Handwritten Zip Code Recognition with Multilayer Networks

Y. Le Cun, O. Matan, B. Boser, J. S. Denker, D. Henderson,
R. E. Howard, W. Hubbard, L. D. Jackel and H. S. Baird
AT&T Bell Laboratories, Holmdel, N. J. 07733

Abstract

We present an application of back-propagation networks to handwritten zip-code recognition. Minimal preprocessing of the data was required, but the architecture of the network was highly constrained and specifically designed for the task. The input of the network consists of size-normalized images of isolated digits. The performance on zipcode digits provided by the U.S. Postal Service is 92% recognition, 1% substitution, and 7% rejects. Structured neural networks can be viewed as "statistical methods with structure" bridging the gap between purely statistical and purely structural methods.

1 INTRODUCTION

A widely accepted approach to pattern recognition, and in particular to handwritten character recognition, is to divide the classification process into a feature extraction, followed by a classification. The feature extractor usually contains most of the problem-dependent information, and is rather specific to the problem at hand. It requires most of the design effort, and determines the performance to a large extent. The classifier, on the other hand, often incorporate a trainable module, and contains little *a priori* knowledge about the task.

The main point of this paper is to show that large connectionist systems (or neural networks) trained with back-propagation (BP) can be applied to real image-recognition problems without a large, complex feature-extraction stage requiring detailed engineering. However, expecting good performance without having to specify any knowledge about the task (relying exclusively on learning) is wishful thinking. Previous work performed on simple digit images [5] showed that the architecture of the network strongly influences generalization performance. Not surprisingly, good generalization can only be obtained by designing a network architecture that contains a certain amount of *a priori* knowledge about the problem. The basic design principle is to minimize the number of free parameters that must be determined by the learning algorithm, without overly reducing the computational power of the network. This principle increases the probability of correct generalization because it results in a specialized network architecture that has a reduced entropy (see [7] and references therein). On the other hand, some effort must be devoted to designing appropriate constraints into the architecture, but this only requires general, high-level knowledge about shape recognition.

The benefit of constraining the architecture is that the network can be directly fed with images, rather than feature vectors. This demonstrates the ability of BP networks to deal with large amount of low-level information. The network effectively synthesizes appropriate local-feature extractors as part of the learning process. The resulting system can be viewed as intermediate between statistical, and structural methods, although it resembles the former more than the latter.

2 ZIPCODES

The problem of handwritten zipcode recognition is an interesting research subject because of its great practical value, and because the large body of literature devoted to the subject makes it a good benchmark for testing new methods [9].

The database we used consists of 9298 segmented numerals digitized from handwritten zipcodes that appeared on U.S. Mail envelopes passing through the Buffalo, N.Y. post office. Examples of such images are shown in figure 1. Since the zipcodes were obtained from standard mail pieces, they were written in a totally unconstrained fashion, by many different people, using a great variety of sizes, writing styles and instruments, with widely varying levels of care. This zipcode database was supplemented by a set of 3349 printed digits coming from 35 different fonts. The training set consisted of 7291 handwritten digits plus 2549 printed digits. The remaining

An early version of this paper was published in reference [9].

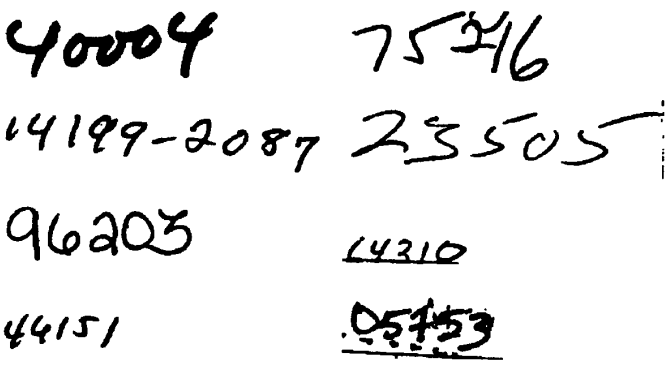

Figure 1: Examples of original zipcodes from the testing set.

2007 handwritten, and 700 printed digits were used as the test set. The printed fonts in the test set were different from the printed fonts in the training set. One important feature of this database, which is a common feature to all large databases, is that both the training set and the testing set contain numerous examples that are ambiguous, unclassifiable, or even misclassified.

