Minimizing Disagreement for Self-Supervised Classification

Virginia R. de Sa
Department of Computer Science
University of Rochester
Rochester, NY 14627
desa@cs.rochester.edu

In natural environments, the sensations arriving at two or more sensory modalities are often correlated. We derive an algorithm for a piecewise linear classifier which uses the relationship between patterns presented simultaneously to two or more networks as a supervisory signal. The algorithm is based on the idea of minimizing the disagreement error — the proportion of patterns disagreed upon — between two or more networks receiving correlated patterns. We test the algorithm on a ten class vowel classification problem and find that it performs better than a hybrid unsupervised/supervised algorithm and, with two iterations, almost as well as a related supervised algorithm (Kohonen's LVQ2.1).

One of the most ubiquitous tasks for both animals and machine recognition systems is to classify novel patterns based on prior experience with other similar patterns. For machine applications, a supervised approach (in which prior patterns are presented together with their classification label or target output) is often used and these systems have produced many successful pattern recognition systems.

Humans and other animals learn to form complicated categories seemingly without this kind of supervision. However, while no explicit labeling signal is present, there is more information available than is modeled with conventional unsupervised approaches. Natural environments are structured such that sensations to different sensory modalities (and sub-modalities) are correlated. For example, "hearing mooing" and "seeing cows" tend to occur together.

We develop a model in which two "modalities" provide the labeling signal for each other. The algorithm is derived by minimizing the disagreement error —the proportion of patterns on which the modalities disagree— between the outputs of the two networks that receive correlated input patterns from a world consisting of a number of discrete classes. Each class has a particular probability distribution for the sensation received by each modality. Figure 1 represents a simple 2-Class world, with two 1-D modalities. Experience with the world does not consist of the usual input sensation and pattern label (as for regular supervised learning), but instead appropriate sensations to two or more modalities. If modality 1 experiences a sensation from its pattern A distribution, modality 2 experiences a sensation from its own pattern A distribution. That is, the world presents patterns from the appropriate corresponding distributions.

MINIMIZING DISAGREEMENT

Consider the task of iteratively learning to discover the optimal class boundary for distinguishing whether Class A or Class B occurred in the one-dimensional problem of two classes in Figure 1. Let the current boundary for modality i be represented with b_i (see Figure 1). The ideal goal is to minimize the probability of misclassified patterns for each modality. That is, to minimize

$$E = \int_{b_1}^{\infty} P(C_A)p(x_j|C_A)dx_j + \int_{-\infty}^{b_1} P(C_B)p(x_j|C_B)dx_j \quad (1)$$

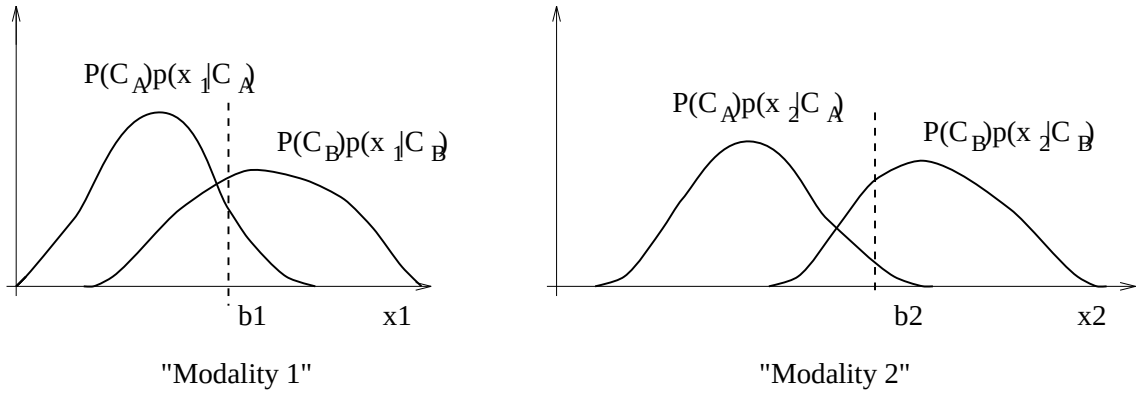

Figure 1: This figure shows an example world as sensed by two different modalities. If modality A receives a pattern from its Class A distribution, modality 2 receives a pattern from its own class A distribution (and the same for Class B). Without receiving information about which class the patterns came from, they must try to determine appropriate placement of the boundaries b_1 and b_2.

