Evaluating an ANN based pattern recognizer

M. Haak, S.Bos, S.Panic

Department of Man Machine Interaction
Delft University of Technology
Delft, NL.

***Abstract* – In this report the pattern recognition abilities of an Artificial Neural Network (ANN) will be explored and evaluated. Experiments have been conducted in which ANN's have been trained and tested on their ability to classify noisy patterns. These experiments focused on examining variables of interest belonging to one of the groups 'Network architecture', 'Network behavior' and 'Network use'. For a classification task where skewed or rotated test patterns are not considered, both a single layer and a dual layer ANN can perform optimally. However the former requires half the amount of neurons (15) of the latter to do so. Variables from the group network use, such as with how much epochs the network is trained, are of crucial influence to the resulting network performance.**

***Index Terms* – Neural networks, pattern recognition, artificial intelligence.**

I. Introduction

Optical character recognition (OCR) is the interpretation of images that represent text, and identify the characters that the text contains. This report will investigate and evaluate the performance of an ANN in a simple character recognition task. After introducing the given OCR problem, the section on experimental approach will elaborate more on the ANN algorithm's variables of interest, as well as the baseline configuration against which the performance of an ANN will be evaluated. The results of the experimentation with the baseline variables will then be presented and discussed, from which a few conclusions will be drawn. This will lead to insights and understanding of how, and how well ANN's can recognize digital characters.

II. Problem statement and evaluation

The given problem definition is to build an ANN based pattern classifier which is capable of recognizing the characterset as defined in Fig. 1. Under the variation of variables of interest, the ability of the network to learn to correctly classify test patterns needs to be investigated. For example the consequences that adding a ratio of noise to the patterns used for training or testing has on the resulting network's performance can be evaluated during the experiment.

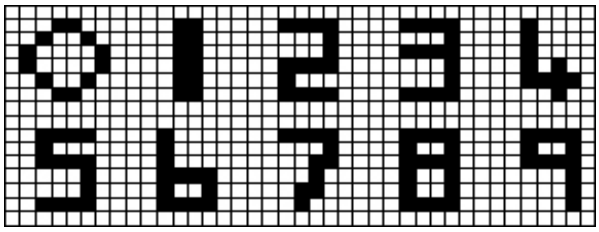

Fig. 1: The 8 by 8 pixel character set that the ANN must recognize.

III. Experimental approach

To investigate the influence that the ANN characteristics have on its learning and classification ability, a number of variables of interest have been considered in our approach. These variables can be grouped together according to their relation to Network architecture, Network behavior and Network use.

A Experimental environment

To facilitate the creation and use of ANN's, MATLAB R2008b [1] is used in combination with its standard Neural Networks Toolbox (NNTOOL) component. This cobination provides the experimenters with an environment that is more than suitable for investigating different configurations of ANN based pattern recognizers.

B Experimental procedure

To facilitate the experiment a program has been written using MATLAB and NNTOOL . The program first generates all the required training and test patterns, after which two neural networks are set up. The program then repeatedly asks the user to choose an action from a list of predefined actions. These actions could be to train a network, to simulate a network and display the results of the simuation, or to display the generated training or testing patterns that were used. The program allows for easy manipulation of standard variables such as the number of neurons, the number of test patterns to generate, or the maximum amount of noise for these patterns.

C Variables of interest

The variables of interest can be classified in three groups. The first group of variables determine the network architecture, such as the number of neurons and layers in the network. The second group of variables determine the network behavior such as activation functions and training

algorithms used during execution of the algorithm. The final group of variables of interest determine network usage, such as the amount and quality of patterns used for training and testing, and the number of epochs used during the training.

The following variables of interest belong to the Network architecture group:

- Consider one or two hidden layers for the network
- Consider different a amount of neurons for each of the hidden layers

The following variables of interest belong to the Network behavior group:

- What is the activation function of the neurons? Consider two types of activation functions for all the neurons: an hyperbolic tangent sigmoid transfer function (tansig) and a log-sigmoid transfer function (logsig).
- Consider two different training algorithms: scaled conjugate gradient backpropagation (trainscg), and gradient descent with adaptive learning rule backpropagation (traingda).

