positive ones. We will use these dynamics to generate large families of sequences. The special viewpoint of the DNF is that each spatiotemporal sequence of that kind is encoded by the specific structures w_{ij} and b_i that were used to generate it. Thus we may encounter a binary spatiotemporal sequence of $N = 40$ neurons and $T = 1000$ time-steps, whose 40,000 entries can be encoded by the 1640 parameters of the relevant DNF. Whereas the complexity of the spatiotemporal sequence may seem uncontrollable, it looks more manageable when being recast in terms of its generators. The particular issue that we will discuss is that of proximity of such sequences, defining the latter in terms of the generating parameters (codes) instead of the spatiotemporal entries.

It is well known [2] that symmetric $\mathbf{w}$ matrices lead to fixed-points or two-cycles, and anti-symmetric matrices can lead [3], at most, to four-cycles. Nonetheless, completely asymmetric matrices [4] can lead to arbitrary large cycles in the large N limit. Studying DNF structures with relatively low N values, we have shown [1] that one may, indeed, obtain large cycles with asymmetric matrices, provided the biases are small. Moreover, we have demonstrated the occurrence of many different sequences, in the thousands for an asymmetric $\mathbf{w}$ matrix when $N = 5$. Asymmetry may be defined by the parameter[1]

$$\alpha = \frac{\sum_{ij} w_{ij} w_{ji}}{\sum_{ij} w_{ij} w_{ij}} \qquad (2)$$

that varies between 1 and -1, with the extremes characterizing the symmetric and anti-symmetric cases. The completely asymmetric case corresponds to $\alpha = 0$.

It should be noted that for each time step the system finds itself in a state $\{n\}$, one of 2^N states. Once we turn to large N values, e.g. $N = 40$, the volume of the space of all states becomes enormous. Moreover, choosing an asymmetric $\mathbf{w}$ and small biases we guarantee obtaining very complex spatiotemporal behavior. Biases are in general limited to lie

[1] In [1] we have used $b_i = R_i - \theta_i$, putting emphasis on the external inputs R_i fed into the neurons. For simplicity we use only one parameter here.

within the range

$$-\sum_j w_{ij}H(w_{ij}) \leq b_i \leq -\sum_j w_{ij}H(-w_{ij}), \qquad (3)$$

otherwise they lead to trivial results, dominating the dynamics of the neurons. Choosing them to lie in the middle of this range, i.e.

$$b_i \approx -\frac{1}{2}\sum_j w_{ij} \qquad (4)$$

guarantees generating the largest possible cycles. This is demonstrated in the next section where we present numerical studies of an $N = 40$ model with $\alpha \approx 0$. We show there that under such conditions one generates very long sequences. Moreover, slight changes in b_i may lead to new sequences. If this is the case, a divergence occurs at one time step, after which the sequences don't bear any resemblance to each other. This is so because, given the large volume of state space (2^N), a small shift of a sequence by one state leads to a completely new path in this space. In other words, sequences generated by near-by biases (reflecting near-by external inputs) may bear no similarity to one another. Nonetheless, sequences generated by a DNF contain information that allows for the reconstruction of the DNF. We will use this fact to define a proximity measure for such sequences.

Given any sequence one may use [1] a perceptron algorithm to reconstruct a DNF that can generate it. This algorithm is described in the next subsection. A precondition for this algorithm to work is that the given sequence obeys linear separability, in a sense to be specified below. This may be guaranteed under certain conditions (see in subsection B below) and is easily obtained if a teacher DNF is employed to create the sequence. This is the method we use in section 2 where we study an $N = 40$ example. For large N, the linear separability conditions can be easily met for randomly generated sequences.

A. The Perceptron Algorithm

Here we recapture a straightforward algorithm for building the DNF from a given sequence [1]. Let us start by defining, for each neuron i, an $N+1$ dimensional vector of perceptron weights $\vec{w}^i$

$$(\vec{w}^i)_j = w_{ij} \quad \text{for } j = 1, \cdots N \qquad (\vec{w}^i)_{N+1} = b_i. \qquad (5)$$