for both modalities ($j = 1, 2$) where $P(C_i)$ is the a priori probability of Class i and $p(x_j|C_i)$ is the conditional density of Class i for modality j.

The problem with this formulation is that the estimation of the conditional probabilities depends explicitly on class information. Instead consider minimizing the Disagreement Error between the networks [1]

$$E(b_1, b_2) = Pr\{x_1 < b_1 \ \& \ x_2 > b_1\} + Pr\{x_1 > b_1 \ \& \ x_2 < b_2\} \tag{2}$$

That is,

$$\begin{aligned} E(b_1, b_2) &= P(C_A)\left[\int_{-\infty}^{b_1} p(x_1|C_A)dx_1 \int_{b_2}^{\infty} p(x_2|C_A)dx_2 + \int_{b_1}^{\infty} p(x_1|C_A)dx_1 \int_{-\infty}^{b_2} p(x_2|C_A)dx_2\right] \\ &+ P(C_B)\left[\int_{-\infty}^{b_1} p(x_1|C_B)dx_1 \int_{b_2}^{\infty} p(x_2|C_B)dx_2 + \int_{b_1}^{\infty} p(x_1|C_B)dx_1 \int_{-\infty}^{b_2} p(x_2|C_B)dx_2\right] \end{aligned} \tag{3}$$

This heuristic measure is motivated by the fact that in the case where both modalities have distributions in which the Bayes optimal border is such that the fraction of misclassification is equal for both classes (as for example in identical symmetric distributions), if (3) has a single non-trivial local minimum it is the same as the minimum of (1). But the key point is that (3) does not depend on the class information as can be seen by letting

$$f(x_1, x_2) = P(C_A)p(x_1|C_A)p(x_2|C_A) + P(C_B)p(x_1|C_B)p(x_2|C_B) \tag{4}$$

and rewriting (3) as

$$E(b_1, b_2) = \int_{-\infty}^{b_1}\int_{b_2}^{\infty} f(x_1, x_2)dx_1dx_2 + \int_{b_1}^{\infty}\int_{-\infty}^{b_2} f(x_1, x_2)dx_1dx_2 \tag{5}$$

[1] Actually we are interested in finding a non-trivial local minimum of the Disagreement Error. Such a local minimum exists when there exists a pair (b_1^*, b_2^*) such that $\int_{-\infty}^{b_2^*} f(b_1^*, x_2)dx_2 = \int_{b_2^*}^{\infty} f(b_1^*, x_2)dx_2$ and $\int_{-\infty}^{b_1^*} f(x_1, b_2^*)dx_1 = \int_{b_1^*}^{\infty} f(x_1, b_2^*)dx_1$ and $\partial^2 E(b_1^*, b_2^*)/\partial b_1^2 \times \partial^2 E(b_1^*, b_2^*)/\partial b_2^2 - (\partial^2 E(b_1^*, b_2^*)/\partial b_1 \partial b_2)^2 > 0$ and $\partial^2 E(b_1^*, b_2^*)/\partial b_1^2 > 0$. A local minimum is not guaranteed to exist even for distributions for which the optimal boundary could be found with a supervised algorithm.

It is not necessary to know the class labels as $f(x_1, x_2)$ is the joint density for the inputs of the two modalities.

