Self-Organized-Expert Modular Network for Classification of Spatiotemporal Sequences

Sylvian R. Ray and William H. Hsu

Department of Computer Science
University of Illinois at Urbana-Champaign
1304 West Springfield Avenue
Urbana, IL 61801
(ray | bhsu)@ cs.uiuc.edu

Abstract

We investigate a form of modular neural network for classification with (a) pre-separated input vectors entering its *specialist* (expert) networks, (b) specialist networks which are self-organized (radial-basis function or self-targeted feedforward type) and (c) which fuses (or integrates) the specialists with a single-layer net. When the modular architecture is applied to spatiotemporal sequences, the Specialist Nets are recurrent; specifically, we use the Input Recurrent type.

The Specialist Networks (SNs) learn to divide their input space into a number of equivalence classes defined by self-organized clustering and learning using the statistical properties of the input domain. Once the specialists have settled in their training, the Fusion Network is trained by any supervised method to map to the semantic classes.

We discuss the fact that this architecture and its training is quite distinct from the *hierarchical mixture of experts* (HME) type as well as from *stacked generalization*.

Because the equivalence classes to which the SNs map the input vectors are determined by the natural clustering of the input data, the SNs learn rapidly and accurately. The fusion network also trains rapidly by reason of its simplicity.

We argue, on theoretical grounds, that the accuracy of the system should be positively correlated to the product of the number of equivalence classes for all of the SNs.

This network was applied, as an empirical test case, to the classification of melodies presented as direct audio events (temporal sequences) played by a human and subject, therefore, to biological variations. The audio input was divided into two modes: (a) frequency (or pitch) variation and (b) rhythm, both as functions of time. The results and observations show the technique to be very robust and support the theoretical deductions concerning accuracy.

Keywords: modular neural networks, temporal sequences, fusion, multichannel signals.

1. Introduction

The primary objective of modularity is the reductionist ideal, to divide the complete task into subtasks, each of which can be more simply learned and then combined.

1.1 Modular Networks

Modular networks are frequently realized as a *hierarchical mixture of experts* (HME), cf. [1, 2, 10]. In that version, each expert network learns what region of the input space it is assigned to. The expert outputs are combined linearly in an integration or fusion network, the weighting being supplied by the gating network, which is also trained during the learning process.

Another approach to modularity is to divide the training set into M batches that are assigned to M experts [2]. In this case, each of the M experts receives full-width input vectors.

In both of these approaches, the total complexity, measured as the total number of outputs of the expert networks, is $M \cdot N_S$, where N_S, is the number of semantic classes.

An alternative form of the modularity problem, the *Specialist-Fusion* (SF) architecture, which we pursue here, begins with the assumption that each mode (i.e., specialist) is associated with a distinct, physically separate sensory input from the outset. Correlated external stimuli may excite input vectors simultaneously in all modes. This structuring of the modularity problem not only resembles a natural system where the sensory inputs are large scale classes such as visual, auditory, and somatosensory but also corresponds well with a multichannel data acquisition system [11].

This form of the problem has the distinct advantage that each specialist can be trained by self-organization to use only as many equivalence classes as are needed for the particular task. This potentially reduces the complexity (in the sense of network size) as compared to the HME form where every expert must, in practical realizations, provide one output for each semantic class.

The specialist outputs are then combined by a relatively simple fusion network, usually, but not limited to, a single layer.

We study here the ramifications of this SF architecture for modularity, the objective being to understand and quantify the complexity issues and to test the conclusions arrived at.

1.2 Architecture of Specialist-Fusion Networks

The neural network that we are investigating, a Specialist-Fusion network, receives its input as M distinct sets of feature vectors, $X^1 \cdots X^M$. Each set originates from a unique sensor (conceptually), and is processed by one specialist network as depicted in Figure 1. The dimensionality of the input vectors for each mode is arbitrary.