3 PREPROCESSING

Acquisition, binarization, location of the zipcode on the envelope, and segmentation of the zipcode into individual digits were performed by Postal Service contractors [10]. Further segmentation, notably to eliminate extraneous marks, were performed in our laboratory. Segmentation of totally unconstrained digit strings is an extremely difficult problem, and several ambiguous characters in the database are the result of mis-segmentation (especially broken 5's) as can be seen in figure 2.

The segmented digits vary in size, but are typically around 40 by 60 pixels. The size of the characters is then normalized to 16 by 16 pixels using a linear transformation. This transformation preserves the aspect ratio of the character, and is performed after extraneous marks in the image have been removed. Because of the linear transformation, the resulting image is not binary but has multiple gray levels, since a variable number of pixels in the original image can fall into a given pixel in the target image. The gray levels of each image are scaled and translated to fall within the range -1 to 1. Note that *no* skeletonization was performed. The printed digits were artificially generated using a stochastic model of the printing process and of the acquisition system.

4 THE NETWORK

The remainder of the recognition is entirely performed by a multi-layer neural network. All of the connections in the network are adaptive, although heavily constrained, and are trained using back-propagation. This is in contrast with our earlier work [2] where the first few layers of connections were chosen by hand and were not subject to learning. The input of the network is a 16 by 16 normalized image and the output is composed of 10 units: one per class. When a pattern belonging to class i is presented, the desired output is $+1$ for the ith output unit, and -1 for the other output units.

Back-Propagation networks are composed of several layers of interconnected elements arranged in a feed-forward architecture: connections can only go from lower layers to higher layers. Each element resembles a "soft" linear classifier, computing a weighted sum of its input, and transforming this sum through a non-linear squashing function (usually a sigmoid function such as tanh). Learning is performed by iteratively modifying the weights on each connection so as to minimize an objective function. A popular objective function is the mean squared error between the actual output of the network and a desired output. Minimizing the objective function is performed by a gradient descent procedure which requires to compute the gradient of the objective function with respect to connection weights. Back-Propagation is just an efficient way to compute this gradient. It was popularized by [8] but similar procedures were proposed earlier in various contexts for various purposes [4].

A naive approach to our task would use a large, fully-connected (unstructured) backpropagation net, where all the units in a layer are connected to all the units in the following layer(s). This approach has severe deficiencies. If the number of weights is kept reasonably small, the network cannot

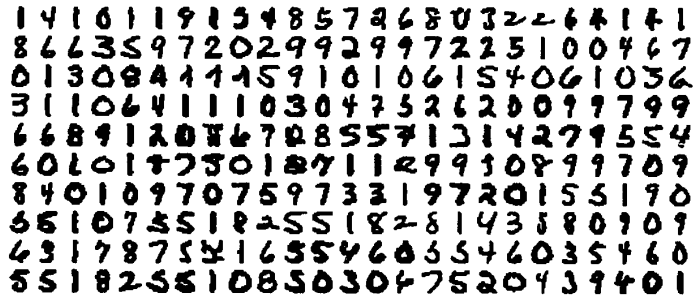

Figure 2: Examples of normalized digits from the testing set.

even learn the training set accurately. On the other hand, if it is made big enough to learn the training set, the excessive number of parameters causes overfitting, with devastating effect on the generalization performance. Structuring the network can solve this conflict, as we show in the following.

One alternative to a fully connected net is of course a locally connected net. The design of the connection pattern must be guided by our knowledge about shape recognition. Because there are well-known advantages to performing shape recognition by detecting and combining local features, our network has only *local connections* in all but the last layer. Furthermore, salient features of a distorted character might be displaced slightly from their position in a typical character, or the same feature can appear at different locations in different characters. Therefore a feature detector that is useful on one part of the image, is likely to be useful on other parts of the image as well. Specifying this knowledge we can be performed by forcing a set of units, located at different places on the image, to have identical weight vectors. The outputs of such a set of neurons constitute a *feature map*. A sequential implementation of this would be to scan the input image with a single neuron that has a local receptive field, and store the states of this neuron at corresponding locations in the feature map. This operation is equivalent to a convolution with a small size kernel, followed by a squashing function. The process can be performed in parallel by implementing the feature map as a plane of neurons that *share* a single weight vector [1]. That is, units in a feature map are constrained to perform the same operation on different parts of the image (see figure 3).