The derivative of (3) with respect to b_1 is

$$\partial E/\partial b_1 = \int_{b_2}^{\infty} f(b_1, x_2)dx_2 - \int_{-\infty}^{b_2} f(b_1, x_2)dx_2 \tag{6}$$

and similarly with respect to b_2

$$\partial E/\partial b_2 = \int_{b_1}^{\infty} f(x_1, b_2)dx_1 - \int_{-\infty}^{b_1} f(x_1, b_2)dx_1 \tag{7}$$

Using an extension by Wassel and Sklansky [1972], to the stochastic approximation method [10] gives the following iterative calculation for b_1 (see the Appendix for the derivation) where the hypothesized output class from the other modality network replaces the correct label.

$$b_1(n+1) = b_1(n) + \alpha(n)Z_n(X_1(n), X_2(n), b_1(n), b_2(n), c(n)) \tag{8}$$

with

$$Z_n = \begin{cases} 1 & \text{for } X_2(n) < b_2(n), |X_1(n) - b_1(n)| < c(n) \\ -1 & \text{for } X_2(n) > b_2(n), |X_1(n) - b_1(n)| < c(n) \end{cases}$$

In analogy with Competitive Learning[5, 7, 12] and LVQ2.1 [6] consider moving the borders indirectly through the motion of codebook vectors. The borders arise from the Voronoi tessellations of the codebook vectors. In the 1-D case the borders b_j can be defined by $b_j = (w_{j,1} + w_{j,2})/2$, where $w_{j,k}$ is a codebook vector for modality j. Thus the codebook vectors move according to

$$\partial E/\partial w_{j,1} = \partial E/\partial w_{j,2} = .5\partial E/\partial b_j\,.$$

Rewriting (8) in terms of the codebook vector motion gives

$$w_{1,i}(n+1) = w_{1,i}(n) + \alpha_2(n)\frac{(X_1(n) - w_{1,i}(n-1))}{|X_1(n) - w_{1,i}(n)|}$$

$$w_{1,j}(n+1) = w_{1,j}(n) - \alpha_2(n)\frac{(X_1(n) - w_{1,j}(n-1))}{|X_1(n) - w_{1,j}(n)|}$$

if $X_1(n)$ lies in a window of width $2c(n)$ centred at $b_1(n)$; $w_{1,i}$ and $w_{1,j}$ are the two closest codebook vectors to $X_1(n)$; $w_{1,i}$ is a codebook vector belonging to the same class as the closest codebook vector to $X_2(n)$ in Modality 2; and $w_{1,j}$ belongs to a different class [2]. Otherwise no updates are made. Note that here we have removed the restriction that Class A lies to the left of Class B.

Expanding the problem to more dimensions, and more classes with more codebook vectors per class, complicates the analysis as a change in two codebook vectors to better adjust their border affects more than just the border between the two codebook vectors. However ignoring these effects, a first order approximation suggests the algorithm shown in Figure 2. Note that this algorithm is identical to the modified LVQ2.1 [6] algorithm in [3] except that the labeling signal is not the actual class but the class hypothesized by the other modality (and the unsupervised initialization).

The label initialization algorithm mentioned in the second step above, can best be described by considering the network shown in Figure 3. Each output class has associated with it a vector in the space of codebook vector activations. These vectors can be represented as weights from hidden neurons (whose weights are the

[2] How the unsupervised labeling initialization of the codebook vectors is accomplished, is discussed at the end of this section.

