Data Driven Learning for Feature Binding and Perceptual Grouping with the Competitive Layer Model

Der Technischen Fakultät
der Universität Bielefeld
vorgelegt von

Sebastian Weng

zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften

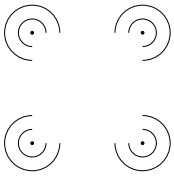
August 2005

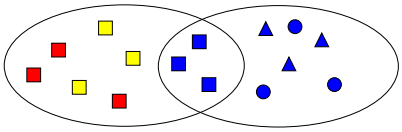
[1]

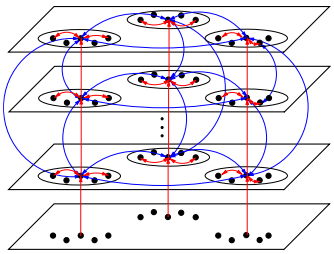
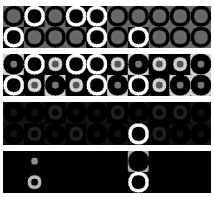
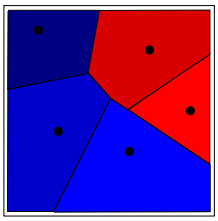
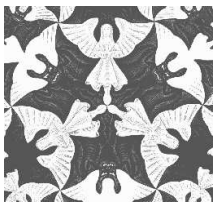
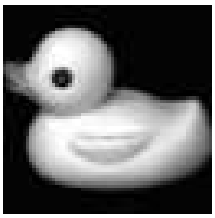
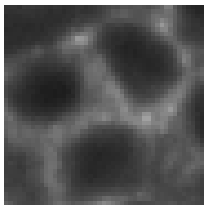
[1] Gedruckt auf alterungsbeständigem Papier „ISO 9706“.

Acknowledgement

This work was carried out in the Neuroinformatics group, headed by Prof. Dr. Helge Ritter, at the Faculty of Technology, University of Bielefeld. It was supported by the DFG grand GK-231 “Strukturbildungsprozesse“.
First, I want to thank Helge, whose challenging lectures introduced me to the field of neural networks. He provided an excellent workspace with a friendly and informal atmosphere. Helge is the architect of the different projects in the group. His ideas constitute the basic cornerstone of this work.
The theoretical fundament of this work was laid in the work of Dr. Heiko Wersing. Heiko was so kindly to shared his knowledge and experiences. His friendly and optimistic comments, as his always constructive criticism, were of extreme value for me.
Additional help came from other members of the group. Junior-Prof. Dr. Tim Nattkemper and Jörg Ontrup provided their expertise in medical image processing, texture segmentation and early implementations of the Competitive Layer Model. This manuscript is build on the foundation walls of Heiko’s, Tim’s and Jörg’s work. It was on me to construct new building blocks of knowledge on this basis. Therefore, Thorsten Twellmann gave me a useful tool for my work with his hints on Support Vector Machines, and Kai Essig gave me an insight to the world of eye-tracking.
Further, I thank Petra Udelhoven for her helping hand in organizing formal things, as the other members of the group. It was motivating to watch their advances on their projects.
Special thanks goes to Dr. Jochen Steil, who was my primary advisor during my whole time in the group. Jochen was always willing to discuss new ideas and to enrich them with his great scientific knowledge and personal experiences. I want to thank him for his patient and friendly way, which advised me, when I was building too careless on my work or when I was afraid, that the construction could break down.
Finally, Ingo Bax, Jochen Steil und Heiko Wersing read parts of this manuscript and tested it’s consistency and stability. I hope the construction is strong enough, such that other can build on it.
I could never have worked on this manuscript without my family and friends, who supported me all my life. Thanks a lot.





Contents

Chapter 1

Introduction

1.1 Scope and Goals

One of the key features of intelligent systems is the ability of perception. It translates the information of the surrounding world, given by sensory input, to an internal representation which is a precondition for developing plans, formulating goals and deciding about actions to influence the world according to these goals. Further, it gives the possibility to observe and control own actions and to evaluate their conformity with the intended goals.