An interesting side-effect of this *weight sharing* technique, already described in [8], is to reduce greatly the number of free parameters, since a large number of units share the same weights. In addition, this builds a certain level of shift invariance into the system. In practice, multiple feature maps, extracting different features types from the same image, are needed. The learning algorithm is not significantly modified by the weight sharing technique.

The idea of local, convolutional feature maps can be applied to subsequent hidden layers as well, to extract features of increasing complexity and abstraction. Interestingly, higher level features require less precise coding of their location. Reduced precision on the position is actually advantageous, since a slight distortion or translation of the input will have reduced effect on the representation. Thus, each feature extraction in our network is followed by an additional layer which performs a local averaging and a subsampling, reducing the resolution of the feature map. This layer introduces a certain level of invariance to distortions and translations. The resulting architecture is a "bi-pyramid": The loss of spatial resolution in the feature maps (due to subsampling) is partially compensated by an increase in the number of feature types. This is reminiscent of the Neocognitron architecture [3], with the notable difference that instead of using unsupervised learning, we use backpropagation learning, which we feel is more appropriate to this sort of classification problem.

The network architecture, represented in figure 4, is a direct extension of the ones described in [5; 6]. The network has four hidden layers respectively named H1, H2, H3, and H4. Layers H1 and H3 are shared-weight feature extractors, while H2 and H4 are averaging/subsampling layers.

Although the size of the active part of the input is 16 by 16, the actual input is 28 by 28 to avoid boundary problems. H1 is composed of 4 groups of 576 units arranged as 4 independent 24 by 24 feature maps. These feature maps will be designated by H1.1, H1.2, H1.3 and H1.4. Each unit in a feature map takes its input from a 5 by 5 neighborhood on the input plane. As described above, corresponding connections on each unit in a given feature map are constrained to have the same weight. In other

[1] A better name for a feature map would be "regiment" since all units are controlled by the same kernel.

words, all of the units in H1.1 uses the same set of 26 weights (including the bias). Units in another map (say H1.4) share *another* set of 26 weights. Layer H2 is an averaging/subsampling layer. It is composed of 4 planes of size 12 by 12. Each unit in one of these planes takes inputs on 4 units on the corresponding plane in H1. Receptive fields do not overlap. All the weights are constrained to be equal, even within a single unit, except the bias. Therefore, H2 performs a local averaging and a 2 to 1 subsampling of H1 in each direction. Layer H3 is composed of twelve 8 by 8 feature maps. As before, these feature maps will be designated as H2.1, H2.2 $\cdots$ H2.12. The connection scheme between H2 and H3 is quite similar to the one between the input and H1, but slightly more complicated because H3 has multiple 2-D maps. Each unit receptive field is composed of one or two 5 by 5 neighborhoods centered around units that are at identical positions within each H2 maps. Of course, all units in a given map are constrained to have identical weight vectors. Layer H4 plays the same role as layer H2, it is composed of 12 groups of 16 units arranged in 4 by 4 planes.

The output layer has 10 units and is fully connected to H4. In summary, the network has 4635 units, 98442 connections, and 2578 independent parameters. This architecture was derived using the OBD technique [7] starting from a previous architecture [6] that had 4 times more free parameters.

5 RESULTS

After 30 training passes the error rate on the training set (7291 handwritten plus 2549 printed digits) was 1.1% and the MSE was .017. On the whole test set (2007 handwritten plus 700 printed characters) the error rate was 3.4% and the MSE was 0.024. All the classification errors occurred on handwritten characters (no error were made on the printed characters).

In realistic applications, substitutions are traded for rejections. Our rejection criterion was that the difference between the activity levels of the two most-active output units should be larger than a threshold. The best percentage of rejections on the complete test set was 5.7% for 1% substitution error. On the handwritten zipcode set only, the result was 9% rejections for 1% substitutions, and 6% rejections for 2% substitutions. About half the substitutions were due to segmentation problems. Most remaining errors were due to erroneous assignment of the desired category, or low-resolution effects.

