K-Winner Machines for Pattern Classification

Sandro Ridella, *Member, IEEE*, Stefano Rovetta, *Member, IEEE*, and Rodolfo Zunino, *Member, IEEE*

***Abstract*—The paper describes the K-winner machine (KWM) model for classification. KWM training uses unsupervised vector quantization and subsequent calibration to label data-space partitions. A K-winner classifier seeks the largest set of best-matching prototypes agreeing on a test pattern, and provides a local-level measure of confidence. A theoretical analysis characterizes the growth function of a K-winner classifier, and the result leads to tight bounds to generalization performance. The method proves suitable for high-dimensional multiclass problems with large amounts of data. Experimental results on both a synthetic and a real domain (NIST handwritten numerals) confirm the approach effectiveness and the consistency of the theoretical framework.**

***Index Terms*—Pattern classification, supervised learning, unsupervised learning, vector quantization.**

I. Introduction

THE ESTIMATION of generalization error is the most critical issue in classifier design. An empirical approach to this task consists in splitting available samples into a training and a test set for cross-validation [1]–[3]. Otherwise, a large variety of methods have been proposed to characterize a classifier's generalization ability theoretically [4]–[7]. Among these, the formulation based on the properties of the growth function and of the Vapnik–Chervonenkis (VC) dimension [8] offers a most general theoretical foundation. A crucial feature of Vapnik's approach is that the theoretical result stems from a worst-case analysis. The estimated classification accuracy often falls in a pretty wide range, which may be of limited usefulness in practical applications. Therefore, Vapnik recently proposed a domain-oriented method named support vector machine (SVM) [9]–[11]. The design criterion for SVM classifiers is to maximize the error margin, a quantity involving distances between training patterns and the separation surfaces among classes. The SVM paradigm joins theoretical validity with practical impact, especially in complex domains involving huge data sets.

The domain-oriented perspective that inspires structural risk minimization [9], [10] leads the trend in classifier design: application performance is privileged, yet preserving consistency within a theoretical framework. In this context, the present paper describes a classification model called *K-winner machine* (KWM), whose novel conceptual contribution lies in combining unsupervised with supervised training. Vector quantization (VQ) provides the basic paradigm, as the unsupervised part of the design criterion requires that the distortion of data representation be minimized.

In a VQ-based approach, the data space is mapped by a set of reference vectors of "prototypes," which lie at significant locations and span a partitioning schema over the pattern distribution. The algorithms so far proposed in the literature to position prototypes pursue topological consistency [12], [13], uniform occurrence probability [14], uniform coverage of the data space [15], or uniform approximation for the probability distribution [16]. The basic model of KWMs is actually independent of the specific prototype-positioning algorithm adopted.

The training of a KWM is straightforward: in the unsupervised training phase, VQ prototypes are placed according to the spatial pattern distribution, regardless of pattern classes; then, in the supervised training phase, prototype calibration exploits the class information contained in each data-space partition.

In compliance with the principle of structural risk minimization, a K-winner machine includes a set of nested classifiers (k-winner classifiers or kWCs) characterized by specific bounds to the generalization error. The elementary kWC classifies a pattern by checking the classes associated with the k best-matching prototypes: if all the classes agree, the pattern is classified accordingly, otherwise it is discarded. The overall KWM chooses, for each test pattern, the most confident classifier, that is, the kWC that does not discard the test pattern, and that is characterized by the best estimated generalization error. As a result, a KWM associates each classification output with the tightest bound to the expected generalization error, and is not subject to the problem of rejected patterns.

The paper shows that it is possible to characterize the growth function of a kWC. The computed value depends on the number of prototypes and is independent of the data dimensionality. Such a property can notably improve classification performance, especially in applications involving high-dimensional domains. The properties of the growth functions of kWCs make it possible to label each location in the data space by a bound to the expected generalization error.

The validity of the KWM model is first demonstrated experimentally on an artificial testbed in a two-dimensional (2-D) domain, allowing a visual interpretation of results. The practical impact is then evaluated on a real testbed, i.e., the NIST handwritten numerals database, involving large numbers of patterns from a high-dimensional data space for a complex, multiclass problem.

Section II describes k-winner classifiers and the KWM model. Section III analyzes the growth function of a kWC and the generalization performance of a KWM. Section IV reports on experimental results. Finally, some concluding remarks are made in Section V.

Manuscript received March 22, 1999; revised September 14, 2000. This work was supported in part by the Italian Ministry for the University and Scientific and Technological Research (MURST).

The authors are with DIBE, Department Biophysical and Electronic Engineering, University of Genoa, 16145 Genova, Italy (e-mail: ridella@dibe.unige.it; rovetta@dibe.unige.it; zunino@dibe.unige.it).

Publisher Item Identifier S 1045-9227(01)01405-9.

II. The K-Winner Machine (KWM)

The KWM adopts a dual-paradigm approach: first, a VQ schema uses available prototypes to render the probabilistic pattern distribution; then, each prototype is labeled by the predominant class within its data-space partition. Decoupling vector positioning from subsequent class assignment plays a key role in design aimed at generalization performance. This approach prevents an uncontrolled (and practically detrimental) explosion of the classifier's growth function.

A. Training Procedure for the KWM

The KWM design criterion is to optimize the representation of the data distribution by a VQ mechanism. The D-dimensional data space is partitioned by a set of prototypes, $\mathrm{W} = \{\mathbf{w}_n \in \Re^D,\ n = 1, \ldots, N_h\}$, which lie at "significant" positions in the data space; each prototype covers the patterns lying within its associate partition. The process assigning a prototype to each pattern follows a best-match criterion minimizing a distortion cost. Euclidean metrics is usually adopted to measure distortion, hence a data pattern, $\mathbf{x} \in \Re^D$, is associated with the prototype, $\mathbf{w}^*(\mathbf{x}) \in \mathrm{W}$, that satisfies

$$\mathbf{w}^*(\mathbf{x}) = \arg\min_{\mathbf{w}\in\mathrm{W}} \left\{ \|\mathbf{x} - \mathbf{w}\|^2 \right\}. \tag{1}$$

The VQ-representation problem implies finding the optimal set of prototypes, W, that minimizes the overall distortion: $E(\mathrm{W}) = \int_{\Re^D} \|\mathbf{x} - \mathbf{w}^*(\mathbf{x})\|^2 p(\mathbf{x})\, d\mathbf{x}$. The pattern distribution is not known *a priori*, hence the integral cannot be computed analytically in any but very peculiar cases. Therefore, one usually resorts to an empirical estimation of the involved distortion: a set of training patterns, $\mathrm{X} = \{\mathbf{x}_l \in \Re^D,\ l = 1, \ldots, N_p\}$, drives vector positioning to minimize the empirical cost

$$\hat{E}(\mathrm{W}) = \frac{1}{N_p} \sum_{l=1}^{N_p} \|\mathbf{x}_l - \mathbf{w}^*(\mathbf{x}_l)\|^2. \tag{2}$$

Searching for the global minimum of (2) is very expensive from a computational point of view, hence a large variety of iterative approaches have been proposed in the literature. The k-means algorithm [17] provides the classical pattern-recognition approach to the problem. Neural models exploit interneuron connectivity to derive a topologically consistent mapping of training data [12], [13]. Some methods privilege probabilistic aspects and aim at an accurate rendering of the data distribution [14], [16]; others tend to average distortion over available prototypes through a uniform coverage of training patterns [15], [18].

As far as the KWM model is concerned, the only constraint on the applied VQ algorithm is that it should support unsupervised training. In principle, models privileging a consistent mapping of the data distribution should be preferred; empirical practice, however, did not point out any significant differences in the several alternatives considered. The research presented in this paper adopted the plastic neural gas model [19] for two reasons: 1) given a constraint on final distortion, the VQ model can simultaneously assess the number and positions of prototypes and 2) an efficient hardware implementation of the training algorithm has been developed to reduce the considerable computational cost [20]. For the reader's convenience, a sketch of the PGAS algorithm is given in the Appendix.

In order to build up a classification machine based on the previous unsupervised representation, assume now that some criterion is available to assign a class to each prototype after it has been positioned. This process is conventionally named "calibration" [12]. From a general perspective, the calibration mechanism is required to label the tessellation produced by the VQ training process. In KWM training, such a process labels a prototype according to the majority of patterns covered by the prototype; a possible "tie" case is solved by choosing a class at random from among the best candidates. The following definitions help describe the KWM training process formally:

$\mathrm{C} = \{c^{(k)},\ k = 1, \ldots, N_c\}$	set of categories from which patterns are drawn;
$\mathrm{W} = \{\mathbf{w}_n \in \Re^D,\ n = 1, \ldots, N_h\}$	prototype set mapping the data space;
$\mathrm{X} = \{(\mathbf{x}_l, c_l),\ \mathbf{x}_l \in \Re^D,\ c_l \in \mathrm{C},\ l = 1, \ldots, N_p\}$	set of labeled training patterns;
$P_n = \{\mathbf{x} \in \Re^{\mathrm{D}} : \mathbf{w}^*(\mathbf{x}) = \mathbf{w}_n\}$	data space partition that is spanned by the nth prototype;
$\alpha_n^{(k)},\ k = 1, \ldots, C$	shares of patterns lying in P_n and belonging to each class; $\sum_{k=1}^{N_c} \alpha_n^{(k)} = 1$.

The KWM construction procedure combines unsupervised VQ training with subsequent calibration.

