VECTOR QUANTIZATION FOR ARBITRARY DISTANCE FUNCTION ESTIMATION

İ. Kuban Altınel *
Dept. of Indus. Eng.
Boğaziçi University
İstanbul, TÜRKİYE
altinel@boun.edu.tr

John Oommen †
Sch. of Comp. Sci.
Carleton University
Ottawa, CANADA
oommen@scs.carleton.ca

Necati Aras *
Dept. of Indus. Eng.
Boğaziçi University
İstanbul, TÜRKİYE
aras@boun.edu.tr

Abstract— *In this paper we apply the concepts of Vector Quantization (VQ) for the determination of arbitrary distance functions – a problem which has important applications in Logistics and Location Analysis. The input to our problem is the set of coordinates of a large number of nodes whose inter-node arbitrary "distances" have to be estimated. To render the problem interesting, non-trivial and realistic, we assume that the explicit form of this distance function is both unknown and uncomputable. Unlike traditional Operations Research methods, which compute aggregate parameters of functional estimators according to certain goodness-of-fit criteria, we have utilized VQ principles to first adaptively polarize the nodes into sub-regions. Subsequently, the parameters characterizing the sub-regions are learnt by using a variety of methods (including, for academic purposes a VQ strategy in the meta-domain). The algorithms have been rigorously tested for the actual road-travel distances involving cities in Türkiye. The results obtained are not only conclusive, but also the best currently available from any single or hybrid strategy.*
Keywords— *Artificial Intelligence, Location, Neural Networks, Road Transportation, Self Organizing Maps, Vector Quantization.*

1 Introduction

An enormous amount of work has been done in designing neural networks using a variety of learning principles. Among these, in this paper we shall concentrate our attention on the Vector Quantization (VQ) methodologies [20, 23, 24, 27, 33] as adapted by Kohonen in his reputed Self Organizing Map (SOM).

The SOM has been used in a variety of applications. In statistical pattern recognition it has been used in the recognition of Finnish and Japanese speech [22, 25, 26], sentence understanding [45], in classification of sea-ice [39] and even in the classification of insect courtship songs [38]. From an hardware point of view the SOM has been used in the design of algorithms which at the lowest level can control the production of semiconductor substrates [34, 47] and at a higher level the synthesis of digital systems [21]. It has also been used in solving certain optimization problems such as the Traveling Salesman Problem ([41] pp.341).

The beauty of the SOM is the fact that the individual neurons adaptively tend to learn the properties of the underlying distribution of the space in which they operate. Additionally, they also tend to learn their

*Partially supported by the Turkish Scientific and Technical Research Council (TÜBİTAK) Grant TBAG-1336
†Partially supported by the National Sciences and Engineering Research Council (NSERC) of Canada

places topologically. This feature is particularly important for problems which involve two and three-dimensional **physical** spaces, and is indeed, the principal motivation for the SOM being used in path planning and obstacle avoidance in Robotics [18, 19, 35, 42, 43, 44] . In this paper, we shall use the principles of the SOM (or more precisely, the principles of Vector Quantization (VQ)) in the estimation of arbitrary distance functions – a problem which has received much attention in Logistics and Location Analysis [16, 31].

Consider the situation in which a user is given a set of N nodes (cities and towns), G, located in a multi-dimensional "physical" space. We assume that there is an unknown arbitrary distance function Φ between the nodes. By arbitrary, we mean that the set of inter-node distances dictated by Φ may or may not satisfy all the rigorous properties of a well-defined mathematical norm. Furthermore, the triangular inequality may also be violated. However, to keep the informal concepts of a distance measure valid, we impose the requirement that Φ is loosely related to the Euclidean norm as follows. First of all, $\Phi(P_i, P_i)$ must be zero, and $\Phi(P_i, P_j)$ must be symmetric. Furthermore, let P_i, P_j, P_m and P_n be any four nodes in G. Then, informally speaking, if the pairs (P_i, P_j) and (P_m, P_n) are "close" to each other in the physical world, the respective arbitrary distances between (P_i, P_m) and (P_j, P_n) must be correspondingly of similar magnitude. We formalize these concepts below.

Definition : A function Φ is defined to be a valid arbitrary distance function if for every P_i, P_j, P_m and P_n in G, the following is satisfied :

1. $\Phi(P_i, P_i) = 0$,
2. $\Phi(P_i, P_j) = \Phi(P_j, P_i)$, and,
3. For every $\epsilon > 0$ there exists a $\delta > 0$ such that
   $\| P_i - P_j \| < \epsilon$ and $\| P_m - P_n \| < \epsilon \Rightarrow |\Phi(P_i, P_m) - \Phi(P_j, P_n)| < \delta$.

The principles of VQ lend themselves naturally to the domain of arbitrary distance estimation. Indeed, in the solution which we have proposed, a sequence of pattern recognition and polarizing modules are implemented using VQ. It would appear as if the fact that we are working with a "real-life" physical world would make the SOM a natural tool to achieve complete learning, classification and estimation. While this is, of course, true from a philosophic point of view, the fact that the arbitrary function Φ is not explicitly related to their geographical (Euclidean) "as the crow flies" distances complicates the problem. Indeed, our results tend to show that we now have a scenario when an all-neural approach ([24] pp.82) is sometimes recommendable (as opposed to the speech recognition example discussed in [24]). In other cases the hypothesis of Kohonen (that a neural network be followed by a traditional strategy) is clearly validated, because a neural preprocessor followed by a traditional optimization yields even more superior results.

The physical application domain in which we have tested our algorithms involves the actual road distances between the major cities and towns in Türkiye. This has provided us with a platform to verify the power of our algorithms, and also to compare them to the results obtained using the existing techniques.

With regard to the salient contributions of the paper, we believe that our strategy is the first reported technique which tackles distance estimation using an "adaptive" multi-regional approach. This, is indeed, equivalent to approximating the unknown function by a "patchwork" (lattice) of intra-regional and inter-regional explicit subfunctions. In all of the early works subregions are selected *a priori* based on subjective judgements [3, 15]. However, in our method, the region of interest is subdivided into a set of sub-regions adaptively using a VQ method. This imposes an implicit discriminant mapping on the domain. Subsequently, the arbitrary distance function is sub-classified as a set of intra-set and inter-set distance functions each of them being characterized using their own respective parameters. The training sites and their corresponding available coordinates and inter-distances are then used to train the intra and inter-set parameters whence the estimation follows. All of these ideas are novel to the area of distance estimation.

The highlights of our contributions from a neural network perspective will be explained in a subsequent section.

Most of the research that is currently available in distance estimation involves the estimation of geographical road travel distances. Consequently, to place our current work in the right perspective, in the next section we shall define the problem of distance estimation and review briefly the currently available results on the use of distance functions. In Section 3 we shall give an overview of VQ and the SOM and proceed, in Section 4 to show how it can be applied to the estimation of arbitrary distance functions. Section 5 discusses the experimental results and highlights the salient features of our methods in the context of both the optimization and neural network strategies. Section 6 concludes the paper.

