Object segmentation using an array of interconnected neural networks with local receptive fields

Predrag Neskovic and Leon N Cooper
Physics Department and Institute for Brain and Neural Systems
Brown University, Providence, RI 02912
email: *Predrag_Neskovic@Brown.edu* and *Leon_Cooper@Brown.edu*

Abstract

Neural networks (NNs), such as multi layer perceptrons and radial basis function architectures, proved to be powerful tools in many problems where the objective is robust classification. However, in applications that require simultaneous segmentation and recognition, such as speech and handwriting recognition, NNs were used with much less success. In this work, we introduce an architecture for object segmentation/recognition that overcomes some limitations of classical NNs by utilizing contextual information. An important characteristic of our model is that recognition is treated as a process of discovering a pattern rather than a one-time comparison between a pattern and a stored template. Our network implements some properties of human perception and during the recognition emulates the process of saccadic eye movements. We contrast our model to hidden Markov models in application to segmentation/recognition of handwriting and demonstrate a number of advantages.

1 Introduction

Neural networks (NNs), such as multi layer perceptrons and radial basis function architectures, proved to be powerful tools in many problems where the objective is robust classification. However, in many situations, such as speech and handwriting recognition, segmentation seems to be as important and difficult problem as classification itself.

For many years, the dominant paradigm in speech and handwriting recognition was based not on NNs but on hidden Markov models (HMMs) [6]. Although HMMs are able to successfully deal with segmentation problems, their discriminative powers are not as good as NNs [3]. When applied to handwriting, this means that given a local section of the input pattern a NN can better classify it as one of the letters from the alphabet than an HMM-based system. The straightforward approach to handwriting recognition problem would be to cover the pattern with local NNs (letter detectors) and then just read their outputs. However, the problem with using an array of spatially arranged local NNs is how to connect their often conflicting outputs since they are operating independently of one another. One of the simplest solutions for selecting the "correct" outputs is to use a dynamic programming technique [7]. Another possibility of connecting the NNs is to convert NNs posterior probabilities into emission probabilities and thus "embed" the NNs into a HMM-based system [2, 8, 3] creating so called hybrid NN/HMM systems [3, 8].

In this work, we present a new model for connecting an array of independently processing local NNs using the weights that carry contextual information. In addition, we introduce a new approach for pattern segmentation/recognition in which the network explores a pattern in a way qualitatively similar to human perception by emulating saccadic eye movements. Although for the description of our model we will mostly use handwriting examples, the method we present is quite general and is currently being applied to recognition of vehicles from real-time video streams.

The paper is organized as follows. In Section 2 we illustrate some of the problems associated with pattern segmentation and identification of local regions and motivate the construction of our network. In Section 3 we present the network architecture and define the equations that govern the processing of the units of each layer of the network. In Section 4 we describe the recognition process - an algorithm for hierarchical exploration of the pattern that focuses attention on most salient regions. We show that the definition of "saliency" is dynamically modified by contextual influences during the recognition process. And finally in Section 5 we present the results of our system in application to recognition of on-line handwriting and show

0-7803-7044-9/01/$10.00 ©2001 IEEE

some of the advantages of our model over the HMMs.

2 Background

One of the problems related to the segmentation of an object into its parts is that local regions of the pattern can be interpreted in many different ways. This can be easily seen from the example shown in Figure 1, a word "account" written by one of the writers from our database. The region of the pattern, marked with the rectangle, can be most likely identified as the letter "u" or the letter "n". However, we associate this region with two letters "cc" based on contextual information. The fact that this region of the pattern is surrounded by the letters "a", "o", "u" and "t" that can be fairly clearly identified, influences the perception so that the region is then associated with letters "cc". Similarly, the reason why we don't identify this region as the letter "u" or "n" is because these letters do not have support from the surrounding letters.

Figure 1: An example of a pattern in which different regions have different confidence of representing a given dictionary word.