K-Winner Machine (Training)

0. Input: training set of labeled data, X;
1. (Unsupervised prototype positioning) Apply an unsupervised VQ algorithm (PGAS [19]) to adjust the prototype set, W, minimizing (2);
2. (Calibration) Calibrate W into a labeled set of prototypes, W′, computed as

$$\mathrm{W}' = \{(\mathbf{w}_n, c_n),\ \mathbf{w}_n \in \mathrm{W},\ c_n \in \mathrm{C},\ n = 1, \ldots, N_h\}$$
$$\text{where: } c_n = c_b,\ b = \max_k \{\alpha_n^{(k)}\}.$$

B. Run-Time Operation of a k-Winner Classifier (kWC)

The principle of operation of a kWC lies in checking the agreement of k reference sites in the data space to make a reliable decision on a test location. The kWC classification process, denoted by $k\mathrm{WC}(k, \mathbf{x})$, involves two steps. The first performs an unsupervised categorization of the test pattern, $\mathbf{x}$, with the k best-matching prototypes; the second considers the calibrations of the prototypes and classifies the pattern accordingly.

k-Winner Classifier (run-time operation)
$\equiv k\mathrm{WC}(k, \mathbf{x})$
0. Input: test pattern $\mathbf{x} \in \Re^D$, trained prototype set, W', agreement level k
1. (k-winner unsupervised categorization)
1.1 Sort the set of prototypes, $\mathrm{W}''(\mathbf{x})$, arranging them in order of increasing distance from $\mathbf{x}$

$$\mathrm{W}''(\mathbf{x}) = \{\mathbf{w}_{n_r} \in \mathrm{W}' : r < s \Rightarrow \|\mathbf{x} - \mathbf{w}_{n_r}\| \leq \|\mathbf{x} - \mathbf{w}_{n_s}\|, r = 1, \ldots, N_h\};$$

1.2 Extract the set $\mathrm{W}_k(\mathbf{x}) \subseteq \mathrm{W}''(\mathbf{x})$ including the k best-matching prototypes with respect to $\mathbf{x}$

$$\mathrm{W}_k(\mathbf{x}) = \{\mathbf{w}_{n_r} \in \mathrm{W}''(\mathbf{x}), r = 1, \ldots, k\};$$

2. (k-agreement classification)
If $\exists c^* : \forall\, \mathbf{w}_{n_r} \in \mathrm{W}_k(\mathbf{x}) c_{n_r} = c^*$
then: Classify $\mathbf{x}$ as belonging to class c^*
else: Discard $\mathbf{x}$.

The K-Winner Machine (KWM)
0. Input: a test pattern, $\mathbf{x}$; a trained and calibrated set of prototypes, W'; $k = 1$; $\pi^* = 1$;
1. While: $k\mathrm{WC}(k, \mathbf{x})$ does not discard $\mathbf{x}$
1.a) If $(N(k) > 0)$ and $(\pi(k) < \pi^*)$
$\pi^* = \pi(k)$;
1.b) $k = k + 1$;
2. Output:
2.a) classify $\mathbf{x}$ using $k\mathrm{WC}(k-1, \mathbf{x})$;
2.b) set the expected error probability, π^*, associated with the proposed classification.

The outcome of the kWC classification process depends on the agreement of all the elements of the set of k neighbors associated with a pattern, and exhibits some basic interesting features. First of all, the minimal kWC configuration with $k = 1$ always classifies any test pattern and does not allow a "discard" output. Second, as a consequence of such a property, it is easy to show that if two different kWCs do not discard a pattern, they must necessarily prompt the same classification output, which also coincides with the class prompted by the simplest kWC with $k = 1$. On the other hand, different kWC's are characterized by different generalization abilities, as indicated by the specific growth function of each kWC and the associated bound to its expected generalization performance. The latter quantity depends on the kWC specific level, k, and on the number of patterns, $N(k)$, that the kWC does not discard. In the following, such a bound to the expected generalization error will be denoted by $\pi(k)$. The underlying theory will be provided in Section III.

C. Run-Time Operation of the KWM

The class-agreement principle ruling a kWC offers the possibility of discriminating between space regions on the basis of the associate confidence in their classification. In practice, from the highest prototype agreement, K, attained by the family of kWCs on a given test location, one can infer the level of confidence in the classifier's decision.

Thus, for each point in the space, one can select the kWC that does not discard the test point and makes the smallest estimated generalization error. This variable-confidence mechanism allows one to label data-space regions according to their expected generalization errors. This opportunity is the principle of operation of the K-winner machine, whose algorithm is outlined as follows.

The KWM processes a pattern starting by a kWC with $k = 1$; the associate kWC spans the entire domain space and supports a conventional winner-take-all (WTA) categorization schema. In the subsequent steps, the process tries to improve the confidence in the decision by checking larger and larger sets of neighboring prototypes; if successful, this mechanism has the effect of reducing the bound to the error probability. This fundamental property will be proved in Section III. It is therefore reasonable that the additional information should yield a better estimate, as compared with that of a straightforward WTA-based classifier.

Such features represent the basic difference between KWM and multiple-voter classifiers. KWMs do not involve any majority mechanism in the classification process, and the individual contributions are drawn from the same set of prototypes rather than from multiple and independent experts. In comparison with other approaches [21], the KWM model has the advantage of implicitly and easily facing multiclass classification problems. Indeed, the functioning of a KWM stems from prototype calibration, which supports any arbitrary number of categories.

Thanks to the selection mechanism, the KWM model is not subject to the drawback of rejected patterns. Any input pattern is labeled by a confidence level, K, related to the associated smallest error probability. In the following, we shall denote by K the eventual confidence level determined by the overall KWM, and by k the agreement parameter that characterizes a specific kWC.

III. Theory for the KWM Model

This section presents theoretical properties that describe the behaviors of k-winner classifiers and of the overall KWM. For the sake of the analysis, in the following we will assume that the prototypes have been fixed before the data sample was generated. Theorem proofs are given in Appendix B for clarity.

A. Theoretical Derivation of the Growth Function of a k-Winner Classifier

Computational Learning Theory characterizes a classifier in terms of its VC dimension, which is the largest number of patterns that can be shattered by the classifier [22]. In the peculiar case of $K = 1$, however, the KWM model reduces to a prototype-based nearest-neighbor classifier that categorizes a test pat-

tern with the class of the best-matching prototype. In this case, the following relevant property can be obtained.

Theorem 1: The VC dimension of a k-winner classifier using N_h prototypes and $k = 1$ is $d_{\rm VC}^{(1)} = N_h$.

The relevance of the unsupervised training phase to the above theorem can be highlighted by a simple example. Consider a kWC with $k = 1$ and $N_h = 2$. If prototype positions were set by a supervised training algorithm (e.g., LVQ [12]), then the resulting classifier would be equivalent to a Perceptron [9], and its VC-dim would be equal to $d_{\rm VC} = D+1$, where D is the space dimensionality. By contrast, the VC-dim of the actual kWC with unsupervised prototype positioning amounts to $d_{\rm VC}^{(1)} = 2$.

For reasons that will be clarified later on, the use of unsupervised training inhibits the application of the VC-dim when $k > 1$. However, a theoretically consistent derivation of the growth function of the kWC can be obtained by following its definition [22]: the growth function, GF(n), of a classifier gives the largest number of target configurations that can be processed correctly by the classifier for a certain number of patterns, n. It is worth reminding that the GF of a classifier does not depend on any specific training set but only on its cardinality. Again, the following derivations assume that the prototypes have been set independently of any specific data sample. The following theorem derives the GF, GF$^{(k)}(n)$, of a kWC.

Theorem 2: The growth function of a k-winner classifier using N_h prototypes is ${\rm GF}(n) = 2^{d_{GF}^{(k)}}$, where $d_{\rm GF}^{(k)} = \lfloor N_h/k \rfloor$.

Theorem 2 defines an important feature of the functioning of a kWC, namely, that the growth function of the classifier is independent of the actual dimensionality of the data space. Conversely, the independence of the GF of the number of patterns stems mainly from the unsupervised WTA-based procedure that assigns patterns to the associated data-space partitions. Any configuration of prototypes generates a unique tessellation of the data space. Therefore, once a prototype set is established (either by *a priori* setting or after empirical unsupervised training), boundaries among data partitions do not depend on pattern classes, and the data partition associated with each subgraph remains unchanged for different class assignments. Thus the number of functions that can be supported by a kWC based on a given prototype set is fixed, and will depend only on the number of available prototypes.

As a final remark, we now briefly give the reasons why one cannot define the VC-dim of a kWC with $k > 2$. Unsupervised training requires that the number of patterns, N_p, be equal or larger than the number of prototypes, N_h, in order to avoid the occurrence of dead prototypes. Therefore, the $d_{\rm VC}$ of the kWC must be equal to or larger than N_h. At the same time, the analysis made in the proof of Theorem 2 shows that only $\lfloor N_h/k \rfloor$ patterns can be shattered, which shows that the VC-dim of such a classifier can be determined only for $k = 1$.

B. Generalization Performance of a k-WC

In order to apply Vapnik's theory to the specific k-winner classifier under the assumptions of Theorem 2, one restricts the input space to those points for which the k nearest prototypes are in the same connected component of the graph. The prototype graph can only be labeled with distinct labels on distinct components, so that for the restricted input space the nearest k labels will always be consistent. As a consequence, on the restricted input space the kWC never rejects inputs and so the generalization can be estimated by using Vapnik's theory.

In [9, p. 72–73] Vapnik gives two basic results that provide a worst-case estimate of the classifier's generalization error π on the overall data distribution. The estimator is the empirical training error, ν, defined as the ratio of misclassified training patterns to the total number of training patterns. The first result (expression 3.15 therein) is

$$\pi \leq \nu + \tfrac{1}{2}\sqrt{\varepsilon}. \tag{3}$$

The second result is given by

$$\pi \leq \nu + \frac{\varepsilon}{2}\left(1 + \sqrt{1 + \frac{4\nu}{\varepsilon}}\right). \tag{4}$$

The quantity ε is defined as ([9, 3.14])

$$\varepsilon = \frac{4}{n}\left(\ln GF(2n) - \ln\frac{\eta}{4}\right) \tag{5}$$

where η is the expected confidence in the result.

In order to apply these results to a kWC, we define the following quantities:

- $N(k)$ number of training patterns that are processed by the kWC; this number is equal to or less than the total number of patterns, N_p, because a share of the overall training set may be discarded by the kWC;
- $\Omega(k)$ portion of $N(k)$ training patterns that are misclassified by the kWC;
- $\nu(k)$ empirical error $\nu(k) = \Omega(k)/N(k)$;
- $\Pi(k)$ portion of a test sample, including $N(k)$ patterns, that are misclassified by the kWC, estimated by Vapnik's worst-case theory;
- GF$^{(k)}(n)$ growth function of the kWC; Theorem 2 proves that it is given by: ${\rm GF}^{(k)}(n) = 2^{d_{\rm GF}^{(k)}}$.

By using the result of Theorem 2 in expression (5), one can write

$$\varepsilon(k) = \frac{4}{N(k)}\left[d_{\rm GF}^{(k)}\ln 2 - \ln\frac{\eta}{4}\right]. \tag{6}$$

When $k = 1$, one will use the result of Theorem 2 in (6), as the result of Theorem 1 would lead to a broader bound. The designer will choose the sharper bound between (3) and (4). In fact, Vapnik's bound (4) is usually preferable when dealing with a small error on the training set ($\nu \ll 1$), and will be adopted as a default throughout the paper. By using Theorem 2 and the previous definitions one can prove the following property.