A SN can be any self-organized architecture that maps its individual domain into an arbitrary set of equivalence classes, $C_1^k, C_2^k, \ldots, C_{N_M}^k$. Example architectures are RBF nets [4], Kohonen maps, or feedforward nets trained by the targeted algorithm given below.

Each specialist network, SN_k, divides its input domain into N_k equivalence classes based on the statistical distribution of its input vectors. While the number of self-organized equivalence classes (SOE's), N_k, is arbitrary, it must be large enough to permit adequate discrimination among the semantic classes which are ultimately to be distinguished. Thus, for example, if certain semantic classes cannot ultimately be discriminated without distinguishing between the colors, orange, yellow, and brown, then the SN specializing in color must itself output these three as SOEs.

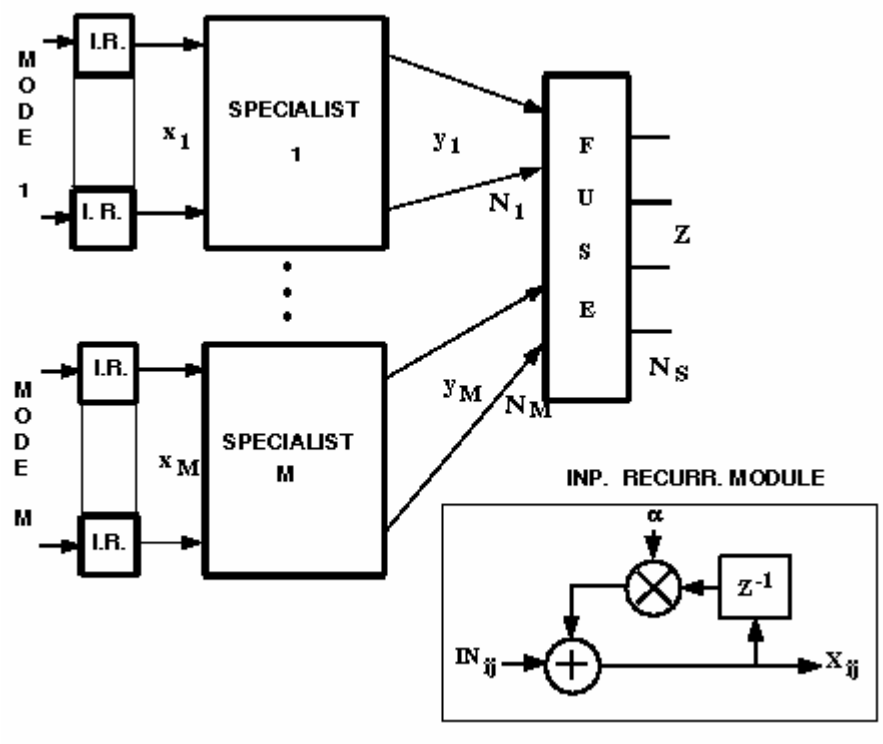

Figure 1. Modular Classifier Network
(numbers of outputs from specialists, N_i, are generally different)

The outputs of the SN's, $y_1,\ldots,y_M$, are concatenated to form the input of the final network, the *fusion* network, which maps the SN outputs into semantic classes, $\{Z_1,\ldots,Z_S\}$. The semantic classes are the only classes that have an ultimate external meaning; they are the target classes supplied in the training set.

We set out to investigate the application of the modularity concept specifically to spatiotemporal sequence classification and recognition. The sequential properties can be introduced by making the SN's simple recurrent networks. We chose to use the Input Recurrent[1] form of SRN to introduce the temporal information. This operation maps the input vectors into various regions of phase space determined by the temporal order of events. If each recurrent vector embodies all of the temporal information in one event, the resulting vector can be treated as a static input vector in the training and/or test sets.

The remaining text is organized as follows. We will first discuss the structure and training of the SNs and their performance in the ideal case, which results in the simplest overall network structure. Then we analyze the near-optimum case(s).

