Handwritten Digit Recognition with Pattern Transformations and Neural Network Averaging

Juan Manuel Alonso-Weber, M. Paz Sesmero,
German Gutierrez, Agapito Ledezma, and Araceli Sanchis

Computer Science and Engineering Department
Universidad Carlos III de Madrid
Avenida de la Universidad 30 Leganés 28911, Madrid (Spain)
jmaw@ia.uc3m.es, {msesmero,ggutierr,ledezma,masm}@inf.uc3m.es

Abstract. Recently there has been a considerable improvement in applications related with isolated handwritten digit and letter recognition supported on the use of deep and convolutional neural networks and other combinations which make use of ensemble averaging. The proposal of the present work is based on a relatively modest sized Neural Network trained with standard Back Propagation and combined with a set of input pattern transformations. Applying ensemble averaging on the trained Neural Networks gives an encouraging error rate of *0.34%* measured on the MNIST dataset.

Keywords: Artificial Neural Networks, Back Propagation, Ensembles, MNIST, Handwritten Digit Recognition.

1 Introduction

Some recent works on Neural Networks applied to handwritten character recognition show an interesting improvement driven by the use of new neural models. Convolutional Neural Networks and Deep Neural Networks are both rather complex structures, which allow reaching a highly respectable performance measured on the popular MNIST Database [1][2]: 0.4% [3] and 0.35% [4]. Combining these architectures with committees or integrating them with ensemble-like structures allows to further improve down to 0.27% [5] or even 0.23% [6]. Using committees with a traditional MLP displays an error rate of 0.39% [7]. Other interesting works which are based on different approaches reach an error rate of 0.40% [8] and 0.32% [9] respectively.

The present work, derived from [10][11], shows a promising approach which advocates for the use of the standard Back Propagation learning algorithm with a relatively modest sized Multilayer Perceptron (MLP). A specific alternative pattern deformation is combined with other usual transformations (displacement and rotation), with an input size reduction and an additive input noise schedule. This helps to avoid local minima and stalling during the learning process.

The outcome is a creditable mean error rate of 0.46% tested on the MNIST Database. Applying ensemble averaging on a collection of trained Neural Networks helps to reduce the final error rate down to 0.34%.

2 Data Processing

The MNIST Database contains 70000 digitized handwritten numerals distributed in ten different classes. The whole dataset is divided into 60000 images for training purposes, and the remaining 10000 are reserved for the test set. The graylevel values of each pixel are coded in this work in the [0,1] interval, using a 0 value for white pixels and 1 for black ones.

An important point for managing a high performance in the learning process is the construction of a useful training set. The 60000 different patterns contained in the MNIST database can be seen as a rather generous set, but evidence shows that the usual learning algorithms run into serious trouble for about one hundred or more of the test set samples [10]. Therefore, some strategy is needed in order to increase the training set cardinality and variability. Usual actions comprise geometric transformations such as displacements, rotation, scaling and other distortions. Here a specific set of transformations is combined with an image size reduction and an additive input noise schedule. In this work displacements and rotations are combined with an alternative deformation procedure that yields rather good results.

A problem with which the Back Propagation algorithm tackles is the relative high input dimensionality for the original 28x28 sized digits. Using downsized images helps to reduce the error rate in a small amount. Therefore, a second version of both the training and test sets are generated where each pattern is downsized through interpolation to 20x20 pixels.

Each digit is randomly shifted zero, one or two pixels both in the horizontal and in the vertical axis. The final performance is rather sensible to the probability distribution of the different displacements. Finding an *optimal* probability distribution is a cumbersome task. An interesting possibility is to design different displacement schemas in order to reduce the error correlation of the trained networks, which in turn can induce an improvement with the ensemble averaging procedure. This is shown further on in the Experimental Results Section.

The most important transformation relies on the so called deformation, which involves pulling or pushing each of the four image corner pixels in a random amount along the vertical and horizontal axis. The rest of the pixels are proportionally displaced simulating an elastic behaviour. This leads to a combination of partial stretching and/or compression of the image. Fig. 1 illustrates this process. For the full sized images the displacement interval of the corner pixels is [-5, +5] (distance measured as pixels). For the 20x20 sized images, the best results are achieved with displacements in the order of [-4, +4] pixels. In parallel with the deformation, a rotation is applied around the image center selecting a random angle between -0.15 and +0.15 radians. For technical reasons, the deformation and the rotation need to be computed in an inverse way.

