A Neural Network for the Travelling Salesman Problem with a Well Behaved Energy Function

Marco Budinich and Barbara Rosario
Dipartimento di Fisica & INFN, Via Valerio 2, 34127 Trieste, Italy
E-mail: mbh@trieste.infn.it

(Contributed paper to MANNA Conference, Oxford, July 3-7, 1995)

Abstract - We present and analyze a Self Organizing Feature Map (SOFM) for the NP-complete problem of the travelling salesman (TSP): finding the shortest closed path joining N cities. Since the SOFM has discrete input patterns (the cities of the TSP) one can examine its dynamics analytically. We show that, with a particular choice of the distance function for the net, the energy associated to the SOFM has its absolute minimum at the shortest TSP path. Numerical simulations confirm that this distance augments performances. It is curious that the distance function having this property combines the distances of the neuron and of the weight spaces.

1 - Introduction

Solving difficult problems is a natural arena for a would-be new calculus paradigm like that of neural networks. One can delineate a sharper image of their potential with respect to the blurred image obtained in simpler problems.

Here we tackle the Travelling Salesman Problem (TSP, see [Lawler 1985], [Johnson 1990]) with a Self Organizing Feature Map (SOFM). This approach, proposed by [Angéniol 1988] and [Favata 1991], started to produce respectable performances with the elimination of the non-injective outputs produced by the SOFM [Budinich 1995]. In this paper we further improve its performances by choosing a suitable distance function for the SOFM.

An interesting feature is that this net is open to analytical inspection down to a level that is not usually reachable [Ritter 1992]. This happens because the input patterns of the SOFM, namely the cities of the TSP, are discrete. As a consequence we can show that the energy function, associated with SOFM learning, has its absolute minimum in correspondence to the shortest TSP path.

In what follows we start with a brief presentation of the working principles of this net and of its basic theoretical analysis (section 2). In section 3 we propose a new distance function for the network and show its theoretical advantages while section 4 contains numerical results. The appendix contains the detailed description of parameters needed to reproduce these results.

2 - Solving the TSP with self-organizing maps

The basic idea comes from the observation that in one dimension the exact solution to the TSP is trivial: always travel to the nearest unvisited city. Consequently, let us suppose we have a smart map of the TSP cities onto a set of cities distributed on a circle, we will easily find the shortest tour for these “image cities” that will give also a path for the original cities. It is reasonable to conjecture that the better the distance relations are preserved, the better will be the approximate solution found.

In this way, the original TSP is reduced to a search of a good neighborhood-preserving map: here we build it via unsupervised learning of a SOFM.

The TSP we consider is constituted of N cities randomly distributed in the plane (actually in the (0,1) square). The net is formed by N neurons logically organized in a ring. The cities are the input patterns of the network and the (0,1) square its input space.

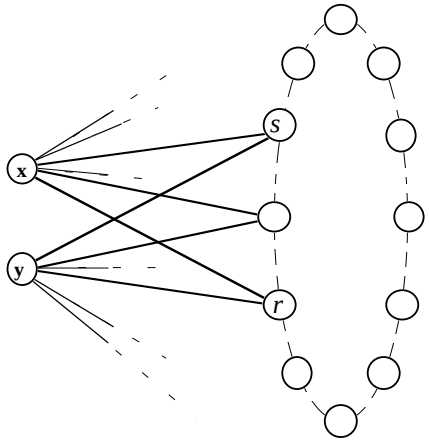

Figure 1 Schematic net: not all connections from input neurons are drawn.

Each neuron receives the (x, y) = $\vec{q}$ coordinates of the cities and has thus two weights: $(w_x, w_y) = \vec{w}$. In this view both patterns and neurons can be thought as points in two dimensional space. In response to input $\vec{q}$, the r-th neuron produces output $o_r = \vec{q} \cdot \vec{w}_r$. Figure 1 gives a schematic view of the net while figure 2 represents both patterns and neurons as points in the plane.