Let us define vectors $\vec{x}^i(t)$, for each neuron i, as follows

$$(\vec{x}^i(t))_j = n_j(t)(2n_i(t+1)-1) \quad \text{for } j = 1, \cdots N\ , \qquad (6)$$

$$(\vec{x}^i(t))_{N+1} = (2n_i(t+1)-1).$$

The vector $\vec{x}^i(t)$ represents the state of all neurons that form the input at time t to neuron i, weighted by a positive or negative sign depending on whether the target neuron i at the next time step is 1 or 0 respectively. With these definitions, all constraints of the problem at hand can be written as T perceptron inequalities

$$\vec{w}^i \cdot \vec{x}^i(t) > 0 \quad \text{for all } t = 0, 1, \cdots T-1. \qquad (7)$$

Now one can use the perceptron learning rule [2], [5]

$$\Delta\vec{w}^i = \eta\vec{x}^{i,k}(t)H(-\vec{w}_i \cdot \vec{x}^{i,k}(t)) \qquad (8)$$

while iterating the system, time and again, over all T states. The Heaviside function guarantees that $\vec{w}^i$ gets modified at a given iteration by the vectors $\vec{x}^i(t)$ that do not satisfy the inequality. This algorithm converges [2] if the system of inequalities is soluble.

Adhering to a notation where all vectors lie in an $N+1$ dimensional space we define also

$$(\vec{n}(t))_i = n_i(t) \quad \text{for } i = 1, \cdots N \qquad (\vec{n}(t))_{N+1} = 1, \quad (9)$$

which allows rewriting Eq. 1 as $n_i(t+1) = H(\vec{w}^i \cdot \vec{n}(t))$.

B. Network Size

Given data of N neurons in T time steps one may wonder if they could be generated by a DNF. It is straightforward [1] to show under which conditions the sequence would fit a DNF even if the states were generated randomly. Each neuron has to satisfy T inequalities in an $N+1$ dimensional space. Hence, strict matching for any set of such data requires (see, e.g., [2])

$$\mathcal{A}: \qquad N \geq T-1. \qquad (10)$$

In the large N limit one can apply Cover's result [6], saying that a solution may be found for

$$\mathcal{B}: \qquad T \leq 2(N+1) \qquad \text{or} \qquad N \geq \frac{1}{2}(T-2). \qquad (11)$$

This would be the case when the sequence is constructed of random states. Clearly, if the sequence is generated by a (teacher) recurrent network, the inequalities expressing linear separability will be obeyed for any length of the sequence. In other words, trying to reconstruct a DNF from given data, we may run into problems at $T \approx 2N$, if the data were not generated by a DNF. Otherwise, we should be able to reconstruct the DNF, and improve on it, as T increases.

II. Numerical studies

A. Apparent Randomness

We run a network of $N = 40$ neurons with $w_{ij} \sim \mathcal{N}(0,1)$ and b_i chosen according to Eq. 4. Under these conditions, the average activity of the neurons is 1/2, and the resulting spatiotemporal sequence has a very long cycle. A characteristic raster plot of this system is shown in Fig. 1. This is the kind of spatiotemporal pattern we wish to study. Checking 1200 time steps, we find that the sequence never repeats itself, i.e. it belongs to a cycle that is larger than 1200.

A very large cycle, whose states belong to a space of size 2^{40}, gives the impression of apparent random behavior. This behavior is also reflected in another aspect of the system. As implied by its name, the DNF may be viewed as a dynamic filter from external input space, or bias space formed by of all b_i, to its spatiotemporal representation for a given fixed $\mathbf{w}$. If the resulting spatiotemporal sequence has an apparent random structure, one may expect small shifts in b_i to lead to divergence of the resulting sequence. This is indeed the case, as can be seen in Fig. 2. Here we show the Hamming distance

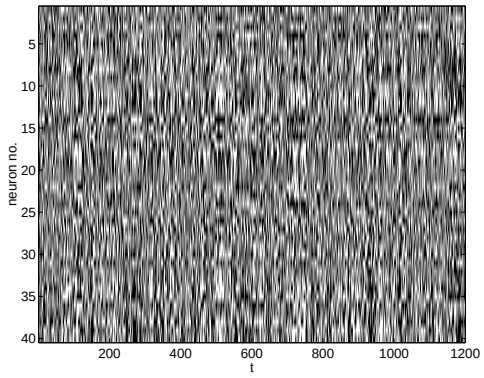

Fig. 1. A raster plot of an $N = 40$ DNF, demonstrating a spatiotemporal sequence over $T = 1200$ time steps without any repetition.