2 Road Travel Distance Estimation

2.1 Distance Estimation Problem

The *actual distance* between any two points on the earth surface is the length of the shortest road connecting them. Since it is often not feasible to measure the *actual distances* for all pairs of points, it is a common practice to use distance estimators. Then the question is to choose a good estimator so that accurate distance approximations are obtained.

A good estimation of actual distances is critical in many applications. Almost all of the location problems, distribution problems such as the transportation problem, its generalization the transshipment problem, the traveling salesman problem, and the vehicle routing problem assume the knowledge of actual distances in their formulations. For example, in their recent simulation study to determine the number of fire-stations in İstanbul, Erkut and Polat multiply the Euclidean distance by an inflation factor, which they call the *road coefficient*, in order to estimate the actual distance between the fire-station and fire area [14].

We can define the problem of distance estimation formally as follows: Let us say that P_a and P_b are two points on the Cartesian plane with coordinates $P_a = (x_{a1}, x_{a2})^T$ and $P_b = (x_{b1}, x_{b2})^T$. The aim is to build an estimator $\Phi(P_a, P_b|\boldsymbol{\theta})$ of the actual distance between P_a and P_b. Let $\pi_i = \langle P_{i1}, P_{i2} \rangle$ be the ith **pair** of points, and let r_i be the actual distance between P_{i1}, P_{i2}. The set of all pairs and the corresponding distances is given by S as :

$$S = \{(P_{i1}, P_{i2}, r_i) : 1 \leq i \leq n\}, \ \textit{where}, \quad n = \binom{N}{2} \tag{1}$$

$\boldsymbol{\theta}$ is a vector of parameters estimated using S with respect to the following goodness-of-fit criterion :

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \left[E[\delta(\Phi(P_{i1}, P_{i2}|\boldsymbol{\theta}, S), r)]\right] = \arg\min_{\boldsymbol{\theta}} \left[\frac{1}{n}\sum_{i=1}^{n} \delta\left(\Phi(P_{i1}, P_{i2})|\boldsymbol{\theta}, r_i\right)\right] \tag{2}$$

$\delta(\cdot)$ is the difference measure. One possibility, originally proposed by Love and Morris [28], is the absolute value of the deviation:

$$\delta\left(\Phi(P_{i1}, P_{i2}|\boldsymbol{\theta}), r_i\right) = |\Phi(P_{i1}, P_{i2}|\boldsymbol{\theta}) - r_i| \tag{3}$$

According to this criterion, a distance function must estimate greater actual distances relatively more accurately than shorter distances. This is a drawback if we are more interested in proportional deviations than absolute deviations. Another error measure, also proposed by Love and Morris [28], is normalized by dividing pairwise estimation errors by the square root of the actual distance between them :

$$\delta\left(\Phi(P_{i1}, P_{i2}|\boldsymbol{\theta}), r_i\right) = \left[\frac{\Phi(P_{i1}, P_{i2}|\boldsymbol{\theta}) - r_i}{\sqrt{r_i}}\right]^2 \tag{4}$$

Although both criteria provide ample insight in their own right, the latter one is superior not only because it gives importance to proportional errors but also because of the following three reasons. First, most of the experimental results in the literature use the second criterion, e.g., [4, 5, 11, 28, 32, 48] and hence serve as an excellent benchmark. Furthermore, it has important statistical properties which leads to statistical tests for comparing the accuracy of distance functions under certain normality and independence assumptions and thus the results obtained can be statistically justified. Finally, it is a continuous and differentiable function of the parameter vector which enables the use of gradient descent minimization strategies important in various domains including neural network learning.

The standard approach for distance estimation uses estimators that are parameterized functions of certain "easy-to-obtain" pieces of information, namely the coordinates of the points. This approach has been widely used ever since the first work by Love and Morris [28] because it provides simple analytical closed form expressions of the coordinates once the values of the parameters have been determined. As in any parametric method, the concept works well with small samples, but the accuracy may not be high if the assumed form of the function is not appropriate.

In the recent work by Alpaydın *et. al.* [2] the problem of estimating distances has been viewed in the context of function approximation or nonlinear regression and perceptron based estimators have been applied for this task of estimating $\Phi(P_{i1}, P_{i2}|\boldsymbol{\theta})$. These methods, being nonparametric, have the advantage that they do not assume any *a priori* model and are trained directly from a training sample. They, of course, necessitate larger training samples and more computer time as the simplicity of a parametric model with just a few parameters does not exist anymore. Although, perceptron based non-parametric estimators perform better compared to parametric distance functions, (i.e., they yield smaller errors), the results can be improved further if the cities are clustered adaptively using a VQ method prior to any estimation attempt as we will see in Section 3. Indeed, as we shall philosophically justify, VQ seems to be hybrid between the parametric and non-parametric families of algorithms.

2.2 Distance Functions

A generally used method for estimating actual distances between any pair of points is to make approximations by means of a distance function, which is a parameterized function of the planar coordinates of the two points. These functions can be classified in three major groups with respect to the type of coordinates they use. The members of the first group use spherical coordinates for the purpose of introducing the spherical effect of the earth surface into the distance estimation [28, 29]. Although this idea provides certain additional accuracy, the contribution has been experimentally reported to be minor by Love and Morris [28]. The second group consists of functions which use polar coordinates [36, 40]. The motivation is based on the observation that the roads in historically older cities are not usually planned according to a rectangular grid structure and consequently, distances are often better approximated by a ring-radial measure. This approach seems to be very accurate especially for a spider's web-like road network structure. The third group contains some simple functions of the Cartesian coordinates. These are mostly norms or norm-based functions, and the ones we have adopted are listed in Table 1. Indeed, in the literature these are the most important ones, because of their wide usage in location and distribution problems [16, 31].

The parameters, which should be nonnegative, k, p, and s constitute $\boldsymbol{\theta}$ and are estimated over the sample to provide good approximations and as such, encode geographical characteristics of the region where they are used. There is a large literature on the determination of these parameters and the comparison of the parametric distance functions. Astonishingly enough, some of the conclusions drawn in these papers are conflicting [5, 6, 7, 9, 28, 29, 30].

For all practical purposes, the function chosen to estimate actual road distances should be as accurate as possible. In their early study, Love and Morris [28, 29] compute the parameters k, p, and s of $\Phi_1(P_1, P_2), \Phi_2(P_1, P_2), \Phi_3(P_1, P_2)$, and $\Phi_4(P_1, P_2)$ for the United States and compare them with respect

Table 1: Distance functions used and their associated parameters.