One of the most impressing examples of perception lies in the area of human vision, which enables us to perceive our environment "with one gaze". We can recognize objects and complex scenes within a few hundred milliseconds [7] without any conscious effort. Thereby, our perception is very robust against difficult circumstances, like changing illumination, distances and different points of view. Further, we have the ability to detect new, yet unseen, objects and to learn concepts and categories for them to identify similar objects in new situations.

Obviously, it is an important scientific task to transfer the powerful ability of perception to artifical systems. This task motivates wide areas of research, like the fields of computer vision, image processing and pattern recognition. One branch of research in these fields is motivated from the example of the brain. Using information from neurobiology about the structure and functionality of single neurons and from neurophysiology about the connections within and between areas of the brain, the field of artifical neural networks tries to develop mathematical and algorithmic models to reproduce functionalities of the brain. Famous results from this area of research are the Perceptron the Multi Layer Perceptron, the principle of Hebbian Learning and Self-Organizing Maps. These architectures and concepts are well investigated and understood (see e.g. [2], [15], [25] for reference). However, because of their simple feed forward character, they are limited in their abilities.

To improve their abilities it is possible to extend such models by recurrent feedback connections between the neurons, which leads to the area of recurrent neural networks. Through their dynamical character these kind of networks are more pow-

erful and flexible in their abilities. On the other hand, it is also much more difficult to predict and control their behavior. Clear statements about their functionality and methods of learning can only be made for very small networks or under very restrictive constraints about the structure and connections of neurons. The probably most common example from this area is the family of associative memory networks with the famous representative of Hopfield nets [15],[19]. These networks have the property to converge to a set of stable attractor states representing a set of stored patterns, where the process of storing patterns in the weight matrix of the network is well understood and related to the principle of Hebbian Learning.
In this work, a similar recurrent neural network architecture is investigated, the Competitive Layer Model (CLM), proposed by Ritter [50]. The CLM is able to solve sensory segmentation, perceptual grouping and feature binding tasks by converging to stable attractor states that represent a consistent separation of an input set into coherent groups. The theoretical properties of the CLM were already well investigated in a number of previous studies [61], [62]. Wersing and Ritter proved the convergence of the CLM to stable states and described the dynamical behavior of the CLM through an eigenvalue analysis of it's weight matrix. Further, they showed it's application to various grouping and labeling problems and made a first suggestion for an automatic learning algorithm.
This work continues this line of research, putting the focus on the advancement of the learning process. Starting from practical observations according to the previous analysis in [61], [62], the learning algorithm is simplified. The abilities of the CLM are extended from the grouping and segmentation of data to the ability of classification, based on different grouping behaviors. The theme of this work can be summarized in the question: How deep can the process of perception be modeled with the CLM under the special aspect of learning? The approach to this question is presented in the following.

1.2 Plan of the Manuscript

The first chapter gives a short overview over the scope and content of this work in the area of perception and recurrent neural networks.
The second chapter deepens this introduction to perception processes, describing the relevance of grouping and segmentation processes for perception. It starts from some observations in psychology about natural grouping principles in human perception, formulated as Gestalt laws. Further it gives a rough overview over the wide range of segmentation algorithms and related processing stages that are applied in computer vision to implement the different Gestalt laws. Finally, it notes, that in this work all the different grouping principles are approached with a single architecture, the Competitive Layer Model, and describes, how these principles are encoded into pairwise compatibilities of elementary objects, called features.
Then chapter three describes the concrete grouping behavior of the CLM. It mainly presents the previous theoretical and practical results and summarizes the appli-