The following variables of interest belong to the Network use group:

- The total number of training epochs used. Consider training the network repeatedly instead of just once.
- The total number of training patterns used. Consider using multiple input patterns to train for classifying a single class.
- The amount of noise in the training patterns. Consideer using multiple input patterns for each class, where each of the inputs is subject to an increasing amount of noise
- The amount of noise in the testing patterns. Consideer using multiple input patterns for each class, where each of the inputs is subject to an increasing amount of noise

D Comparison with a baseline configuration

To investigate the influence that each of the variables of interest has on the performance of the resulting network, a baseline configuration has been identified for the network. This baseline configuration specifies how to configure a single or multilayer network that, when given around 500 epochs to train with supervision, can almost perfecly classify noisy test patterns.

The single layer network has 30 neurons, while the multilayer network has 20 and 10 on each of its hidden layers. The neurons on a hidden layer have the tansig activation function, and neurons on an output layer have the logsig activation function. The traingda method is used to train the network. Furthermore, for training the network to recognize each of the character classses, for each of the

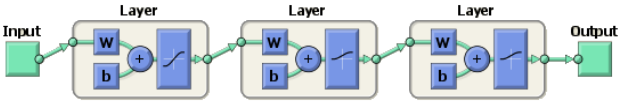

Fig. 2: An Artificial Neural Network with two hidden layers and one output layer. Each layer is defined by its connection bias, activation function and the number of neurons it contains

classes 4 training patterns are used with no noise. The test patterns also consist of 3 patterns per character class, also with no noise. Both single layer and dual layer neural networks with this baseline cconfiguration have been able to perfectly identify noisy test patterns when trained for around 500 epochs (or until the error gradient is smaller than 1e-10).

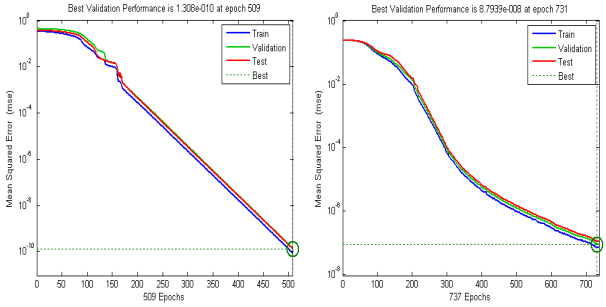

Fig. 3: Performance of an ANN with baseline configuration. The left graph shows the performance of the single layer ANN, while the right graph shows the performance of a multilayer ANN.

IV Results

In this section the results of the experiments with the previously mentioned variables of interest will be mentioned. Starting from a baseline configuration the consequences for the ANN performance of varying each of the mentioned variables has been evaluated.

A One or two hidden layers

The ANN is able to learn to classify noisy character patterns with both one and two hidden layers. If all other variables are left at their baseline value, the single layer ANN performs a bit better in overall. Where the single layer network is capable of perfectly classifying 30 out of 30 test patterns, the multi layer network reaches at most 29 out of 30 correct classifications. Under the given baseline configuration (with an equal amount of neurons for the entire ANN) the performance of the multilayer ANN appears to be much less on average.

B Number of neurons and their distribution

Table I lists the results for varying the number of neurons of a baseline ANN with one hidden layer, and evaluating the resulting performance. A distinction in performance is made between the noise used in the

generated training and testing patterns. The performance was evaluated with patterns that contain 8% and 6% noise for the training and testing patterns, and no noise at all. In the case of no noise in the patterns used, the performance of the ANN was perfect and unaffected by a drastic reduction of neurons. In the case of noise in the patterns, there was a varying reduction of performance. Only with the minimum amount of 5 neurons did the performance on the task of classifying noisy patterns suffer dramatically as only 14 out of 30 test patterns were correctly classified.

TABLE I
EFFECT OF NEURON COUNT ON THE CORRECT CLASSIFICATION OF 30 TEST PATTERNS BY AN ANN WITH A SINGLE HIDDEN LAYER. THE NETWORK WAS TRAINED AND TESTED WITH PATTERNS WITH AND WITHOUT A RATIO OF NOISE (8% FOR TRAINING, 6% FOR TESTING)

	Neurons					
	30	25	20	15	10	5
No noise	30	30	30	30	30	30
Noise	30	26	29	30	28	14

Under variation of the number and distribution of neurons in a multi layer ANN, the performance shows much more variation. Table II shows that for non-noisy patterns, the performance of the ANN varied dramatically. The optimal baseline peformance is maintained with less neurons on the hidden layers (15-10). The performance with noisy patterns degrades under all deviations from the basaline configuration.