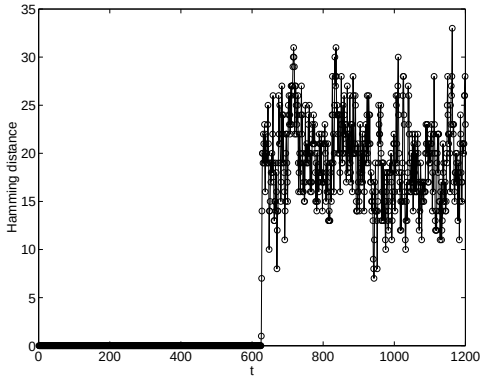

Fig. 2. Hamming distance between two $N = 40$ spatiotemporal sequences, generated by sets of b_i values that were identical but for a slight change in one of them.

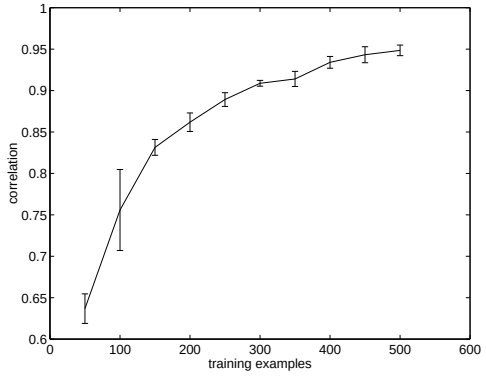

Fig. 3. The correlation between teacher and student DNF in the $N = 40$ problem, using the perceptron algorithm in 10 different teacher-student experiments.

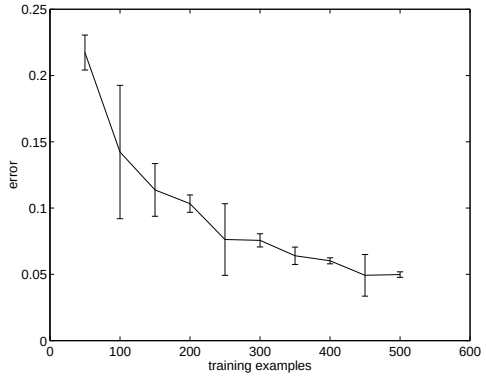

Fig. 4. Convergence of the errors of student networks on one-step prediction tasks.

between two sequences generated by the same $N = 40$ DNF. All b_i values for both sequences were identical, but for one that was slightly shifted. The resulting sequences started diverging very late, because the shift in the parameter was very small, but once they diverged they quickly reached the maximal Hamming distance, which, in a system of binary 1/0 neurons whose average activity is 1/2, is $N/2$.

B. Reconstructing a DNF from its Sequences

As explained in the Introduction we will associate a spatiotemporal sequence with the DNF generating it. To recapture the $\mathbf{w}$ matrix of the DNF we use the perceptron algorithm. Thus the DNF that served to generate the sequence may be regarded as a teacher network training DNF students. The results of this training are shown in Fig. 3 where we display averages of the correlations $\vec{w}^i$ for each neuron, $1/N \sum_i corr(\vec{w}^i_T \vec{w}^i_S)$, between teacher (T) and student (S) recurrent networks. The training converges nicely after a few hundred steps. The error bars represent standard deviations over 10 different student-networks, trained on 10 different sequences of the type shown in Fig. 1. Another measure of successful learning is given by the results of one-step predictions of the student. By one-step prediction we refer to the final state $\{n(1)\}$ produced by the system for a random initial state $\{n(0)\}$. Comparing the results of the student with those of the teacher, and averaging over 10 different sessions of teacher-student tests, we obtain the errors shown in Fig. 4, displaying rapid convergence as function of the training period.

Thus we see that the given temporal sequence of Fig. 1 defines, through the perceptron algorithm, a quite unique $\mathbf{w}$ matrix. Its uniqueness depends on the size of the training set, i.e. the number of time steps of the given sequence. The smaller this size, the wider will be the range of $\mathbf{w}$ that can reproduce a given sequence. One should notice that, in any case, the matrix $\mathbf{w}$ is defined only up to a set of N scale transformations. As can be seen from Eq. 1, every factor multiplying any $\vec{w}^i$ of Eq. 5, leads to the same dynamics. This arbitrariness of scale is factored out once we calculate the correlations $1/N \sum_i corr(\vec{w}^i_T \vec{w}^i_S)$ between teacher (T) and student (S) displayed in Fig. 3.

