Object Recognition Using a Radial Basis Neural Network With a Rotation Mechanism

Art Blevins

December 19, 1997

Contents

1 Abstract

This paper describes an extention to Poggio and Edelman's Gaussian Radial Basis Function (GRBF) network for 3D object recognition. To achieve greater generalization, the new architecture utilizes a biologically plausible rotation mechanism which is iterative, giving an explicit measure of the network's response time. In most object recognition models, an object is determined to be different from the learned objects when the recognition method fails. The proposed architecture includes a mechanism to determine recognition failure without having to wait for *all* of the rotation alignments to be evaluated, thus some "different" decisions can be made quicker that some "same" decisions.

2 Introduction

Poggio and Edelman have suggested a Gaussian Radial Basis Function (GRBF) neural network for object recognition [6, 7, 8]. Their network uses a set of 2D coordinates for n features that appear in a 2D image of the object. The GRBF neural network them "compares" the input view with several stored views to make a recognition decision. Each radially symmetric basis function represents a single study view of a specific object. The function compares the views by computing a distance metric between the corresponding features of the test view and a specific study view of the object. That distance metric is extended over the m objects views using

$$S = \sum_{i=1}^{m} w_i G(||F - C_i||)$$

where S is a measure of similarity, F is the set of coordinates of the test view, and $C_i, 1 \leq i \leq m$ are vectors representing the feature coordinates in the m study views. The function G can be any radially symmetric function, but is usually chosen to be the Gaussian.

This paper describes a variant of the GRBF model which incorporates the following features.

- An alignment mechanism is used to improve generalization performance, and is implemented in a biologically plausible manner.
- An explicit measure of the model's response time is based on the alignment process.
- A neural mechanism has been designed to allow quick negative recognition decisions.

This variant of the GRBF model is called PERM (**P**oggio and **E**delman's architecture with a **R**otation **M**echanism). The nomenclature brings to mind a process by which hair styles are artificially created. While the object recognition process described here has nothing to do with hair, the analogy is not totally irrelevant. Both refer to artificial processes that are attempting to simulate natural human characteristics.

3 PERM Notation and Architecture

3.1 Network Nodes

The PERM neural network is constructed using the 2D coordinates of the n corresponding features visible in each of the study views of the object. These are not actually network nodes, but are used as inputs to some nodes in the hidden layers in the PERM architecture. These feature vectors are denoted

$$C_i \in R^{2n}, 1 \leq i \leq m$$

where m is the number of study views and C_i is the vector containing the 2D coordinates of the n features in study view i. Thus, C_i has the form

$$C_i = (x_1, y_1, x_2, y_2, ..., x_n, y_n)^T$$

where T denotes the transpose operator. The notation

$$C_{ij}, 1 \leq i \leq m, 1 \leq j \leq 2n$$

will also be used to refer to individual elements from C_i. The jth element of C_i is denoted by C_{ij}. Similarly, $F \in R^{2n}$ is defined as the vector containing the 2D coordinates of the n features in the test view of the object, and F_j is the jth element of F. The network input consists of $F_j, 1 \leq j \leq 2n$. The model assumes the coordinates of the features are relative to the center of mass of the n features.

The first hidden layer of nodes in the network is denoted by

$$r_j^{(t)}, 1 \leq j \leq 2n, t \geq 0$$

where t is a meaure of time, here representing a specific iteration number.

The m Gaussian radial basis functions comprise the second hidden layer of nodes of the neural network. These are denoted

$$g_i^{(t)}, 1 \leq i \leq m, t \geq 0.$$

There are two additional "direction" nodes: d_1 and $d_2^{(t)}$ used in the architecture. The node d_1 is connected to the input $F_j, 1 \leq j \leq 2n$ and denotes the initial direction of rotation. The value of d_1 will remain constant during the operation of the network (i.e. it does not vary with t). The node $d_2^{(t)}$ denotes the network's direction of rotation at time t. Inputs to node $d_2^{(t)}$ come from $r_j^{(t)}, 1 \leq j \leq 2n$.

The output layer consists of two nodes: $D^{(t)}$ and $S^{(t)}$. The node $D^{(t)}$, called the "different" output node, is connected to nodes d_1 and $d_2^{(t)}$ and is used to determine if the input object is *different* than the object represented by the stored views. The node $S^{(t)}$, called the "same" output node, is connected to the basis functions $g_i^{(t)}$ and is used to determine if the input view represents the same object.

3.2 Network Connections

There are two sets of weights in the PERM network. The first is a set of $4n$ weights denoted

$$v_{jk}, 1 \leq j \leq 2n, 1 \leq k \leq 2.$$

These weights interconnect nodes $r_j^{(t)}, 1 \leq j \leq 2n$ in the first hidden layer, and are responsible for rotating the representation of the test view of the object.