- Generate initial codebook vectors (currently randomly chosen data vectors)
- Initialize labels of codebook vectors using the network shown in Figure 3 with the algorithm described below
- Repeat for each presentation of input patterns $X_1(n)$ and $X_2(n)$ to their respective modalities
  - find the two nearest codebook vectors $w_{1,i_1^*}, w_{1,i_2^*}, w_{2,k_1^*}, w_{2,k_2^*}$ to the respective input patterns
  - Find the hypothesized ouput class (C_A, C_B) in each modality (as given by the label of the closest codebook vector)
  - For each modality update the weights according to the following rules (Only the rules for modality 1 are given)

    If neither or both w_{1,i_1^*}, w_{1,i_2^*} have the same label as w_{2,k_1^*} or $X_1(n)$ does not lie within c(n) of the border between them no updates are done, otherwise

$$w_{1,i^*}(n) = w^*_{1,i}(n-1) + \alpha(n)\frac{(X_1(n) - w_{1,i^*}(n-1))}{||X_1(n) - w_{1,i^*}(n-1)||}$$

$$w_{1,j^*}(n) = w^*_{1,j}(n-1) - \alpha(n)\frac{(X_1(n) - w_{1,j^*}(n-1))}{||X_1(n) - w^*_{1,j}(n-1)||}$$

    where w_{1,i^*} is the codebook vector with the same label, and w_{1,j^*} is the codebook vector with another label.
  - update the labeling weights as described below

Figure 2: Minimizing Disagreement (M-D) Algorithm

codebook vectors) to output neurons—one for each class as shown in the figure. The weights are trained with a Competitive learning [5, 7, 12] rule which tends to connect co-active neurons (representing closest codebook vectors in each modality) to the same output class. During this stage the codebook vectors themselves are not modified.

During the main part of the algorithm described in the third step of Figure 2, the connections to the output neurons are also adjusted in the same way by increasing the weight to the output class hypothesized by the other modality, from the neuron with the closest codebook vector. Experiments show that this makes the algorithm more robust because codebook vectors that are not able to find one particular boundary (due perhaps to no local minimum in the Disagreement Error) may be reassigned.

SIMULATIONS

The algorithm was tested using a version of the Peterson and Barney vowel formant data [3]. The dataset consists of the first and second formants for ten vowels in a /hVd/ context from 75 speakers (32 males, 28 females, 15 children) who repeated each vowel twice [4]. For comparison with other algorithms, the training set and test set were the same. In order to facilitate evaluation, the algorithm was tested by giving the same distributions to each modality. That is $p(x_1|C_j) = p(x_2|C_j)$ for all j. A pattern presentation consisted of randomly choosing one of the data patterns for the first modality and randomly choosing a pattern from the same class for the second modality.

The algorithm obtained an average performance of 74% with 30 codebook vectors and 40000 pattern

[3] obtained from Steven Nowlan

[4] 3 speakers were missing one vowel and the raw data was linearly transformed to have zero mean and fall within the range $[-3, 3]$ in both components

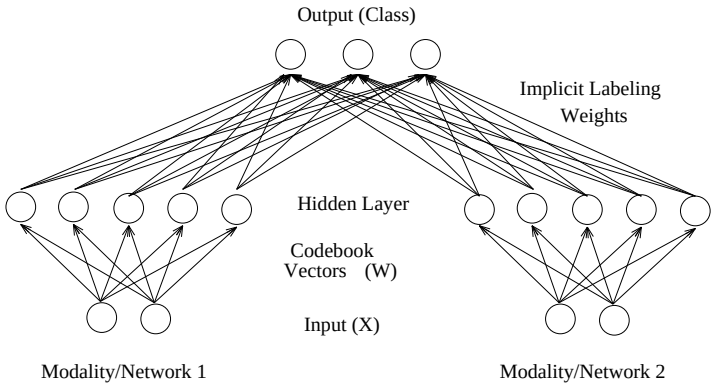

Figure 3: This figure shows a network for learning the labels of the codebook vectors. The weight vectors of the hidden layer neurons represent the codebook vectors while the weight vectors of the connections from the hidden layer neurons to the output neurons represent the output class that each codebook vector currently represents. In this example there are 3 output classes and two modalities each of which has 2-D input patterns and 5 codebook vectors.
.