cation of the CLM and related architectures in several further works to give an overview about the present knowledge about the CLM at the beginning of this work. First the problem domain, the architecture and the notation of the CLM is introduced, giving an overview over the dynamics of the CLM and describing, how a specific grouping behavior is encoded in the weights of the CLM by pairwise compatibilities between feature representing neurons. Then the main theoretical results from the proof of convergence and the eigensubspace analysis of the CLM-dynamics are presented, arising in an algorithmic description for the simulation of the CLM grouping process. This algorithm includes a special annealing technique, that is applied to prevent suboptimal grouping results in the attractor states of the CLM. Further this chapter reviews several examples for the application of the CLM on different grouping problems and gives the original formulation of the learning method, that was suggested by Wersing [62] and is advanced in this work.
Chapter four demonstrates the theoretical knowledge about the grouping and annealing process of the CLM practically, describing the course of the CLM-dynamics for an idealized grouping problem. The observations made are discussed and lead to assumptions about the relation between the structure of the weight matrix, the annealing process, the robustness of grouping results and the speed of convergence. Experiments show, that during the annealing process the groups in the input manifest in a predefined ordering according to the structure of the weight matrix in the CLM. The assumptions and observations of this chapter provide motivations for the approaches in the following chapters.
In chapter five, a new learning algorithm for the CLM is developed, which uses the original formulation of the learning problem by Wersing, but is based on the observations, that the relevant part of the weight matrix can be formulated as a correlation matrix of special pattern vectors, constructed from goal states of the CLM. This approach shows parallels to storing a set of pattern vectors in the weight matrix of a Hopfield network. Learning is achieved by the projection of the components of the correlation matrix onto pairwise relations between the elementary features, like local distance, distance in color or distance of orientation. A similar weight matrix for a new pattern can then be constructed by projecting the pairwise relations within the new set of features backwards onto the correlation components. A control parameter for the level of segmentation is introduced to give the user the possibility to adjust the learned grouping behavior towards a rougher or finer segmentation without repeating the learning process.
The practical properties of the new learning algorithms are discussed in chapter six, where it is applied to the grouping problems of point clustering, texture segmentation, fluorescence cell image segmentation and contour grouping.
While chapter five and six show the adaptation of the CLM weights to a single grouping behavior, chapter seven shows an extension of the abilities of the CLM to classification of different objects by the competition of several object specific grouping behaviors. These behaviors can be learned by the algorithm proposed in chapter five. However, they have to be weighted suitably against each other in an additional learning phase. Chapter seven shows two approaches to this learning

phase. The first formulates a constrained optimization problem on the additional weights from the structure of the weight matrices of the different grouping principles. The second can be implemented by a variant of an on-line error correction rule similar to the perceptron learning rule.
Chapter eight describes, how the Competitive Layer Model can be extended by a model of attention, modifying the weight matrix of the CLM with an attention map. As a result of this modification, the ordering in which the groups in the input manifest during the annealing process can be changed according to the focus of attention.
Chapter nine discusses alternative implementations of the simulation algorithm. The main goal of this chapter is to reduce the computational effort of the simulation for patterns that consist of a high number of features. This effort can be divided into two parts: the precomputation of the weights in the CLM, which mainly depends on the number of features in the patterns, and the iteration of the CLM-dynamics, which depends on the number of neurons in the network and the speed of convergence of the neurons. There the key question is: How much can the effort in the first part be reduced by omitting the computation of weights without increasing the effort in the second part too much? The two approaches to this question are the computation of sparse weight matrices and the on-line computation of weights during the simulation process, which enables a parallel implementation of the CLM on distributed systems that do not have a shared memory for all activities and weights within the CLM.
Finally, chapter ten summarizes the results of this work and gives an outlook to interesting aspects of research for future works on the topic of the Competitive Layer Model.