Finally, two example experiments are discussed, the first being selected to demonstrate and verify the optimum complexity case, and the second of which supports the complexity relationship deduced by the theoretical analysis.

[1] *Input recurrent* networks are members of the family of simple recurrent networks [5, 6], including Elman networks [9]. Their construction is documented in Appendix A; they were previously investigated by Principé and Lefebvre [13], and by Mozer [6] and Mehrotra *et al* [5] under the term *exponential trace memories*. The term *input recurrent*, however, is not standard; we present a formal definition here.

2. Training the Specialist Networks

The primary function of an expert network in the HME model is to map a region of the total input space, selected by the gating network, into the output vector. Each expert network output is a vector of dimension, N_S = the number of semantic classes.

On the other hand, the primary function of an specialist network, SN_k, in our SF case is to differentially segment its (individual) input domain into N_k specialist equivalence classes. The number of equivalence classes, N_k, is, in the present study, the subject of experiment but, typically, we can expect $N_k < N_S$.

The SN architecture may be chosen arbitrarily from among the network types that provide *self-organized clustering* of the input domain. For example, training an RBF network by *k*-means clustering or Kohonen mapping is a practical choice [4]. A feedforward network trained to target equivalence classes derived from statistical clustering of the training set is also a satisfactory SF network.

The number of specialist equivalence classes per mode, N_k, is a design parameter which has a major effect on the overall classification accuracy and which we will investigate in a subsequent section.

2.1 Specifics of Training the Network

All SNs are trained first, followed by training of the fusion network.

Using any clustering algorithm, the training set for mode *k* is divided into N_k self-organized equivalence (SOE) classes with centers, $\mu_1, \ldots, \mu_j, \ldots, \mu_{N_k}$.

If the SN's are chosen to be RBF networks, the μ_j are the RBF unit centers.

In the case that a SN is chosen to be an RBF net:

Each RBF unit outputs a scalar value

$$y_i^k = \frac{\exp\left(\left\| x^k - \mu_i^k \right\|\right)}{\sigma_i^k}$$

where x^k = the input vector for the *k*th specialist net
and μ_i^k = the center of element RBF_I for the *k*th specialist.

For the case of self-targeted feedforward network style of SN, the training is performed as follows. Each input vector receives a label corresponding to its (arbitrarily assigned) SOEC number. The SN is then trained using a supervised learning algorithm using the SOEC labels as the target class. The particular self-organizing algorithm we used is of little relevance but it is given in the appendix. We chose to take the SNs output vectors, y_i^k, as linear or softmax vectors rather than WTA in order to preserve more information. The output of the *k*th SN is a vector, $y^k \varepsilon R^{N_k}$.

In order to analyze the system theoretically, however, the one-of-C case is used for clarity.

The Fusion Network

The concatenated output of the SN's is

$$Y = \left[y^1, y^2, \ldots, y^K \right]$$

which forms the input of the fusion network.

Each SN in the HME architecture has N_S outputs resulting in a total number of outputs of $(M+1)\cdot N_S$ but in our SF network, the number of outputs per specialist is typically much less than N_S.

If the SN's perform the segmentation of their individual domains with Sufficiently fine granularity, the task of the Fusion network is relatively simple and it can be accomplished using a single layer network trained by delta rule. If the SN's outputs are not sufficiently segmented, a two-layer network might be necessary.

To get the deepest insight into the operation of this architecture, we next discuss the network with some idealizations.

3. Properties of the Modular Network

To make the argument clear and simple, we assume the SN outputs are hard-limited (i.e., a single winner-take-all neuron is selected to obtain test predictions) and that their targets (semantic class) are 1-of-C coded (cf. [12]).

We define the complexity, C, of the network as the total number of outputs from all SNs plus Fusion net, that is:

$$C = \sum_{i=1}^{M} N_i + N_S$$

where

- M = the number of specialists
- N_S = the number of semantic classes to be recognized
- N_i = the number of SOECs for the *i*th expert network.

