Classification of Non-Linear-Separable Real-World-Problems Using Δ-Rule, Perceptrons, and Topologically Distributed Encoding

Martin Eldracher *†
Technische Universität München
Institut für Informatik
Arcisstr.21, 8000 München 2, Germany
e-mail: eldrache@informatik.tu-muenchen.de

1 Abstract

We describe how to solve linear-non-separable problems using simple feed-forward perceptrons without hidden layers, and a biologically motivated topologically distributed encoding for input data. We point out why neural networks have advantages compared to classic mathematical algorithms without loosing performance. The Iris-dataset from Fisher [3] is analyzed as a practical example.

Keywords: Classification, perceptrons, topologically distributed encoding, Iris dataset

2 Introduction

In this paper we examine the performance of neural classification networks dealing with real world problems. We show that neural networks can provide results comparable to mathematical methods (c.f. [10]). But in contrast to mathematical methods neural networks do not require applicability preconditions, which in most cases are not fulfilled. To clarify this we will give two examples of well known analytic methods.

The Bayesian decision theory has two severe drawbacks: the a-priori-probabilities of all possible patterns must be known and the statistical independence of patterns must be guaranteed [1]. Additional limitations arise from practical considerations because of the exponentially growing need for training patterns with increasing problem complexity. In addition noise free patterns are required [15]. Because of theoretical considerations the full accuracy of the procedure cannot be achieved, as known theoretical distributions provide insufficient approximations for the real distributions of the patterns [1, 14]. Furthermore in most cases too few patterns are known to solve the equations exactly. Even worse, to obtain good results highly overdetermined equation systems are needed [1].

Polynomial classifiers also suffer from high computational costs, if complete polynomials of a high degree are used (direct approach). By using a wrong, incomplete polynomial of low degree an existing solution might not be found [17]. Using an implicit approach implies that the same preconditions as for the Bayesian decision theory are required. Therefore all the already mentioned disadvantages for Bayesian decision theory arise [1, 17].

In contrast neural networks show some advantages. There is no need to define an often artificial distance measurement for symbolic information, e.g. colours. The mapping from pattern space to classification space doesn't need to be calculated as a transparent function, but the neural net forms its opaque mapping by itself [15]. As all trainable classifiers neural networks do not require any assumptions about specific pattern distributions without increasing the misclassification ratio [13].

The Iris-dataset consists of the measurements of the blossoms of fifty plants of each of the three species Iris setosa, Iris versicolor and Iris virginica. The flowers were found growing together in the same colony and measured by Dr. E. Anderson. Four blossom measurements are given for each flower: sepal length, sepal width, petal length and petal width [3]. It should be mentioned that there is some overlap in the distributions of Iris virginica and Iris versicolor. Thus a certain diagnosis of these two species can not be based solely on these four measurements of a single flower taken from a wild-growing plant [3]. This also prevents classification with one-layered perceptrons [16] which can only form linear classifiers [12]. The Iris-dataset was already used for a mathematical analysis [3] and in a comparison of several clustering algorithms [11] (cited from [18]).

The overlapping problem can be described as follows. A linear one-dimensional problem is divided into two intervals each of which containing all samples of one

*This is a revised version of an article, which was originally published in the *"Proceedings of the 1992 ACM/SIGAPP Symposium on Applied Computing"*, Volume II, pp1098-1104, ACM Press, ISBN: 0-89791-502-X.

†Funded by German Ministry for Research and Technology (FKZ ITN9102B).

coordinate				class
1	2	3	4	
5.7	4.4	1.5	0.4	Iris setosa
4.4	2.9	1.4	0.2	Iris setosa
5.9	3.2	4.8	1.8	Iris versicolor
6.3	2.5	4.9	1.5	Iris versicolor
6.7	3.0	5.0	1.7	Iris versicolor
6.3	2.7	4.9	1.8	Iris virginica
6.1	3.0	4.9	1.8	Iris virginica
7.2	3.0	5.8	1.6	Iris virginica

Table 1: *Each coordinate contains pseudo-XOR-problems*

class. The Iris-dataset contains examples which prevent such a division (table 1).

We call this sort of problem pseudo-XOR-problem, because like the XOR-problem it cannot be solved by a linear classificator. For an illustration of a pseudo-XOR-Problem see figure 1.