Chapter 2

Perceptual Grouping

2.1 Human Perception and Gestalt Laws

Grouping processes play an important role for human perception, which is exemplified in the optical illusion in Fig. 2.1 a) and b), developed by Kanizsa [24]. Figure 2.1 a) consists only of black colored dots and circular arcs that surround these points. However, through the special arrangement of these components the human observer perceives a white square that clearly points out from the background, even though the two regions have obviously the same color. Interestingly, this effect vanishes, if the scheme is extended by further line segments in Fig. 2.1 b) that connect the end points of the arcs. This seems to be surprising from a naive point of view, because the new line segments explicitly describe at least a part of the outline of the white square, such that it could be rather expected that they enhance the perception of the square.

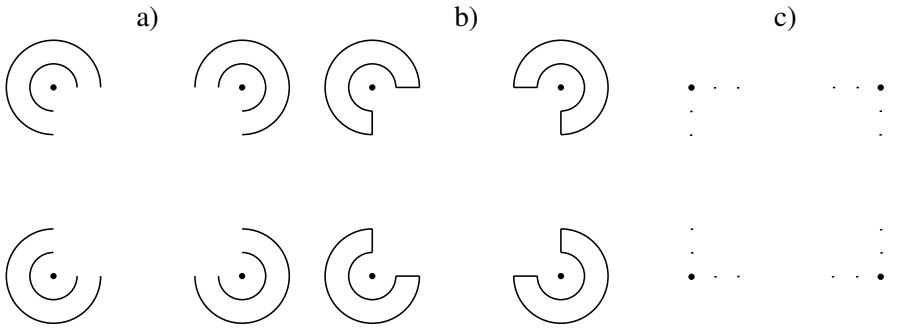


Figure 2.1: Example of an illusion in human perception [24].

Since the schemes only consist of black lines, the reason for this illusion must be somehow connected to the detection of edges and the way these edges are connected to objects or groups. In Fig. 2.1 a), it seems to be reasonable, that the arcs are perceived as coherent groups, because the detected edges along the arcs have a

continuous and smooth curvature. However, at the end points of the arcs there is a clear break in this course, which may be explained by the higher activation of point receptors or edge detectors with orientation perpendicular to the course of the arcs. Assuming, that the end points of the arcs are not perceived as part of the arcs, they form the structure, as shown in Fig. 2.1 c), which can be detected as the outline of a square. Together with some kind of fill-in-mechanisms to complete the arcs to full circles and the structure in Fig. 2.1 c) to the outline of a square, this also might give the impression of depth, imaging, that the white square occludes the circles and the background.
Through the connections of the arc end points by the additional line segments in 2.1 b) each pair of arcs at the corners of the scheme are perceived as a closed contour. This grouping prevents the break out of the end points of the arcs, because they are part of the closed contours, such that the endpoints can no longer be combined to the outline of the square.
This simple example shows the strong influence of the early processing state of grouping on the whole process of perception. Although it is not clear whether the above argumentation describes the correct biological processes in the brain, it conforms with a huge amount of experimental data investigating human vision and grouping behaviors. Based on early results of psychologists like Wertheimer [66], the school of Gestaltists observed and formulated a number of principles, called the Gestalt laws. A schematic illustration of the different Gestalt laws is presented in figure 2.2.
The Gestalt laws describe natural principles for grouping considering various modalities of contextual object properties. For example the principle of proximity only regards the local position of objects stressing that areas of high concentration of objects with small inter-object-distances form groups, while areas with low concentration and high inter-object-distances separate these groups. In contrast, the principle of continuation connects the properties of position and orientation and describes that it is preferred to build coherent contours from line segments that show a continuous and smooth course, while sharp breaks of orientation and gaps in the contours separate different groups. The law of similarity represents a whole class of grouping principles, so similarity can mean similarity by color, texture, shape or even higher order properties which are connected with the functionality of objects.
A special role plays the law of *Prägnanz*, which describes the grouping process, in the case that the application of several Gestalt laws is possible at the same time. In general it says, that always that grouping principle dominates which is most simple for the observer. However, it is not always easy to compare the different modalities of grouping to say which is the simplest. This depends as well on the actual arrangement and graduation of object properties, like color and position, as on the experience of the observer. So, there might be differences in the evaluation of similarity for color between people living in artifical terrains or tropical jungle regions. Another example might be the different levels of granularity in the differentiation of melodies and musical trained and untrained people.