Theorem 3: For any k-WC, the worst-case number of misclassified test patterns, $\Pi(k)$, out of a test sample of $N(k)$ patterns is monotonically nonincreasing as k increases.

Theorem 3 states that increasing k reduces the worst-case number of misclassified patterns in generalization performance. Therefore, the worst-case error probability of a kWC is given by

$$\pi(k) = \frac{\Pi(k)}{N(k)} \leq \nu(k) + \frac{\varepsilon(k)}{2}\left(1 + \sqrt{1 + \frac{4\nu(k)}{\varepsilon(k)}}\right). \tag{7}$$

Expression (7) applies only to the patterns that are not discarded by the kWC and when $N(k) > 0$. The case $N(k) = 0$ denotes a

subsampling phenomenon, which certainly occurs for very large values of k. In such peculiar situations, one cannot characterize the associate kWC, and the KWM resorts to the closest, smallest value of k that makes (7) valid.

C. Relevance to the KWM

Summarizing the above theoretical framework will highlight some basic aspects that may help understand the actual functioning of a KWM. The crucial feature of the KWM model is the preliminary unsupervised training phase, which strongly reduces the number of functions supported by kWCs. As to the supervised training phase, it is worth stressing that the calibration of a prototype considers only the patterns lying in the prototype's partition and does not depend on the classes of all the other patterns in the training set.

Calibrated prototypes support the family of kWCs. According to the classification mechanism, any kWC that does not discard a pattern must necessarily agree with the kWC with $k = 1$. Instead, kWCs differ in their expected generalization performances. The values of $\pi(k)$ depend on k for two main reasons: first, Theorem 2 states that different kWCs have different growth functions; second, the number of covered patterns, $N(k)$, may depend on k for $k > 1$.

The latter property raises the issue of discarded patterns. In particular, one might wonder whether Vapnik's theory still applies when the quantities that describe a kWCs behavior depend on k. In order to prove this property, one should first consider a few fundamental facts.

First, the tessellation of the domain space is uniquely set by prototype positions, and the resulting partitions do not depend on pattern classes. On the other hand, calibration results stem either from fixed settings or from sample-driven training that minimizes the empirical error for each prototype; in any case, class assignment proceeds locally within each partition.

As prototype calibrations are mutually independent, the behavior of a specific kWC can be analyzed independently of the others. The only constraint deriving from different values of k is that $N(k) \geq N(j)\ \forall\ j > k$. Indeed, one can easily verify that the quantity $N(k+1)$ varies in the range $\{0, 1, \ldots, N(k)\}$; in the lack of additional information, one can *a priori* only assume a uniform distribution for $N(k+1)$ within such range. Therefore, all the quantities associated with a kWC [i.e., the number of covered patterns, $N(k)$, and the number of errors, $\Omega(k)$], are determined once and for all when calibration results are set. One can also directly see that $N(k)$ and $\Omega(k)$ are statistically independent of each other: the former is related to the spatial distribution of samples, $p(\mathbf{x})$, whereas the latter results from the class probability, $p(c|\mathbf{x})$, which is independent of $p(\mathbf{x})$. Finally, we also stress that the resulting error rate, $\nu(k)$, is ultimately fixed by calibration and is not subject to any optimization process that takes into account the value of k.

These considerations prove that, for any kWC, the mechanism that selects $N(k)$ patterns out of the original sample of N_p patterns operates independently of the quantity $\nu(k)$. The independence of the relevant quantity of the pattern-selection criterion is the basic prerequisite for performing any estimation on a subset of the overall sample of data. This holds true also to apply Vapnik's theory, which allows one to use any subset of the training set, provided the selection is made independently of the resulting empirical error. The above considerations make it possible to state that the theoretical results (6) and (7) apply consistently if $N(k)$, $\nu(k)$ and $\varepsilon(k)$ are used whenever appropriate.

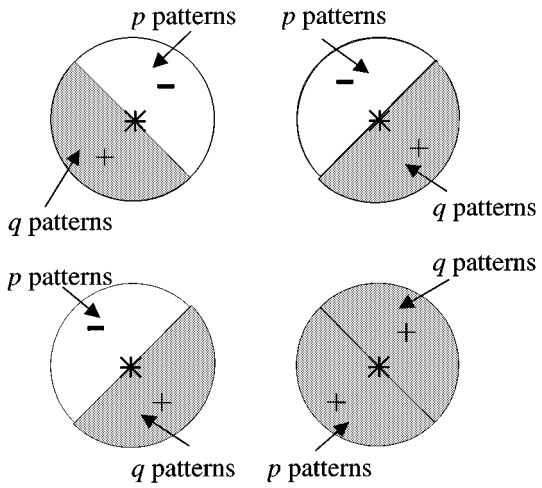

Fig. 1. Sample problem to demonstrate the role of calibration in the KWM operation. Stars indicate the positions of prototypes determined by unsupervised training.

The crucial difference that results from including the value of k in the training process is pointed out by the following example. A two-dimensional (2-D) sample (Fig. 1) is composed of four equiprobable nonoverlapping clusters, whose patterns may belong to two classes ("+" and "−"). In the case $N_h = 4$, unsupervised training places a prototype in the center of each cluster, containing $p + q$ patterns ($p > q$).

Consider now the case of $k = 2$: 1) If one trains a KWM with the procedure described in Section II-A, prototype calibration will give the kWCs results and decision regions shown in Fig. 2(a); the consequent 2-winner classifier will bring about a number of empirical training errors $\Omega(2) = q$, and the number of discarded patterns will be $(p+3q)$. 2) Instead, if one calibrates the prototypes to optimize specifically the kWC empirical error, a better solution for $k = 2$ will be that presented in Fig. 2(b): the number of misclassified patterns will be $\Omega(2) = 0$, whereas the number of discarded patterns will still amount to $(p+3q)$. This implies that solution 1) is nonoptimal as long as $\nu(k)$ is considered; in case 2), however, the calibration settings to minimize $\Omega(2)$ are chosen *after* evaluating the situation for $k = 2$. As the kWC training affects the calibration process of individual prototypes, the theoretical results reported in Sections II-A and B are no longer valid.

D. Extension to a Multiclass Case

The properties of the KWM model presented in the previous sections can be extended to multiclass problems. The theory for a multiclass case has been developed in the literature [23], [24], stating that the definition of the growth function for multiclass problems is analogous to that for a binary-classification case: the number of target configurations for a number of patterns, n, belonging to $N_c > 2$ possible classes is given by $(N_c)^n$ rather than 2^n.

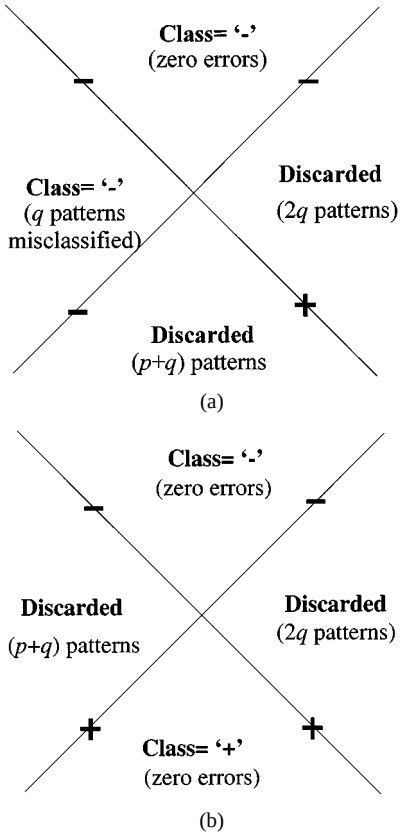

Fig. 2. Decision regions and results of the two-winner classifiers trained by different strategies. (a) After KWM calibration. (b) After calibration tuned to the kWC ($k = 2$). Case (b) yields a smaller empirical error but does not allow the application of KWM theory.

Theorem 4: The growth function of a k-WC using N_h prototypes in a multiclass case is $\mathrm{GF}(n) = N_{\mathrm{c}}^{d_{GF}^{(k)}}$, where $d_{\mathrm{GF}}^{(k)} = \lfloor N_h/k \rfloor$.

By applying the demonstration used for Theorem 2, the proof is straightforward, considering that the calibration phase may assign any of the N_c classes to each prototype. The assertion follows immediately when including this fact in (10) and then substituting the result in to (11).

A pattern is counted in $\Omega(k)$ whenever the kWC assigns it to a class different from its original one. In compliance with the approach described in [23, p. 112], if one includes the result of Theorem 4 in the expressions derived in Section III-B, one can rewrite (6) as

$$\varepsilon(k) = \frac{4}{N(k)} \left[d_{\mathrm{GF}}^{(k)} \ln N_{\mathrm{c}} - \ln \frac{\eta}{4} \right] \tag{8}$$

and Vapnik's generalization theory (6) and (7) can be applied to the multiclass case, as well. In fact, such a property is reported rather seldom in the literature. This depends mainly on the considerable difficulties inherent in the computation of the growth function of a classifier; this opportunity, peculiar to the KWM model, greatly facilitates the theoretical characterization of the overall model.

IV. Experimental Results

This section describes the experimental verification of the KWM model in a synthetic and a real-world domain. Both testbeds entail a multiclass problem: the former involves a 2-D space allowing a visual interpretation of obtained results, the latter addresses a well-known standard database related to a complex and technically very significant recognition problem (OCR).

A. Artificial-Domain Tests: 3-Gaussian Problem

This artificial testbed combines three Gaussian distributions, which have the same variance and are placed symmetrically on a 2-D plane. The experiments aimed to demonstrate the space-labeling ability of KWM's. The Gaussian distributions were centered in $\mathbf{c}_1 = (1, 0)$, $\mathbf{c}_2 = (\cos(2\pi/3), \sin(2\pi/3))$, and $\mathbf{c}_3 = (\cos(4\pi/3), \sin(4\pi/3))$, and gave rise to a three-class problem. The KWM performance was tested in three different experiments. The variance value (equal for the three Gaussians) was progressively increased from 0.1 to 0.5 and up to 1; thus the classification system was tested for problems of increasing difficulty in separating the three classes. For each experiment, 6000 total training patterns were randomly generated [Fig. 3(a)–(c)]. Likewise, an equal number of test patterns were randomly generated to assess generalization performance. In the KWM training process, the number of prototypes used in each experimental run was always set to 30, ensuring the constant ratio 200 of the number of training patterns to that of prototypes.