The weights v_{jk} are initialized as constants, based on the parameter θ.

$$v_{j,1} = \cos(\theta), j = 1, 3..., 2n-1$$

$$v_{j,2} = \sin(\theta), j = 1, 3..., 2n-1$$

$$v_{j,1} = -v_{j-1,2}, j = 2, 4..., 2n$$

$$v_{j,2} = v_{j-1,1}, j = 2, 4..., 2n$$

The second set of weights connects the second hidden layer of nodes, $w_i, 1 \leq i \leq m$, to the output node $S^{(t)}$.

The weights w_i are also constants, initialized to equivalent values.

$$w_i = \frac{1}{m}, 1 \leq i \leq m$$

This strategy weights the study views equally in the recognition decision. If a particular view is thought to be more "canonical" than the others [3], the weights can be adjusted accordingly. Neither the w_i weights or the v_{jk} weights are adjusted during the network's operation.

3.3 Model Parameters

The following is a list of parameters (constants) used by the model.

- σ is sometimes called the basis function "width" and controls the response of the Gaussian basis functions.
- θ is a constant that controls how many degrees the test view features are rotated on each iteration.
- β is a constant bias term that controls how closely a test view and the network basis functions must match before the test view is deemed the same object.

4 PERM Network Activations

4.1 The Hidden Layer Nodes

The first layer of hidden nodes have initial activations

$$r_j^{(0)} = F_j, 1 \leq j \leq 2n$$

but for $t > 0$, the activations of the first hidden layer nodes are defined as follows.

$$r_j^{(t)} = v_{j,1} r_j^{(t-1)} + d_2 v_{j+1,1} r_{j+1}^{(t-1)}, j = 1, 3, ..., 2n-1$$

$$r_j^{(t)} = d_2 v_{j-1,2} r_{j-1}^{(t-1)} + v_{j2} r_j^{(t-1)}, \ j = 2, 4, ..., 2n$$

Thus, at time t, node $r_j^{(t)}$ is based on a weighted sum of its previous value (i.e. at time $t-1$), and the previous value of its "partner node". Recall that the features are represented as 2D coordinates, so the nodes in this layer can be logically grouped in pairs. Here "partner node" refers to the other member of the pair.

The Gaussian radial basis functions in the second hidden layer have activations defined by

$$g_i^{(t)} = exp\left(-\frac{\sum_{j=1}^{2n}\left(r_j^{(t)} - C_{ij}\right)^2}{2\sigma^2}\right) = \prod_{j=1}^{2n} exp\left(-\frac{\left(r_j^{(t)} - C_{ij}\right)^2}{2\sigma^2}\right), 1 \leq i \leq m$$

where σ is a network parameter. These equations illustrate that a multi-dimensional Gaussian can be decomposed into a product of lower dimensional Gaussians. The singly-dimensioned Gaussians will be denoted using two subscripts

$$g_{ij}^{(t)} = exp\left(-\frac{\left(r_j^{(t)} - C_{ij}\right)^2}{2\sigma^2}\right)$$

and will measure the similarity of feature j of the test view with feature j of the ith study view.

4.2 The Direction Nodes

The direction nodes keep track of the direction of rotation. The node d_1 indicates the initial direction of rotation, and node $d_2^{(t)}$ indicates the direction of rotation at time t. In order to determine how to align a test view with a specific study view, it is only necessary to use one feature of the object. Since this model assumes the correspondence problem has been solved, the first feature in the feature vector is used. This choice is arbitrary. Other strategies for choosing the feature to use for alignment could include choosing a feature at random, or using some metric such as the feature with the greatest distance from the center of mass of the object features.

Once a single feature has been identified, the model estimates which of the study views is "closest" to the test view. The model will rotate in the direction of the smallest angle toward this "closest" test view. Estimating the closest view is trivial, given the decomposition of the multi-dimensional Gaussians into sinlge-dimensional Gaussians. The output of the Gaussian $g_{ij}^{(t)}$ is a measure of how close feature j in the study view is to feature j in study view i. Given a specific feature, j, the maximum over the set $g_{ij}^{(t)}, 1 \leq i \leq m$ can be used as an estimate of the closest study view, here denoted by i_{max}. Currently, the model chooses i_{max} as follows.

$$i_{max} = \{i | g_{i,j}^{(t)} = \max_k g_{k,1}^{(t)}\}$$

There are several ways to determine a maximum in a neural architecture. Edelman and Weinshall set a global threshold to an initial value larger than any possible node output [4]. Their strategy for identifying the maximum involves iteratively lowering the threshold until a "winner" emerges. This "winner" node is the maximum.

Lippmann details another strategy for identifying a node whose output is maximum. He describes a neural network structure called MAXNET which is a layer of nodes interconnected with inhibiting weights [5]. After the initial activations of the nodes in this layer, they iteratively inhibit the activations of their neighbors until the competition has converged to a configuration where only one node has a non-zero activation. This single node will be the one whose initial activation was the highest.

Either of these strategies could be used in this network, and future implementations of PERM will incorporate such biologically plausible mechanisms. Currently, however, i_{max} is found by using the **max** function available in the java programming language.