To reduce the number of errors due to the low-resolution of the normalized images, a new network with a 20x20 pixel input was tested. Except for the size of the various feature maps, the architecture was identical to the previous one. The network was trained for about 20 passes through the 7291 handwritten zipcode digits. The performance was then 9% rejects for 1% substitutions. The network was then further trained for 9 passes on 36300 additional handwritten digits (not coming from zipcodes), and retrained for 4 passes on the handwritten zipcodes. The performance on the 2007 zipcode test set improved to 7% rejects and 1% substitutions.

Figure 5: Atypical data. The network classifies these correctly, even though they are quite unlike anything in the training set.

It is interesting to note that the learning takes only a few passes through the training set. Even though a second-order version of back-propagation was used, we think this can be attributed to the large amount of redundancy present in real data. A complete training session (30 passes through the training set plus performance measure on the test set) takes about 3 days on a SUN SPARCstation 1 using the SN2 connectionist simulator [1]. In fact, after about a dozen passes the performance improves only marginally, and the learning can be stopped. After training, a recognition (including size normalization) is performed in about 1.5 second on a SUN 3 with FPA.

In some experiments, the four kernels of the first layer were *initialized* to Sobel-like edge detectors (horizontal, vertical and two diagonals). They were of course allowed to evolve during training. This did not significantly improve the performance of the system, but made the interpretation of the internal state much easier.

After successful training, the network was implemented on a commercial Digital Signal Processor board containing an AT&T DSP-32C general purpose DSP chip with a peak performance of 12.5 million 32-bit floating point multiply-adds per second. The DSP operates as a coprocessor in a PC connected to a video camera. The PC performs the digitization, binarization and segmentation of the image, while the DSP performs the size-normalization and the classification. The overall throughput of the recognizer including image acquisition is 10 to 12 characters per second and is limited mainly by the size normalization step. The size normalization is performed in about 60ms, while the network classification is performed in less than 30ms.

6 CONCLUSION

Back-propagation learning was successfully applied to handwritten zipcode recognition. The network had many connections but relatively few free parameters. The bi-pyramidal architecture and the constraints on the weights were designed to incorporate geometric knowledge about shape recognition. Because of this architecture, the network could be trained on normalized images without requiring a predetermined feature extractor. The network was able to synthesize a hierarchy of appropriate feature detectors.

Because of the redundant nature of the data and because of the constraints imposed on the network, the learning time was relatively short considering the size of the training set. The final network of connections and weights obtained by back-propagation learning was readily implementable on commercial digital signal processing hardware. Throughput rates, from camera to classified image, of more than 10 characters per second were obtained.

At first glance. multi-layer nets resemble purely statistical techniques. However, specifying knowledge by constraining the architecture introduces some high-level structure in the process, thereby restricting the set of implicit "rules" that the network can generate. Neural networks heavily rely on learning, so they require large training sets. On the other hand, their adaptability is very well suited to problems with high variability and/or noise. Additionally, the amount of task-specific information needed is minimal; the same system can be retrained to recognize other symbols with no modification. Preliminary results obtained on alphanumeric characters confirm this fact, and show that the method can be readily extended to larger tasks.

The main limitation of our approach comes from the necessity to normalize the size of the pixel image, which is quite an expensive operation. However, recent progress in the implementation of neural-net chips suggest that "brute force" solutions are possible. The raw speed of the chips allows to run several networks in parallel at various scales and positions on the region of interest. We do not believe that our approach can be applied to very complex objects with wide variation of scale and rotation, even if these objects have little shape variability, but it seems very well suited to handwritten characters.

Acknowledgments

We thank the US Postal Service and its contractors at SUNY Buffalo for providing us with the zipcode database.

References

[1] L.-Y. Bottou and Y. Le Cun. *SN2: A Simulator for Connectionist Models.* Neuristique SA, Paris, France, 1989.

[2] J. S. Denker, W. R. Gardner, H. P. Graf, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, H. S. Baird, and I. Guyon. Neural network recognizer for hand-written zip code digits. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 1, pages 323–331, Denver, 1988, 1989. Morgan Kaufmann.

[3] K. Fukushima and S. Miyake. Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position. *Pattern Recognition*, 15:455–469, 1982.

[4] Y. Le Cun. Learning processes in an asymmetric threshold network. In E. Bienenstock, F. Fogelman-Soulié, and G. Weisbuch, editors, *Disordered systems and biological organization*, pages 233–240, Les Houches, France, 1986. Springer-Verlag.