presentations. Performance was averaged between modalities [5] and over 30 runs from different initial codebook vector positions. This was a big improvement over the initial accuracy after the label initialization step (where the positions of the codebook vectors are still random but the output weights have been trained) which had an average performance of 59%. The final performance figures were positively correlated with the performance after the label initialization step, which in turn was correlated with (in fact bounded by) the best performance possible with the initial codebook vectors (as measured independently with optimal labels). Improved methods of choosing the initial codebook vectors and algorithms for label initialization result in improved performance. For example, using the final codebook vectors from a run of the M-D algorithm as the initial codebook vectors for another run (replacing the first step in Figure 2) results in a final performance of 76%. This can be seen in Figure 4 which shows performance after the initial labeling, first application of the M-D algorithm, and second application of the M-D algorithm.

The algorithm's performance compares favourably with the 72% resulting from a hybrid unsupervised-supervised algorithm — Kohonen feature mapping algorithm (with 30 codebook vectors) followed by optimal labeling of the codebook vectors. (In fact if the same optimal labeling algorithm is applied to the codebook vectors resulting from the M-D algorithm, an average performance of 76% and 78% (for applying after one or two iterations respectively) results.) Performance is not as good as that of the related fully supervised algorithm, LVQ 2.1 which achieved an average performance of 80%, but is comparable to the performance of (supervised) back-propagation (with 25-200 hidden layer neurons) which obtained average performances of 73.4-78.5% [8]. Nowlan's mixture model (supervised) achieved an average performance of 86.1%.

DISCUSSION

In summary, we have shown that classification borders can be learnt without an explicit labeling or supervisory signal. For the particular vowel recognition problem, the performance of this "self-supervised" algorithm is almost as good as that achieved with supervised algorithms. This algorithm would be ideal for tasks in which signals for two or more modalities are available, but labels are either not available or expensive to obtain. One specific task is learning to classify speech sounds from images of the lips and the acoustic

[5] Each modality was tested separately for its ability to output the same label for all occurrences of the same vowel.

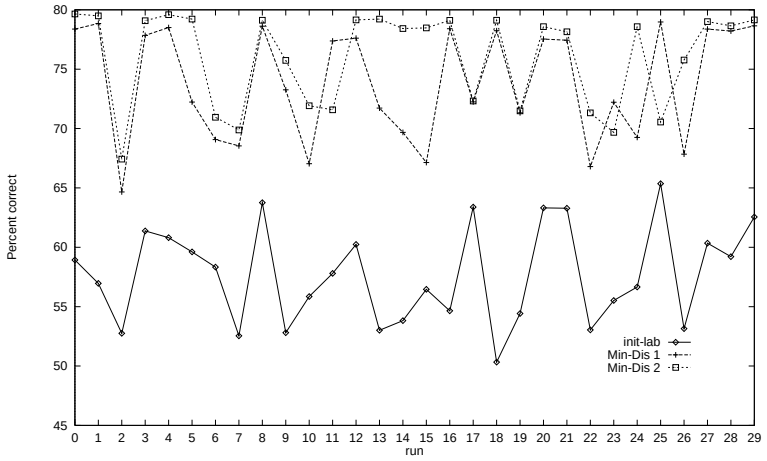

Figure 4: The performance (for 30 different initial configurations) after: initial labeling (init-lab), one application of the M-D algorithm (Min-Dis 1), and two applications of the M-D algorithm (Min-Dis 2)

signal. Stork et. al. [1992] performed this task with a supervised algorithm but one of the main limitations for data collection was the manual labeling of the patterns [David Stork, personal communication, 1993].

The algorithm could also be used for learning to classify signals to a single modality where the signal to the other "modality" is a temporally close sample. As the world changes slowly over time, signals close in time are likely from the same class. This approach should be more powerful than that of [4] as signals close in time need not be mapped to the same codebook vector but the closest codebook vector of the same class.