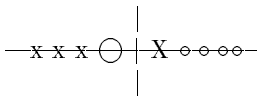


Figure 1: *A simple pseudo-XOR-problem*

The two classes, ∘ and x, are separable using the vertical line marking the place of a hyperplain (here a simple dot) as a linear classifier. Similarly ◯ and X may be separated using the same classificator. Regarding ◯ and X as special patterns of ∘ and x no linear classificator can be found but at least four intervals must be formed. Pseudo-XOR-problems can only arise regarding the collection of all patterns. Pseudo-XOR-problems probably may not be visible using a special (non-representative and/or small) set of training patterns.

The topology of our networks is the simplest possible. We use fully connected feed-forward perceptrons without hidden layers (figure 2). The units' sigmoid squashing function is approximated by three straight lines (figure 3). All networks were trained with the △-rule only.

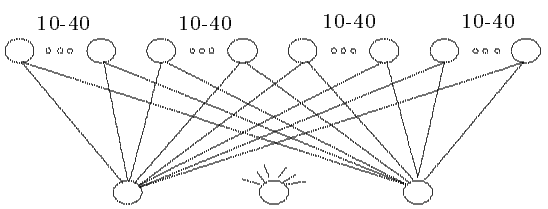


Figure 2: *The network's topology: 10-40 input units for each of the four measurements, one output unit for each of three possible classes*

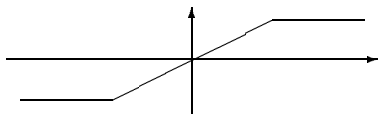


Figure 3: *Units' squashing function*

Details of the unit's model and the basic neural network simulator used for this work can be found in [19].

3 Topologically Distributed Encoding

Especially for the classification of not clearly separated classes very exact encoding of numerical values is frequently needed to find the small area where it is possible to place the classifier.

If the numerical values are encoded in single neurons using real numbers the net is dependent on the computer's arithmetic. Furthermore only one weight can be adapted per numerical value.

When encoding the numerical values using a channel model, one neuron is needed for each possible input value. This may not be feasible. Since especially real world data is normally strongly clustered, not all possible values will occur. Thus many of the units are never activated. Therefore the network's overall activity is low and the overall learning rates in Δ-rule are small. Beyond that an over-fitting on the training data is easily achieved. As a result of this the generalization with this input encoding is poor. Besides, the question arises whether pattern encodings are meaningful, which have a much higher dimension in pattern space than the pattern's degree of freedom [6]. Both encodings are biologically implausible.

The topologically distributed encoding model adopted here [5] uses the resources economically, provides sufficient changeable weights and achieves a high overall-accuracy with comparatively inaccurate encoding in the single neurons [7]. Topologically distributed encoding is biologically motivated, because it is somewhat similar to the way acoustic information is coded in the cochlea of mammals [5]. For every variable x a number of neurons ordered along a one-dimensional axis is used. To encode a value x1 the corresponding neurons are activated according to their "receptive fields" consisting of overlapping Gaussian curves. Since several neurons are always active numerical information is encoded in topologically distributed fashion (figures 4 and 5).

Using only two encoding-neurons, each having a resolution of 256 steps, real numbers between [0,10] (as in Iris-dataset) can be coded with a resolution of 0.05 us-

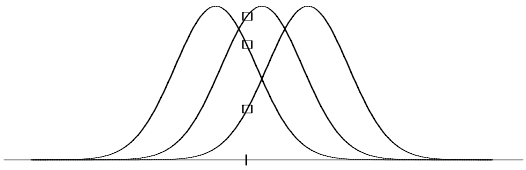


Figure 4: *Topologically distributed encoding: X-axis: numerical value, y-axis: "receptive fields" of three units (Gaussian curves) and their activation for a specific value (squares)*

ing the topologically distributed encoding with fitting parameters σ and μ for the Gaussian curves [7]. Because the Iris-dataset is discrete with step size 0.1 this appears to be enough to encode the data and solve the problem if it were linearly separable.

The more neurons are used to encode the numerical values, the better two single values can be distinguished, provided the "receptive fields" always cover the same number of centers of "receptive fields" of other neurons (figure 6 a). This results from the fact that two values are easily distinguishable whenever they cause a significantly different overall activation. Because our model uses discrete values to encode the input, this may be achieved with a neuron which is activated for one value and deactivated for the other value (figure 6 b).

4 Experiments and Results

We attempted to train the described networks in a way so that each training pattern causes full activation in the output-neuron representing the pattern's class and zero activation in the other two output-neurons. To achieve this the training patterns were presented in episodes until all patterns showed the desired output-activation. In one episode each pattern was presented to the network, but only trained if the output was not already correct.