TABLE II
EFFECT OF NEURON COUNT ON THE CORRECT CLASSIFICATION OF 30 TEST PATTERNS BY AN ANN WITH TWO HIDDEN LAYERS. THE NETWORK WAS TRAINED AND TESTED WITH PATTERNS WITH AND WITHOUT A RATIO OF NOISE (8% FOR TRAINING, 6% FOR TESTING)

	Neurons on each of the hidden layers					
	20 - 10	24 - 6	15 - 10	15 - 5	10 – 5	5 – 5
No noise	30	21	30	6	15	18
Noise	30	22	24	15	9	3

C *Activation function for the neurons*

The effects of changing the activation function for the neurons has been examined, by varying the activation functions between a hyperbolic tangent sigmoid (tansig) and a log sigmoid (logsig). These activation functions can be assigned to either a hidden layer or an output layer of the ANN. The effects on the baseline performance are different for networks that use a single layer and ones that use a multi layer.

Table III shows that for single layer ANN's, there is a significant performance drop when the tansig function is used for neurons on both the hidden layer and the output layers. The three other combinations of tansig/logsig do not degrade the capabilities of the ANN to learn to classify the input patterns correctly. Table IV shows that for multi layer ANN's, there is a performance drop when the logsig function is used for neurons on the hidden layer, while the tansig function is used for neurons on the output layers. The three other combinations of tansig/logsig do not degrade the capabilities of the multi layer ANN to learn to classify the input patterns correctly. For both these exceptional combinations in Table III and Table IV, during the training of the network it was noted that the error gradient was not monotone decreasing, as seemed the case during at least some other trainings that were observed.

TABLE III
EFFECT OF NEURON ACTIVATION FUNCTION ON THE CORRECT CLASSIFICATION OF 30 TEST PATTERNS BY AN ANN WITH A SINGLE HIDDEN LAYER

		Hidden layer	
		logsig	tansig
Output layer	logsig	30	30
	tansig	30	18

TABLE IV
EFFECT OF NEURON ACTIVATION FUNCTION ON THE CORRECT CLASSIFICATION OF 30 TEST PATTERNS BY AN ANN WITH MULTIPLE HIDDEN LAYERS

		Hidden layer	
		logsig	tansig
Output layer	logsig	30	30
	tansig	24	30

D *Training function for the network*

The 'gradient descent with adaptive learning rule backpropagation' (traingda) training function is used in the baseline configuration of the ANN's. When given a sufficient amount of training epochs, both single and multi layer networks were able to learn to classify the test patterns.

When the scaled conjugate gradient backpropagation (trainscg) training function was used, the single layer ANN training suffered from many premature stops. The training was halted after a few epochs because one of the stop criteria has been met. The performance of the ANN on the task of classifying the test patterns was very poor, until the network was forced to train for a number of more epochs. After around 200 epochs the network would not train any more because all the stop criteria have been met. When using the trainscg training function the single layer ANN was able to correctly classify 24 out of 30 test patterns. The multi layer ANN, which did not suffer from premature

training stops and trained 'in one go', was able to correctly classify 30 out of 30 after around 200 epochs, much less then is required with the traingda training function.

E Number of training epochs for the network

The number of training epochs seems to be crucial for the ANN's ability to classify test patterns correctly. The training method provided by NNTOOL iteratively trains an ANN and performs a number of validations. Such a training step is repeated until a fixed number of validations is reached (6), until the error gradient reaches a minimum (1e-10) or until a maximum number of epochs is reached (1000). Due to these criteria it is possible for an ANN to be undertrained when the training procedure is executed.

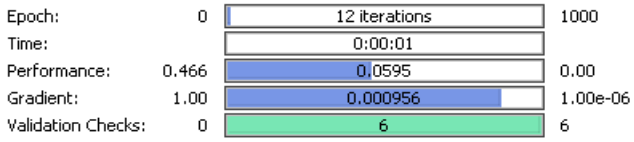

Fig. 4: Example training results of a baseline configuration. After only 12 epochs the validation stop criteria has been reached, while the minimal error gradient criteria has not been reached yet.