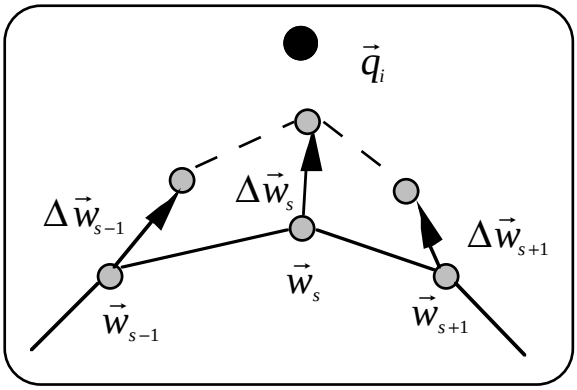

Figure 2 Weights modification in a learning step: neurons (small gray circles) are moved towards the pattern $\vec{q}_i$ (black circle) by an amount given by (1). The solid line represents the neuron ring. The shape of the deformation of the ring is given by the relative magnitude of the $\Delta\vec{w}$ that is in turn given by the distance function h_{rs}.

Learning follows the standard Kohonen algorithm [Kohonen 1984]: a city $\vec{q}_i$ is selected at random and proposed to the net; let S be the most responding neuron (i.e. the neuron nearest to $\vec{q}_i$) then all neuron weights are updated with the rule:

$$\Delta\vec{w}_r = \varepsilon h_{rs}\left(\vec{q}_i - \vec{w}_r\right) \quad (1)$$

where $0 < \varepsilon < 1$ is the learning constant and h_{rs} is the distance function.

This function determines the local deformations along the chain and controls the number of neurons affected by the adaptation step (1); thus it is crucial for the evolution of the network and for the whole learning process (see figure 2).

Step (1) is repeated several times while ε and the width of the distance function are being reduced at the same time. A common choice for h_{rs} is a Gaussian-like function like $h_{rs} = e^{-\left(\frac{d_{rs}}{\sigma}\right)^2}$ where d_{rs} is the distance between neurons r and s (the number of steps between r and s) and σ is a parameter which determines the number of neurons r such that $\Delta\vec{w}_r \neq 0$; during learning $\varepsilon, \sigma \to 0$ so that $\Delta\vec{w}_r \to 0$ and $h_{rs} \to \delta_{rs}$.

After learning, the network maps the two dimensional input space onto the one dimensional space given by the ring of neurons and neighboring cities are mapped onto neighboring neurons. For each city its image is given by the nearest neuron. From the tour on the neuron ring one obtains the path for the original TSP [1].

The standard theoretical approach to these nets considers the expectation value $E\left[\Delta\vec{w}_r|\vec{w}_r\right]$ [Ritter 1992]. In general $E\left[\Delta\vec{w}_r|\vec{w}_r\right]$ cannot be treated analytically except when the input patterns have a discrete probability distribution as it happens for the TSP. In this case, $E\left[\Delta\vec{w}_r|\vec{w}_r\right]$ can be expressed as the gradient of an energy function, i.e. $E\left[\Delta\vec{w}_r|\vec{w}_r\right] = -\varepsilon\nabla_{\vec{w}_r}V(W)$, with

$$V(W) = \frac{1}{2N}\sum_{rs} h_{rs} \sum_{q_i \in F_s} \left(\vec{q}_i - \vec{w}_r\right)^2 \qquad (2)$$

where $W = \{\vec{w}_r\}$ and the second sum is over the set of all the cities $\vec{q}_i$ having $\vec{w}_s$ as nearest neuron, i.e. the cities contained in F_s , the Voronoi cell of neuron $\vec{w}_s$. On average, $V(W)$ decreases during learning since $E[\Delta V|W] = -\varepsilon\sum_r ||\nabla_{\vec{w}_r} V||^2$.

Substantially, in this case, there exists an energy function which describes the dynamics of the SOFM and which is minimized during learning; formally, the learning process is the descent along the gradient of $V(W)$.