[5] Y. Le Cun. Generalization and network design strategies. In R. Pfeifer, Z. Schreter, F. Fogelman, and L. Steels, editors, *Connectionism in Perspective*, Zurich, Switzerland, 1989. Elsevier.

[6] Y. Le Cun, L. D. Jackel, B. Boser, J. S . Denker, H. P. Graf, I. Guyon, D. Henderson, R. E. Howard, and W. Hubbard. Handwritten digit recognition: Applications of neural net chips and automatic learning. *IEEE Communication*, pages 41–46, November 1989.

[7] Yann Le Cun, J. S. Denker, and S. Solla. Optimal brain damage. In David Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2, (Denver, 1989), 1990. Morgan Kaufman.

[8] 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*, volume I, pages 318–362. Bradford Books, Cambridge, MA, 1986.

[9] C. Y. Suen, editor. *Frontiers in Handwriting Recognition.* CENPARMI, Concordia University, Montréal, 1990.

[10] C. H. Wang and S. N. Srihari. A framework for object recognition in a visually complex environment and its application to locating address blocks on mail pieces. *International Journal of Computer Vision*, 2:125, 1988.

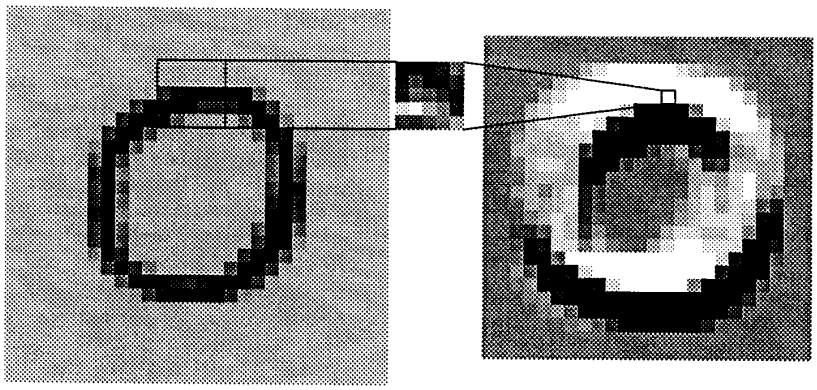

Figure 3: Input image (left), weight vector (center), and resulting feature map (right). The feature map is obtained by scanning the input image with a single neuron that has a local receptive field, as indicated. White represents -1, black represents +1.

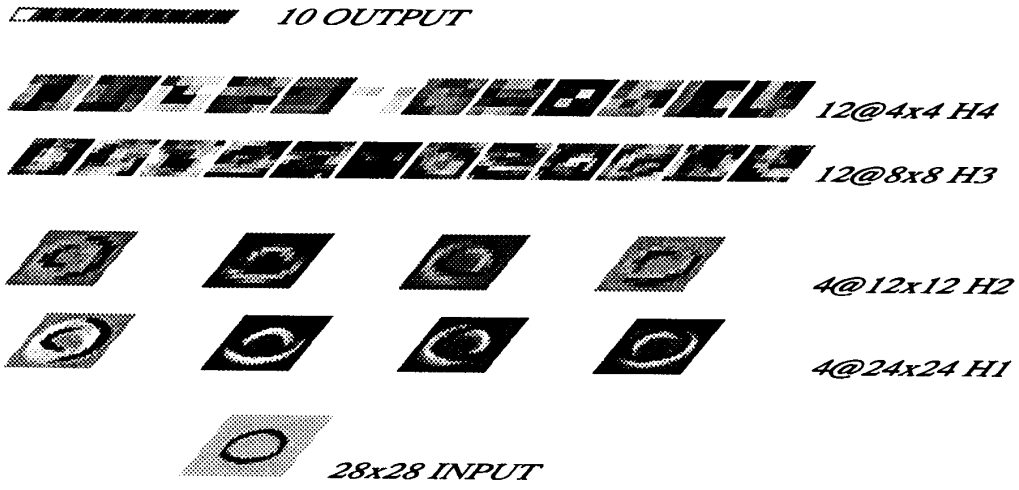

Figure 4: Network Architecture with 5 layers of fully-adaptive connections.