The number of distinct states of **Y**, the concatenated SN output vector, is $N = \prod_{i=1}^{M} N_i$, which, optimally, equals the number of semantic states, N_S.

Equating N_S to the number of distinct states of Y,

$$C = \sum_{i=1}^{M} N_i + \prod_{i=1}^{M} N_i .$$

With N_S fixed, the minimum complexity then reduces to the well-known problem of minimum-perimeter-for-fixed-area, which occurs when all N_i are equal and their product is N_S. At minimum complexity, $N_i = \sqrt[M]{N_S}$, and the complexity itself is: [2]

$$\min(C) = M\sqrt[M]{N_S} + N_S$$

[2] The equality holds strictly only when $\sqrt[M]{N_S}$ is an integer.

.

Quite simply, the minimum complexity under these ideal conditions required to discriminate between N_S classes occurs when there is (1) *an equal number of self-organized equivalence classes for each specialist* and, (2) the product of all SOEC sizes equals the number of semantic classes.

This ideal classification problem corresponds to the fusion net consisting merely of N_s, N_Y-input logical AND circuits, the M-dimensional version of a 2-D selection matrix so well known in computer memory system (RAM/ROM) architecture. If $N_Y = \sum_{i=1}^{M} N_i$, we have, by definition, the *square-orthogonal* case. It is illustrated in Figure 2 for the case M=2, $N_1 = N_2 = 4,$ and $\mathrm{N_S} = 16$.

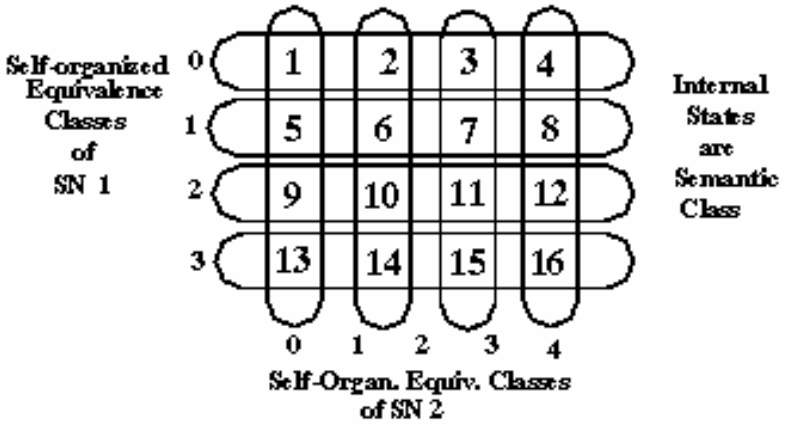

Figure 2. Ideal Orthogonal Separation of Classes for M = 2

An ideal multimodal input implies that each input mode carries *equal discriminatory power* so that the division into the same number of equivalence classes for each mode is the correct granularity to distinguish among semantic classes. More realistically, we would expect $N_i \neq N_j$, for $i \neq j$ in general, requiring that some SNs need to divide their feature space more finely than others to achieve the desired N_S. This case is discussed next.

3.1 Approximately Ideal Case

For a semi-ideal, but more probable, application of modularity, the granularity for each SN should different, that is, $N_i \neq N_j, i \neq j$ generally. For example, if color and shape were the two modes, and the problem consists of classifying every (color,shape) pair as a semantic class, then with only 3 distinct colors and 15 shapes, the shape granularity needs to be at least 5 times greater than the color mode granularity. In this case. the minimum complexity is

$$C = N_1 + N_2 + N_S = 3 + 15 + 45 = 63.$$

For this case in general, namely when $N_i \neq N_j$, when $(i \neq j)$, the minimum complexity reduces to

$$C = \sum_{i=1}^{M} N_i + \prod_{i=1}^{M} N_i = \sum_{i=1}^{M} N_i + N_S .$$

The first term is the total number of outputs of the SNs and the second term is the total number of distinct states which the fusion network can deliver, using the simplifying assumption that the SN output vectors are *1-of-C coded* [12].