DISTANCE FUNCTION	PARAMETERS (θ)
$\Phi_1(P_1,P_2) = k(\lvert x_{11}-x_{21}\rvert + \lvert x_{12}-x_{22}\rvert)$	k
$\Phi_2(P_1,P_2) = k(\lvert x_{11}-x_{21}\rvert^2 + \lvert x_{12}-x_{22}\rvert^2)^{1/2}$	k
$\Phi_3(P_1,P_2) = k(\lvert x_{11}-x_{21}\rvert^p + \lvert x_{12}-x_{22}\rvert^p)^{1/p}$	k,p
$\Phi_4(P_1,P_2) = k(\lvert x_{11}-x_{21}\rvert^p + \lvert x_{12}-x_{22}\rvert^p)^{1/s}$	k,p,s

to the accuracy they provide. The important conclusion of this study is the superiority of $\Phi_4(P_1,P_2)$ over the other three. The second best approximating function seems to be $\Phi_3(P_1,P_2)$.

At the end of their study on the road network of the former Federal Republic of Germany (FRG), Berens and Körling [6] and Berens [5] conclude that the accuracy provided by the weighted Euclidean norm $\Phi_2(P_1,P_2)$ is sufficient and the use of $\Phi_3(P_1,P_2)$ is not worth the extra computational effort necessary for calculations. However in a further study over the largest 25 cities of FRG, Love and Morris [30] report conflicting results which demonstrate that the accuracy of the weighted L_p norm, $\Phi_3(P_1,P_2)$, is remarkably higher than the accuracy provided by $\Phi_2(P_1,P_2)$. Although it supports the early findings of Berens and Körling [6] for FRG, the study by Berens [5] includes mixed results when it is enlarged to cover 11 other countries; the relative improvement introduced by $\Phi_3(P_1,P_2)$ over $\Phi_2(P_1,P_2)$ ranges within 0.00% and 11.27%. Finally, Berens and Körling [7], in their last comment, state that, if the accuracy is of primary interest, the empirical distance functions should be tailored for the regions they are to be used for. Currently, there is no single general distance function which provides the same accuracy all over the world.

There are also distance measures which do not fit completely in any of the above mentioned three groups. They can be included in the last category but they are not always simple functions of the coordinates and require additional information such as a rotating angle for the coordinate axes [11, 29] or vectors for possible directions on a typical road [48, 49]. All of them are based on the idea that a travel has two major components; rectilinear and Euclidean, and the actual distance between any pair of points can be modeled as their non-negative linear combination. Ward and Wendell [48] initiate this hybrid idea by suggesting the weighted one-infinity norm and observe that the accuracy of this function is relatively close to the accuracy of the weighted L_p norm, $\Phi_3(P_1,P_2)$, based on the data set of Love and Morris [28]. In their later work, Ward and Wendell generalize the one-infinity norm to obtain the family of block norms in which the accuracy of the approximation depends on possible travel directions [49]. They report that the approximations obtained by the weighted L_p norm are more accurate than those obtained by a two-parameter block norm, which is actually the weighted one-infinity norm, and the accuracy of the weighted L_p norm is slightly worse than the one of eight-parameter block norms. Similar conclusions have been obtained also by Love and Walker [32] in their detailed empirical study on block and round norms. Block norms play an important role in location models because they lead to linear programming problems for certain objective functions, such as the minimax distance function; but the size of the linear program can easily become very large.

Another hybrid distance function is due to Brimberg and Love [10]. It is called the weighted one-two norm since the rectilinear and Euclidean elements of the travel are presented respectively by the weighted L_1 and L_2 norms. The authors suggest its use to approximate $\Phi_3(P_1,P_2)$ in estimating distances. The weighted one-two norm provides also good approximations for the probabilistic L_p norm [12]. Besides, its parameters can be calculated easily by simple linear regression [9].

Having briefly surveyed the field we are now in a position to explain how VQ and an adaptive multi-regional strategy can be applied to the estimation of arbitrary distance functions.

3 Vector Quantization and the Self-Organizing Map

The foundational ideas motivating VQ and the SOM are classical concepts that have been applied in the estimation of probability density functions. Traditionally, (in the realms of both statistical analysis and statistical pattern recognition) distributions have been represented either parametrically or non-parametrically. In the former, the user generally assumes the form of the distribution function and the parameters of the function are learnt using the available data points. In pattern recognition (classification), these estimated distributions are subsequently utilized to generate the discriminant hyperspheres (or hyperellipsoids) whence the classification is achieved.

As opposed to the latter, in non-parametric methods, the practitioner assumes that the data must be processed in its entirety (and not just by using a functional form to represent the data). The corresponding pattern recognition (classification) algorithms which result are generally of the nearest neighbor (or k-nearest neighbor) philosophy and are thus computationally expensive. The comparison of these two perspectives is found in standard pattern recognition textbooks [13, 17] and bounds on the classification error rate of non-parametric strategies (as compared to the optimal Bayesian) parametric strategies have also been derived.

The concept of VQ can be perceived as a compromise between the above two schools of thought. Rather than representing the entire data in a compressed form using only the estimates (and in the estimate domain), VQ opts to represent the data in the actual feature space. However, as opposed to the non-parametric methods which use all the data in the training and testing phases of classification, VQ compresses the information by representing it using a "small" set of vectors, called the **code-book** vectors. These code-book vectors are migrated in the **feature** domain so that they collectively represent the distribution under consideration. We shall refer to this phase as the *Intra-Regional Polarizing* phase. In a multi-class problem the code-book vectors for each region are subsequently migrated so as to ensure that they adequately represent their own regions and furthermore distinguish between the other regions. This phase, which we refer to as the *Inter-Regional Polarizing* phase, also implicitly learns the discriminant function to be used in a subsequent classification module. Note that these discriminant functions are of a nearest neighbor philosophy, except that the nearest neighbors are drawn from the set of code-book vectors (as opposed to the entire set of training samples). They thus drastically reduce the computational burden incurred in the testing of traditional non-parametric methods. It is not appropriate that we explain the details of VQ and the SOM here; they can be found in [23] and in an excellent survey by Kohonen [24]. However, in the interest of completeness and continuity, we shall in all brevity, explain the various phases of the VQ modules.

3.1 Intra-Regional Polarizing

We assume that we are to estimate the distance $\Phi(P_j, P_m)$ between any two points P_j, P_m in the set of points G. We also assume that we are given (the training set) L, a subset of G and the inter-node distances for the nodes in L (i.e., $\{\Phi(P_j, P_m) : P_j, P_m \in L\}$).

The basic hypothesis in distance estimation using a multi-regional approach is that G can be partitioned into a set of smaller regions whence intra-regional and inter-regional approximates of Φ can be obtained. Thus, in the training phase[1], we partition L into R subsets, $C_k = \{P_{k,i} : 1 \leq i \leq N_k\}(1 \leq k \leq R)$ each containing N_k points.

Our primary aim is to represent each C_k by M representative points ($M \ll N_k$)[2] $\{Q_{k,j} : 1 \leq j \leq M\}$. The set of code-book vectors $\{Q_{k,j} : 1 \leq j \leq M\}$ are first initially randomly assigned positions within or close to their respective regions. In the intra-regional polarizing the algorithm is repeatedly presented with a node $P_{k,i}$ from C_k. The closest code-book vector, $Q_{k,j}$, to $P_{k,i}$ is determined and this vector is

[1] In what follows, as opposed to the notation of Section 2.1, $P_{k,i}$ will represent the ith point in the kth region.