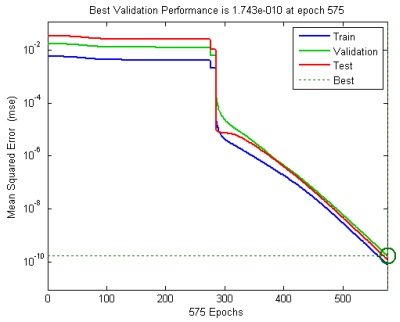

Fig. 5: A significant performance improvement after consecutive training of the ANN. In general ANN's were trained until more than 400 epochs and/or a small error rate gradient (<10^3) was reached . This approach gave the best performance results

According to the NNTOOL documentation this procedure should help with preventing overlearning of the ANN from occuring. However it can also stop prematurely and cause the exact opposite effect of underlearning. After a relatively small number of epochs the validation stopping criteria are met although the error gradient is still relatively high. In such cases the resulting ability to classify test patterns was very poor (less than 10 out of 30 at most). The training of these badly performing networks was continued until either 500 epochs were reached, or the error gradient criteria was reached. When the classification abilities of the ANN was tested again, it was almost always greatly improved, up until near-perfect classification as was the case with the baseline configuration.

The way that NNTOOL has implemented the training procedure can prove to be a barrier for training a network until the error gradient is below a certain level. During our experiments it happened often that after relatively few epochs the algorithm would not train the network any further, even though the error gradient was still relatively high. The result of this insufficient training is ofcourse a decreased performance on classifying the test patterns. This should be taken into consideration when interpreting the further test results.

F Number of training patterns used

The number of training patterns used has a drastic effect on the ability of the ANN to correctly classify the test patterns. However it should be noted that the training methods provided by NNTOOL limits how the training patterns passed on to the ANN are interpreted. They are automatically subdivided in three groups of patterns: training, validation and testing. This division is done according to a fixed ratio, of 0.6, 0.2 and 0.2 for each of the respective groups. As a result, if the network is presented with one training pattern for each character to recognize, the network will not be trained with all these patterns and thus not be able to recognize all characters. For the network to be trained with patterns for all 10 classes of characters, the number of training patterns presented must exceed 34.

These underlying training pattern mechanics are exposed when varying the number of training patterns presented to the ANN. Its resulting performance ranges from very bad when few training patterns are presented, to near perfect when enough training patterns are presented.

G Noise in the training and testing patterns used

In the used baseline configuration for the ANN's, both the character patterns used for training as well as testing contain no noise. In this setup the ANN must learn to match inputs of 64 precise numbers (pixels) to 10 output numbers which identify the class. This is a relatively straightforward analytical and computational task, which both the single and multi layer ANN can adapt to perfectly (30 out of 30 correct classifications).

The first variation tried was to introduce 6% noise in the test patterns, without using any noise in the training patterns. Both single and dual layer ANNs were still able to correctly classify 30 out of 30 test patterns. When the noise ratio was increased to 8%, the single layer ANN could classify 28 out of 30 patterns, and the multi layer correctly identified 29 out of 30 patterns. However, to a human the noisy patterns might also appear ambiguously and make the same mistake with detecting a character 8 when the original pattern was for the character 3 (See Fig. 6).

The ANN was also trained with 8% noisy patterns, while being tested with patterns which contain no noise.

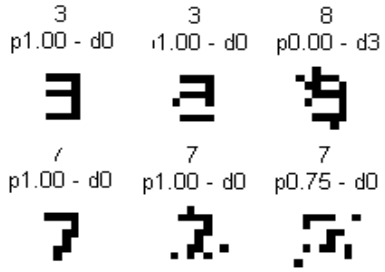

Fig. 6: 8% Noisy test patterns that were used on an ANN that was trained with patterns with no noise. One pattern of the character 3 is wrongly classified as an 8. The distance between the 3 and 8 pattern is 3 (changed pixels).

The training procedure for both the single as well as the dual layer network, consisted of more premature stops than before. It took quite some more repeated trainings to reach the baseline configuration of around 500 epochs or an error gradient of 1e-10. However in both cases the ANN was still able to perfectly classify 30 out of 30 test patterns.

The last experiment with the noise variable was to introduce noise to both the training as well as the testing patterns.The training patterns that were generated for each character class had an increasing noise ratio of up to 8%. Similarly test patterns contained up to 6% noise. When the ANN was trained and tested, the single layer variant was able to still correctly classify 30 out of 30 test patterns. The multilayer ANN performed much worse with only correctly classifying 21 out of 30 patterns. However the multilayer ANN could not be easily trained with a focus on error gradient, since the training algorithm would stop because the validation criteria have been met. Thus it is more difficult to sufficiently train a multilayer ANN. There have been cases where the multilayer ANN was able to classify 29 out of 30 test patterns (see Fig. 7).

Fig. 7: Noisy test patterns that can be classified by an ANN with the baseline configuration. The single layer correctly classifies 30/30 patterns, and the dual layer 29/30.