From the previous example we can make the following observations: The region of the pattern within the rectangle can be associated with the letters "cc" with very low confidence. On the other hand, this region can be identified as part of the word "account" with very high confidence. After the pattern is recognized as the word "account" one can think of the letters "cc" merely as the symbols with which we label the region of the pattern that is part of the word "account". In fact, even if one of the letters "c" was completely missing we would still be able to associate the pattern with the word "account"" and the label of the region would still be "cc".

In order to remove the ambiguities associated with a local region of a pattern, one should increase the size of the input vector to the NN and thus include the contextual information. Ideally, the input to the NN should be the whole pattern. However, there are several problems with this approach. Fist, the size of the pattern representing the same dictionary word significantly varies from one writer to another depending on the writing style. Since a NN has to be trained on the fixed size input vector the question is which size should the network be trained on? Second, training a neural network using a very large input vector (equivalently a very large feature vector) that would safely cover patterns of all possible sizes is practically impossible due to the "curse of dimensionality" [1]. It can be shown that the number of training examples increases exponentially with the number of features that represent the pattern therefore limiting the size of the feature vector that can be used for all practical purposes. This is one of the reasons why NNs were never successful in applications such as speech and handwriting recognition. Instead, as we mentioned in Section 1, the researchers resorted to the use of HMMs or a combination of HMMs and local NNs.

3 The Architecture of the Network

In this section, we will present an architecture that provides a possible way of connecting local NNs so that they can exchange contextual information. The network has a hierarchical structure and consists of several layers of processing units. The units of each subsequent layer of the network have progressively larger receptive fields and therefore process the larger portions of the input pattern.

The first layer consists of an array of spatially arranged NNs that operate independently of one another and are selective to features of specific classes. When applied to handwriting recognition, these units are selective to different letters from the alphabet whereas in the application to recognition of vehicles from video streams the simple units are selective to different parts of the vehicle. The second layer units, called simple units, are also selective to features of specific classes but include contextual influence from one feature called the central feature. Referring to the example in Fig. 1, if the central feature is the letter "o", then the simple units, selective to letters "a", "c", "c", "u", "n" and "t" would be influenced by the presence of the detected letter "o". The third layer units, complex units, incorporate contextual information in the other direction. Referring again to the same example in Fig. 1 the complex unit associates the region of the pattern identified as the letter "o" as part of the word account based on the fact that the letters "a", "c", "c", "u", "n" and "t" are detected at their corresponding locations. At the top of the pyramid are the object units that integrate information coming from different complex units and

therefore different regions of the pattern.

For illustration purposes and in order to simplify the description, we will present the 1D network architecture that can be applied to segmentation/recognition of patterns such as handwriting. Therefore, features will be letters and an object unit will be called a word unit.

The first layer units, called letter detectors, have restricted and overlapping receptive fields and are arranged in such a way that they completely cover the input pattern. Each letter detector is in itself a NN and in our application to handwriting, the architecture is based on a weight sharing technique proposed by Rumelhart [7].

Let us denote by $\vec{y}$ the input pattern and by $\Delta\vec{y}$ a section of the pattern whose center is at location x with respect to an arbitrary reference point. Then the output of the letter detector positioned over the section $\Delta\vec{y}$ estimates the probability $d^c(x) = p(\alpha^c|\Delta\vec{y})$ that the section represents a letter of specific class c, $c \in [1, ..., 26]$ and $\alpha^c \in [a, b, c, ..., z]$. This estimate therefore represents a level of confidence with which the section $\Delta\vec{y}$ can be associated with a specific letter from the alphabet but it doesn't tell us to which word this letter belongs. For example, a letter "a" detected at certain location can be part of the word "act", or the word "cat", etc. In the rest of the paper, we will use an alternative notation $d^i_n(x)$, which will denote the output of the letter detector that is selective to the letter whose class is the same as the class of the i^{th} letter of the n^{th} dictionary word [1]. One should keep in mind that the symbol $d^i_n(x)$ does not represent the probability that the section of the pattern over which the detector is positioned represents part of the n^{th} dictionary word but that the section represents a specific letter.