[2] Although strictly speaking, we can represent a set C_k by M_k points (where M_k increases with N_k), in the interest of simplicity, in this paper we have assumed that the number of representative code-book vectors for all the classes is the same.

moved in the direction of this data point. Indeed, this is achieved by rendering the new $Q_{k,j}$ to be a convex combination of the current $Q_{k,j}$ and the data point $P_{k,i}$. More explicitly, the updating algorithm is as follows :

$$Q_{k,j}(t+1) = \begin{cases} (1-\alpha)Q_{k,j}(t) + \alpha P_{k,i} & if\ \ Q_{k,j}\ \ is\ \ the\ \ closest\ \ code-book \\ & vector\ \ to\ \ the\ \ data\ \ point\ \ P_{k,i} \\ Q_{k,j}(t) & otherwise \end{cases} \tag{5}$$

where 't' is the discretized (synchronized) time index.

Note that this is the traditional SOM strategy [20, 23, 24, 27, 33] except that we have consistently restricted the radius of the "bubble" of interest used by Kohonen to be unity. The reasons for this are two-fold :

1. Since we are attempting to represent the nodes in C_k by a set of representative code-book vectors, the *topological ordering* of these code-book vectors is absolutely irrelevant. This, in turn, makes the algorithm computationally extremely inexpensive, because, at each time we need only locate the nearest code-book vector and do not have to find **all** the code-book vectors within the bubble of activity.

2. In a typical application, the number of code-book vectors must be kept **extremely small**. This is because, we want to partition G into R sub-regions and thus, effectively, we are attempting to approximate function Φ using $(M \times R + 1) \times (M \times R)/2$ "patched" functions. If each of these functions has 3 parameters, the number of parameters to be estimated becomes prohibitively large. Thus, if R is 4 and M is 3, the number of parameters is 234. Indeed, if we represented each region by $M = 4$ code-book vectors, the number of parameters involved would be 408. Indeed, considering the extremely small values of M encountered in this application domain, (we have used $M = 3$ per region) rendering the radius of the bubble of activity to be unity is far from unreasonable. Furthermore, as mentioned above, it only hastens the rate of convergence of the scheme.

In (5) above, we have decremented α linearly from unity for the initial learning phase and then switched to small values of α which decrease linearly from 0.2 for the fine-tuning phase. This is as recommended in the literature [23, 24]. Each region is subjected to the intra-regional polarizing before the next phase, the inter-regional polarizing is invoked.

3.2 Inter-Regional Polarizing

After the individual regions have been represented by M code-book vectors using the above migration strategy, the code-book vectors are tested using L to see whether they adequately classify the points within their respective sub-regions. To achieve this, we resort to an algorithm analogous, in principle, to the LVQ3 algorithm suggested by Kohonen [24]. Every data point in the training set, L (not just in the individual clusters, C_k), is tested against the set of Q_k's to see whether their nearest code-book vector falls within their partition. Thus, unlike the previous phase, where the code-book vectors found their respective places by learning only from the location of the data points **within their** own respective classes, in this phase, these representative vectors are migrated so that they polarize **away from** the data points of the **other competing clusters**. The principle by which this is done is as follows.

Let us suppose that we examine a point $P \in C_k$. Also let us suppose that the two closest code-book vectors to P (among **all** the code-book vectors) are Q_a and Q_b. If both Q_a and Q_b do not belong to the cluster C_k, clearly, the information content in P (with respect to Q_a and Q_b) is misleading, and so it is futile to migrate Q_a and Q_b using this information. However, if both of them are intended to represent C_k, clearly, the information in P can be used to achieve an even finer tuning to their locations. Thus, in this scenario, both Q_a and Q_b are moved marginally from their current locations along the hyperline

towards P. The final scenario is the case when one of them, Q_a (Q_b), correctly belongs to C_k, and the other, Q_b (Q_a), belongs to a different partition. In this case, the information in P can be used to achieve an even finer tuning to their locations by migrating Q_a (Q_b) marginally from its current location along the hyperline **towards** P, and migrating the other code-book vector Q_b (Q_a) marginally from its current locations along the hyperline **away from** P.

Since we do not want the "straggler" points (the points which are misclassified, but which probably would not have been correctly classified even by an optimal classifier) to completely dictate (and thus, disturb) the polarizing, this migration is invoked only if the node P lies within a pre-specified window of interest, W. This restriction has also been recommended in the literature [23, 24], and typically, this window, W, is a hypersphere centered at the bisector between the code-book vectors Q_a and Q_b. Also, as recommended in the literature, the polarizing of **both** Q_a and Q_b (when both of them correctly classify P) is made to be of much smaller magnitude than in the scenario when either of them misclassifies it. These steps are formally given below.

If Q_a and Q_b are the two closest code-book representative vectors to a given point $P \in C_k$

$$\begin{array}{rcll}
Q_a(t+1) & = & (1-\epsilon\alpha)Q_a(t) + \epsilon\alpha P & if \;\; Q_a, Q_b \in C_k \;;\; P \in W \\
Q_b(t+1) & = & (1-\epsilon\alpha)Q_b(t) + \epsilon\alpha P & if \;\; Q_a, Q_b \in C_k \;;\; P \in W \\
Q_a(t+1) & = & (1-\alpha)Q_a(t) + \alpha P & if \;\; Q_a \in C_k \;;\; Q_b \in C_j \neq C_k \;;\; P \in W \\
Q_b(t+1) & = & (1+\alpha)Q_b(t) - \alpha P & if \;\; Q_a \in C_k \;;\; Q_b \in C_j \neq C_k \;;\; P \in W \\
Q_a(t+1) & = & (1+\alpha)Q_a(t) - \alpha P & if \;\; Q_a \in C_j \neq C_k \;;\; Q_b \in C_k \;;\; P \in W \\
Q_b(t+1) & = & (1-\alpha)Q_b(t) + \alpha P & if \;\; Q_a \in C_j \neq C_k \;;\; Q_b \in C_k \;;\; P \in W \\
Q_a(t+1) & = & Q_a(t) \;;\; Q_b(t+1) = Q_b(t) & otherwise
\end{array} \tag{6}$$

In (6) above, we have maintained α at a constant value of 0.1 (as opposed to varying it as recommended in [23, 24]), and kept ϵ to be 0.25.

Observe that after the intra-regional polarizing and the inter-regional polarizing, the representative code-book vectors impose a set of piece-wise linear boundaries which assign the original nodes, L, into potentially slightly different regions than that which was initially assigned. Thus, although the initial demarcation boundaries may have been incorrectly assigned, the sequence of polarizing operations tends to re-allocate them. The effect of this boundary re-allocation will be discussed in greater detail in the section describing our experimental results.

After the training points have been allocated and the set of code-book vectors for each cluster has been learnt, the patchwork of functions approximating Φ is now learnt. This can be done using either an independent optimizing strategy or a VQ scheme. We shall now demonstrate how these are achieved.