Experiments with single neurons to code the numerical values were aborted after 800 training episodes with 75 training patterns. The system never had correctly learned more than 25 of the training patterns after an episode. This was not surprising because of the non-linear-separability.

Experiments with the channel-model didn't show good results either. As expected because of the Iris-dataset's quality (strong clusters, pseudo-XOR-problems), training was very short taking three episodes with 75 training patterns. The whole 150 patterns could be learned in ten episodes. However the overall classification failure having learned 75 patterns (showing the ability of generalization) was high (25%).

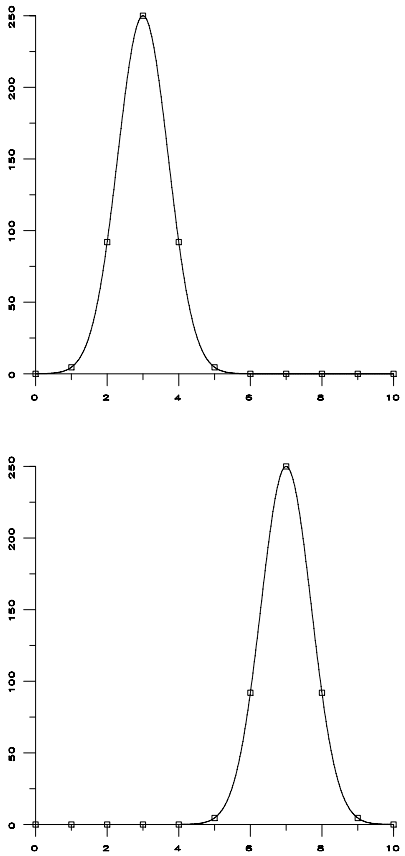
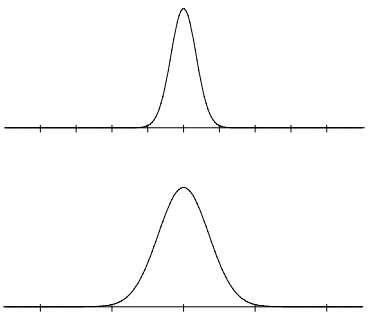


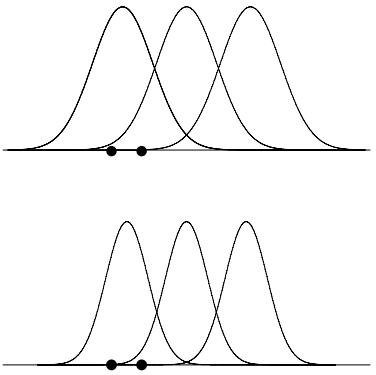
Figure 5: *Topologically distributed encoding: overall activity for encoding "3" (upper) and "7" (lower) in eleven encoding neurons coding the interval [0,10]. The Gaussian curve shows (y-axis) the activities if indefinitely many encoding neurons encode the value at the maximum. Squares show the activity for the actual encoding neurons showing maximal activity for numerical values 0,1,...(x-axis).*

Several series of experiments were done with topologically distributed coding using different numbers of coding-neurons. Each series consisted of experiments using an increasing number of training patterns (the same numbers from each class). The remaining patterns were used as test patterns and the ratio of correct classification on the whole dataset was measured (table 2).

Generally networks using less coding-neurons have to be trained with more episodes, because their few synapses must be tuned very exactly. Because of the strongly clustered data, they offer better generalization when trained with few examples (uncommon training examples hardly disturb the classification of the bigger part of the data on the wide "receptive fields"). Trying to train more patterns results in quasi exponential increase of necessary episodes until the resolution for the



a) *Different overall activations for a certain value for more (upper picture) resp. less (lower picture) encoding neurons*



b) *In the lower case, distinction of the two values is easily possible, but not in the upper case with greater* σ

Figure 6: *Effects of different number of encoding neurons (a) and different* σ *(b)*

pseudo-XOR-problems is reached. Therefore learning was aborted after 800 episodes.