This case can be visualized simply as the semantic classes being located at the intersection of N_i planes in each of the M dimensions, as shown in Figure 3. We will refer to this as the *orthogonal* (or *rectangular*) case. The fusion elements would merely need to perform binary logical AND operations.

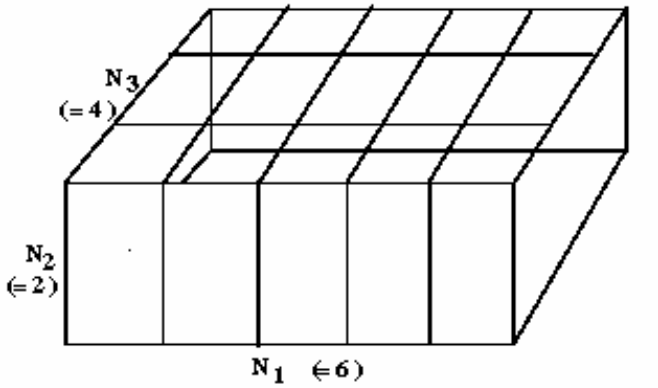

Figure 3. Near-Ideal Orthogonal Case for 3 Specialists (M = 3).
A semantic class occurs at each planar intersection.

3.2 The General Case

In general, we allow the outputs, y, of the SNs to be real numbers and require the fusion network elements to compute general threshold functions but compete to be winner, as usual for classification service.

The question we ask is: what is the relationship between the number of SOECs of the SN's and the classification accuracy?

We limit ourselves to M=2 (two modes) for simplification of the following argument.

Consider the case: $N_1 N_2 >> N_S$. The number of the specialist (SOE) classes is large enough to easily provide a unique combination of outputs from SN_1 and SN_2 to distinguish each semantic class. If, in addition, N_1 and N_2 are both large enough that every semantic class can be uniquely described by an intersection of the two modes, the accuracy of the complete network will tend to be high.

Consider on the other hand, the case $N_1 N_2 < N_S$. Now, there are not enough combinations of the SOECs to uniquely classify all semantic classes. Hence the accuracy of the complete net will tend to be low.

Interpolating between these two cases, we conclude that one should expect a positive correlation between the product of the number of SOECs, $\prod_{i=1}^{M} N_i$, and the overall classification accuracy. This assumes that every SN has enough equivalence classes to segment its input space with sufficiently fine

granularity to make it possible to distinguish the semantic classes. In general, we therefore expect Classification Accuracy $\approx \prod_{i=1}^{M}(N_i)$.

The foregoing deduction can be roughly summarized graphically as shown in Figure 4.

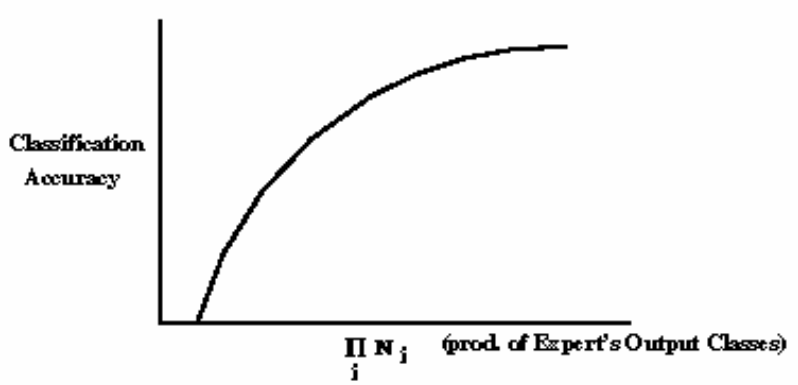

Figure 4. Expected Correlation between Number of Self-Organized Equivalence Classes and Classification Accuracy

4. Experimental Evaluation

4.1 Experiment 1: Approximately Orthogonal Temporal Sequences

The objective of our first experiment was to select two modes of a signal, which are nearly orthogonal in order to test the foregoing deduction.