3.3 Parameter Learning Using VQ

After the individual regions have been represented by the various codebook vectors (using the above migration strategies), we are now in a position to estimate Φ. The basic assumption in this phase, is that we can approximate Φ by a patchwork (or lattice) of intra-regional and inter-regional functions. In this phase, we shall attempt to learn these respective approximating functions using the code-book vectors and the given distances $\Phi(P_i, P_j)$, P_i, $P_j \in L$.

Let P_i and P_j be any two nodes in G. Obviously, if only the points in G are of interest to us, Φ can be approximated (indeed, exactly represented) in terms of all the inter-node Euclidean norms as follows :

$$\Phi(P_i, P_j) = k_{i,j}||P_i - P_j|| \tag{7}$$

Notice that since Φ is symmetric, only roughly half of these coefficients will have to be estimated. Clearly, such a representation defeats the fundamental purpose of a distance estimation strategy, for it would necessitate the learning of all the $\{k_{i,j} : 1 \leq i, j \leq N; i > j\}$ coefficients. Our intention is to approximate (7) by hypothesizing that the constants $\{k_{i,j}\}$ are dependent on the code-book representative

vectors. Thus, rather than specify $\Phi(P_i, P_j)$ using (7) above, we assume that $\Phi(P_i, P_j)$ can be reasonably approximated by locating the closest code-book vectors for P_i and P_j and evaluating a simple function between these respective points. Thus, we approximate $\Phi(P_i, P_j)$ by using (8) below :

$$\Phi(P_i, P_j) \cong k_{a,b} \parallel P_I - P_J \parallel \tag{8}$$

where, closest code-book to P_i and P_j are Q_a and Q_b respectively. The problem that is now before us is one of determining the set of parameterizing constants $\{k_{a,b} : 1 \leq a, b \leq R \times M; a \leq b\}$.

There are at least three distinct schemes for evaluating the above set of parameterizing coefficients, $\{k_{a,b}\}$ using the training set, L and their corresponding true distances.

The first is by a simple averaging strategy. For every pair of nodes in the training set, a cumulative sum of the ratio of their true distance to their Euclidean distance is maintained. This ratio is called the *directional bias* by Brimberg and Love [10] and Brimberg and Wesolowsky [12]. This sum is associated with the pair of closest code-book vectors. The cumulative sum divided by the number of pairs using these code-book vectors yields the average value of $k_{a,b}$ for these code-book vectors Q_a and Q_b.

An alternate strategy to obtain the set of parameterizing coefficients is to perform a VQ learning algorithm in the space involving the coefficients themselves. We explain this strategy as follows. Let us suppose that we have a current value of $k_{a,b}$. When a new pair of points in L is examined, if the codebook vectors are Q_a and Q_b respectively, the updated value of $k_{a,b}$ is obtained by moving the current value towards the value estimated using just this set of points. This updating is done along the hyperline joining the two. This is formally described below.

Algorithm : GetCoeffByVQ
Input : The set of codebook vectors, the training set L and the distances $\Phi(P_i, P_j)$ for all P_i, $P_j \in L$.
Output : The set of parametizing coefficients, $\{k_{a,b} : 1 \leq a, b \leq R \times M; a \leq b\}$.
Begin
    For $a = b$ to $R \times M$ **Do**
        $\lambda_{a,b} = 1$
        $k_{a,b} = 0$
    EndFor
    Repeat until satisfied
        Get any distinct pair of points P_i, P_j in L
        If Closest code-book vectors to P_i and P_j are Q_a and Q_b respectively **Then**
          $k_{a,b} = (1 - \lambda_{a,b})k_{a,b} + \lambda_{a,b}(\Phi(P_i, P_j)/ \parallel Q_a - Q_b \parallel)$
          Decrease $\lambda_{a,b}$
        EndIf
    EndRepeat
End GetCoeffByVQ

It is easy to see that if $\lambda_{a,b}$ decreases inversely with the number of samples encountered (which have Q_a and Q_b as the code-book vectors) the above VQ strategy converges **exactly** to the average value of $k_{a,b}$ computed earlier. Any other updating method for $\lambda_{a,b}$ would converge to an alternate (hopefully, closer-to-optimal) value of the $k_{a,b}$. In all our experiments, we have used an inverse decreasing function for $k_{a,b}$. Indeed, in this setting, our results tend to show that, (as opposed to the speech recognition example discussed in [24]) we now have a scenario when an all-neural approach ([24] pp.82) is recommendable.

The third approach involves explicit optimization. Here, each inter-node distance is specified by a function which is completely defined by the closest code-book vectors, and whose functional form is one of those types tabulated in Table 1. The question of estimation is now reduced to one of optimization as has been done in the literature [2, 28, 29]. Here, Kohonen's recommendation of using a traditional scheme subsequent to a neural strategy has proven to be superior.

4 IMPLEMENTATION RESULTS

4.1 Database

We have collected two distinct samples by pairing respectively 80 and 23 cities and towns of Türkiye for training and test sets. The first is used for estimating the parameters and the second one to assess their performances. The training and test data sets contain planar coordinates and intercity distances for 80 and 23 cities respectively which make 3160 and 253 data pairs. The third dimension is ignored since previous empirical studies have shown that the effect of elevation in the accuracy of the estimators in Türkiye is almost null [4]. The values we report in the following sub-sections are average error per pair in kilometers on both the training and the test sets. Recall that the error per pair is measured by the normalized error measure given in Equation (4).

4.2 Distance Functions

By looking at properties of the application, it is realistic to eliminate some of the distance functions *a priori* by judging them with the structural properties of the actual road network. First of all, the road structure in Türkiye has been developed arbitrarily rather than rectilinearly or ring-radially. This arbitrariness makes the identification of a fixed pattern for possible travel directions within the country impossible; this is crucial for the use of hybrid norms. Besides, the area is too small to require the consideration of the earth's roundness in estimating actual distances. Being convinced by these observations, it is rational to concentrate on functions $\Phi_2(P_a, P_b), \Phi_3(P_a, P_b), \Phi_4(P_a, P_b)$ and compute the best possible value of the parameters k, p, and s. In spite of this fact, we also computed the value of k for $\Phi_1(P_a, P_b)$, since it has been heavily considered in the related literature [16, 31].