The artificial nature of the problem allows one to work out the optimal error rate, attained by a Bayesian classifier using the correct decision surfaces. Such a classifier is easily implemented by placing three prototypes in the centers of the Gaussian distributions. An alternative classical approach involves the 1-nearest neighbor (1-NN) classifier, which classifies each test pattern according to the best-matching pattern in the training set. These methods were compared with the 1-winner classifier. Table I gives the classification errors incurred by the different approaches, and confirms the efficacy of the KWM model, even in the basic case of $K = 1$.

The spatial effect of the variable confidence was analyzed by an exhaustive procedure. Each point of the data space underwent a KWM-based classification and was labeled by the associate level of confidence, K. Such an approach made it possible to plot homogeneous regions of the data space, that is, those holding equally labeled points. The resulting graph provides a "confidence map" and allows an easy and intuitive interpretation of the KWM functioning. The results for the three variance settings are presented in Fig. 4. Each gray level indicates the KWC ruling a specific region of the data space; darker points indicate tighter bounds to the generalization error. Thus each graph suggests the appropriate value of K to be used at each space location. As such, the confidence map can also be regarded as a map of variable bounds (7) to the error probability.

In the "simple" situation [Fig. 4(a)], the symmetrical spatial configuration proves a consistent separation of the classes. It is worth noting that the dark, uniform regions extend almost to the separation boundaries; this is the consequence of the compact data distributions. The actual class-separation area is indicated by bright gray and corresponds to the region of maximum uncertainty in the decision ($K = 1$). The fact that intermediate gray levels are virtually absent witnesses the minimum

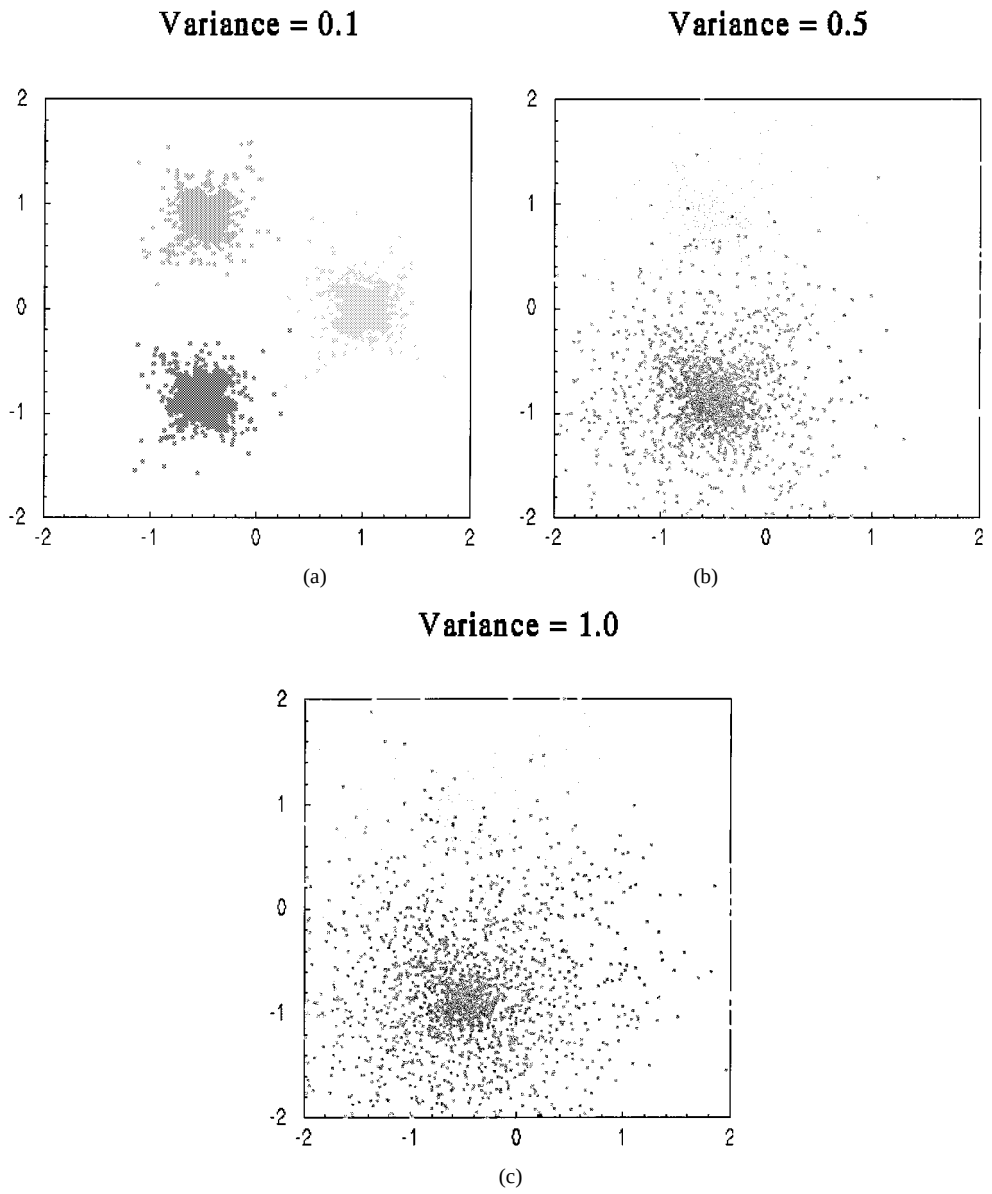

Fig. 3. Different versions of the artificial three-Gaussian problem. (a) Simple; (b) Medium; (c) Difficult.

TABLE I
THE THREE-GAUSSIAN EXPERIMENTS: COMPARISON OF THE ACCURACIES OF DIFFERENT CLASSIFIERS

		Bayesian	1-NN	1-WC
$\sigma^2 = 0.1$	Training set	0.03%	-	0.05%
	Test set	0.07%	0.03%	0.05%
$\sigma^2 = 0.5$	Training set	6.08%	-	7.00%
	Test set	6.15%	9.15%	6.88%
$\sigma^2 = 1.0$	Training set	14.6%	-	15.80%
	Test set	15.13%	21.73%	16.42%

overlap among the Gaussian distributions. "Medium" and "complex" situations are depicted in Fig. 4(b) and (c), respectively. The dark, "certain" areas shrink when the overlap among the classes increases, the highest-uncertainty (brightest) region becomes larger and larger, and intermediate confidence levels assume greater importance. The appearances of the decision surfaces tend to a smooth but steady degradation. Such results confirm the direct connection between the relative distribution of uncertainty and the problem complexity. The ability of KWM's to associate space locations with confidence levels may prove very useful in high-dimensional domains, where class inspection is often desirable but a visual interpretation is not feasible.

Finally, the theoretical analysis of generalization performance was validated by classifying the 6000 random patterns not included in the training set. The experimental results in the three situations can be deduced by comparing the experimental error rate with the theoretical bound, $\pi(k)$, to the generalization error (7). As the latter quantity is subject to a statistical fluctuation, its 5% confidence ranges are also provided, and are worked out [25], when applicable, as

$$\Delta err(k) = \sqrt{\frac{\ln(2/\eta)}{2N(k)}}; \qquad N(k) > 0; \qquad \eta = 0.05. \quad (9)$$

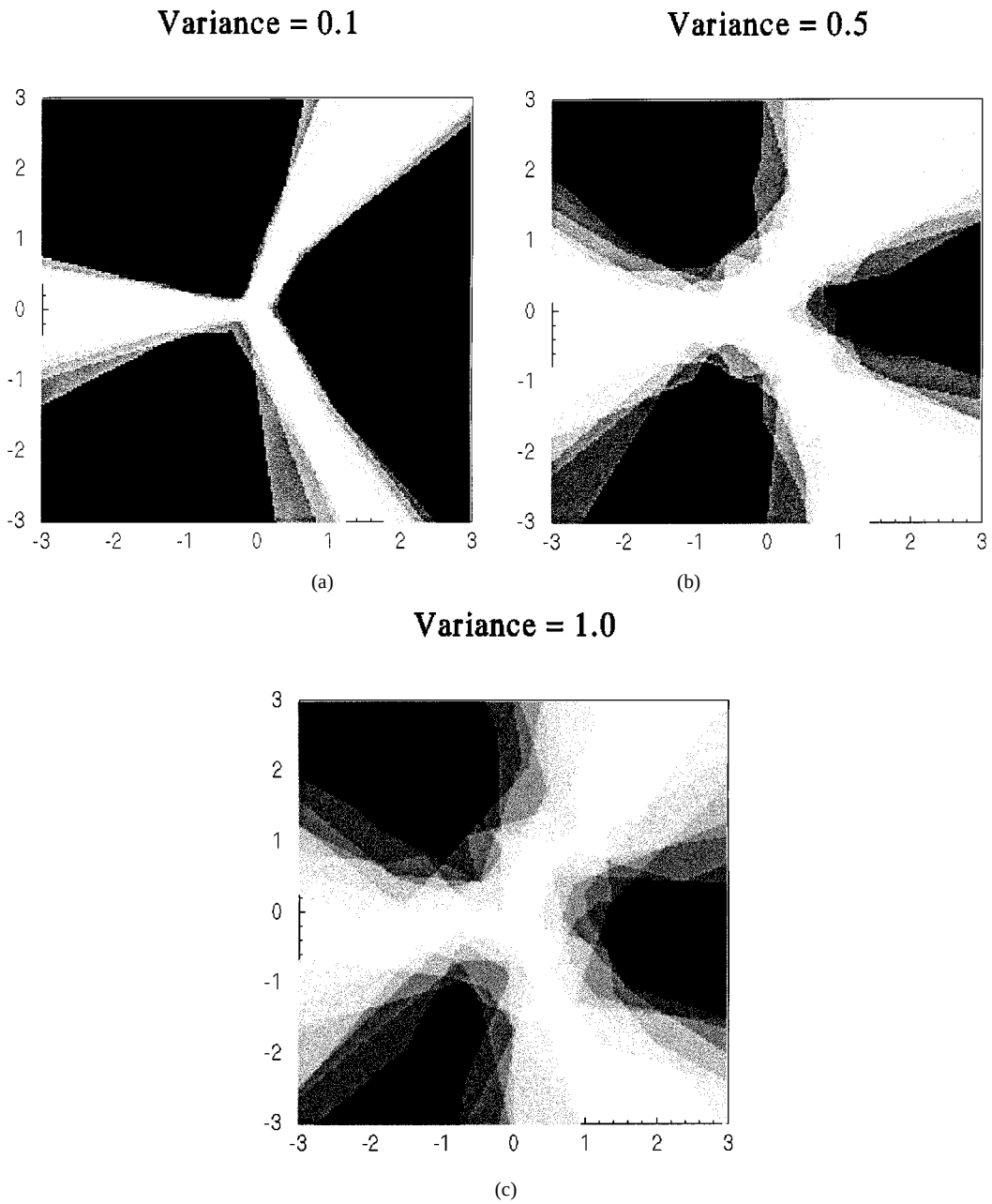

Fig. 4. Confidence maps for the three-Gaussian experiments; darker areas indicate higher confidence. (a) Simple; (b) Medium; (c) Complex.

