MHC - A Connectionist architecture for hybrid training

José M. Ramírez
Paradigma C.A.
Apartado 67079, Caracas 1061
Venezuela
Phone: +58-2-2836942, Fax: +58-2-2832689

Abstract

We present a Connectionist architecture called Modified Hyperspherical Classifier (MHC) based on: 1) the work of Cooper [5][6] and Batchelor [2][3][4] about Hyperspherical Classifiers, 2) the RCE paradigm as described by Scofield et al. [10] and 3) some new considerations derived from the search of an efficient model to perform heterogeneous pattern processing using supervised and/or unsupervised training algorithms depending on the nature of the problem, the availability of the correct output during the training process and the operational state of the network. We use the term Hybrid Training to define the use of a supervised or an unsupervised strategy to train the same network; this definition differs from the presented by Hertz et al. [7] as Hybrid Learning, which refers to layered networks with different learning strategies for each layer (e.g. Counterpropagation).

Conclusions about the performance of the architecture are presented, based on the execution of tests using tasks related with Familiarity detection, Principal component analysis, Clustering, Prototyping, Feature mapping and Pattern transformation.

Keywords: Connectionist Architectures, Classifiers, Neural Networks.

HC, RCE and MDC

Like the nearest-neighbor classifier, the HC is based upon the storage of patterns that represent points in a space. The association of an unknown pattern with a known category is made by the distance function (Cartesian, Hamming, etc.). The main difference with the nearest-neighbor model is that each point has a region of influence that is defined by a sphere with the pattern´s location point as the center. Each stored pattern with its region of influence defines a decision region associated with the category or class of the stored point.

RCE networks can be viewed as special cases of HC in that the stored patterns are stationary in the space and the regions of influence can only shrink and can not expand, this approach is also known as DSND (Disjoint Spheres/ No Drift) because the training process try to disjoint the spheres of different categories to avoid confusion in the classification but the stored point, the center of each sphere, remains stationary.

The MDC model uses the N-dimensional feature space as RCE and HC, but its functionality is based on the definition of at least one prototypical point for each category to be considered, this definition is made in the initialization stage. The classification is performed based on a distance metric between the prototypes and the pattern being processed. details of MDC can be found in [10],[12] and [13].

Modified Hyperspherical Classifier

The proposed architecture (MHC) has the following properties:

1. High storage density based on the N-dimensional feature space model.

2. The stored points (hidden layer units) act like discriminat functions.

3. The connections between the input and hidden layer are weighted. The weights represent the relative importance of each feature (input unit) to classification . This importance can be set "a priori" or by an analysis (e.g. covariance) of the input patterns.

4. The hidden units contain the location of the stored points associated and the size of the region of influence (radius of the sphere).

5. The connections between the hidden layer and the output layer are weighted. the weights are the result of a probabilistic density function applied to the region of influence of the unit. If an unit H1 shares part of the space with an unit H2 of another category, the weight of the connections of H1 and H2 with the corresponding output units will be <1 and will depend on the probability that a point in the shared region belongs to H1´s or H2´s category. The density function can be easily implemented as a count of the correct classifications performed over patterns that "fires" both neurons H1 and H2.

6. The spheres can shrink and expand during the training process to form the category areas in the feature space. Several spheres can be summarized into one or an sphere can be moved to a different location.

7. All the adjustments concerning the classification of the pattern being processed are made before processing the next pattern. The convergence of the net is always reached in just 2 epochs.

8. The initial radius of the spheres are not initially set to a fixed default value, but to the maximum value that can be assigned without include a point of a different category.