The calculation of the parameters with respect to any of the goodness-of-fit criterion introduced in Section 2 requires minimizing an error function similar to that of Equation (2). Since we use the normalized error function given in Equation (4), the minimization problems are continuous in the parameters. Karush-Kuhn-Tucker first order conditions for the first two problems have analytical solutions and the values of k which minimize error function (4) can be easily obtained by using the following equalities:

For $\Phi_1(P_{i1}, P_{i2})$[3] :

$$k = \frac{\sum_{i=1}^{n} \left(|x_{i,11} - x_{i,21}| + |x_{i,12} - x_{i,22}| \right)}{\sum_{i=1}^{n} \left(|x_{i,11} - x_{i,21}| + |x_{i,12} - x_{i,22}| \right)^2 / r_i} \tag{9}$$

and for $\Phi_2(P_{i1}, P_{i2})$:

$$k = \frac{\sum_{i=1}^{n} \left(|x_{i,11} - x_{i,21}|^2 + |x_{i,12} - x_{i,22}|^2 \right)^{1/2}}{\sum_{i=1}^{n} \left(|x_{i,11} - x_{i,21}|^2 + |x_{i,12} - x_{i,22}|^2 \right) / r_i} \tag{10}$$

However, the calculations for k and p of $\Phi_3(P_a, P_b)$, and k, p and s of $\Phi_4(P_a, P_b)$ are slightly more complicated. They require the solution of the following unconstrained optimization problems:

$$\min_{k,p} \sum_{i=1}^{n} \left[\frac{k \left(|x_{i,11} - x_{i,21}|^p + |x_{i,12} - x_{i,22}|^p \right)^{1/p} - r_i}{\sqrt{r_i}} \right]^2 \tag{11}$$

$$\min_{k,p,s} \sum_{i=1}^{n} \left[\frac{k \left(|x_{i,11} - x_{i,21}|^p + |x_{i,12} - x_{i,22}|^p \right)^{1/s} - r_i}{\sqrt{r_i}} \right]^2 \tag{12}$$

Although they are quite simple with respect to the number of variables (which is two or three), the number of nonlinearities induced by these problems can be very large depending on the size of the pairs

[3] We would like the reader to recall that $\pi_i = \langle P_{i1}, P_{i2} \rangle$ is the ith **pair** of points.

Table 2: Average error on the test set per pair using the four parametric distance functions.

	DISTANCE FUNCTION			
RESULTS	$\Phi_1(P_a, P_b)$	$\Phi_2(P_a, P_b)$	$\Phi_3(P_a, P_b)$	$\Phi_4(P_a, P_b)$
Parameters	k=1.082	k=1.320	k=1.286, p=1.688	k=1.60, p=1.769, s=1.829
Average Error (Training Set)	8.024	3.65	3.36	3.18
Average Error (Test Set)	12.396	8.41	8.94	8.14

Table 3: Average error of the neural network and combining. estimators.

ESTIMATOR	**TRAINING ERROR**	**TEST ERROR**
Multi-layer perceptron	2.63	7.91
Regression neural network	2.38	8.49
Voting	2.26	7.63
Stacking	3.81	7.41

of points within the training set. Their minimizations can be carried out by using any known non-linear optimization package, such as MINOS 5.1 [37]. The results using distance functions are given in Table 2. Observe that the rectilinear distance metric has the worst performance. Meanwhile the accuracy of $\Phi_4(P_a, P_b)$ is the highest. These facts definitely support our previous inference on the arbitrariness of the road structure in Türkiye.

4.3 Perceptron Based Methods

In their recent study Alpaydın *et. al.* [2] employed a multi-layer perceptron with one hidden layer with the back-propagation learning rule. In this work the best input representation was determined as the four data values, and the Euclidean distance in between was supplemented as a hint:

$$\boldsymbol{u} = \psi_i(P_a, P_b) = (x_{a1}, x_{a2}, x_{b1}, x_{b2}, \|P_a - P_b\|)^T$$

In terms of output representation, the authors found out that learning the ratio of actual distance to Euclidean distance is better than learning the actual distance itself, namely the directional bias [10, 12] :

$$y = \chi(r) = r/\|P_a - P_b\|$$

This can be perceived as a case of extending $\Phi_2(P_a, P_b)$ in the parametric case. Instead of computing a single global constant k, it is as if the neural network computes a continuous function $k(P_a, P_b)$ by which it scales the Euclidean distance. Note that in these two cases, they can take advantage of the *a priori* knowledge that $\Phi(P_a, P_b) = \Phi(P_b, P_a)$ and effectively double the training set. Details on the implementation, and results with regression neural networks [46] and combining estimators, which use voting [1] and stacking [8, 50] strategies can be found in the same work. We summarize these results in Table 3. The training data they have used is a subset of ours, although the test set is the same.

Table 4: Average error : Single functional form for each subregion

ESTIMATOR	TRAINING ERROR	TEST ERROR
Simple Average	3.34	7.83
$\Phi_2(P_a, P_b)$	3.12	8.05
$\Phi_3(P_a, P_b)$	2.93	7.80
$\Phi_4(P_a, P_b)$	2.77	7.60

4.4 Vector Quantization and The Self-Organizing Map

To implement the algorithms presented in this paper we first divided Türkiye into four geographical subregions. The initial partitioning into regions was done rather arbitrarily. A plot of the Turkish towns used for training is shown in Figure 1 where the latitude and longitude of the bounding rectangle are 36^o N, 26^o E, 42^o N and 45^o E. The towns themselves are marked with an '$\times$'. Note that the actual map of Türkiye has not been superimposed in the figure to avoid cluttering it. The twelve squares, '$\Box$' represent the final positions of the code-book vectors.

The initial partitioning essentially divided the country into four rectangles and shown in the figure each containing 20 towns in the training set L. To demonstrate the power of the strategy, we used a VQ strategy with **only** 3 code-book vectors in each region which were initialized to be on the border of their representative regions. During the intra-regional polarizing phase the VQ algorithm was invoked with a value of α which started at unity and decreased linearly to 0.9 in 1,000 iterations. As expected, most of the learning was accomplished in this phase. Thereafter, in the fine tuning phase the value of α was drastically switched to 0.2 and decreased linearly to attain to 0.1 in 2,000 time steps. In the inter-polarizing phase, the value of α was maintained to 0.1. The constant for the migration of code-book vectors from the same class, ϵ was maintained at 0.25 and done for 2000 iterations (i.e., 25 cycles of all the 80 training sites) Indeed, the entire convergence for both these phases took only a matter of a couple of seconds. As mentioned earlier, the final partitioning (after the code-book vectors converged) was fully determined by the discriminant function implicitly created by the bisectors of the lines joining the code-book vectors. This partitioning is adaptively learnt and is shown in Figure 1 in bold lines. Observe the power of the adaptive regional partitioning scheme.

After the intra and inter-regional learning the constants for the underlying patchwork functions were estimated using the training sites and their corresponding recorded distances. The estimates we computed were of two sorts. First of all, to show the power of the multi-regional approach, we assumed that the distances within each region and the distances between the regions were each characterized by a single functional form. Thus, since we partitioned G into four sets, a functional form of the type (7) involved 10 constants. These constants were estimated by both a simple averaging scheme and a VQ method as explained in Section 3.3. When the explicit form of each intra and inter-regional function was of the types $\Phi_3(P_a, P_b)$ and $\Phi_4(P_a, P_b)$ (where the parameters to be estimated were $\{k, p\}$ and $\{k, p, s\}$), the total number of parameters to be estimated was 20 and 30 respectively. In the latter two cases, the optimization was done independently, and this was typically far a little more time consuming because it involved invoking MINOS 5.1 [37]. The results which we have obtained are quite remarkable and are tabulated in Table 4, where the average training and testing errors are recorded. For example, when the functional form was assumed to be of type $\Phi_2(P_a, P_b)$, the average error obtained by averaging k (which was exactly the error obtained by a VQ algorithm in the k-space) was 7.83. This decreased to 7.80 and 7.60 for the cases when the cases when the parameters were $\{k, p\}$ and $\{k, p, s\}$ respectively.