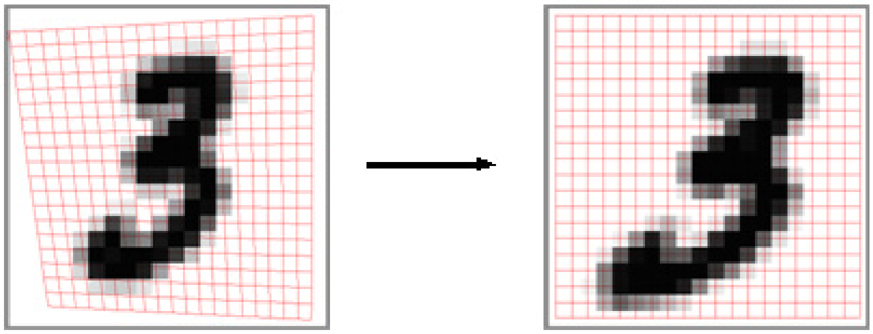

Fig. 1. The deformation is shown as an inverse mapping, at the left the original digit, at the right the deformed one

Finally, noise is added to each image previous to the training process. Noise injection has been extensively related with other techniques such as *weight decay*, and is known to provide a better generalization under certain circumstances [12][13]. Here, best results are obtained with a specific variant of input noise addition that uses annealing: starting with high noise rates that are lowered at a small amount after each learning cycle, and ending with a zero noise rate.

Including the descending input noise schedule improves the MLP precision, on behalf of a longer learning process. A noiseless training requires about 500 cycles, whereas adding the input noise extends the learning up to 1000 cycles. In this circumstance, convergence is tempered down towards the end of the noise schedule. Usually few improvements are achieved during the 100-200 last cycles, and virtually none after the noise scheme is extinguished. The relation between these parameters is $R=N_0/T_{max}$, where T_{max} stands for the number of training cycles, R is the noise reduction value and N_0 is the initial noise value. Using an initial noise value of $N_0=1.0$ adds to each pixel a uniform random value from the interval [-1, +1].

The noise addition and the geometrical transformations are applied to each pattern for each Back Propagation training cycle. Hence, the MLP sees each pattern just once.

The construction of an ensemble with a set of trained Neural Networks is performed with the averaging of the networks outputs which is a simple but effective procedure [14]. For a better performance, it is desirable that the errors committed by the MLPs should be uncorrelated, i.e. the networks should be at most *precise* and *diverse* [15]. Inducing diversity through some random process, for example, training networks with different weight initialization is a correct procedure [14] but has a limited effectiveness [16]. A higher diversity can be accomplished through a guided design of the ensemble [17][18] or inducing additional differentiation in the training process [16], for example modifying the input space. Here both approximations are tried out: a) with a ranked selection methodology, and b) with a set of displacement schemas.

3 Experimental Setup

All the experimentation is built up around training at first a collection of Multilayer Perceptrons, which are afterwards used to apply and evaluate the proposed ensemble averaging.

The MLPs have a fixed size by default: $784{\times}300{\times}200{\times}10$, for the full sized image database. The output class is coded in the usual way as a one-out-of-n schema. For the downsized $20{\times}20$ images the only change is for the input layer, which needs 400 input units. The training process is performed using online Back Propagation, where all patterns are presented to the input, structured in cycles. In this case, each pattern is processed previously, applying the above mentioned geometrical transformations combined with the noise addition. The weight initialization is restricted to the [-0.3, +0.3] interval, and the Learning Rate is set to 0.03.

A subset of the training patterns (10000 out of 60000) is randomly removed for validation purposes. This validation set can be used in several ways during the neural network training facing the posterior ensemble averaging: at first, for determining the stopping point of the learning process, and secondly as a criterion for establishing a ranking inside a set of trained neural networks.

As already stated, including the descending input noise schedule improves the final MLP precision, at the cost of a longer lasting learning process. As a rule, the annealing scheme lasts $T_{max}=1000$ cycles, and 100 noiseless cycles are added at the final stage in order to ensure a stable convergence. The initial noise value is $N_0=1.0$, and the descending noise rate $R=0.001$.

The training process of the MLPs was performed on Intel Core i7 and equivalent Xeon processors. Each processor allows to train up to 8 MLPs in parallel without a noticeable loss of performance. Given the MLP size, the size of training set and the needed cycles, the whole learning process lasts about 20 hours for the 20x20 images, and 24 hours for the full sized images.

4 Experimental Results

This section presents the experiments performed at first with the full sized images, and then with the downsized images that achieve a slightly better performance.

At first, a set of 90 MLPs are trained with 50000 images from the MNIST database, leaving 10000 randomly chosen digits for validation. Applying ensemble averaging on the whole MLP set gives an error rate of 0.39%. The drawback of this procedure is that adding or deleting members of the ensemble can vary the results within a range. Leaving out one member selected at random varies the final output between 0.38% and 0.41%.