Unfortunately $V(W)$ escapes analytical treatment until the end of the learning process when some simplifications are applicable. Since at the end of learning $h_{rs} \to \delta_{rs}$, we can suppose h_{rs} is significantly different from zero only for $r = s, s \pm 1$; in this case (2) becomes

$$V(W) \cong \frac{1}{2N}\sum_s \sum_{q_i \in F_s}\left[h_{s-1,s}\left(\vec{q}_i - \vec{w}_{s-1}\right)^2 + h_{ss}\left(\vec{q}_i - \vec{w}_s\right)^2 + h_{s+1,s}\left(\vec{q}_i - \vec{w}_{s+1}\right)^2\right] \qquad (3)$$

---

[1] The main weakness of this algorithm is that, in about half of the cases, the map produced is not injective. The definition of a continuous coordinate along the neuron ring solves this problem yielding a competitive algorithm [Budinich 1995].

In addition, simulations support that, at the end of learning, most neurons are selected by just one city to which they get nearer and nearer. This means that F_s contains just one city, let's call it $\vec{q}_{i(s)}$, and that $\vec{w}_s \to \vec{q}_{i(s)}$, consequently

$$V(W) \cong \frac{1}{2N}\sum_s \left[h_{s-1,s}\left(\vec{q}_{i(s)} - \vec{q}_{i(s-1)}\right)^2 + h_{s+1,s}\left(\vec{q}_{i(s)} - \vec{q}_{i(s+1)}\right)^2 \right] \quad (4)$$

and assuming h_{rs} symmetric i.e. $h_{s-1,s} = h_{s+1,s} = h$, we get

$$\begin{aligned} V(W) &= \frac{h}{2N}\sum_s \left[\left(\vec{q}_{i(s)} - \vec{q}_{i(s-1)}\right)^2 + \left(\vec{q}_{i(s)} - \vec{q}_{i(s+1)}\right)^2 \right] \\ &= \frac{h}{N} L_{TSP^2} \end{aligned}$$

where L_{TSP^2} is the length of the tour of TSP considering the squares of the distances between cities. Thus the Kohonen algorithm for TSP minimizes an energy function which, at the end of the leaning process, is proportional to the sum of the squares of the distances. Numerical simulations confirm this result.

3 - A new distance function

Our hypothesis is that we can obtain better results for the TSP using a distance function h_{rs} such that, at the end of the process, $V(W)$ is proportional to the simple length of the tour L_{TSP}, namely $V(W) \propto \sum_s \left(\vec{q}_{i(s)} - \vec{q}_{i(s-1)}\right) = L_{TSP}$ since, in general, minimizing L_{TSP^2} is not equivalent to minimizing L_{TSP}.

We thus consider a function h_{rs} depending both on the distance d_{rs} and on another distance D_{rs} defined in weight space:

$$D_{rs} = \sum_{j=r+1}^{s} |\vec{w}_j - \vec{w}_{j-1}|$$

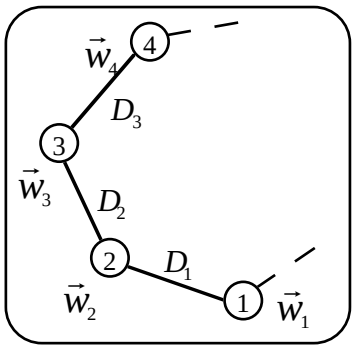

Figure 3 Distances between neurons s=4 and r=1: $D_{1,4} = D_1 + D_2 + D_3$ and $d_{1,4}$=3.