Instead of associating a section of the pattern with a letter from the alphabet, our goal is to estimate the confidence with which the section can be viewed as part of a word of a given class. Once we design a network that can give us that estimate, we can use it to "scan" the whole pattern or some sections of the pattern. The confidence that the pattern represents an object of a given class will then be calculated as an average confidence over all local regions that have been investigated.

The second layer units, the simple units, receive inputs from letter detectors and provide the connections between different letter detectors. The connections among the units of the first three layers of the network are illustrated in Fig. 2. We denote by g the weights that connect the letter detector d^i_n to the surrounding letter detectors. Let us denote by x_i the location of the center of the letter detector d^i_n with respect to some reference point. Then the weight $g^{ij}_n(x_i, x_j)$ connects the letter detector positioned over the location x_i with the letter detector positioned over the location x_j. This weight represents the probability of detecting the j^{th} letter of the n^{th} dictionary word at location x_j given that the i^{th} letter of the n^{th} dictionary word that has already been detected at location x_i. One of the simplest ways for approximating pairwise probabilities $g^{ij}_n(x_i, x_j)$ is to first find a distribution of widths for each letter from the alphabet and then from single letter distributions calculate nearest neighbor pairwise probabilities. Knowing the nearest neighbor location estimates, it is then easy to propagate them and find the pairwise probabilities between any two letters of every dictionary word.

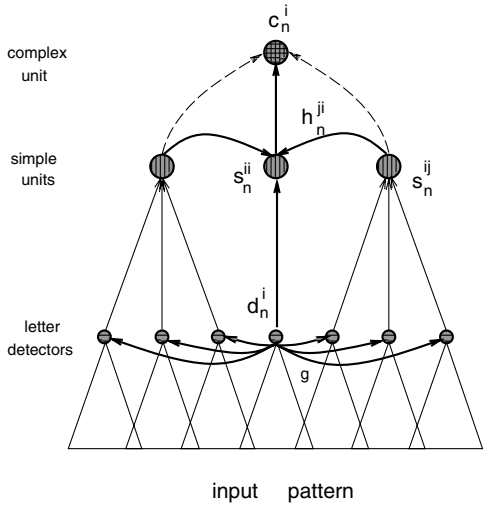

Figure 2: This figure illustrates the connections among the units of the first three layers of the network. The first layer consists of units (NNs) with localized and overlapping receptive fields. The second layer units, the simple units, combine the outputs of the units within their receptive fields (NNs) with the contribution coming from the central unit, d^i_n, through the weights g. The simple units then in turn influence the central unit s^{ii} through the weights h that carry contextual information. The output of the complex unit c^i represents a level of confidence with which the region of the pattern over which the central unit is positioned can be associated with n^{th} dictionary word.

[1] The indices i and j will be used to number the letters within a dictionary word.

A simple unit located above the letter detector d_n^i is called the central unit and the simple units around the central unit are called the surrounding units. Every simple unit is class dependent and the region where it expects to see the letter to which it is selective is called the *receptive field* (RF). A surrounding unit, of the i^{th} central unit, combines the contextual expectation ($g_n^{ij}(x_i, x_j)$) and the information supplied by the input ($d_n^j(x)$), and chooses the *location* of the letter by finding the maximum of its input elements weighted by the "expectation" weights

$$s_n^{ij}(x_i, x_j) = \max_{x_j \in RF} [g_n^{ij}(x_i, x_j) d_n^j(x_j)]. \quad (1)$$

Since the weight $g_n^{ii}(x_i, x_i) = 1$ it follows that $s_n^{ii} = d_n^i(x_i)$ meaning that the activation of the central unit s_n^{ii} is the same as the activation of the letter detector $d_n^i(x_i)$. The fact that a simple unit *chooses* one letter from its receptive field means that given the location of the central unit, x_i, simple units segment a pattern (from the point of view of the central letter). In the rest of the paper, we will simplify the notation and use $s_n^{ij} = s_n^{ij}(x_i, x_j)$ keeping in mind that each simple unit contains information about the location of the letter it has selected.