It is important to realize that sequences may be very uncorrelated in their spatiotemporal structure and yet may be very close in their generating $\mathbf{w}$s. Consider a sequence of the type shown in Fig. 1. Generating a $\mathbf{w}$ from its first 500 time steps, and another $\mathbf{w}$ from the next 500 steps, we obtain correlations close to 1 between these two $\mathbf{w}$s, as expected from Fig. 3. Yet the two spatiotemporal patterns are completely uncorrelated as far as their neural realization is concerned. In other words, while Hamming distance shows no proximity the dynamics are as close as they can be! This exemplifies the power of the DNF as a tool for defining proximity between

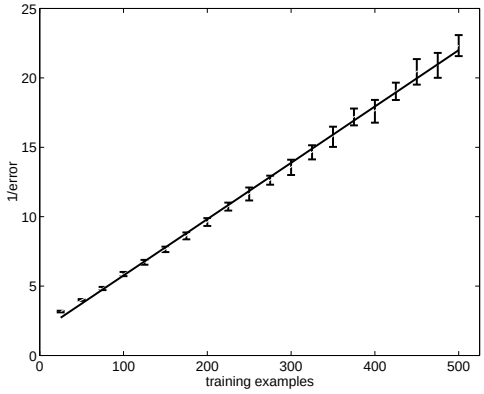

Fig. 5. The inverse error is proportional to the number of training examples, as shown here for the averages of 10 teacher-student experiments.

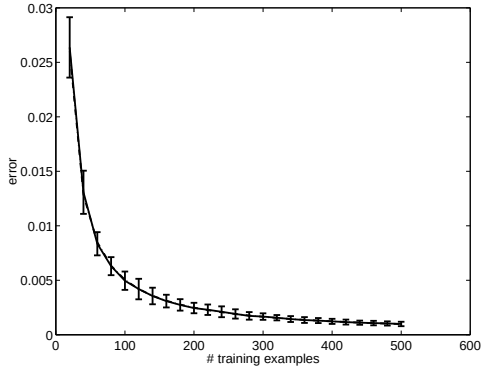

Fig. 6. The error of reconstructing the bias as function of the number of training examples. Here it is assumed that the synaptic matrix is known. Shown are averages and standard deviations of 10 teacher-student experiments.

spatiotemporal patterns.

C. The Generalization Error

In Fig. 4 we have tested the performance of the student networks on examples on which they have not been trained. Hence the error measured here is the generalization error. Given the fact that the problem we study can be viewed as a set of perceptrons that are being trained together, as is quite evident from the perceptron algorithm that we use in the training procedure, we may expect the generalization error to follow that of the perceptron. It is well known [7] that once the number of training examples, presented here by the time duration T of the spatiotemporal training sequence, is larger than twice the dimension of the input samples $T > 2N$, the asymptotic behavior sets in and the error decreases as $1/T$. Testing the large T range in Fig. 5 we find that this rule applies to our system as expected.

D. Reconstructing the Biases

Of particular interest may be the problem of bias reconstruction. Imagine the following encryption problem: given a weight matrix, that is known to both sender and receiver of an encrypted message, the sender transmits a spatiotemporal sequence from which the receiver has to extract the vector of biases, which is the encrypted message. How long should this sequence be?

Obviously the error in the extracted bias is expected to behave like the generalization error of the previous subsection. But, given the much smaller number of variables, we expect it to converge much faster. The numerical results are shown in Fig. 6. The averages decrease in proportion to the inverse of T, the length of the transmitted sequence. The scale of the error is, as expected, much smaller than that of Fig. 5 because of the much simpler problem.

E. Reconstructing a DNF from One-Step Data

Although the main emphasis of this paper is on sequences, we digress in this subsection to point out that learning a teacher DNF, can be better done by using random initial states for one-step dynamics, than by employing a long sequence. This

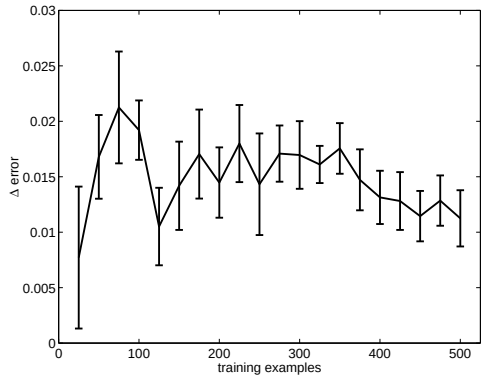

Fig. 7. The difference between errors in networks trained on sequences and those trained on random one-step trials shows the advantage of the latter scheme.