The full power of the multi-regional approach is clearly displayed if we "patch" the distance function

Table 5: Average error : A separate sub-function for each code-book vector

ESTIMATOR	TRAINING ERROR	TEST ERROR
Simple Average	2.42	7.69
$\Phi_2(P_a, P_b)$	2.36	7.90
$\Phi_3(P_a, P_b)$	2.11	7.48
$\Phi_4(P_a, P_b)$	1.93	7.12

using a separate sub-function **for** each code-book vector and **between** each code-book vector. In this case, since we partitioned G into four sets with 3 code-book vectors in each region, we would involve 78 explicit sub-functions. A functional form of the type (7) would now involve 78 constants, and when the forms of each intra and inter-regional function were of the types $\Phi_3(P_a, P_b)$ and $\Phi_4(P_a, P_b)$ (where the parameters to be estimated were $\{k, p\}$ and $\{k, p, s\}$, the total number of parameters to be estimated was 156 and 234 respectively. Again, as in the above, the latter two cases involved an independent optimization using MINOS 5.1 [37] . The results which we have obtained are truly amazing, and are given in Table 5.

In the most conservative case, the testing error is only 7.69, and in the case when the functions are characterized by $\{k, p, s\}$ the test error went as low as 7.12. The power of the scheme is clear ! Note that the most time consuming phase of the learning is the optimization stage. But since this is done only once (during the training phase) the work done is well worth its while. Subsequently, in the testing phase, the estimation of the distance between any two points merely involves computing the Euclidean distance between them and invoking computation of the functional form associated with their nearest code-book vectors. We are currently investigating how the optimization phase (in $\{k\}$, $\{k, p\}$ or $\{k, p, s\}$) can itself be circumvented by using a VQ algorithm in the corresponding parameter space. This will, of course, involve a gradient descent type of algorithm for each node-pair and distance processed. But since the criterion functions are highly nonlinear deriving such a gradient method is not trivial.

5 CONCLUSIONS AND DISCUSSIONS

In this paper we have studied the problem of estimating arbitrary distance functions using the concepts of Vector Quantization (VQ). This problem has received much attention in Logistics and Location Analysis. The assumptions made on the arbitrary distance function, Φ, are quite relaxed : The set of inter-node distances dictated by Φ may or may not satisfy all the rigorous properties of a well-defined mathematical norm. Furthermore, the triangular inequality may also be violated. However, to keep the informal concepts of a distance measure valid, we impose the requirement that Φ is loosely related to the Euclidean norm, and so if P_i, P_j, P_m and P_n be any four nodes, if the pairs (P_i, P_j) and (P_m, P_n) are "close" to each other, the respective arbitrary distances between (P_i, P_m) and (P_j, P_n) must be correspondingly of *similar* magnitude. Also, we assume that the explicit form of this distance function is both unknown and uncomputable. Unlike traditional Operations Research methods, which use parametric distance functions, we have utilized VQ principles to first adaptively polarize the nodes into sub-regions. Subsequently, the parameters characterizing the sub-regions are themselves learnt using by a variety of methods including a distinct VQ strategy in the (meta) parameter-domain.

The algorithms have been rigorously tested for the actual road-travel distances involving cities and towns in Türkiye. They converge very quickly – in a matter of seconds, and the numerical results obtained are conclusive. Indeed, they are the best results currently available from any **single or hybrid** strategy. The results of Alpaydın *et. al.* [2] show how a combination of learning strategies can be used to yield superior

results by incorporating stacking and voting principles. Clearly, such principles can be used subsequent to our current results to yield even smaller estimation errors.

With regard to the salient features of our present work, to the best of our knowledge, our technique is the pioneering work which uses an "adaptive" multi-regional approach to distance estimation. The regions are learnt adaptively using discriminant functions derived implicitly from the code-book vectors. Subsequent to the partitioning, the actual parameters of the intra-regional and inter-regional functions can be obtained either by optimization (in a non-all-neural approach, for example when k, p and s are parameters) or by using a VQ algorithm in this parameter space itself. This is also novel, because the problem lends itself to both philosophies of learning. Finally, we believe that the VQ (and Kohonen's algorithm) is superior to the Perceptron based methods because unlike the latter, the distance function Φ itself is defined on a well-defined Euclidean space. Consequently, learning the weights (the code-book vectors) within this space is a much more natural characterization than learning it in a space where the weight vectors have no physical significance.

Acknowledgements

We thank Mr. Daryl Graf for various discussions.

References

[1] Alpaydın, E., "Multiple Networks for Function Learning," *IEEE International Neural Network Conference*, San Francisco, vol 1, 9–14, March 1993.

[2] Alpaydın, E., Altınel, İ. K., and Aras, N.,"Parametric Distance Metrics vs. Nonparametric Neural Networks for Estimating Road Travel Distances," Research Paper Series No : FBE-IE-11/94-14, Department of Industrial Engineering, Boğaziçi University, İstanbul,1994.

[3] Altınel, İ. K., and Aras, N., "Estimating Road Distances in İstanbul with Single and Multi Regional Models," Research Paper Series No : FBE-IE-05/94-05, Department of Industrial Engineering, Boğaziçi University,İstanbul, 1994.

[4] Altınel, İ. K., Aras, N., Alie, A., Cangür, G.,Özel, R., and Yücel, A., "Estimating Road Travel Distances in Türkiye", Research Paper Series No : FBE-IE-03/94-03, Department of Industrial Engineering, Boğaziçi University, İstanbul, 1994.

[5] Berens, W., "The Suitability of the Weighted L_p norm in Estimating Actual Road Distances," *European Journal of Operational Research* 34 (1988) 39–43.

[6] Berens, W., and Körling, F., "Estimating Road Distances by Mathematical Functions," *European Journal of Operational Research* 21 (1985) 54–56.

[7] Berens, W., and Körling, F., "On Estimating Road Distances by Mathematical Functions–A Rejoinder," *European Journal of Operational Research* 36 (1988) 254–255.

[8] Breiman, L., "Stacked Regression," TR-367, Department of Statistics, University of California, Berkeley (1992).

[9] Brimberg, J., Dowling, P.D., and Love, R. F., "The Weighted One-two Norm Distance Model : Empirical Validation and Confidence Interval Estimation," *Location Science* 2 (1994) 91–100.

[10] Brimberg, J., and Love, R. F., "A New Distance Function for Modeling Travel Distances in a Transportation Network," *Transportation Science* 26 (1992) 129–137.

[11] Brimberg, J., Love, R. F., and Walker J. H., "The Effect of Axis Rotation on Distance Estimation," *European Journal of Operational Research* 80 (1995) 357–364.

[12] Brimberg, J., and Wesolowsky, G. O., "Probabilistic L_p Distances in Location Models," *Annals of Operations Research* 40 (1992) 67–75.