In order to include global (contextual) information into processing of the central unit, the surrounding units are connected to the central unit through the weights h. We denote by the symbol h^{ji} the weight that connects j^{th} unit to the i^{th} unit. The outputs of all the surrounding units are used for calculating the activation of the complex unit

$$c_n^i(\vec{x}) = d_n^i(x_i) \sum_{j=1, j\neq i}^{L_n} h^{ji} s_n^{ij}, \quad (2)$$

where $d_n^i(x_i)$ is the probability of detecting the i^{th} letter of the n^{th} dictionary word (at location x_i), L_n represents the number of letters in the word and $\vec{x} = (x_1, ..., x_{L_n})$ is a particular configuration of the locations of the letters that are selected by the simple units.

We can think of the processing of a complex unit as transforming the estimate that a section of the pattern represents a letter of specific class to the estimate that the section is part of an object of a specific class. The term $d_n^i(x_i)$ is the probability that the local section of the pattern over which the letter detector is positioned represents a letter from the alphabet and the second term (the sum) represents the contextual influence coming from the "rest" of the pattern. The weights h^{ji} allow the surrounding units to have different importance in influencing the processing of the central units and can be obtained from the training data. In our implementation applied to handwriting recognition the contributions of all the surrounding units are the same $h^{ji} = h^{kl}$ and in order to ensure that the contextual influence for every central unit is in the same range of values, the weights h^{ji} satisfy the equation $\sum_{j=1, j\neq i}^{L_n} h^{ji} = 1$.

The outputs of the simple units can be combined in a different way (as shown using dashed lines in Figure2), resulting in the following equation

$$c_n^i(\vec{x}) = d_n^i(x_i) \sum_{j=1, j\neq i}^{L_n} h^{ji} s_n^{ij} + \sum_{j=1, j\neq i}^{L_n} s_n^{ij}. \quad (3)$$

According to this equation, the output of the complex unit c_n^i can be interpreted as the level of confidence with which the region covered by the receptive fields of the simple units is associated with an object of a certain class. The simple unit c_n^i in this interpretation represents a word from the point of view of one letter.

Notice that in this description the activation of the central unit ($s_n^{ii} = d_n^i$) is combined with contextual information coming from *all* the surrounding units (the first sum on the right) whereas the activation of each surrounding unit (s_n^{ij}) contains a contribution (g_n^{ij}) from only the central letter. Since the influence of the central letter on the surrounding units depends on the distance of the unit from the central letter, the reliability of the estimate of the surrounding unit that is further away progressively decreases with its distance from the central unit. In the rest of the paper, we will use the activation of the complex unit as defined by Eq. (2).

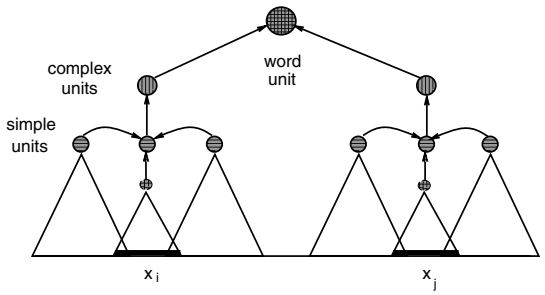

Figure 3: The outputs of the complex units are supplied to the common word unit.