This work is similar in general motivation to that of [1, 2], but differs significantly in the derivation and resulting algorithm. Both algorithms are able to self-supervise from spatial/temporal/multi-modality relationships between their sub networks. This algorithm, however is restricted to classification tasks but requires less computation, memory storage and communication complexity. The different advantages and disadvantages with both algorithms are analogous to those between their related supervised algorithms, LVQ [6] and back-propagation [11].

Appendix: Derivation of the Minimizing Disagreement Algorithm

The algorithm we derive for minimizing (3) is patterned after [14] and uses their extension to the stochastic approximation method [10]. In stochastic approximation, iterative approximations are made by adding to the current estimate an appropriately scaled random variable (which is a function of the received input) whose expected value is the gradient, at the current estimate, of the function to be minimized. In this case the function to be minimized is a regression function. Wassel and Sklansky [1972] extended this method to work for functions that are the limit of a sequence of regression functions. Thus the solution to the minimization of (3) involves finding a sequence of functions whose limit is given by (6)(and similarly for (7)).

Consider the function

$$\hat{g}(b_1, b_2, c) = \int_{-\infty}^{\infty} \Psi(x_1 - b_1, c) \left[\int_{b_2}^{\infty} f(x_1, x_2) dx_2 - \int_{-\infty}^{b_2} f(x_1, x_2) dx_2 \right] \tag{9}$$

where $\Psi(x - z, c)$ is a Parzen window function [9] centred at z with width parameter c that satisfies the following conditions

$$\Psi(x - z, c) \geq 0 \quad \forall x, z \quad c > 0$$

$$\int_{-\infty}^{\infty} \Psi(x-z,c)dx = 1 \quad \forall z \quad c>0$$

$$c\int_{-\infty}^{\infty} \Psi^2(x-z,c)dx < \infty \quad \forall z \quad c>0$$

$$\lim_{c\to 0} \Psi(x-z,c) = \delta(x-z) \quad \forall x,z \quad c>0$$

then

$$\begin{aligned} \lim_{c\to 0} \hat{g}(b_1,b_2,c) &= \int_{b_2}^{\infty} f(b_1,x_2)dx_2 - \int_{-\infty}^{b_2} f(b_1,x_2)dx_2 \\ &= \partial E/\partial b_1 \end{aligned} \tag{10}$$

We now show that $\hat{g}$ is a regression function by showing that there is a random variable Z such that $\frac{1}{2c}E(Z|B_1=b_1) = -\hat{g}(b_1,b_2,c)$

Let

$$Z_i = 2cS(X_2(i))\Psi(X_1(i)-B_1,c)) \tag{11}$$

Then,

$$\frac{1}{2c}E(Z|B_1=b_1) = E(S(X_2)\Psi(X_1-b_1,c)) = \lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n} S(X_2(i))\Psi(X_1(i)-b_1,c)$$

where $X_j(i)$ is the sample data point at time i presented to modality j, and $S(X_2(i))$ is $+1$ if $X_2(i) < b_2$ and -1 if $X_2(i) > b_2$.

But,

$$\begin{aligned} E(S(X_2)\Psi(X_1-b_1,c)) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x_1,x_2)S(x_2)\Psi(x_1-b_1,c)dx_1dx_2 \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{b_2} f(x_1,x_2)\Psi(x_1-b_1,c)dx_1 - \int_{-\infty}^{\infty}\int_{b_2}^{\infty} f(x_1,x_2)\Psi(x_1-b_1,c)dx_1 \\ &= -\hat{g}(b_1,b_2,c) \end{aligned}$$

Thus (6) is the limit of a sequence of regression functions and the position of b_1 can be iteratively calculated according to[6]

$$b_1(n+1) = b_1(n) + \alpha(n)Z_n(X_1(n),X_2(n),b_1(n),b_2(n),c(n)) \tag{12}$$

where

$$Z_n = \begin{cases} 2c(n)\Psi(X_1(n)-b_1(n),c(n)) & \text{for } X_2(n) < b_2(n) \\ -2c(n)\Psi(X_1(n)-b_1(n),c(n)) & \text{for } X_2(n) > b_2(n) \end{cases}$$