If we define

$$h_{rs} = \left(1 + \frac{D_{rs}}{\sigma}\right)^{-d_{rs}^2} \quad (5)$$

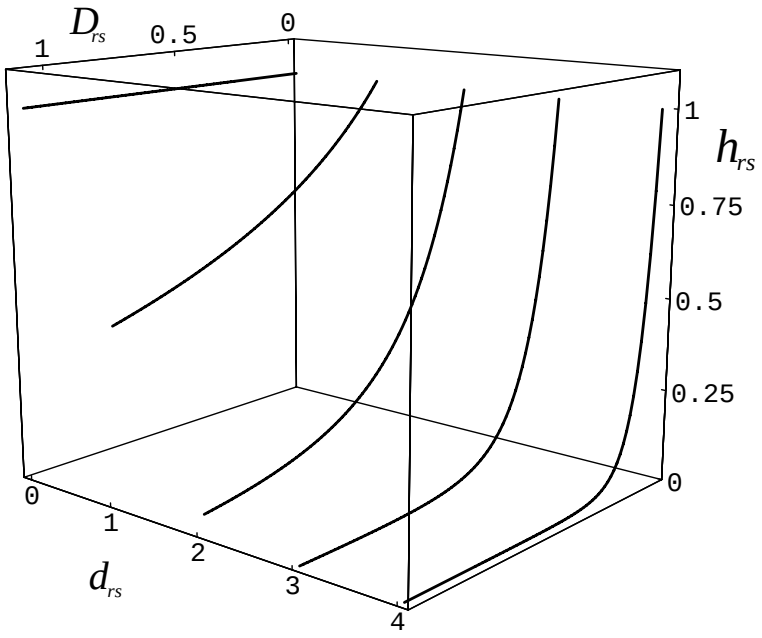

Figure 4 Set of h_{rs} given by (5) for $0 < D_{rs} < 1$ and d_{rs} = 0, 1, ... ,4.

when $\sigma \to 0$, we get for $h_{s\pm1,s}$

$$h_{s\pm1,s} = \left(1 + \frac{D_{s\pm1,s}}{\sigma}\right)^{-1} \cong \frac{\sigma}{D_{s\pm1,s}} = \frac{\sigma}{\left|\vec{w}_s - \vec{w}_{s\pm1}\right|} \cong \frac{\sigma}{\left|\vec{q}_{i(s)} - \vec{q}_{i(s\pm1)}\right|}$$

and substituting this expression in (4) we obtain

$$V(W) \cong \frac{1}{2N}\sum_s \left[\frac{\sigma}{\left|\vec{q}_{i(s)} - \vec{q}_{i(s-1)}\right|}\left(\vec{q}_{i(s)} - \vec{q}_{i(s-1)}\right)^2 + \frac{\sigma}{\left|\vec{q}_{i(s)} - \vec{q}_{i(s+1)}\right|}\left(\vec{q}_{i(s)} - \vec{q}_{i(s+1)}\right)^2 \right]$$

$$= \frac{\sigma}{N}\sum_s \left|\vec{q}_{i(s)} - \vec{q}_{i(s+1)}\right|$$

$$= \frac{\sigma}{N} L_{TSP}$$

With this choice of h_{rs} the minimization of the energy $V(W)$ is equivalent to the minimization of the TSP path.

We remark that the introduction of the distance D_{rs} between weights is a slightly unusual hypothesis for this kind of nets that usually keep well separated neuron and weight spaces in the sense that the distance function h_{rs} depends only on the distance d_{rs}.

4 - Numerical results

Since the performances of this kind of TSP algorithms are good for problems with more than 500 cities and more critical in smaller problems [Budinich 1995], we began testing the performances produced by the new distance function (5) in problems with 50 cities.

We compared the quality of TSP solutions obtained with this net to those of two other algorithms both deriving from the idea of a topology preserving map and that both actually minimizes L_{TSP^2}: the elastic net of Durbin and Willshaw [Durbin 1987] and this same algorithm with a standard distance choice.

As a test set, we considered the very same 5 sets of 50 randomly distributed cities used for the elastic net.

Table 1 contains a comparison of the best TSP path obtained in several runs of the different algorithms expressed as percentual increments over the best known solution for the given problem.