The final layer of the network consists of the word units. These units receive inputs from complex units

$$w_n(\vec{x}) = \sum_{i=1}^{L_n} c_n^i(\vec{x}), \quad (4)$$

as illustrated in Figure 3. From Eq. (4) we see that the activation of the word unit is a function of specific segmentation $\vec{x}$ (locations of the letters selected by the complex units). Therefore, the locations of the complex units shown in Figure 3 present just one possible choice for positioning them over the pattern. The goal of the recognition algorithm is to find the optimal configuration of their positions such that Eq. (4) is maximized.

4 Recognition Process

In many recognition systems in use today, recognition is treated not as a process but as a one-time comparison between the stored templates and the given pattern. In addition, each region of the input is given the same importance and is processed in the same manner and with the same resolution.

An important characteristic of human recognition, on the other hand, is that it is an active process of exploring patterns and discovering features. Our eyes constantly move, probe and analyze different regions of the input at different times, and our brain integrates this information into the perception of an object.

Inspired by these properties of human perception, we have constructed an algorithm for selection and integration of spatially distributed features and for pattern classification. As opposed to the classical neural network approach, where the whole pattern is supplied to the network which then classifies it as one of the memorized objects, our network investigates local regions of the pattern by repositioning central units over the pattern emulating the process of saccadic eye movements. At each point in time all the central units are positioned over the central letter while the spatial arrangement of the receptive fields of the simple units, with respect to each other, remains fixed.

In the beginning of the recognition process, the system focuses attention on the most interesting region of the pattern. Assuming that there is no prior knowledge of what the pattern represents, "interesting" is defined as the region that elicits the highest activation in the letter detector that is positioned over the region. At this point, the region is associated with one letter from the alphabet and this letter becomes the central letter. The system then positions all of its central units over the central letter. The central letter, through the activation of the letter detector above it, now affects the processing of the surrounding units (through the weights g) and they "see" a different pattern than before the central letter was detected. Each surrounding unit of a given complex unit selects one letter from its receptive field and in turn influences the processing of the central unit. At the end, one complex unit among all complex units (of all the word units) wins, and the region of the pattern over which the central unit is positioned is associated with one dictionary word.

The system then chooses the next interesting region on which to saccade. However, this time the landscape of the activations of the letter detectors is changed, due to the presence of the discovered central letter. Therefore, the next target letter (on which the system will saccade) combines in itself the contextual expectations from the central letter (top-down information) and the information coming from the input (bottom-up information). One can imagine that with each saccade the landscape of activations becomes increasingly more complex due to contextual influences of discovered letters. Recognition becomes an active process of searching for specific features at expected locations and confirming or replacing the initial hypothesis of the object's class.

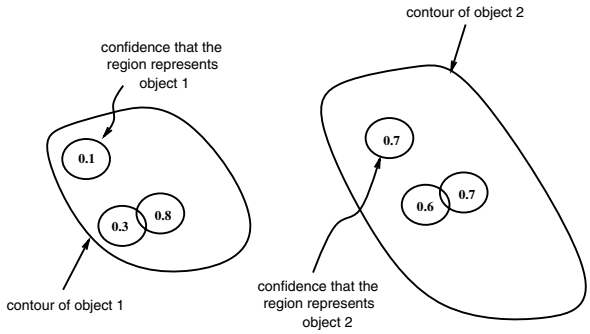

Figure 4: Two different partial coverages of the pattern, each associated with a different object. A number in each of the local regions represents a level of confidence that the region is part of a given object.

Recognition can be viewed as a process in which with each saccade a new region is discovered and assigned a level of confidence that it represents part of a given object. If we define a collection of local regions that have been discovered as a coverage of the pattern, then the goal of recognition is to find a maximal coverage value such that the sum of activations of the covered regions is maximal, Figure 4. Notice however that each configuration of selected features, or each segmentation of the pattern in terms of features, leads to different activations of the local regions. Since the correct segmentation of the pattern always results in higher coverage value and vice versa, segmentation and recognition can

be viewed as the two sides of the same process.