Figure 1. illustrates a MHC network

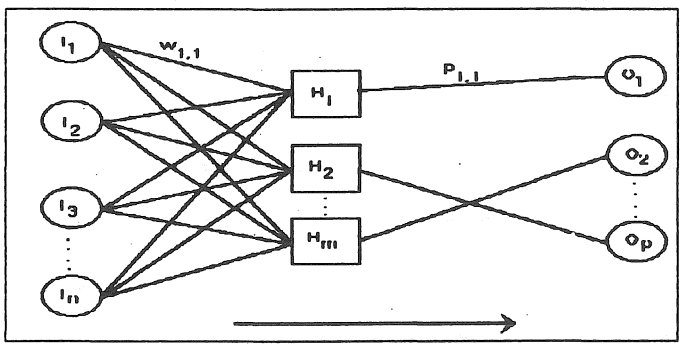

Figure 1.

Each input unit corresponds to a feature and each output unit corresponds to a pattern category. The hidden units are the stored points (spheres).

The key characteristics of MHC are: Feedforward, High storage density, Reduced connectivity, Partially distributed, Dynamic category learning and hybrid training.

Operation and Training of the MHC

When a pattern is presented to the network the activation of the hidden units will be given by:

$$H_j = G((SW_{ij}*(X_i-P_i)^2)^{1/2})- T)$$

$$G(x) = \quad 0 \text{ if } x > 0$$

1 if x <= 0

where:

H_j is the activation of the jth hidden layer

X_i is the ith feature of the input pattern

P_i is the ith component of the stored point

W_{ij} is the connection between the ith input unit and the jth hidden unit

T is the threshold or radius of the sphere (the region of influence)

The activation of the output units will be given by:

$$O_j = H_i * P_{ij}$$

where:

P_{ij} is the weight of the connection between the hidden unit and the associated output unit. Corresponding to the probabilistic density function.

The function G(x) associates each hidden unit with a region in the feature space. The location of the region is stored in the hidden unit and the size of the region is determined by the threshold also stored in the hidden unit. Any input pattern falling within the region of influence of a given unit will cause the unit to transfer output to the corresponding output unit. A pattern will cause the firing of all the units that share the region where the pattern

is located, if the hidden units project theirs output to more than one output unit (categories) the result will be an "ambiguous" classification; if only one output unit fires the answer of the network will be "known", the other possible output is "unknown", this results when no output unit fires.

The training process of MHC networks involves the execution of several actions: a) Creation of hidden units (spheres) and/or output units (categories). b) Adjustment of the spheres, traduced in shrinking, expansion, movement or jointing and c) Adjustment of the hidden to output connections using a probabilistic density function.

The conditions to perform the above actions are different in each training strategy. The control over their execution resides in a preprocessing module that determines if the pattern is prepared (or intended) for unsupervised or supervised training.

Hybrid training is necessary since there are problems where the solution is poorly known or just a subset of the possible categories are needed; in this case the network can be trained in a supervised way using the patterns which solution is known or just with patterns representing the desired categories. This process initializes the network and then a unsupervised training can take place until the entire feature space is covered. If the solution of a problem is totally unknown (e.g. exploratory data analysis) the network can be trained in an unsupervised way, using all the patterns available. The network will form clusters that can be analyzed for the experts in the domain and the categories can be defined to proceed with supervised training over the same network or a new one.

In the supervised strategy, if the output of the network is "unknown", a new hidden unit is created using the location of the current pattern and this unit is associated to the output unit of the correct category, if the category is not in the network (there is no output unit for this category) a new output unit is created, this process is known as dynamic category learning. the weight of the connection between the hidden unit and the output unit will be 1 and the size of the region of influence (threshold) will be the maximum value that does not include a point of another category. In an "ambiguous" situation, the spheres that are not associated to the correct category are shrunk so that they no longer cover the current pattern. If that size of the spheres reached the minimum value and remains the overlapping with spheres of another category, the probabilistic density function is used to adjust the connections between the overlapping units and the associated output units.

In the unsupervised strategy, if the output of the network is "unknown" a new category is created (hidden unit and output unit) using the same procedure described above, the category identifier is generated by the network. If more that one output unit fires, the categories involved are ranked according to the computed degree of belonging (as in MDC). The initial size of the spheres in the unsupervised strategy is settled to a default maximum value given by the user, obviously the maximum possible size can not be used as in the supervised strategy.