Following the idea behind the statement that "*many could be better than all*" [17][18], a methodology for selecting then MLPs for the averaging procedure is proposed: validation errors are used in order to establish a ranking for the trained MLPs. The 20 best ranked MLPs are distributed into four subsets named *p, q, r* and *s*, where *p* contains the five neural networks that perform best on the validation set, and *s* the worst. Several ensembles are then built starting with the best subset (*p*), and progressively adding the subsets *q*, *r* and *s*. Table 1 shows the Test Errors committed for these ensembles. Also shown is the evolution of the Mean Test Error for the selected MLPs, which is lower for the better ranked ones. Experiments performed on various MLP sets with different cardinalities suggest that using different seeds for weight initialization derives in a limited differentiation, i.e. the individual MLPs have highly correlated errors. The ensemble averaging tends to acquire the best performance with nine to fifteen members.

In order to establish a reference for evaluating the heuristic, another procedure for building ensembles is included: four *K*-sized ensembles (*K*=5, 10, 15, 20) whose members are randomly selected on 1000 trials from the whole MLP set (i. e. 90 members). This allows establishing a mean value for reference. The results in Table 2 show that this value converges steadily to 0.39%. The ranked selection methodology gives a slightly better performance at 0.36%.

Table 1. Test Errors (in %) for the ensembles built with the progressive addition of the MLPs with the best validation values

		Ranked MLPs				**All**
	(sub)set:	**P**	**p,q**	**p,q,r**	**p,q,r,s**	**90**
MLPs	Mean	0.488	0.485	0.495	0.492	0.51
	Min	0.46	0.45	0.45	0.45	0.43
	Max	0.52	0.53	0.61	0.61	0.61
Ensembles		0.38	**0.36**	**0.36**	0.40	0.39

Table 2. Test Errors (in %) for 1000 ensembles built with *K* randomly selected MLPs

		Randomly selected MLPs				**All**
	K **=**	**5**	**10**	**15**	**20**	**90**
Ensembles	Mean	0.406	0.397	0.394	0.393	0.39
	Min	0.33	0.33	0.33	0.33	-
	Max	0.49	0.47	0.45	0.45	-

Whereas the MLPs trained on the original MNIST database achieve a mean error rate of 0.51%, using the *20×20* downsized images allows for a slightly lower 0.46%. The following experiment shows that using these MLPs for building ensembles indeed improves the results.

Although the use of the continuous random deformations generates a training set with a virtually unlimited number of patterns, in practice, leaving out a fraction of the original pattern set for validation purposes leads to a descent in performance, especially with the *20x20* sized image set. Therefore, in the following experiments the whole original training set without any geometrical transformations is used for validation purposes. The drawback is that this *particular* validation set does not allow using the ranked selection method on the trained MLPs: the number of validation errors seems to have low relation with the behaviour of the MLPs on the test set in the averaging procedure.

In order to increase the diversity between the trained MLPs, different displacement schemas are established: each image is shifted along the vertical and horizontal axis combined with the remaining deformations, prior to the training process. Each schema determines a different probability distribution of the possible displacements (shown in Figure 2 and Table 3).

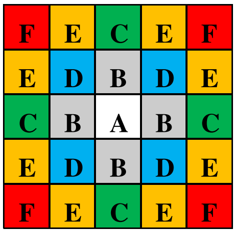

Fig. 2. Different displacement schemas. All the image pixels are displaced the same distance and in the same direction from position A to position types B/C/D/E/F with a certain probability (shown in Table 3). Destination A implies no displacement.

Table 3. Probability distribution of the displacement schemas

	Displacement Schema					
Dest.	**D1**	**D2**	**D3**	**D4**	**D5**	**D6**
A	0.43	0.27	0.22	0.20	0.16	0.11
B	0.57	0.49	0.45	0.32	0.41	0.30
C	-	0.16	0.22	0.32	0.05	0.15
D	-	0.08	0.11	0.16	0.31	0.20
E	-	-	-	-	0.08	0.18
F	-	-	-	-	-	0.05

For this experiment 10 Neural Networks for each displacement schema were trained (on *20×20* images). The Mean Test Errors for the six groups varies between 0.465% and 0.496% (shown in Table 4). Averaging each MLP group provides a decrease in the error rates that varies between 0.36% and 0.41%. The full ensemble contains 60 members and displays an error rate of 0.34%. For a more precise result, ensembles are generated with the random deletion of 1 or 2 members of each displacement schema (on 1000 trials), giving a mean error rate of 0.345% and 0.348% respectively.