Fig. 5(a)–(c) displays the measured and expected generalization errors in the three experiments. The comparison of theoretical expectations with empirical evidence supports the results reported in Section III. The bound (7) is a reasonable estimator of the overall error rate; the curves show that, thanks to the limitation imposed by the unsupervised training phase on the GF, the achievements of computational learning theory can have an effective practical impact.

B. Real-Domain Test: The NIST Handwritten Numerals Database

The NIST handwritten digits database represents a significant testbed for the experimental verification of the KWM model in a real-world domain of high practical interest [26]. This multiclass problem involves three distinct data sets, which, in the following, will be denoted by "LS" (learning set), "VS" (validation set), and "TS" (test set), respectively. LS and VS consist of 60 000 patterns each, and TS is composed of 58 646 patterns

After normalization and slant correction [27], the database included B/W bitmaps holding 40×32 bits, further compressed into lower-dimensional patterns by averaging squares of 4×4 pixels into single pixels with a four-bit resolution.

The set of prototypes for vector-quantizing the data space was generated by the plastic neural gas algorithm [19], which evaluated the appropriate set cardinality. The unsupervised fitting procedure exploits a basic property of VQ training models: the distortion error (2) saturates up to an asymptotical value as the number of neurons increases. Fig. 6 shows experimental distortion errors (LS, VS, TS) versus prototype-set cardinality. In the

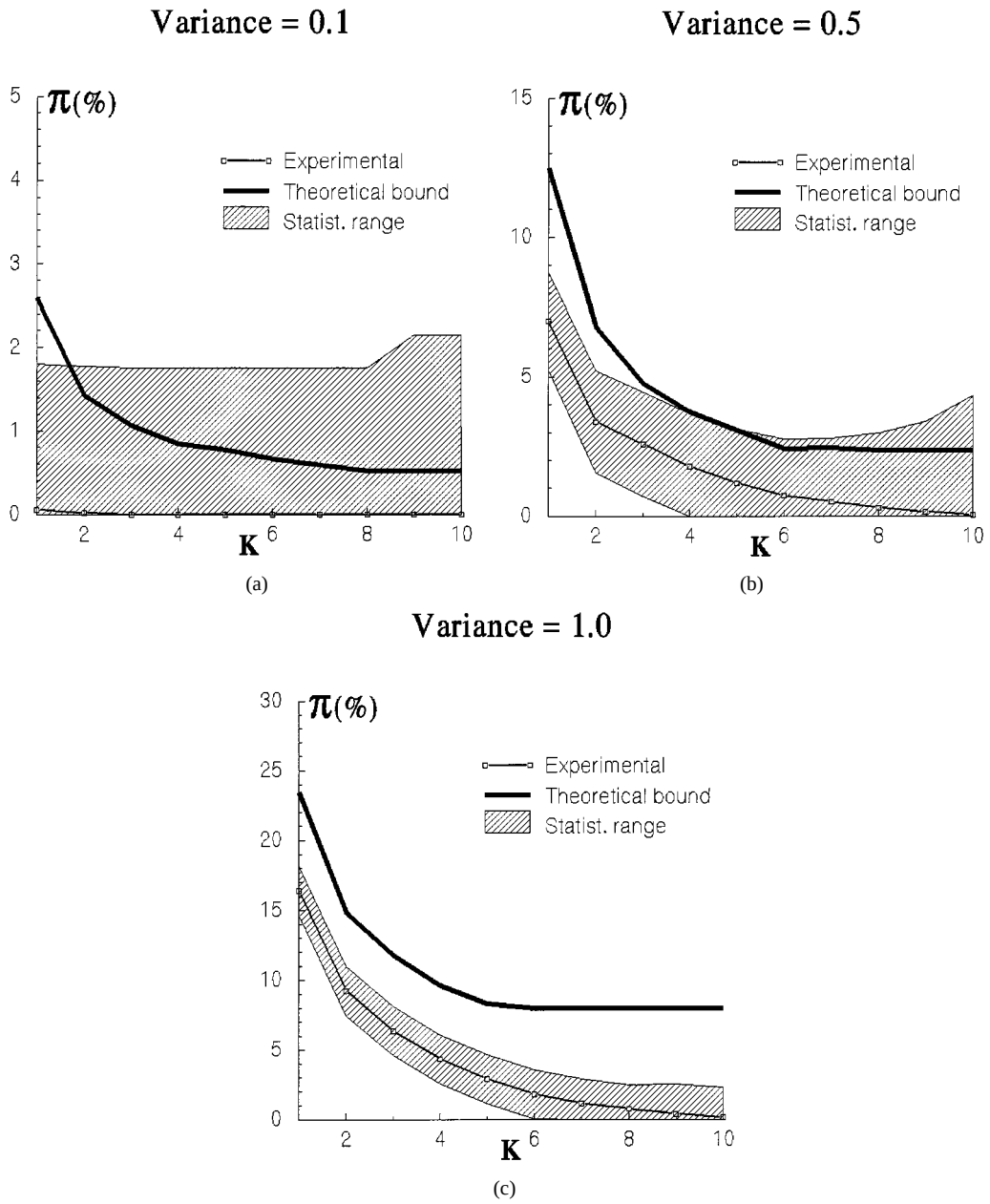

Fig. 5. Generalization performances of the KWM in the three-Gaussian experiments (a) simple, (b) medium, and (c) complex.

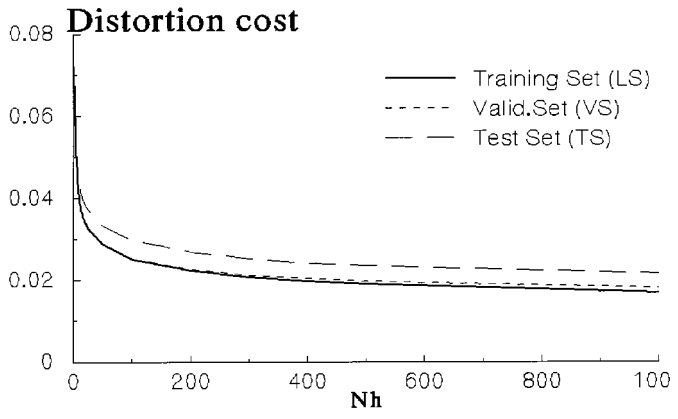

Fig. 6. VQ costs (training, validation) versus numbers of prototypes. Quantization distortion saturates for large values of N_h.

prototype training, the progress of the empirical distortion cost on VS was taken into account, disregarding pattern classes: the number of prototypes $N_h = 300$ seemed to allow a suitable tradeoff between representation accuracy and prototype-set complexity, so it was always used as a default in all the experiments.

The subsequent calibration phase assigned categories to the prototypes; the resulting class distribution is given in Table II. In fact, calibration results provided a crucial, preliminary basis for the practical applicability of the KWM method. The graph in Fig. 7 is the plot of sorted prototype reliability, α_n, $n = 1, \ldots, N_h$, defined as the share of the predominant class for the nth prototype: $\alpha_n = \max_c\{\alpha_n^{(c)}\}$. The curve shows that the class agreement among the patterns belonging to the same partition is very high: almost all the prototypes ($292/300 = 97.3\%$) exhibit a reliability higher than 0.80, thus suggesting that there is a good matching between the spatial positions of prototypes and the associate class regions. The varying numbers of prototypes for the different classes is the consequence of the different complexities of the digits' graphic patterns. Simple patterns ul-

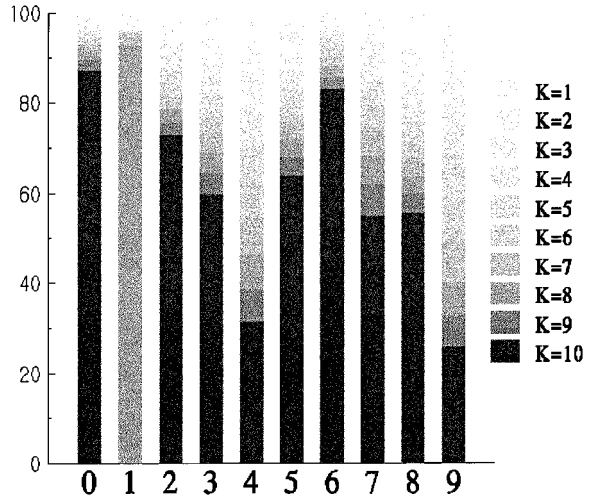

Fig. 7. NIST testbed: calibration results.

TABLE II
CALIBRATED CLASS DISTRIBUTION

Class	# Prototypes
0	24
1	12
2	42
3	29
4	35
5	35
6	29
7	25
8	41
9	28

timately give rise to more aggregate clusters of patterns, and therefore require a smaller number of prototypes. For instance, class "1" clearly appears more aggregate than class "2" or "8."

By analogy to the comparative approach followed in the three-Gaussian experiment, the NIST database was processed by a 1-NN classifier, using the 60 000 training patterns to categorize the other data sets. The results are given in Table III; the generalization performance attained by the 1-NN classifier confirms a peculiarity of NIST databases that has been reported previously [28]: LS and VS are drawn from a similar distribution, whereas such a property does not seem to hold for TS. This motivates the considerable generalization error of any classifier using TS as a testbed, after having been trained with LS.

The data dimensionality of the NIST testbed prevents one from drawing a confidence map for visual interpretation, like the map used for the 2-D problem. Nevertheless, the feasibility of labeling space locations by confidence values proved useful in clarifying interesting aspects. The basic idea is to check space sites with nonnull probability, hence training patterns were used to spot "landmarks" in the confidence map. For each training pattern, the level of confidence resulting from the KWM algorithm was evaluated. In principle, it is not ensured that larger K values lead to tighter bounds to the generalization error, due to a possible subsampling in (7). In fact, the experiments on the NIST testbed always indicated that there is a direct connection between the agreement level, K, and the confidence in the classification outcome.