Using rectangular Parzen window functions, the above simplifies to

$$Z_n = \begin{cases} 1 & \text{for } X_2(n) < b_2(n), |X_1(n)-b_1(n)| < c(n) \\ -1 & \text{for } X_2(n) > b_2(n), |X_1(n)-b_1(n)| < c(n) \end{cases}$$

[6]The theorem in [14] guarantees convergence under the condition that the function to be minimized has a single, minimum; as we are searching for a local minimum, convergence is only guaranteed if (b_1,b_2) stay within the area in which (b_1^*,b_2^*) is a local minimum and the only local extremum. . The following conditions are also required : $\alpha(n), c(n) > 0$; $\lim_{n\to\infty}\alpha(n) = 0$; $\lim_{n\to\infty} c(n) = 0$; $\Sigma_{n=1}^{\infty}\alpha(n)c(n) = \infty$; $\Sigma_{n=1}^{\infty}\alpha(n)^2c(n) < \infty$; and $p(x_i|C_j) < \infty$, $i,j = A,B$

Acknowledgements

I would like to thank Steve Nowlan for making the vowel formant data available to me. Many thanks also to Dana Ballard and Geoff Hinton for their helpful conversations and suggestions.

References

[1] S. Becker and G. E. Hinton, "A self-organizing neural network that discovers surfaces in random-dot stereograms," *Nature*, vol. 355, pp. 161–163, Jan. 1992.

[2] S. Becker, "Learning to categorize objects using temporal coherence," in *Advances in Neural Information Processing Systems 5* (C. Giles, S.J.Hanson, and J. Cowan, eds.), pp. 361—368, Morgan Kaufmann, 1993.

[3] V. R. de Sa and D. H. Ballard, "a note on learning vector quantization," in *Advances in Neural Information Processing Systems 5* (C. Giles, S.J.Hanson, and J. Cowan, eds.), pp. 220—227, Morgan Kaufmann, 1993.

[4] P. Földiák, "Learning invariance from transformation sequences," *Neural Computation*, vol. 3, no. 2, pp. 194–200, 1991.

[5] S. Grossberg, "Adaptive pattern classification and universal recoding: I. parallel development and coding of neural feature detectors," *Biological Cybernetics*, vol. 23, pp. 121–134, 1976.

[6] T. Kohonen, "Improved versions of learning vector quantization," in *IJCNN International Joint Conference on Neural Networks*, vol. 1, pp. I–545–I–550, 1990.

[7] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological Cybernetics*, vol. 43, pp. 59–69, 1982.

[8] S. J. Nowlan, *Soft Competitive Adaptation: Neural Network Learning Algorithms based on Fitting Statistical Mixtures*. PhD thesis, School of Computer Science, Carnegie Mellon University, 1991.

[9] E. Parzen, "On estimation of a probability density function and mode," *Annals of Math. Stat.*, vol. 33, pp. 1065—1076, 1962.

[10] H. Robbins and S. Monro, "A stochastic approximation method," *Annals of Math. Stat.*, vol. 22, pp. 400—407, 1951.

[11] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, eds.), vol. 1, pp. 318–364, MIT Press, 1986.

[12] D. E. Rumelhart and D. Zipser, "Feature discovery by competitive learning," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, eds.), vol. 2, pp. 151–193, MIT Press, 1986.

[13] D. G. Stork, G. Wolff, and E. Levine, "Neural network lipreading system for improved speech recognition," in *IJCNN International Joint Conference on Neural Networks*, vol. 2, pp. II–286—II–295, 1992.

[14] G. N. Wassel and J. Sklansky, "Training a one-dimensional classifier to minimize the probability of error," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-2, no. 4, pp. 533—541, 1972.