Results and Conclusions

1. The MHC adapt to a wide range of applications, since there´s no restriction derived from the size of the problem (storage requirement) or nature (unsupervised or supervised).

2. The convergence time is reduced to 2 epochs, no matter what training strategy is used.

3. The design of the network is simplified, due to the evolving nature of the model. The network designs itself.

4. The HMC is able to learn to separate very complex decision surface.

5. The hardware implementation of HMC is easy, compared to parametric feedforward networks (e.g., backpropagation).

Acknowledgments

Thanks to my colleague Mauricio Paletta who wrote most of the C++ code for RCE and also the base code for MDC.

References

[1] J. A. Anderson et al. "Categorization and Selective Neurons". Parallel Models of Associative Memory. pp. 213-236. 1.981. Earl Baum. N.J.

[2] B. Batchelor, B. Wilkins. "Adaptive Discriminant Functions". Pattern Recognition. 1.968 IEE Conf. Publication, 42, pp 168-178.

[3] B. Batchelor. "Learning machines for pattern recognition". Ph.D. Dissertation, Southampton.

[4] B. Batchelor. "Practical approach to pattern classification". 1.974. New York. Plenum Press.

[5] P. Cooper. "The Hypersphere in pattern recognition". 1.962. Information and Control, 5, pp. 324-346.

[6] P. Cooper. "A note on an adaptive hypersphere decision boundary". 1.966. IEEE Transactions on Electronic Computers, pp. 948-959.

[7] J. Hertz et al. "Introduction to the theory of Neural Computation", Addison-Wesley, 1.991.

[8] S. Grossberg. "Adaptative Pattern Classification and Universal Recoding: Part I". Parallel Development and Coding of Neural Feature Detectors. Biological Cybernetics. Vol. 23. pp. 121-134. 1.976.

[9] T. Kohonen. "Clustering, Taxonomy and Topological Maps of Patterns". Proceedings of the sixth International Conference of Pattern Recognition. pg. 114-125. 1.982. IEEE Press.

[10] **N. Nilsson.** "Mathematical Foundations of Learning Machines". 1.990. Morgan-Kaufmann.

[11] **J. Ramírez.** "Connectionist Learning Methods". Tutorial in the 1st. World Congress on Expert Systems Dec. 1.991. Orlando, Florida.

[12] **J. Ramírez.** "Use of Connectionist Classifiers to detect regularities in data". 1.991. Intevep S.A. tech. report.

[13] **J. Ramírez, I. Torres.** "Conclusions of the use of connectionist models as feature detectors", Intevep S.A., 1.991. tech. report.

[14] **J. Ramírez, I. Torres.** "Connectionist architectures for real-time pattern processing: Design issues", Intevep S.A., 1.991. tech. report.

[15] **D. Reilly et al.** "A neural model for category learning". 1.982. Biological Cybernetics, 45, pp. 35-41.

[16] **D. Reilly et al.** "Learning system architectures composed of multiple learning modules". 1.987. Proc. IEEE First International Conference on Neural Networks, San Diego, CA, June 21-24.

[17] **D. Reilly, L. Cooper.** "An overview of neural networks: early models to real world systems". 1.990. An introduction to Neural and Electronic Networks. Academic Press.

[18] **D. Rumelhart, J. Mc Clelland.** "Parallel Distributed Processing Explorations in the Microstructure of Cognition". Vol. I, II & III. 1.988. MIT Press.

[19] T. Sanger, "Optimal Unsupervised Learning in a Single-layer Linear Feedforward Neural Network", Neural Networks, vol. 2 pp. 459-473, 1.989.

[20] C. Scofield et al. "Pattern class degeneracy in an unrestricted storage density memory". 1.988. Neural Information Processing Systems, pp. 674-682. New York American Institute of Physics.

[21] C. Scofield. "Learning Internal Representations in the Coulomb Energy Network", Cybernetics, 1.987.