Closer analysis of learning shows that the number of learning episodes until learning success was achieved jumps between "plateaus". This means after a nearly constant number of training episodes (while increasing the number of training patterns), suddenly this number increases significantly (table 2). While enlarging the learning set on "plateaus" the number of learning episodes does not increase but sometimes even decreases. The jumps are a result of a new pseudo-XOR-problem arising with the new training patterns. Taking some more training-patterns facilitates the learning task in that way, that other new patterns may fall into the same region and therefore this specific pseudo-XOR-problem is trained several times during one episode. This can reduce the number of episodes slightly. We conclude this from the fact,that some

number of learned patterns	number of episodes				number of correctly classified patterns			
	10	20	30	40	10	20	30	40
12	3	4	1	1	126	136	103	110
15	3	3	1	1	127	135	103	110
18	4	3	1	2	135	135	124	112
72	37	13	17	11	144	144	140	143
75	36	13	17	11	144	144	140	143
78	33	13	11	10	144	144	140	144
81	186	67	25	12	145	144	144	144
84	456	23	42	16	143	146	146	146
87	—	33	40	15	—	143	147	145
99	—	38	43	12	—	148	146	147
102	—	—	164	41	—	—	147	148
126	—	—	558	60	—	—	150	149
141	—	—	559	60	—	—	150	149
144	—	—	559	60	—	—	150	150

Table 2: *Results of the experiments with topolocical distributed encoding*

patterns are trained in each episode, while others are never trained. This means they can already be correctly classified from knowledge learned with other patterns. These learning "plateaus" can also be found using polynomial classifiers [17].

The "plateaus" raised the assumption that the training set might be ordered by increasing difficulty. Experiments with randomly chosen training patterns and a network with 40 coding neurons showed two results. On one hand the maximum overall misclassification percentage with 75 training patterns was always below 7%. On the other hand the number of training episodes depends significantly on the learning spot check (figure 7). Using 54 and 96 training examples yielded similar results.

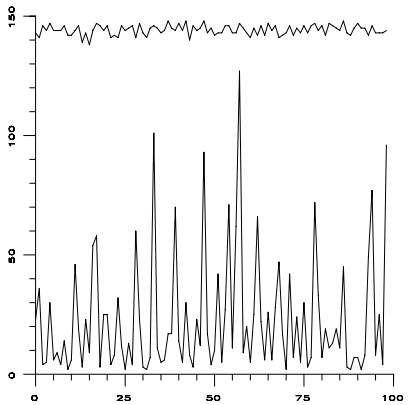


Figure 7: *Y-axis: Number of training episodes (lower curve) and correct classified patterns (upper curve) using 100 times 75 random training patterns out of the 150 dataset patterns, x-axis: number of test*

Experiments with different values for σ could not pro-

vide any insight for choosing an optimal σ_{opt}. Theoretically there would have been two possible hypotheses:

1. Greater σ results in fewer learning episodes because information is represented more distributed. This was regarded as positive because single neurons coding the values showed bad results.

2. Greater σ results in more episodes because the differences in the neurons activities become too small and, in the extreme case, all neurons are fully activated.

After having completed all our experiments it was obvious that an optimal σ could not be found with the Iris-dataset because of the pseudo-XOR-problems. As shown in figure 6 slightly different σ may provide or hinder the networks ability to solve a special pseudo-XOR-Problem, thus being able to solve the whole task. This effect does not depend on the absolute value of σ.

Though excellent classifying ratios were achieved an improvement for better use of the resources comes to mind by looking at the learned connection matrix (figure 8).

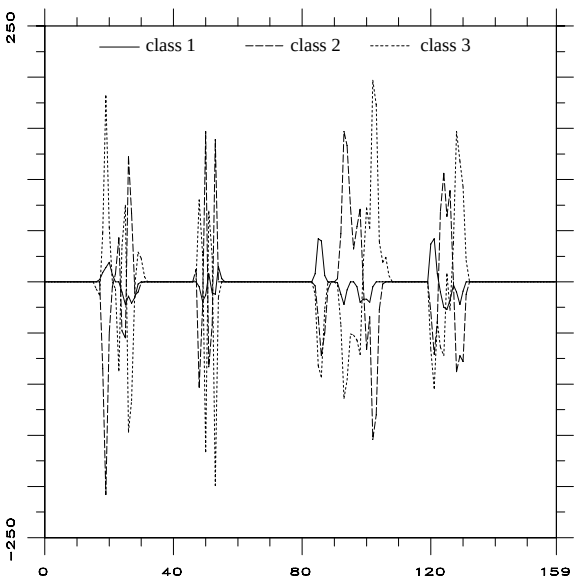


Figure 8: *Connection matrix (strength of connections) of a network with 40 encoding neurons for each of the four values after training with all patterns. X-axis: Number of input unit, y-axis: Strength of connection to output unit for the different classes*