Table 4. Ensembles built with six MLP sets trained each with a different displacement schema

	Displacement Schema					
	D1	**D2**	**D3**	**D4**	**D5**	**D6**
MLPs	0.487%	0.468%	0.469%	0.465%	0.485%	0.496%
Ensembles	0.39%	0.36%	0.36%	0.36%	0.41%	0.36%
Full Ensemble	**0.34%**					

Averaging all the MLPs trained with the six different displacement schemas shows a lower error rate than those committed by the best ensemble based on an individual displacement schema.

5 Conclusions

This work shows that training traditional Neural Networks with the standard Back Propagation algorithm in combination with an averaging procedure can provide fairly low error rates that lie not too far away from other leading results that rely on rather more complex neural models. The key issues are the generation of an adequate training pattern set, and the use of an additive input noise schedule. For an increased benefit in the ensemble averaging procedure, both a ranked selection methodology and a set of displacement schemas have been presented.

Acknowledgement. This work has been supported by the Spanish Government under projects TRA2011-29454-C03-03 and TRA2011-29454-C03-02.

References

1. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998)
2. LeCun, Y., Cortes, C.: THE MNIST DATABASE of handwritten digits, `http://yann.lecun.com/exdb/mnist/`
3. Simard, P.Y., Steinkraus, D., Platt, J.C.: Best practices for convolutional neural networks applied to visual document analysis. In: Seventh International Conference on Document Analysis and Recognition, vol. 1, pp. 958–963 (2003)
4. Ciresan, D.C., Meier, U., Gambardella, L.M., Schmidhuber, J.: Deep, Big, Simple Neural Nets for Handwritten Digit Recogntion. Neural Computation 22(12), 3207–3220 (2010)
5. Ciresan, D.C., Meier, U., Gambardella, L.M., Schmidhuber, J.: Convolutional Neural Network Committees for Handwritten Character Classification. In: International Conference on Document Analysis and Recognition, vol. 10, pp. 1135–1139 (2011)
6. Ciresan, D., Meier, U., Schmidhuber, J.: Multi-column Deep Neural Networks for Image Classification. In: IEEE Conf. on Computer Vision and Pattern Recognition, CVPR, pp. 3642–3649 (2012)
7. Meier, U., Ciresan, D.C., Gambardella, L.M., Schmidhuber, J.: Better Digit Recognition with a Committee of Simple Neural Nets. In: International Conference on Document Analysis and Recognition, vol. 1, pp. 1250–1254 (2011)
8. Ranzato, A.M., Poultney, C., Chopra, S., LeCun, Y.: Efficient Learning of Sparse Representations with an Energy-Based Model. In: Advances in Neural Information Processing Systems 19, NIPS 2006 (2006)
9. Cruz, R., Cavalcanti, G., Ren, T.: Handwritten digit recognition using multiple feature extraction techniques and classifier ensemble. In: International Conference on Systems, Signals and Image Processing (IWSSIP), pp. 215–218 (2010)

10. Sesmero, M.P., Alonso-Weber, J.M., Gutiérrez, G., Ledezma, A., Sanchis, A.: A new artificial neural network ensemble based on feature selection and class recoding. Neural Computing and Applications 21(4), 771–783 (2010)
11. Alonso-Weber, J.M., Sanchis, A.: A Skeletonizing Reconfigurable Self-Organizing Model: Validation Through Text Recognition. Neural Processing Letters 34(1), 39–58 (2011)
12. Matsuoka, K.: Noise Injection into Inputs in back-propagation learning. IEEE Transactions on Systems, Man, and Cybernetics 22(3), 436–440 (1992)
13. An, G.: The Effects of Adding Noise During Backpropagation Training on a Generalization Performance. Neural Computation 674, 643–674 (1996)
14. Opitz, D., Maclin, R.: Popular ensemble methods: An empirical study. Journal of Artificial Intelligence Research 11(1), 169–198 (1999)
15. Dietterich, T.: Ensemble methods in machine learning. In: Multiple Classifier Systems, pp. 1–15 (2000)
16. Brown, G., Wyatt, J., Harris, R., Yao, X.: Diversity creation methods: a survey and categorisation. Information Fusion 6(1), 5–20 (2005)
17. Perrone, M.P., Cooper, L.N.: When networks disagree: Ensemble methods for hybrid neural networks. In: Mammone, R.J. (ed.) Neural Networks for Speech and Image Processing, ch. 10. Chapman-Hall (1993)
18. Sharkey, A.J.C., Sharkey, N.E., Gerecke, U., Chandroth, G.O.: The 'Test and Select' Approach to Ensemble Combination. In: Multiple Classifier Systems, pp. 30–44 (2000)