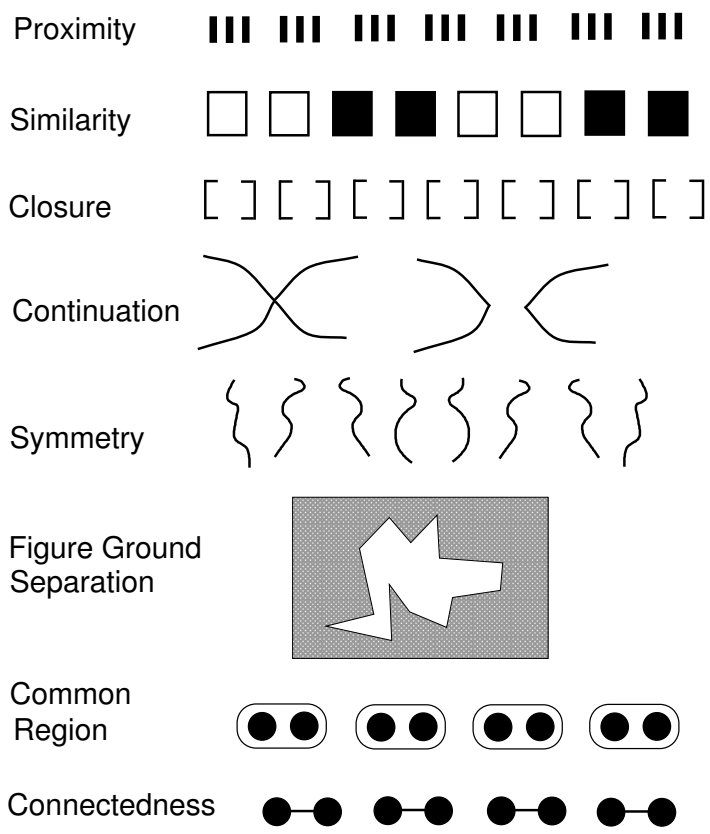


Figure 2.2: Table of Gestalt laws

This overview over the Gestalt laws illustrates, that the related grouping principles are only described on a phenomenological level giving no details about the concrete implementation. The Gestalt laws inspired a lot of algorithmic approaches for the technical realization of grouping and segmentation tasks in computer vision. Through the different modalities and the high number of possible feature combinations the number of developed algorithms increases very fast in the course of present research, often resulting in very problem specific realizations. The following section shall give a rough overview over different approaches that are most relevant in relation to this manuscript.

2.2 Grouping Algorithms in Image Processing

A classical bottom up image processing architecture shows the following processing stages: It starts from the respective application specific radiographic or visualization technique which represents each point in the image by a low level feature vector. The image is often preprocessed by image enhancement methods to reduce the influence of noise and errors. Then the image is separated into coherent regions, which might for example represent relevant objects in the image. To analyze these objects closer, in the next stage higher level features of the different regions are extracted and fed into a classification algorithm to reveal the rough class of the displayed objects. Finishing the process of pattern recognition, the exact configuration of the objects can be investigated and represented by suitable data structures. Afterwards higher order knowledge based processing steps can follow that deduce contextual meanings of the objects and formulate the semantic information of the image in an internal representation.
Obviously, the stage of image segmentation based on grouping principles, like the Gestalt laws, has a high influence on the following steps, such that the segmentation method has to be chosen carefully concerning the image representation by local features and the requirements and abilities of further processing steps, like object classification. The three areas of feature extraction, segmentation algorithms and object classification are highly connected and are reviewed in the following.