is shown in Fig. 7, where we compare the average errors of ten students using random one-step pairs, with those using data generated by a sequence. In the figure we plot the difference between errors of the latter and the former. This suggests there is an advantage to using random pairs. The reason for this difference is that the states read off from a sequence are biased by the fact that they are generated by the teacher. The weights $\mathbf{w}$ of the teacher induce correlations in input samples of the student in the sense that the activities $n_i(t)$ and $n_j(t)$ are correlated due to the generating weights $\vec{w}^i$ and $\vec{w}^j$. This is in accord with known understanding of the dynamics of recurrent neural networks [10], [11], [12].

III. The Optimal Network

In Fig. 3 we see that an increase in training examples leads to better and better correlation of the student DNF with the teacher DNF. It is quite evident that, for a given number of training examples, there exists a large number of possible $\mathbf{w}$ matrices that can generate it. One may then ask if there exists an optimal choice of $\mathbf{w}$, not necessarily the one generated by the perceptron algorithm that we have used so far. Optimality may be defined as stability toward small (noisy) change of data. In this case, the answer can be obtained by using the maximal margin approach of Support Vector Machines (SVM) [8], [9]. This is a well-developed theory within which one can

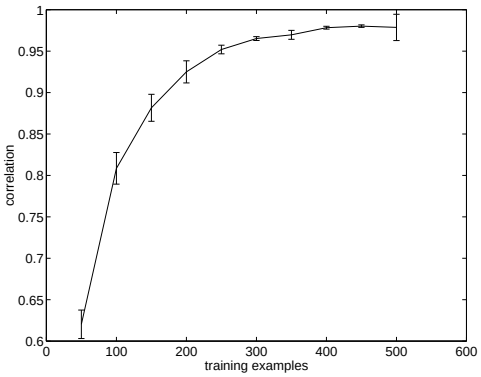

Fig. 8. The correlation between teacher and student DNF using the SVM learning algorithm. The error bars represent standard deviations of 10 different teacher-student experiments.

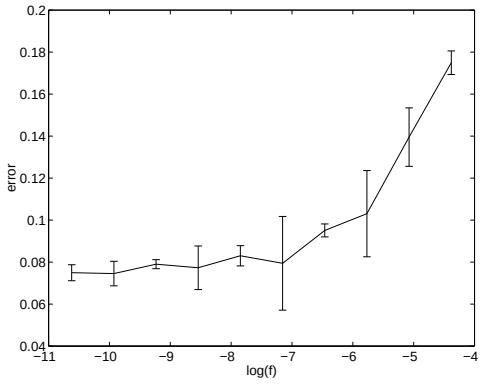

Fig. 9. Error of student performance in an $N = 40$ problem as function of the noise in its data. The training sequence is of length 250. f is the error frequency in the data. The highest noise value corresponds to 1/80, i.e. half a flip on average for each state of the trained system.

discuss and describe issues of classification in general and linear separability in particular.

A teacher DNF produces a sequence that is linearly separable. The data that the student network is presented with consists of all initial state vectors $\vec{n}(t)$, with t varying from 0 to $T-1$, and their corresponding final (target) values that are specified by $\vec{n}(t+1)$. To cast our problem into the SVM formulation consider the set of all input samples $\vec{n}(t)$, and their corresponding target values $n_i(t+1)$ on neuron i as expressed by $\vec{x}^i(t)$ in Eq. 6. They form a linearly separable data set in the N-dimensional input space, separated according to whether the target value is 0 or 1. The SVM algorithm specifies the weight vector of neuron i in terms of a few of these inputs,

$$\vec{w}^i = \sum_{t=0}^{T-1} \beta^i(t)\vec{x}^i(t). \tag{12}$$

The values of the parameters β are obtained by solving a quadratic optimization problem [8], [9]. Most of the β values vanish. The few positive weights $\beta^i(t)$ specify the t values which define the support vectors of the problem of neuron i. The resulting $\vec{w}^i$ is guaranteed to specify the largest margin, i.e. this is the direction such that the hyperplanes perpendicular to it, that close on the two classes of data, are separated by the largest distance (margin).