In some parts the connections were changed through learning, in others they still had the initializing value zero. Zero-connections do not have any influence on the output-neurons' activations, therefore they are useless and may be omitted to accelerate learning. The other areas with changed weights probably need more encoding neurons to solve difficult pseudo-XOR-problems. Thus an adaptable density of encoding neurons would be desirable. This is easily achieved with a hidden layer (figure 9). The result after learning (figure 10) is a local distortion of the input space which provides both the sufficient distances (in Hamming-distance) for classification and the needed neighborhoods for generalization. For the output-neurons the hidden layer can provide an encoding as if the centers of the encoding neurons' "receptive fields" were transformed in a way to solve difficult problems through shifting density from one part of input-space to an other. In some way this equals self-organizing feature-maps [9].

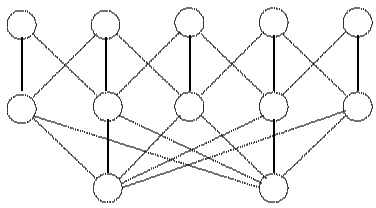


Figure 9: *Possible network's topology with an extra hidden layer*

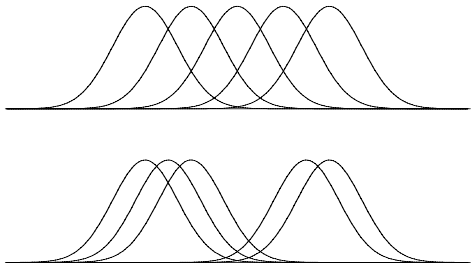


Figure 10: *Possible "locations" of coding-neurons before (upper) and after (lower) training*

5 Comparison to other Works

We intended to keep the neuron-models, network topologies and learning algorithms as simple as possible to solve a given task with minimum efforts, and allow solution of real world tasks with huge datasets in acceptable time. Besides, a system should be able to solve a whole class of problems rather than one specific problem [5]. So our system is in some way opposite to highly

specialized systems, as for example the "Neocognitron" [4].

Enforcing a certain output by learning with the Δ-rule, the human view of a problem is entrained, which in most cases is desired. Mathematical methods do not always show this feature. The average linking algorithm [8] (cited from [18]) as an example forms five clusters out of the Iris-dataset. These clusters partly contain patterns of two of the three real world classes.

Fisher [3] examined the two classes Iris setosa and Iris versicolor using mathematically exact techniques. He calculated that these two classes could be separated with a linear classifier with an misclassification ratio of only $^1/_{3000000}$, provided a nearly normal distribution for the patterns. Because of class intersections Iris versicolor and Iris virginica cannot be reliably separated using only the four measurements of the single flowers. Assuming approximately equal standard deviations and the calculated classes' centers of gravity [3], this results in a misclassification ratio of about 1.25% per class. In this work overall misclassification ratio of 2.5% was achieved, using 90 training and 60 test patterns. It should be mentioned that Fisher's calculations refer to the complete dataset while only 60% of the dataset were used to train the network.

6 Conclusion

Utilizing a sophisticated, biologically motivated input encoding of the numerical values we solved a non-linearly separable problem using a perceptron without hidden layers. This allows us to use the Δ-rule to train the networks with the double advantage of efficient learning and the guarantee of finding a possibly existing optimal solution.

The results show in practice that neural networks' classification ratios are approximately comparable with classical methods (c.f. [13][10]). Neural networks can solve real world tasks without being provided with the huge number of training examples which are necessary to solve theoretically the problem's equation system presented in a specific input space representation.

Generalization of the obtained results may be done without further ado. On one hand other, classified datasets of radar echoes and gas chromatography were treated in an equally successful manner. On the other hand nothing was known about the dataset when the work was done. Just when we were working on some theoretical aspects we incidentally found that the dataset we had been working on was the Iris-dataset.

As a whole this work shows that the capabilities of even the most simple neural nets are not yet fully exploited. Apart from the development of new techniques one should therefore use older, less sophisticated paradigms (probably enriched with some tricky ideas) because of their conceptual and computational simplicity, wherever they can offer optimal solutions. Furthermore we showed one possible reason why humans are still so much smarter than neural networks. The classification of the Iris dataset with the used resources was only possible because of the topologically distributed encoding. This encoding is also used by humans, and surely strong preprocessing allows us to solve more complex tasks with probably simpler connection structures than assumed nowadays.