TABLE III
NIST DATABASES: ACCURACIES OF DIFFERENT CLASSIFIERS

	1-NN	1-WC
Training Set (LS)	-	1.92%
Valid. Set (VS)	0.76%	1.94%
Test Set (TS)	4.18%	7.34%

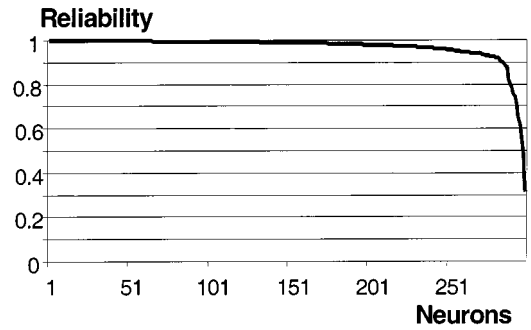

Fig. 8. Confidence distribution over classes.

Grouping patterns of the same class but with different K values made it possible to count the different confidence levels within each class, and therefore to deduce the K distribution within the class itself. The results of these measurements are given in Table IV. Fig. 8 allows a visual assessment of the confidence distribution: more than 85% of the patterns belonging to class "0" were correctly classified with $K = 10$. No pattern of class "1" could be classified with $K = 10$, due to the few (12) prototypes calibrated with that class. The graph helps understand the mapping supported by the calibrated prototype set. Classes "0" and "6" cover the most "certain" areas of the data space, though one might say that the most reliable classification concerns class "1" as the percentage of patterns correctly processed with $k = 8$ exceeds 92%. Classes "4" and "9" exhibit critical situations; one might ascribe them to strong overlaps with neighboring classes. Measuring uncertainty in this way might help the designer to build a classifier that accounts for such a distribution, for example, by treating critical classes selectively.

The validation of the space-varying confidence mechanism can also be attained by visualizing patterns lying in different areas of the data space. Fig. 9 presents, for each confidence level in the range [1, 10], patterns chosen at random from those counted in Table IV. A comparative inspection of the example images shows that the qualities of the pattern appearances increase as K increases. The confidence in classification increases with the visual quality of a pattern. Such a nontrivial result supports the "natural" representation paradigm that underlies the KWM model.

The final experimental phase for the NIST database addressed the verification of the theoretical predictions concerning the expected generalization error. Both VS and TS were used to measure the classification error, although TS should be regarded as a more reliable generalization testbed for two reasons: 1) VS was used to cross-validate prototype positioning during unsupervised VQ training; nevertheless, it is worth recalling that

TABLE IV
DISTRIBUTION OF CONFIDENCE LEVELS WITHIN CLASSES (NO. OF PATTERNS)

	"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"
$K = 10$	5249	0	4378	3597	1902	3839	5004	3310	3354	1560
$K = 9$	146	2	160	287	426	253	144	416	249	408
$K = 8$	116	5580	187	266	466	226	147	369	231	448
$K = 7$	96	164	178	241	498	240	133	390	239	550
$K = 6$	84	19	182	228	489	235	136	298	227	580
$K = 5$	73	14	177	277	460	223	104	305	257	570
$K = 4$	55	9	180	248	481	214	91	294	333	516
$K = 3$	42	10	170	260	470	188	77	224	335	479
$K = 2$	60	61	160	274	424	336	65	190	338	410
$K = 1$	79	141	228	322	384	246	99	204	437	479

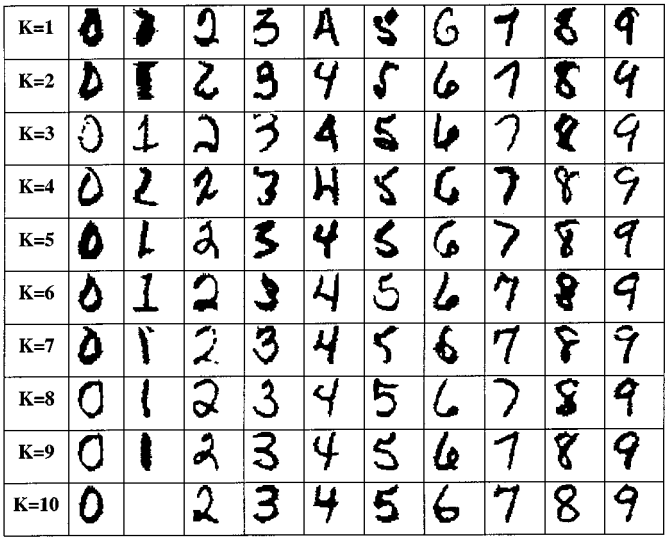

Fig. 9. Sample patterns for increasing levels of confidence.

pattern classes were disregarded while evaluating the distortion cost; 2) more importantly, VS was drawn from a distribution very close to that of LS, hence results appear strongly correlated. Conversely, as anticipated by the results of the 1-NN classifier, TS was not a twin of the training set [28]. This fact had also been confirmed (Fig. 6) by comparing the unsupervised distortion cost related to TS with those relevant to {VS, LS}.

The evaluation procedure was the same as used for the 2-D three-Gaussians problem: for each test run, the theoretical bound, $\pi(K)$, to the generalization error (7) was compared with the empirical error rate, corrected for statistical fluctuations according to (9). Fig. 10(a) and (b) present the obtained results on VS and TS, respectively. The curves witness the different distributions of the processed patterns, as VS seems a definitely easier problem than TS; in this respect, TS is preferable as it is a more reliable test set. The comparison between empirical costs and the theoretical bound shows that, for the NIST testbed, too, computational learning theory turned out to yield an expected result not too far from empirical evidence.

V. DISCUSSION AND CONCLUSION

The basic idea to use a prototype-based representation paradigm to span a data space for classification purposes is not entirely new, especially when WTA mechanisms are applied to drive space-dependent decision-making processes. In this context, a novel aspect of the presented research lies in using several space locations to derive a measure of confidence in the classification outcome. The KWM training procedure is also quite standard (VQ prototype positioning plus calibration), hence the related model can fully benefit from the vast literature on such subjects.

The crucial issue associated with the KWM model lies in defining and characterizing a family of classifiers, which always agree in the classification outcome but differ in their growth functions and consequent expected generalization errors. This makes it possible to assign a confidence level to each point in the data space; such a level is obtained by selecting the most appropriate classifier providing the tightest bound to the generalization performance. From this viewpoint, both theoretical and practical demonstrations have shown that there is a sharp relationship between class overlap (problem complexity) and the shape of the related confidence map (expected generalization error). Such features endow KWMs with the classification accuracy of classical surface-based representation models, and also provide the local-level inspection ability typical of prototype-based paradigms.

An important advantage of the method is the model's independence of both data dimensionality and the possible multiclass nature of the classification problem; as a result, the KWM approach applies effectively to masses of data. This arouses great interest in this method for critical practical applications. By contrast, the model might not work properly in undersampled domains, where an accurate estimation of the empirical classification error is unfeasible and the generalization performance is often difficult to evaluate experimentally.

The simplicity of the model also favors efficient implementations of the overall model in dedicated hardware circuitry. In particular, both the pattern-prototype distance computation and the prototype-sorting process based on the extraction of the K

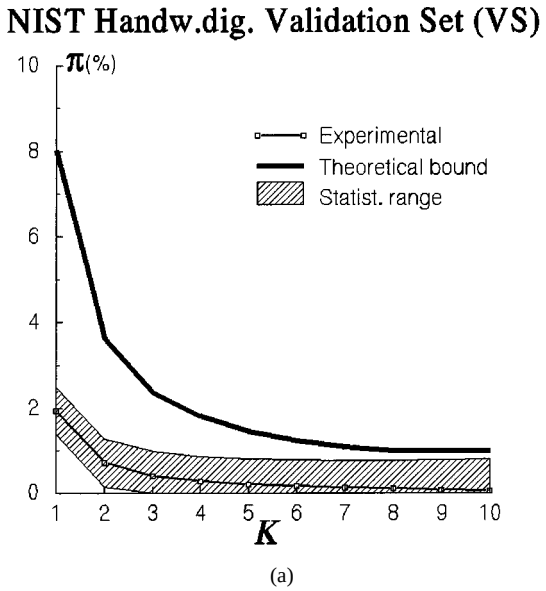

Fig. 10. NIST database: bounded and experimental generalization performances. (a) Results on the validation set. (b) Results on the test set.

best-matching candidates have already been adopted in VLSI architectures [20], [29]. The reduction in the basic VQ computational cost in both training and run-time operation represents a crucial effort devoted to a technically valid realization of the overall approach. At the same time, the theoretical analysis is currently oriented toward a more extensive study of the properties of KWMs within the framework of structural risk minimization.

APPENDIX A
OVERVIEW OF THE UNSUPERVISED PROTOTYPE-POSITIONING ALGORITHMS

The "Neural Gas" (NGAS) model was defined in [15] as an effective unsupervised algorithm to position prototypes so as to optimize the uniform coverage of the data space. The algorithm peration can be summarized as follows.

The "*Neural Gas*" Algorithm

0. Input: prototype set, W; total number of iterations, I; final and initial learning rates, $\eta_I < \eta_0 \leq 1$; final and initial decay parameters, $\lambda_I < \lambda_0$;
1. Initialize prototype positions (possibly random)
2. For $i = 1$ to I
   2.a Get a sample pattern $\mathbf{x}_i$;
   2.b Compute vector distances from $\mathbf{x}_i$: $d_n^{(i)} = \|\mathbf{w}_n - \mathbf{x}^{(i)}\|$, $n = 1, \ldots, N_h$;
   2.c Sort vector list $k_n(d_n^{(i)}) \in \{0, \ldots, N_h - 1\}$ such that: $d_j^{(i)} \leq d_{j+1}^{(i)} \Rightarrow k_j \leq k_{j+1}$;
   2.d Compute:

$$\eta_i = \eta_0 \cdot \left(\frac{\eta_I}{\eta_0}\right)^{i/I}; \qquad \lambda_i = \lambda_0 \cdot \left(\frac{\lambda_I}{\lambda_0}\right)^{i/I};$$

$$h^{(i)}(k_n) = \exp\left(\frac{-k_n}{\lambda_i}\right), \qquad n = 1, \ldots, N_h;$$

   2.e Adjust vector positions: $\Delta\mathbf{w}_n^{(i)} = \eta_i \cdot h^{(i)}(k_n) \cdot [\mathbf{w}_n^{(i-1)} - \mathbf{x}^{(i)}]$, $n = 1, \ldots, N_h$;;
3. Output: prototype set, W.