Using the SVM algorithm to solve the problem of Fig. 3, we obtain the results shown in Fig. 8. Although the results of the SVM algorithm are slightly better, their general trend is the same as that of the perceptron algorithm solution. Also here the correlations start out low, because, whereas the student finds an optimal solution (in the SVM sense) the teacher is not optimal. In fact, a DNF with a given $\mathbf{w}$, leads through any particular sequence that it generates, to a family of different optimal solutions as function of the length T of that sequence. Only in the limit $T \to \infty$ does the optimal solution converge to that of the teacher DNF. For a moderate value of training examples it may approximate it quite well.

IV. The Noisy Spatiotemporal Sequence

An interesting case is that of a spatiotemporal sequence that is DNF generated, but is subject to a noisy channel, inverting some fraction of its bits. The question is if, given such a sequence, one can rebuild the original $\mathbf{w}$.

Clearly some of the errors will just mislead the learning algorithm, but others may introduce violations of linear separability. The latter pose a problem for the perceptron algorithm, but can be readily handled by the SVM formalism through the introduction of slack variables leading to an upper bound on $0 \le \beta^i(t) \le C$ [8], [9].

In Fig. 9 we display the effect of noise on the teacher-student task of recovering the synaptic matrix $\mathbf{w}$. We employed the same methodology as described in previous experiments, using the SVM algorithm with a constant slack parameter $C = 1$. As can be seen, this system works quite well, leading to relatively small errors. Thus we see that noise that flips on average one neuron in each state of $N = 40$ neurons leads to a prediction error of order 14%, roughly twice that expected from a network trained on a noiseless sequence of the same length of 250. The error bars reflect the standard deviation of ten teacher-student sessions.

These results are very encouraging. They imply that even at a relatively high error rate one can obtain reasonable predictive power from the analysis of the data. In other words, one may trust the meaningfulness of the $\mathbf{w}$ deduced by the student. Comparing two different sequences, this implies that one can deduce from our analysis whether they are generated by a similar $\mathbf{w}$ or not.

V. The Ill-Defined Problem

Sequences or spatiotemporal patterns that are not generated by a DNF may not be amenable to DNF representations that are limited to the number N of observed neurons. There may be two sources of disagreement:

1. Occurrence of contradictory repetitions, i.e. $\vec{n}(t_1) = \vec{n}(t_2)$ while $\vec{n}(t_1+1) \neq \vec{n}(t_2+1)$.
2. Occurrence of XOR-like configurations that violate the linear separability condition that is implied by the dynamics of Eq. 1.

Simple examples of such sequences can be generated by a DNF with a number of neurons N' that is larger than the number of observed neurons N in terms of which the sequence

is presented to the student network. This hints at the way to solve the two problems mentioned above: add hidden neurons [1]. This, however, poses a difficult problem: not only has one to learn a larger synaptic matrix but one has also to choose appropriate values $n_i(t)$ of the hidden neurons during the time steps of the available sequence. We leave open the questions of what is an efficient algorithm to achieve these goals, and how one may guarantee that the minimal N' is obtained. We can state, however, that for any spatiotemporal sequence of length T composed of arbitrary states, a DNF representation exists under the conditions discussed in subsection B of the Introduction (N' of order $T/2$).

Problem number 2 of the XOR-like configurations, can be handled by generalizing the DNF of Eq. 1 into a non-linear SVM kernel formulation,

$$n_i(t+1) = H(K(\vec{w}^i, \vec{n}(t))) \tag{13}$$

with the kernel replacing the scalar product $\vec{w}^i \cdot \vec{n}(t)$. Appropriate kernels can be powers $K(\vec{a}, \vec{b}) = (\vec{a} \cdot \vec{b} + 1)^p$ or Gaussians $K(\vec{a}, \vec{b}) = e^{\frac{(\vec{a}-\vec{b})^2}{2\sigma^2}}$ [8], [9]. Once the separability of a series is resolved, Eq. 13 can be used as the definition of a Recurrent Support Vector Network (RSVN).

There exists still the first problem, that of contradictory repetitions, when the same state vector leads to different results occurring at different time steps. When such a situation occurs one cannot escape the necessity of invoking hidden neurons. It is however straightforward to decide on their minimal number since all that is needed is to guarantee the disappearance of contradictory repetitions.