V. Discussion and conclusion

The results of the experiment have presented some data that has been gathered. The process of generating a set of noisy patterns, training the network, and simulating and testing the network is nondeterministic. This implies that when an ANN program is executed within MATLAB, the performance results may be different with every single run. Although this is an inevitable consequence, all the experiments were done multiple times, to get an impression of average performance. When this seemed the case on at least two runs, the performance data was noted in tables I to IV. This procedure ensured that an ANN created with the specified parameters, would perform about equally well with different sets of generated (noisy) patterns.

From the results presented in the previous section, a few observations are of particular interest. The task itself is fairly limited. Although the ANN is expected to be able to deal with noise, it is not expected to deal with rotated or skewed input patterns. This simplifies the classification task, since the pattern matching only requires to test for a specific sequence of bytes in the input pattern. If classification under rotation and skew would be considered, the recognition of features (such as lines and arcs) becomes more important in the process of proper identification and classification. Since rotation and skew of test patterns are not to be considered in this assignment, the actual classification task becomes simple(r) in the computation required. This may present a bias for the ANN towards a single layer architecture versus a multi layer one. The addition of an extra layer in the network introduces a computational overhead of translating and interpreting the learned knowledge in layer 1 into values presented to the neurons on the output layer. Although using a second hidden layer brings this computational overhead, there might be advantages to having a second layer. These advantages are related to the possibilities they introduce for the ANN to learn computational solutions to the given problem. A single layer (non-recurrent) network must, in one step, make a classification based on an input pattern. The introduction of a second layer allows at least part of the neurons in the first layer to be trained to recognize features in the input patterns, such as lines and arcs. Knowledge about these features could be passed on to the second layer, which then makes a classification. Although using a second hidden layer to solve the classification problem as presented seems to be a more inefficient one (more neurons required for optimal performance than with single layer), when the ANN should also be able to classify skewed or rotated patterns, a multi layer ANN might be the preferred choice for the best performance.

The collected data on the number of neurons and their distribution shows some interesting results. With only half of the neurons as in the baseline configuration, a single layer ANN still performed optimally, both with noisy an non-noisy patterns. The performance of a dual layer ANN changed more drastically under influence of the amount and distribution of neurons. The baseline configuration proved the only configuration that performed optimally on both noisy and non-noisy patterns. A single layer ANN could perform the classification task with less neurons (and thus computationally more efficient) than a dual layer ANN.

The amount of training patterns as well as the noise in these patterns, also seem to influence the performance. Increasing the amount of noise in a pattern might decrease

the distance between classes of patterns, causing both the ANN as well as a human observer to get confused. As an example, the distance between the character class 8 and 9 is 5 pixels. So a 64 pixel pattern for the character 8 might be changed into a pattern for the character 9 by introducing noise that coincidentally changes the right pixels. A noise ratio of 8% might already cause this. So whenever a pattern 8 is presented to the ANN or a human test subject, both will have to take into account that it does not actually represent a character class 8 but could also represent a 9 but with 5 noisy pixels. A surprising result was that the ANN was able to correctly classify noisy test patterns even when trained only with non-noisy ones.

When conducting the experiments it gradually became clear that one configuration of a network might not be better than others in absolute value. This means for instance that although the baseline performance (using traingda training fuction) is optimal, the results show that using the trainscg training function) might also result in a good performance, but just less often. The number of training epochs used, how NNTOOL randomly divides the training patterns into 3 sets, and how NNTOOL decides to stop the training (after only a few epochs with a large error gradient remaining, to prevent overlearning) give a limited amount of control on these processes. This makes it hard to conclude if that particular configuration of the ANN is better or worse than an alternative, since it could just be a case of improper training.

In conclusion, for the classification task that was originally considered, a single layer ANN outperforms a dual layer ANN. However, when this classification task was a bit more realistic in that it should also be able to detect rotated or skewed characters, due to computational complexity of the required task the single layer ANN might be outperformed by a multi layer one. From the three groups 'Network architecture', 'Network behavior' and 'Network use', variables of interest belonging to the latter group seemed the most influential in the network's performance. Although a single layer ANN with 15 neurons seems to outperform a multi layer one with 10+5 neurons, these results might be the case of an insufficient training. However it is assumed that although the latter ANN might be able to perform optimally, when compared to a single layer ANN it becomes much harder and time and computation consuming to obtain these results.

References

[1] The Mathworks, “MATLAB: The language of technical computing” 2008. Available: http://www.mathworks.com/products/matlab/. [Accessed: Mar. 27, 2009].