The chosen data base for this experiment consisted of 16 audio "tone sentences" that constitute the semantic classes to be recognized. A *tone sentence,* S_i consisted of 3-6 *words* drawn from a set of seven words, $\{W_0 W_1 \ldots W_6\}$ (see Figure 5). Each word is a frequency-continuous tone, having the specific frequency profile shown. The 16 tone-sentences used as database are listed in Table 1. The word-sequence column is the sequence of words, W_i, used to define each sentence-class. A rough example of a tone-sentence in nature (selected merely to convey the concept) is the call of a great horned owl where each separable "hoot" sound is a "word". Our data was generated not by an owl, however, but by a human playing a slide whistle. Consequently, each instance of a word or sentence is slightly different.

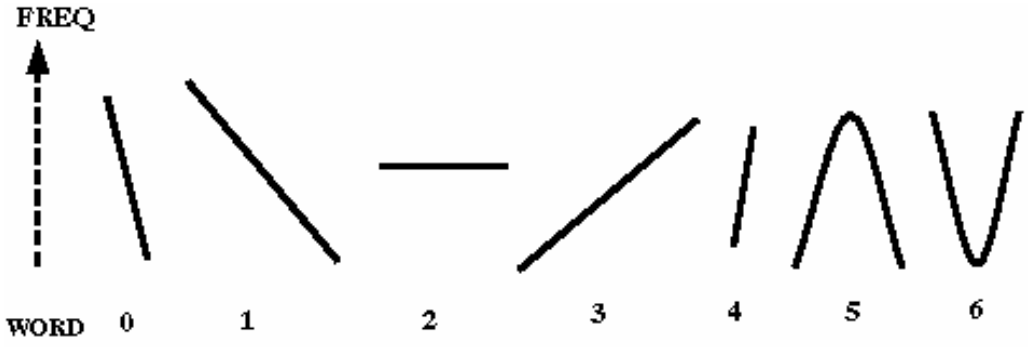

Figure 5. Frequency Contour of the 7 Words for Sentence Formation
e.g., word 5 is a rising + falling continuous pitch

Simulating Multiple Sensory Modes and Extracting Multiple Modes of Information

Sentence Class	Word Sequence (Ideal)
1	2,2,2
2	3,1,3
3	6,2,3
4	3,3,5
5	2,2,2,2
6	3,1,3,1
7	6,2,3,6
8	3,3,5,2
9	2,2,2,2,2
10	3,1,3,1,3
11	6,2,3,6,2
12	3,3,5,2,1
13	2,2,2,2,2
14	3,1,3,1,3,1
15	6,2,3,6,2,3
16	3,3,5,2,1,2

Table 1. Nominal Word Sequence for the 16 Sentences Used in Experiment 1

One of the two modes of information extracted from the signals (i.e., sentences) is *rhythm*. The rhythm vector reduced from the audio signal in this case is merely the sequence of relative lengths (in milliseconds) of [w*ord*(1), *gap*(1), *word*(2), *gap*(2),...*word*(*n*)].

By design, sentences $S_1 - S_4$ formed a single rhythm class, each vector having n=3 words (and 2 gaps). Similarly, the remaining twelve sentences formed three more classes in the rhythm mode, namely, $(S_5 - S_8)$ containing 4 words, $(S_9 - S_{12})$ containing 5 words, and $(S_{13} - S_{16})$, containing 6 words (separated by
n-1 = 5 gaps). Ideally, these 4 rhythm classes should resolve into 4 SOECs for the Rhythm specialist network (R-mode). However, the actual number of SOECs, in our experiment, depended on the clustering algorithm used on the training set.

The second of the two modes extracted from the sentences was the temporal sequence of word-classes, which is essentially *frequency* as a function of time, and it is represented as word-