City set	Min. length	[Durbin 1987]	[Budinich 1995]	This algorithm
1	5.8358	2.47 %	1.65 %	0.96 %
2	5.9945	0.59 %	1.66 %	0.31 %
3	5.5749	2.24 %	1.06 %	1.05 %
4	5.6978	2.85 %	1.37 %	0.70 %
5	6.1673	5.23 %	5.25 %	0.43 %
Average		*2.68* %	*2.20* %	*0.69* %

Table 1 Comparison of the best TSP solution obtained in 10 runs of the various algorithms. Rows refer to the 5 different problems each of 50 cities randomly distributed in the (0,1) square. Column 2 reports the length of the best known solution for the given problem. Columns 3 to 5 contain the best lengths obtained by the three algorithms under study expressed as percentual increments from the minimal length; the number of runs of the algorithms is respectively: unknown, 5 and 10. Last row gives the increment averaged over the 5 city sets.

Another measure of the quality of the solution is the mean length obtained in the 10 runs. The percentual increment of these mean lengths, averaged over the 5 sets, was for this algorithm 2.49%, showing that even the averages found with the new distance function are better than the minima found with the elastic net.

These results clearly show that distance choice (5) gives better solutions in this SOFM application, thus supporting the guess that an energy $V(W)$ directly proportional to the length of the tour L_{TSP}, is better tuned to this problem.

One could wonder if adding weight space information to the distance function could give interesting results also in other SOFM applications.

Appendix

Here we describe the network setting that produces the quoted numerical results. Apart from the distance definition (5) we apply a standard Kohonen algorithm and exponentially decrease parameters ε and σ with learning epoch n_e (a learning epoch correspond to N weights update with rule (1))

$$\varepsilon = \varepsilon_0 \ \alpha^{\,n_e} \qquad\qquad \sigma = \sigma_0 \ \beta^{\,n_e}$$

and learning stops when ε reaches $5{\cdot}10^{-3}$. Numerical simulations clearly indicate that best results are obtained when the final value of σ is very small ($\approx 5{\cdot}10^{-3}$) and when ε and σ decrease together reaching their final value at the same time. Consequently given values for α and σ_0 one easily finds β.

In other words there are just three free parameters to play with to optimize results, namely ε_0 and σ_0 and α. After some investigation we obtained the following values that produce the quoted results: $\varepsilon_0 = 0.8$, $\sigma_0 = 14$ and $\alpha = 0.9996$.

References:

[Angéniol 1988] B. Angéniol, de G. La Croix Vaubois and J.-Y. Le Texier, *Self Organising Feature Maps and the Travelling Salesman Problem*, Neural Networks **1** 1988 pp. 289-293;

[Budinich 1995] M. Budinich, *A Self-Organising Neural Network for the Travelling Salesman Problem that is Competitive with Simulated Annealing*, to appear in: Neural Computation;

[Durbin 1987] R. Durbin and D. Willshaw, *An Analogue Approach to the Travelling Salesman Problem using an Elastic Net Method*, Nature **336** 1987 pp. 689-691;

[Favata 1991] F. Favata and R. Walker, *A Study of the Application of Kohonen-type Neural Networks to the Travelling Salesman Problem*, Biological Cybernetics **64** 1991 pp. 463-468;

[Johnson 1990] D. S. Johnson, *Local Optimization and the Traveling Salesman Problem*, in: *Proceedings of the 17th Colloquium on Automata, Languages and Programming*, 1990 Springer-Verlag New York, pp. 446-461;

[Kohonen 1984] T. Kohonen, *Self-Organisation and Associative Memory*, 1984 (3rd Ed. 1989) Springer-Verlag Berlin Heidelberg;

[Lawler 1985] E. L. Lawler, J. K. Lenstra , A. G. Rinnoy Kan and D. B. Shmoys (editors), *The Traveling Salesman Problem - A Guided Tour of Combinatorial Optimization*, John Wiley & Sons, New York 1990, IV Reprint, pp. x 474;

[Ritter 1992] H. Ritter, T. Martinetz and K. Schulten, *Neural Computation and Self-Organizing Maps*, Addison-Wesley Publishing Company, Reading Massachusetts 1992, pp. 306.