7 Acknowledgements

The project, on which the work reported here is based, has been partly funded by the German Ministry for Research and Technology (BMFT) under FKZ ITN9102B. The author is solely responsible for the contents of this publication.
I want to thank the ASCOT-neural-networks-team of Kratzer Automatisierung GmbH, Maxfeldhof 5-6, 8044 Unterschleißheim, Germany, namely Dr. Hans Geiger, Dr. Lilian Bungarten, Thomas Waschulzik, Dagmar Böller, Dieter Butz, Renate Deffner, Sylvester Diehl, Klaus Eder, Ralf Kahler, Margit Kinder, Dieter Kotzick, Wolfram Stanger for helpful discussions and comments on my master thesis at Technische Universität München, which was the basis for this article. Also I want to thank my colleagues Daniel Hernández, Margit Kinder and Max Moser as well as Richard L. Kopec for very useful comments on the manuscript. Last but not least I may not forget, that this manuscript was created in the laboratory of Prof. Dr. W. Brauer, who not only gave a lot of administrative help but also gladly accepted remarks on the content. Furthermore Prof. Dr. W. Brauer was the advisor of my masters thesis [2] on which this article was based.

References

[1] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*. John Whiley Publications, New York, 1973.

[2] Martin Eldracher. Klassifikation großer Datenmengen anhand dynamisch extrahierter relevanter Merkmale mit neuronalen Netzen. Master's thesis, Technische Universität München, Institut für Informatik, 1990.

[3] R. A. Fisher. The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, VII(II):179–188, 1936.

[4] K. Fukushima. Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position. *Biologic Cybernetics*, 36:192–202, 1980.

[5] Hans Geiger. Storing and Processing Information in Connectionist Systems. In R. Eckmiller, editor, *Advanced Neural Computers*, pages 271–277. Elsevier Science Publishers B.V. (North-Holland), 1990.

[6] H. Glünder. *Invariante Bildbeschreibung mit Hilfe von Autovergleichsfunktionen.* PhD thesis, Technische Universität München, Fakultät für Elektrotechnik, 1988.

[7] Margit Kinder. Repräsentation mehrdimensionaler, funktionaler Abhängigkeiten in konnektionistischen Systemen. Master's thesis, Technische Universität München, Institut für Informatik, 1990.

[8] B. F. King. Stepwise Clustering Procedures. *Journal of the American Statistical Association*, 62:86–101, 1967.

[9] Teuvo Kohonen. Self-Organized Formation of Topologically Correct Feature Maps. *Biologic Cybernetics*, 41:42–50, 1982.

[10] Ulrich H.-G. Kreßel. The Impact of the Learning-Set Size in Handwritten-Digit Recognition. In O.Simula T. Kohonen, K. Mäkisara and J. Kangas, editors, *Artificial Neural Networks*, pages 1685–1689. Elsevier Science Publishers B.V. (North-Holland), 1991.

[11] J. E. Mezzich. *An Evaluation of Quantitive Taxonomic Methods.* PhD thesis, Ohio State University, 1975.

[12] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry.* MIT Press, Cambridge, MA, 1969.

[13] G. Nagy. State of the Art in Pattern Recognition. In *Proceedings of the IEEE, Vol.56, No.5*, pages 836–862, 1968.

[14] N. J. Nilsson. Adaptive Pattern Recognition: A Survey. In H. L. Oestreicher and D. R. Moore, editors, *Cybernetic Problems in Bionics*, pages 103–146. Gordon and Breach Science Publishers Inc., New York, London, Paris, 1966.

[15] Yo-Han Pao. *Adaptive Pattern Recognition and Neural Networks.* Addison-Wesley Publishing, Inc., 1989.

[16] F. Rosenblatt. The Perceptron – A Percieving and Recognition Automation. Technical Report 85-460-1, Cornell Aeronautical Laboratory, Ithaca New York, 1957.

[17] Jürgen Schürmann. *Polynomklassifikatoren für die Zeichenerkennung.* R. Oldenbourg Verlag, München, 1977.

[18] H. Solomon. Data Dependent Clustering Techniques. In J. van Ryzin, editor, *Classification and Clustering*, pages 155–173. Academic Press, New York, 1977.

[19] Thomas Waschulzik. Optische Mustererkennung in neuronalen Architekturen. Master's thesis, Technische Universität München, Institut für Informatik, 1988.