2.2.1 Feature Extraction

The probably most common representation of images are gray scale images, where each pixel in the image is described by a gray value according to the intensity of light at its position. The appearance of gray scale images depends on the chosen resolution of the image and the number of available gray levels, describing a finer or rougher quantification of the intensity values, which both have an influence on the detail-level of the displayed objects.
Feature extraction in gray scale images is often realized by the convolution of the image with specialized filter masks. One of the most common application is the detection of edges, represented by discontinuities in the course of intensity. A simple approach is to approximate the first order derivative of the intensities along the x- and y-axis of the image by Sobel filters to estimate the direction and magnitude of the intensity gradient. This approach has to deal with several problems like the tradeoff between sensitivity and robustness, the accuracy in the estimation of edge orientation, the preference to certain direction of edges and computational artifacts, like the double response of Sobel filters on both sides of edges. Subject to research are more sophisticated filters and thresholding methods on the gradient magnitudes to decide whether an edge is present at a certain position or not [28].
The information about intensity in the image can be extended by more color information, like the description of hue and saturation of pixels. This raises the question of color encoding, which can be implemented in various color spaces,

like the technical red-green-blue (RBG) channeling, the hue-saturation-intensity (HSI) description or other normalizations, e.g. according to a uniform intensity to be resistant against changes in illumination. The detection of edges can also be extended to these additional color dimensions.

Besides intensity, color and edges, texture is an important feature in images. However, texture is not a feature of a single point, but arises from regular structures in whole regions. Possible ways to describe textures are measures of energy, contrast and homogeneity of color transition matrices [13] according to some neighborhood of a pixel. Another possibility is the convolution with Gabor filters [8] that detect the occurrence of periodical structures with a certain frequency and orientation. The texture for a pixel is represented by a vector of filter responses from a set of Gabor filters with various orientations and frequencies. This turns the different number of filters in orientation and frequency into degrees of freedom in the encoding of textures. In general, texture depends strongly on the size of the observed neighborhood of a pixel.

The observation of stereo images or image sequences allows the extraction of depth or motion information, based on the disparity of pixels [54]. However, this demands the ability of matching points from different images, solving the correspondence problem to measure the amount of translation between two images [23].

An approach to extract object specific features is to create a sample set of subimages from training images and to apply a method of dimension reduction, like mayor Principle Component Analysis (PCA) [22], [15], on it to reveal filter maps that describe the main variance of the subimages. The idea of this approach is, that some of the learned principle components become selective for special shape features of the presented objects. Various approaches that substitute the Principle Components Analysis by more sophisticated methods follow the aim to generate features that show a sparse activation from the different objects [39] and to find hierarchical combinations of these low-level features to build features of higher order for bigger image patches [64].

Finally, multidimensional features can arise directly from imaging techniques applied in special branches of science, like medical image processing. One example is the generation of time series in magnetic resonance imaging [67], where the temporal change in the concentration of a contrast medium indicates different types of tissues. An other example can be found in [53], where probes of lymphocyte cells are treated by different contrast mediums that highlight special types of proteins within the cells. In both examples, the images are analyzed by parallel processing on multiple image dimensions.

Features can describe a wide range of modalities of information, like color, orientation, texture, speed or shape, by a single value or vectorial representation. These different modalities also demand different approaches of segmentation or grouping methods to bind these features into coherent groups or regions of interest, as it is discussed in the next section.

2.2.2 Segmentation Algorithms

Let us start the investigation of technical implementation of the Gestalt laws with the principle of proximity. Proximity is implemented for instance by clustering algorithms, like the k-Means or LBG-Algorithm [27]. Clustering is a general technique in datamining that works on arbitrary multidimensional input spaces. The goal is to reduce a high number of data vectors to a low number of prototypic reference vectors with an as small as possible average distortion between the prototypes and the data vectors according to problem specific distance measures. The input data can then be partitioned by the next neighbor criterion, assigning each data vector to it's nearest prototype. The output of clustering techniques mainly depends on the type of distance measures that is used and the scaling of these measures.