VI. Discussion and Conclusions

The DNF serves as a general framework for the generation and study of binary spatiotemporal patterns. $N = 40$ is a large enough system to demonstrate very large cycles (over 1000), leading to apparent random spatiotemporal sequences. Although such patterns, and even different sections of the same sequences, look quite different from one another, there exists a dynamical proximity measure in terms of the synaptic matrices $\mathbf{w}$ that generate them.

In our numerical studies we have demonstrated the complexity of the sequences, and have shown that in spite of this complexity some order may be found in terms of their inferred $\mathbf{w}$. Regarding the given spatiotemporal sequence as a teacher we trained on it student networks. We have demonstrated the correlations between $\mathbf{w}$s of teacher and student networks using two algorithms: the perceptron algorithm of [1] and an SVM algorithm suggested here. The latter may serve to define an optimal $\mathbf{w}$ in the sense of providing the largest margin in any classification algorithm of the linearly separable data. We have seen that it is relatively easy to get very high correlations between the $\mathbf{w}$s in our $N = 40$ examples. The scale of the required length of the series is set by $T_C = 2N$, or 80 in this case, when a very large sector of all $\mathbf{w}$-space can accommodate the data, as expected from the Cover limit $\mathcal{B}$ of Eq. 11. For $T > 2N$ generalization sets in, causing an ever increasing correlation between the $\vec{w}^i$ vectors of synaptic weights of the teacher and the student. Here, a few hundred learning steps were sufficient to obtain correlations above 90%. Thus, given two sequences of length $T = 250$, generated by two DNFs with $N = 40$, we may deduce, to an accuracy of order 80%, the correlations between the two original $\mathbf{w}$s.

These results can be used to emphasize the interesting coding property that the DNF suggests for spatiotemporal sequences: a sequence may be defined by the $\mathbf{w}$ used to generate it. If it is long enough, this $\mathbf{w}$ may be reproduced by the algorithms mentioned above, as long as all neurons are observed. Proximity of sequences is then defined by proximity in $\mathbf{w}$ values. One special case is that of keeping all w_{ij} fixed and varying only the b_i values. We saw in Fig. 2 that a small change in these parameters suffices to induce divergence in Hamming distance. Nonetheless, the b_i values may be reconstructed easily, as seen in Fig. 6.

There are two characteristics of DNF-generated sequences that are important: 1. non-recurrence of a state (unless it reappears in a cyclic recurrence), and 2. linear separability of all states observed by each neuron. Once not all neurons are displayed, these characteristics may disappear when only the observed neurons are considered. In principle, any finite spatiotemporal binary sequence can be generated by a DNF provided sufficient hidden neurons are provided. However, this extension and, in particular, finding the minimal DNF that can accommodate a given sequence [1], is a hard problem.

References

[1] B. Quenet and D. Horn, 2003. The Dynamic Neural Filter: A Binary Model of Spatiotemporal Coding. *Neural Computation* **15** 309–329.

[2] Hertz, J., Krogh, A. & Palmer, R., G., 1991. *Introduction to the Theory of Neural Computation.* Addison Wesley Pub. Comp.

[3] P. Peretto, 1992. *An Introduction to the Modeling of Neural Networks.* Cambridge University Press.

[4] Gutfreund, H., Reger, D. J. & Young, A. P. 1988. The nature of attractors in an asymmetric spin glass with deterministic dynamics. *J. Phys. A: Math. Gen.* **21** 2775-2797.

[5] F. Rosenblatt, 1962. *Principles of Neurodynamics.* New York: Spartan.

[6] T. M. Cover, 1965. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE trans. Elect. Comp.* **14** 326-334.

[7] H. Sompolinsky, N. Tishby and H. S. Seung, 1990. Learning from Examples in Large Neural Networks. *Phys. Rev. Lett.* **65** 1683–1686.

[8] V. Vapnik 1995. *The Nature of Statistical Learning Theory*, Springer, New-York.

[9] C. J. C. Burges 1998. A Tutorial on Support Vector Machines for Pattern Recognition. *Knowledge Discovery and Data Mining*, 2(2).

[10] S. Amari and K. Maginu, 1988. Statistical neurodynamics of associative memory. *Neural Networks* 1, 63-67.

[11] W. G. Gibson and J. Robinson, 1992. Statistical analysis of the dynamics of a sparse associative memory. *Neural Netowrks* 5, 645-661.

[12] D. Willshaw and P. Dayan, 1990. Optimal Plasticity from Matrix Memories: What Goes Up Must Come Down Neural Computation 2, 85-93.