The "plastic neural gas" algorithm was proposed in [20] as the adaptive version of the NGAS method, to which it added the ability of dynamically creating and deleting prototypes. The PGAS method is guaranteed to converge in a finite number of steps and, as compared with the NGAS approach, does not suffer from the problem of "dead vectors." The principle of operation of the PGAS method lies in adding prototypes in those regions of the data space that are represented with insufficient accuracy by the current prototype set; at the same time, the prototypes whose local accuracy appears satisfactory, are deactivated and do not enter in the VQ optimization process. The following definition are given:

P_n: partition of patterns covered by $\mathbf{w}_n$: $\mathrm{P}_n = \{\mathbf{x} \in \mathrm{X}: \|\mathbf{x} - \mathbf{w}_n\| < \|\mathbf{x} - \mathbf{w}_j\| \ \forall\ j \neq n, j = 1, \ldots, N_h\}$;

$E_N(\mathbf{w}_n)$: local distortion associated with prototype $\mathbf{w}_n$,:

$$E_N(\mathbf{w}_n) = \frac{1}{D} \sum_{l=1}^{|\mathrm{P}_n|} \|\mathbf{x}_l - \mathbf{w}_n\|^2; \qquad \mathbf{x}_l \in \mathrm{P}_n;$$

the index N indicates that the current set, W, contains N prototypes.

$f(\alpha, \mathbf{w}_n)$: "prototype activation" function, defined as: $f(\alpha, \mathbf{w}_n) = 1 \Leftrightarrow E_N(\mathbf{w}_n) \leq \alpha$; $f(\alpha, \mathbf{w}_n) = 0$ otherwise;

$\mathbf{b}(\mathrm{P}_n)$: centroid of the data partition covered by $\mathbf{w}_n$:

$$\mathbf{b}(\mathrm{P}_n) = \frac{1}{|\mathrm{P}_n|} \sum_{k=1}^{\mathrm{P}_n} \mathbf{x}_k; \qquad \mathbf{x}_k \in \mathrm{P}_n;$$

$E(\mathrm{W})$: total distortion associated with the prototype set:

$$E(\mathrm{W}) = \frac{1}{N_p} \sum_{n=1}^{N} E_N(\mathbf{w}_n).$$

See the algorithm at the bottom of the page.

APPENDIX B
PROOFS OF THEOREMS 1, 2, AND 3

Theorem 1: The VC dimension of a k-Winner Classifier using N_h prototypes and $k = 1$ is $d_{\mathrm{VC}} = N_h$.

Proof: Assume to set up a set of prototypes, W, with N_h prototypes. Consider a data set, X, holding $N = N_h$ patterns: there exists a positioning schema for the N_h prototypes such that each prototype covers one pattern, hence $\forall\, (\mathbf{x}_a, \mathbf{x}_b) \in \mathrm{X}^2$: $\mathrm{W}_1(\mathbf{x}_a) \cap W_1(\mathbf{x}_b) = \emptyset$. Therefore, the number of possible target configurations that can be supported by the 1-WC classifier is 2^N; it follows that: $d_{\mathrm{VC}} \geq N_h$.

On the other hand, consider a larger data set, $\mathrm{X}' = \mathrm{X} \cup \{\mathbf{x}'\}$, holding $N_h + 1$ patterns. The NN prototype that classifies $\mathbf{x}'$ must be drawn from the available set, W. Thus, there exists at least one pair of patterns $(\mathbf{x}_a, \mathbf{x}')$ in X′ such that: $\mathrm{W}_1(\mathbf{x}_a) \cap \mathrm{W}_1(\mathbf{x}') \neq \emptyset$. But this implies that there is a target configuration (in particular, when $c_a \neq c'$) that cannot be classified correctly according to the NN criterion, hence the data set X′ cannot be shattered by the 1-WC. Therefore, $N_h \leq d_{\mathrm{VC}} < N_h + 1 \Leftrightarrow d_{\mathrm{VC}} = N_h$. Q.E.D.

Theorem 2: The growth function of a k-winner classifier using N_h prototypes is $\mathrm{GF}(n) = 2^{d_{GF}^{(k)}}$, where $d_{\mathrm{GF}}^{(k)} = \lfloor N_h/k \rfloor$.

Proof: Using the previous notations, let X be a data set holding n patterns, and let W be a set of N_h prototypes. Build the graph $\mathrm{G} \equiv \mathrm{W} \times \mathrm{W}$ that pairwise connects all the prototypes in W: the prototypes represent the nodes in G, and the arcs are represented by pairs of nodes; G is fully connected. Using a similar derivation to that used for Theorem 1, further assume that the graph structure is fixed before the data sample was generated.

Extract from G the subgraph G′, defined as $\mathrm{G}' = \{(\mathbf{w}_a, \mathbf{w}_b) \in \mathrm{G} : \exists \mathbf{x} \in \mathrm{X} : \mathbf{w}_a, \mathbf{w}_b \in \mathrm{W}_k(\mathbf{x})\}$. G′ includes the pairs of prototypes (arcs) that contribute to classifying at least one data patterns. Incidentally, note that if $k = 1$, all the prototypes in W are isolated, and G′ is the degenerate subgraph that contains all the nodes and no arcs. Given any subgraph $g \subseteq \mathrm{G}'$, denote by X(g) the set of patterns classified by g; it is defined as: $\mathrm{X}(g) = \{\mathbf{x} \in \mathrm{X} : \exists(\mathbf{w}_1, \mathbf{w}_2) \in g : \mathbf{w}_1, \mathbf{w}_2 \in \mathrm{W}_k(\mathbf{x})\}$. By construction, $\mathrm{X}(g) \neq \emptyset\ \forall\, g \subseteq \mathrm{G}'$.

Consider now the set $\mathrm{G}'' = \{g_j \subseteq \mathrm{G}', j = 1, \ldots, M\}$ whose elements are characterized as follows: 1) $\forall\, g_j \in \mathrm{G}''$, g_j is connected and 2) $\forall\, g_u, g_v \in \mathrm{G}''\ g_u \cap g_v = \emptyset \Leftrightarrow u \neq v$. In other words, G″ contains all the connected subgraphs of G′ that are disjointed. By definition of the kWC and by construction of G′, any subgraph $g \in \mathrm{G}''$ contains at least $k-1$ arcs and k prototypes. More importantly, the fact that g is connected implies that

The "*Plastic Neural Gas*" Algorithm

0. Input: a data set X, maximum tolerated cost, ω; final and initial learning rates, $\eta_I < \eta_0 \leq 1$; final and initial decay parameters, $\lambda_I < \lambda_0$;
1. Draw a random pattern $\mathbf{x}_0$; Set: $\mathbf{w}_1 = \mathbf{x}_0$; $\mathrm{W} = \{\mathbf{w}_1\}$; $\mathrm{Z} = \{0\}$; $N = 1$;
2. For each ith input pattern; $i = 1, \ldots, N_p$
   2.a Compute vector distances: $d_n^{(i)} = \|\mathbf{w}_n - \mathbf{x}^{(i)}\|$; $n = 1, \ldots, N$;
   2.b Sort neuron list $k_n(d_n^{(i)}) \in \{0, \ldots, N-1\}$; B is the index of the "best" prototype: $B \in \{1, \ldots, N\}$: $k_B(d_B^{(i)}) = 0$;
   2.c If $f(\alpha, \mathbf{w}_B) = 1$ (If the best prototype is active, update all active prototpyes using NGAS)

$$\Delta\mathbf{w}_n^{(i)} = \eta_i \cdot h^{(i)}(k_n) \cdot f\left(\alpha, \mathbf{w}_n^{(j-1)}\right) \cdot \left[\mathbf{w}_n^{(i-1)} - \mathbf{x}^{(i)}\right];$$

3. For each neuron $\mathbf{w}_n$ such that $(P_n \neq \emptyset)$: Set $\mathbf{w}_n = \mathbf{b}(P_n)$;
4. For $n = 1, \ldots, N$
   If $f(\alpha, \mathbf{w}_n) = 1$ AND $(P_n = \emptyset)$
   Delete $\mathbf{w}_n$; Set $N = N - 1$;
5. If $(C_N(\mathbf{w}_n) < \varepsilon\ \forall\, n = 1, \ldots, N)$ OR $(C(\mathrm{W}) \in \mathbf{Z})$
   Exit and Return W;
   Else
   6.a Set $\mathbf{Z} = \mathbf{Z} \cup \{C(\mathrm{W})\}$;
   6.b Select the "worst" prototype: $\widehat{\mathbf{w}} \in \mathrm{W}$: $C_N(\widehat{\mathbf{w}}) \geq C_N(\mathbf{w})\ \forall\, \mathbf{w} \in W$;
   6.c Create a new prototype, $\mathbf{w}_{N+1} = \widehat{\mathbf{w}}$;
   6.d Set $N = N + 1$; $\mathrm{W} = \mathrm{W} \cup \{\mathbf{w}_{N+1}\}$;
   6.e Loop to step 2).

$\forall\, \mathbf{x}_1 \in \mathrm{X}(g)\, \exists \mathbf{x}_2 \in \mathrm{X}(g)$: $\mathrm{W}_i(\mathbf{x}_1) \cap \mathrm{W}_i(\mathbf{x}_2) \neq \varnothing$. Disproving this hypothesis implies that $\forall\ \mathbf{x}_1$: $\forall\ \mathbf{x}_2 \in \mathrm{X}(g) \mathrm{W}_k(\mathbf{x}_1) \cap \mathrm{W}_k(\mathbf{x}_2) = \varnothing$; this means that the subgraph $\mathrm{W}_k(\mathbf{x}_1)$ is disjointed from the rest of g, which is impossible because g is connected.

Two basic properties can now be proved.

1) Each subgraph $g \in \mathrm{G}''$ can shatter at most one pattern. Assume that $\mathrm{X}(g)$ includes two patterns, $\{\mathbf{x}_1, \mathbf{x}_2\}$. The subgraph $g \in \mathrm{G}''$ therefore includes only the subgraphs $\mathrm{W}_k(\mathbf{x}_1)$ and $\mathrm{W}_k(\mathbf{x}_2)$. For the shattering property, the classifier must support correctly all target configurations, including $(\mathbf{x}_1, c^{(1)})$, $(\mathbf{x}_2, c^{(2)})$, $c^{(1)} \neq c^{(2)}$. To perform a correct k-Winner classification, the two sets of prototypes, $(\mathrm{W}_k(\mathbf{x}_1), c^{(1)})$ and $(\mathrm{W}_k(\mathbf{x}_2), c^{(2)})$, must be calibrated with different classes, therefore they must be disjointed. This is impossible because g is connected, so $\{\mathbf{x}_1, \mathbf{x}_2\}$ cannot be shattered by g. This property easily applies to the general case of $\mathrm{X}(g)$ including more patterns, as the above analysis can be made for each couple $\{\mathbf{x}_1, \mathbf{x}_2\}$ such that $\mathrm{W}_k(\mathbf{x}_1) \cap \mathrm{W}_k(\mathbf{x}_2) \neq \varnothing$.