Given i_{max}, the closest study view, the direction of rotation is determined by computing a cross product. Consider the vector that originates at the center of mass of the features and extends through the chosen feature (i.e. the one feature being use to determine rotation direction). The cross product of the two versions of this vector, one from the test view and one from the closest study view, reveal the direction of rotation. The cross products are computed by the neural network nodes d_1 and $d_2^{(t)}$ whose activations are

$$d_1 = \Theta(C_{i_{max},1}F_2 - C_{i_{max},2}F_1)$$

$$d_2^{(t)} = \Theta\left(C_{i_{max},1}r_2^{(t)} - C_{i_{max},2}r_1^{(t)}\right)$$

where $\Theta(z) = +1$ if $z \geq 0$ and $\Theta(z) = -1$ otherwise. An output of $+1$ indicates a clockwise rotation, and -1 indicates a counter-clockwise rotation.

Given two different vectors with a common origin in a 2D plane, a rotation in either direction will eventually produce an alignment of the two vectors. In the process of determining the direction of rotation, the model is calculating the direction which requires the *shortest* angular rotation. It can be shown that the cross product described above will always yield the direction of the shortest angular rotation, even if the object view has been distorted by a perspective transformation.

4.3 The Output Nodes

The "different" output node $D^{(t)}$ is computed as follows.

$$D^{(t)} = \Theta\left(-d_1 d_2^{(t)}\right)$$

Thus, as long as the initial direction of rotation and the current direction of rotation are the same, $D^{(t)} = -1$, which is interpreted as "the network cannot

determine if the test view is different at time t." If $D^{(t)} = 1$, then the network has changed direction of rotation from the original direction. This means the two corresponding features have rotated past one another and the network did not respond "same". Thus, $D^{(t)} = 1$ can be interpreted as "the test view is different from the study views."

The "same" output node $S^{(t)}$ is computed using

$$S^{(t)} = \Theta\left(\sum_{i=1}^{m} w_i g_i^{(t)} - \beta\right)$$

where β is the bias, a model parameter. $S^{(t)} = -1$, is interpreted as "the network cannot determine if the test view is the same at time t." $S^{(t)} = 1$ is interpreted as "the test view represents the same object as the study views."

5 PERM Training and Testing

5.1 Training

Each PERM network represents multiple views for a *single* object. Training consists of initializing the m basis function centers, C_i, with the features of the m study views for that specific object.

5.2 Testing

Given the n features of a test view, the network operates as follows.

1. Set $t = 0$
2. Set the network input, F, equal to the feature vector of the test view.
3. Feed the signals through the network
   (a) Determine the value of the direction nodes d_1 and $d_2^{(t)}$
   (b) Determine the value of hidden layer 1: $r_j^{(t)}, 1 \leq j \leq 2n$
   (c) Determine the value of hidden layer 2: $g_i^{(t)}, 1 \leq i \leq m$
   (d) Determine the value of the output nodes: $D^{(t)}$ and $S^{(t)}$
4. If $S^{(t)} = +1$ the answer is "same" (stop and report the answer).
5. If $D^{(t)} = +1$ the answer is "different" (stop and report the answer).
6. Otherwise, the network cannot answer yet. Increment t, and go back to step 3.

When the network answers, the value of t indicates the number of iterations required to arrive at an answer and is a direct measure of the neural network's response time.

6 Future Directions

The current model could be improved by incorporating the following features.

- A biologically plausible *max* operation similar to those described in section 4.2 should be implemented. These iterative strategies for computing a maximum would add another term to the network's response time.
- The current model assumes the origin of the 2D features coincides with the center of mass of the n features. Future implementations of the model could incorporate neural network structures to perform this translation prior to input to the current PERM model. This would provide translation invariance as well as rotation invariance.
- Currently, the model always uses the first of the n features to determine the direction of rotation. Other strategies mentioned in section 4.2 could be imeplemented.
- The current model assumes the correspondence problem has been solved. A neural "pre-processor" could be added to determine feature correspondence, or at least a ***subset*** of corresponding features.

This model could be expanded in several other ways. It is possible to intermix training and testing. When a new training view is seen (or a test view is recognized as the "same"), a node represesnting that view could be added to the architecture. Techniques for dynamically adapting the architecture of a GRBF network have been published [1, 2]. In addition, it is possible to use fewer basis functions than training views. Either a subset of the study views can be selected as canonical views, or the centers can be trained [1]. Training the centers would be equivalent to creating prototypes in memory.

To account for virtual views, again the dynamic architecture could be exploited. For instance, to simulate the effects of apparent motion between two views, the dynamic architecture scheme could be employed to add nodes to the GRBF architecture. These additional nodes could correspond to views that would be seen if the object actually was in motion. Using a rotation scheme similar to the one presented above, the basis function centers, C_i, could be rotated to create the virtual views.

7 A Final Note

A version of PERM, implemented in java, can be run via the World Wide Web. It can be found at the following URL.

`http://turing.cn.edu/research/PERM.htm`