In many situations, the coverage of the pattern is much smaller than the pattern itself. The number of regions that the system investigates depends on the pattern and it is easy to calculate the sufficient condition for the termination of the recognition process. Let us denote by P the collection of regions that the system has explored and by the R the rest of the pattern. Furthermore, let us assume that at this point the pattern is classified as one of the memorized objects (e.g. an object of the n^{th} class). If the total activation of all the letter detectors within the region R is smaller than the activation of the complex units already activated (within the region P)

$$\sum_{x \in R} d(x) \leq \sum_{x_i \in P} c_n^i(x_i), \tag{5}$$

then the recognition process can be safely terminated since investigating the rest of the pattern would not result in different classification of the pattern. However, further investigation of the rest of the pattern could increase or decrease the confidence with which the pattern is associated with the object.

5 Results and Conclusions

In this work, we presented a neural network-based architecture that overcomes the limitations of classical NNs when applied to problems of pattern segmentation. We showed that an array of independent, spatially distributed local NNs can be successfully used for simultaneous segmentation and recognition of complex patterns when the NNs are connected through the weights that carry contextual information.

We also described an algorithm in which pattern recognition is treated as an active process of pattern exploration. The algorithm is inspired by human perception and emulates saccadic eye movements. In addition, it allows parallel processing of information and provides robust classification even when the number of missing features is large.

In comparison to HMMs our approach has several advantages. Unlike HMMs where the recognition proceeds in one direction (usually left to right), our algorithm can explore a pattern in any order depending on the relative importance of the features within the pattern. The algorithm starts with the most salient features and dynamically selects a target feature based on contextual expectations. In contrast to HMMs in which the Viterbi algorithm "covers" the whole pattern in order to find the optimal sequence, our algorithm does not have to explore the whole pattern in order to correctly classify it and in some instances the classification can be made after only few saccades.

We have applied our system to a database of 100,000 on-line words compiled by David Rumelhart [7]. Compared to the dynamic programming-based post-processor, and results reported by Rumelhart [7], our recognition performance is significantly better, and for some writers the relative error rate is reduced by more than 30% [4]. Similarly, we demonstrated that an HMM-based recognition system can be successfully replaced by our model [5]. From the computational point of view, we believe that one of the main advantages of our approach over the HMMs is with respect to duration modeling [5]. In most HMMs, only first or second order dependencies are assumed. Although explicit duration HMMs model data more accurately, the computational cost of such modeling is high. Our model, on the other hand, can easily model arbitrarily complex pair-wise probabilities without an increase in the computational complexity.

References

[1] R. E. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, Princeton, NJ, 1961.

[2] Y. Bengio, Y. LeCun, C. Nohl, and C. Burges. Lerec: A NN/HMM hybrid for on-line handwriting recognition. *Neural Computation*, 7:1289–1303, 1995.

[3] H. Bourlard and C. Wellekens. Links between hidden Markov models and multilayer perceptrons. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:1167–1178, 1990.

[4] P. Neskovic and L. Cooper. Neural network-based context driven recognition of on-line cursive script. In *7th International Workshop on Frontiers in Handwriting Recognition*, pages 352–362, 2000.

[5] P. Neskovic, P. Davis, and L. Cooper. Interactive parts model: an application to recognition of on-line cursive script. In *Advances in Neural Information Processing Systems*, 2000.

[6] L. Rabiner. Tutorial on hidden markov models and selected applications in speech recognition. *Proc. IEEE*, 77:256–286, 1989.

[7] D. E. Rumelhart. Theory to practice: A case study – recognizing cursive handwriting. In E. B. Baum, editor, *Computational Learning and Cognition: Proceedings of the Third NEC Research Symposium*. SIAM, Philadelphia, 1993.

[8] M. Schenkel, I. Guyon, and D. Henderson. On-line cursive script recognition using time delay neural networks and hidden markov models. *Machine Vision and Applications*, 8:215–223, 1995.