2) For any pair of subgraphs, $g_1, g_2 \in \mathrm{G}'' \Rightarrow \mathrm{X}(g_1) \cap \mathrm{X}(g_2) = \varnothing$ (the sets of patterns covered by disjointed graphs are disjointed, too). The fact that $g_1, g_2 \in \mathrm{G}''$ implies that $g_1 \cap g_2 = \varnothing$. Assume that there is a pattern $\mathbf{x}$ such that $\mathbf{x} \in \mathrm{X}(g_1)$ and $\mathbf{x} \in \mathrm{X}(g_2)$. By construction of G′, the subgraph $\mathrm{W}_k(\mathbf{x})$ should appear in both g_1 and g_2, hence the subgraphs would not be disjointed any more, which contradicts the hypothesis.

From Property 1), one concludes that only two target configurations, regardless of the number of patterns in $\mathrm{X}(g)$, can be processed correctly by a kWC based on a subgraph $g \in \mathrm{G}''$; therefore

$$\mathrm{GF}_g(n) = 2\ \forall\, n. \tag{10}$$

From Property 2), one deduces that the GF of the classifier based on the entire graph G′ is given by the product of the individual GF's associated with all the disjointed subgraphs

$$\mathrm{GF}_{\mathrm{G}'}(n) = \prod_{g \in \mathrm{G}''} \mathrm{GF}_{\mathrm{g}}(n) = 2^M. \tag{11}$$

As each disjointed connected subgraph contains at least k prototypes, the largest number of disjointed subgraphs is reached when

$$M = \left\lfloor \frac{N_h}{k} \right\rfloor. \tag{12}$$

Substituting $d_{\mathrm{GF}}^{(k)} = \lfloor (N_h/k) \rfloor$ into (11) proves the assertion. Q.E.D.

Theorem 3: For any k-Winner classifier, the worst-case number of misclassified patterns, $\Pi(k)$, on a test sample of $N(k)$ patterns, is monotonically nonincreasing as k increases.

Proof: First, consider that, for any pattern $\mathbf{x}$, $\mathrm{W}_k(\mathbf{x}) \subset \mathrm{W}_{k+1}(\mathbf{x})$. A pattern that is not discarded by the $(k+1)$-winner classifier is not discarded by the k-winner classifier either. As a consequence, the patterns mistaken by the $(k+1)$-winner classifier are at most as many as the errors made by the k-winner classifier. These properties can be expressed as

$$\forall\, k \geq 1 N(k+1) \leq N(k) \tag{13}$$

$$\forall\, k \geq 1 \Omega(k+1) \leq \Omega(k). \tag{14}$$

By Theorem 2 (12) one has

$$\forall\, k \geq 1\, d_{\mathrm{GF}}^{(k+1)} \leq d_{\mathrm{GF}}^{(k)}. \tag{15}$$

From (6) and (12), it is easy to verify that $\varepsilon(k)N(k)$ is a nonincreasing function of k

$$\forall\, k \geq 1 \varepsilon(k+1)N(k+1)\varepsilon \leq (k)N(k). \tag{16}$$

If one now uses (7), the worst-case number of errors on a test sample of $N(k)$ patterns is

$$\Pi(k) = \Omega(k) + N(k)\frac{\varepsilon(k)}{2}\left(1 + \sqrt{1 + 4\frac{\Omega(k)}{N(k)\varepsilon(k)}}\right) \Leftrightarrow$$

$$\Pi(k) = \Omega(k) + \frac{1}{2}N(k)\varepsilon(k) + \sqrt{\frac{N^2(k)\varepsilon^2(k)}{4} + N(k)\varepsilon(k)\Omega(k)}. \tag{17}$$

Considering inequalities (14) and (16) and the monotonicity of all the terms in (17), one has

$$\forall\, k \geq 1 \Pi(k+1) \leq \Pi(k).$$

Q.E.D

ACKNOWLEDGMENT

The authors wish to thank Dr. A. M. Colla and the staff of Elsag SpA for their valuable assistance in performing the experiments described in the paper.

REFERENCES

[1] R. Hecht Nielsen, *Neurocomputing*. Reading, MA: Addison-Wesley, 1990.
[2] Y. Liu, "Unbiased estimate of generalization error and model selection in neural network," *Neural Networks*, vol. 8, pp. 215–219, 1995.
[3] L. Prechelt, "Automatic early stopping using cross validation: Quantifying the criteria," *Neural Networks*, vol. 11, no. 4, pp. 761–767, 1998.
[4] H. Akaike, "A new look at the statistical model identification," *IEEE Trans. Automat. Contr.*, vol. AC-19, pp. 716–723, 1974.
[5] J. Rissanen, *Stochastic Complexity in Statistical Inquiry*, Singapore: World Scientific , 1989.
[6] ——, "Stochastic complexity," *J. R. Statist. Soc. B*, vol. 49, no. 3, pp. 223–239, 252–265, 1987.
[7] S. Amari, N. Murata, K. R. Mueller, M. Finke, and H. H. Yang, "Asymptotic statistical theory of overtraining and cross-validation," *IEEE Trans. Neural Networks*, vol. 8, pp. 985–996, 1997.
[8] V. N. Vapnik, *Estimation of Dependences Based on Empirical Data*. New York: Springer-Verlag, 1982.
[9] ——, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.
[10] ——, *Statistical Learning Theory*. New York: Wiley, 1998.
[11] C. J. C. Burgess, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 1–43, 1998.

[12] T. Kohonen, *Self-Organization and Associative Memory*, 3rd ed. New York: Springer-Verlag, 1989.
[13] T. Martinetz and K. Schulten, "Topology representing networks," *Neural Networks*, vol. 7, no. 3, pp. 507–522, 1994.
[14] D. DeSieno, "Adding a conscience to competitive learning," in *Proc. IEEE Int. Conf. Neural Networks*, vol. 1, San Diego, CA, 1988, pp. 117–124.
[15] T. M. Martinetz, S. G. Berkovich, and K. J. Schulten, ""Neural gas" network for vector quantization and its application to time-series prediction," *IEEE Trans. Neural Networks*, vol. 4, pp. 558–569, 1993.
[16] M. M. Van Hulle, "Kernel-based equiprobabilistic topographic map formation," *Neural Comput.*, vol. 10, pp. 1847–1871, 1998.
[17] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inform. Theory*, vol. 28, pp. 127–135, 1982.
[18] N. Ueda and R. Nakano, "A new competitive learning approach based on an equidistortion principle for designing optimal vector quantizers," *Neural Networks*, vol. 7, no. 8, pp. 1211–1228, 1994.
[19] S. Ridella, S. Rovetta, and R. Zunino, "Plastic algorithm for adaptive vector quantization," *Neural Comput. Applicat.*, vol. 7, no. 1, pp. 37–51, 1998.
[20] S. Rovetta and R. Zunino, "Efficient training of neural gas vector quantizers with analog circuit implementation," *IEEE Trans. Circuits Syst. II*, vol. 46, pp. 688–698, 1999.
[21] J. Weston and C. Watkins, "Multi-class support vector machines," Royal Holloway Univ., Dept. Comp. Sci., London, U.K., CSD-TR-98-04, 1998.
[22] M. Anthony and N. Biggs, *Computational Learning Theory*. Cambridge, U.K.: Cambridge Univ. Press, 1992.
[23] J. Shawe-Taylor and M. Anthony, "Sample sizes for multiple-output threshold networks," *Network Comput. Neural Syst.*, vol. 2, no. 1, pp. 107–117, Feb. 1991.
[24] D. Haussler, "Generalizing the PAC model: Sample size bounds from metric dimension-based uniform convergence results," in *Proc. 30th IEEE Symp. Found. Comput. Sci.*, 1989, pp. 40–45.
[25] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *J. Amer. Stat. Assoc.*, vol. 58, pp. 13–30, 1963.
[26] A. F. R. Rahman and M. C. Fairhurst, "An evaluation of multiexpert configurations for the recognition of handwritten numerals," *Pattern Recognition*, vol. 31, no. 9, pp. 1255–1273, 1998.
[27] A. M. Colla and P. Pedrazzi, "Binary digital image feature extracting process," U.S. Pat. no. 5 737 444, Apr. 7, 1998.
[28] S. Zunino, "Circular backpropagation networks embed Vector quantization," *IEEE Trans. Neural Networks*, vol. 10, pp. 972–975, July 1999.
[29] ——, "Circuit implementation of the K-Winner Machine," *Electron. Lett.*, vol. 35, no. 14, pp. 1172–1173, July 8, 1999.

Sandro Ridella (M'93) received the Laurea degree in electronic engineering from the University of Genova, Italy, in 1966.

He is a full Professor in the Department of Biophysical and Electronic Engineering, University of Genova, Italy, where he teaches Circuits and Algorithms for Signal Processing. In the last eight years, his scientific activity has been mainly focused in the field of neural networks.

Stefano Rovetta (M'98) received the Laurea degree in electronic engineering and the Ph.D. degree in models, methods, and tools for electronic and electromagnetic systems, both from the University of Genoa, Italy, in 1993 and 1997, respectively.

He held a position as a postdoctoral researcher with the Electronic Systems and Networking Group of the Department of Biophysical and Electronic Engineering, University of Genoa. He has been an Invited Professor of Operating Systems at the University of Siena. He is currently an Assistant Professor at the Department of Computer and Information Sciences of Genoa University. His research interests include electronic circuits and systems, and neural-network theory, implementation, and applications.

Rodolfo Zunino (S'90–M'90) received the Laurea degree in electronic engineering from Genoa University, Italy, in 1985.

From 1986 to 1995, he was a Research Consultant with the Department of Biophysical and Electronic Engineering of Genoa University. He is currently with the same department as an Associate Professor in Electronics and Industrial Electronics. His main scientific interests include electronic systems for neural networks, efficient models for data representation and learning, advanced techniques for multimedia data processing, and distributed-control methodologies.