[13] Duda, R.O., Hart, P.E., *Pattern Classification and Scene Analysis*, John Wiley, 1973.

[14] Erkut, H., and Polat, S., "A Simulation Model for a Urban Fire Fighting System," *Omega* 20 (1992) 535–542.

[15] Fildes, R. A., and Westwood, J. B., "The Development of Linear Distance Functions for Distribution Analysis," *Journal of the Operational Research Society* 29 (1978) 585–592.

[16] Francis, R. L., McGinnis L. F. Jr., and White, J. A., *Facility Layout and Location : An Analytical Approach*, 2nd edition, Prentice Hall, Englewood Cliffs, 1992.

[17] Fukunaga, K., *Introduction to Statistical Pattern Recognition*, 2nd edition, Academic Press, San Diego, 1990.

[18] Graf, D.H., and LaLonde, W.R., "A Neural Controller for Collision-free Movement of General Robot Manipulators," *Proc. IEEE Int. Conf. on Neural Networks*, I77–I-84, 1988.

[19] Graf, D.H., and LaLonde, W.R., "Neuroplanners for Hand/Eye Coordination," *Proc. Int. Joint Conf. on Neural Networks*, II-543–II-548, 1989.

[20] Gray, R. M, "Vector Quantization" *IEEE ASSP Mag.*, vol 1, 4–29,1984.

[21] Hemani, A., and Postula, A., "Scheduling by Self Organisation," *Proc. Int. Joint Conf. on Neural Networks, IJCNN-90-WASH-DC*, II-543-II-546, 1990.

[22] Kohonen, T., "The "Neural" Phonetic Typewriter," *Computer* 21 (1988) 11–22.

[23] Kohonen, T., *Self-Organization and Associative Memory*, 3rd ed. Berlin, Heidelberg, Germany : Springer - Verlag, 1989.

[24] Kohonen, T.,"The Self-Organizing Map," *Proc. IEEE*, vol 78, 1464–1480, 1990.

[25] Kohonen, T., Makisara, K., and Saramaki, T., "Phonotopic Maps – Insightful Representation of Phonological Features for Speech Recognition," *Proc. Seventh. Int. Conf. on Pattern Recognition* 182–185 1984.

[26] Kohonen, T., Torkkola, K., Shozokai, M., Kangas, J., and Venta, O., "Microprocessor Implementation of a Large Vocabulary Speech Recognizer and Phonetic Typewriter for Finnish and Japanese," *Proc. European Conference of Speech Technology* 377–380 1987.

[27] Linde, Y, Buzo, A., and Gray, R. M., "An Algorithm for Vector Quantization," *IEEE Trans. Communication* COM-28 (1980) 84–95.

[28] Love, R. F., and Morris, J. G., "Modeling Inter-City Road Distances by Mathematical Functions," *Operational Research Quarterly* 23 (1972) 61–71.

[29] Love, R. F., and Morris, J. G., "Mathematical Models of Road Travel Distances," *Management Sciences* 25 (1979) 130–139.

[30] Love, R. F., and Morris, J. G., "On Estimating Road Distances by Mathematical Functions," *European Journal of Operational Research* 36 (1988) 251–253.

[31] Love, R. F., Morris, J. G., and Wesolowsky, J., *Facilities Location : Models and Methods*, North - Holland, New York, 1988.

[32] Love, R. F., and Walker, J. H., "An Empirical Comparison of Block and Round Norms for Modeling Actual Distances," *Location Science* 2 (1994) 21–43.

[33] Makhoul, J., Roucos, S., and Gish, H., "Vector Quantization in Speech Coding," *Proc. IEEE*, vol. 73, 1551–1588, 1985.

[34] Marks, K. M., and Goser, K. F., "Analysis of VLSI Process Data Based on Self-Organizing Feature Maps," *Proc. Neuro-Nimes'88*, 337–347, 1988.

[35] Martinetz, J., Ritter, H. J., and Schulten, K. J.,"Three-dimensional Neural Net for Learning Visuomotor Coordination of a Robot Arm," *IEEE Trans. Neural Networks*, 131–136, 1990.

[36] Mittal, A. K., and Palsule, V., "Facilities Location with Ring Radial Distances," *Institute of Industrial Engineers Transactions* 16 (1984) 59–64.

[37] Murtagh, B. A., and Saunders, M. A., "MINOS 5.1 User's Guide," Technical Report No : SOL 83 - 20R, Stanford University, Stanford, California, 1983 (revised 1987).

[38] Neumann, E. K., Wheeler, D. A., Burnside, A. S., Bernstein, A. S., and Hall, J. C., "A Technique for the Classification and Analysis of Insect Courtship Song," *Proc. Int. Joint Conf. on Neural Networks, IJCNN-90-WASH-DC*, II-257-II-262, 1990.

[39] Orlando, G. A., Mann, R., and Haykin, S., "Radar Classification of Sea-ice Using Traditional and Neural Classifiers," *Proc. Int. Joint Conf. on Neural Networks, IJCNN-90-WASH-DC*, II-263-II-266, 1990.

[40] Perreur, J., and Thisse, J., "Central Metrics and Optimal Location," *Journal of Regional Science* 14 (1974) 411–421.

[41] Potvin, J., "The Travelling Salesman Problem: A Neural Network Perspective," *ORSA Journal on Computing* 5 (1993) 328–348.

[42] Ritter, H. J., Martinetz, J., and Schulten, K. J., "Topology Conserving Maps Learning Visuomotor Coordination," *Neural Network* 2 1989 159–168.

[43] Ritter, H. J., and Schulten, K. J., "Topology Conserving Mappings for Learning Motor Tasks," *Proc. Neural Networks for Computing, AIP Conference*, 376–380, 1986.

[44] Ritter, H. J., and Schulten, K. J., "Extending Kohonen's Self-organizing Mapping Algorithm to Learn Ballistic Movements," *NATO ASI Series*, 393–406, 1988.

[45] Samarabandu, J. K, and Jakubowicz, O. E., "Principles of Sequential Feature Maps in Multi-Level Problems," *Proc. Int. Joint Conf. on Neural Networks, IJCNN-90-WASH-DC* II-683-II-686 1990.

[46] Specht, D. F., "A General Regression Neural Network, "*IEEE Transactions on Neural Networks*," 2, 568–576, 1991.

[47] Tryba, V., Marks, K. M., Rückert, U., and Goser, K., "Selbst-organizierende Karten Als Lernende Klassifizierende Speicher," *IFG Fachbericht*, 102, 407–419, 1988.

[48] Ward, J. E., and Wendell, R. E., "A New Norm Measuring Distance which Yields Linear Location Problems," *Operations Research* 28 (1980) 836–844.

[49] Ward, J. E., and Wendell, R. E., "Using Block Norms for Location Modeling," *Operations Research* 33 (1985) 1074–1091.

[50] Wolpert, D.H., "Stacked Generalization," *Neural Networks* 5 (1992) 241–259.