Applied to image data, clustering is often used to describe principles of similarity by proximity in feature space. For example, in [67], it is used to segment MRI data in biomedical image processing, where each point in an image is described by a time series for the concentration of a contrast medium in the observed organs. These time series are clustered and mapped to prototypes, where the prototypes can be associated with different types of tissue. Another example can be found in [18], where clustering is performed according to dissimilarities in responses of Gabor filters to get a set of prototypes that define regions of coherent texture.

If the observed images consist only of intensity or color information, segmentation is often realized by the two traditional branches of region-based and edge-based segmentation approaches. Region based approaches directly try to find regions of coherent color. This can be done by applying threshold methods that map all pixels whose color lies in a certain interval of thresholds to the same label. The thresholds of the intervals can be estimated from a histogram of the observed image, and pixels that are assigned to the same label and are locally connected are merged to a segment. Other methods are split and merge or island growing methods, where an image is first hierarchically split into elementary segments, which are afterwards merged to more complex regions, if the segments are connected and have similar values of average color. An example of this approach can be found in the color structure code (CSC) [47].

Edge based approaches try to find contours as borders between regions and segments in images and therefore rely on the detection of edges. The set of possible edges has to be thinned out to one pixel wide line segments by methods, like non maximum suppression [3]. After this step, the remaining line segments are concatenated according to the principle of continuation and closure [10] to contours. An alternative approach are the more sophisticated methods of active contours, like the snake algorithm [21]. These methods start from initializations of deformable models of contours and match them with existing contours by energy minimization according to image forces based on the magnitude and orientation of edge detectors. These image forces can be extended by interior forces of the contours, like stiffness and tension, which adds additional constraints to the energy function of

the active contour.
A special problem arises, when results from both region based and edge based methods have to be integrated into a consistent segmentation. An example of such an architecture can be found in [29], where segmentation is implemented by a graph partitioning technique for weighted graphs and the weights of the graph are based on the proximity of Gabor responses and the strength of detected edges. A fundamental part of this architecture is a gating process that estimates the relevance of the two possible ways of segmentation and controls their influence on the output segmentation.

2.2.3 Classification

The process of classification often follows the step of image segmentation to reveal the class or category of an observed object from the detected features. The objects are presented by feature vectors $\mathbf{x} \in \mathbb{R}^d$ of dimension d and mapped onto a set of possible object classes $\mathbb{C} = \{C_1, \dots C_N\}$. In it's simplest form, binary classification, and under the assumption, that the two classes are linear separable in the feature space, the separation plane can be described by a straight line in the feature space (see Fig. 2.3).

$$0 = \mathbf{w}^T\mathbf{x} + b. \tag{2.1}$$

From the point of view of artifical neural networks, this can be interpreted as a linear neuron. Early results from Rosenblatt [48] have shown that any kind of classification problem that is linearly separable can be solved by a single layered architecture of linear neurons called perceptron. The necessary weights can be adapted by the perceptron learning rule, which is proved to find a suitable solution in a finite number of steps. The limits of the perceptron of dealing with linear separable data were extended by the development of the Multi Layer Perceptron, which is able to approximate more complex shaped separation planes by the well known Backpropagation learning rule. The well understood abilities of the MLP made it to a standard tool in pattern classification. In recent research, the MLP is often replaced by the more sophisticated Support Vector Machine (SVM). The SVM is in principle also a linear classifier similar to the perceptron, but with the ability to estimate the, in terms of generalization, optimal separation plane from a constrained optimization of the classification weights. As a result of this optimization, only data vectors from the borders of the classes, called the support vectors, contribute to the classification function, while vectors from inside the classes can be omitted. The extension to nonlinear separation planes can be realized within the SVM by transferring the input data to a higher dimensional feature space, where the classes become linear separable and, therefore, can be easily separated. Fortunately, this feature space does not need to be specified explicitly, but can be defined implicitly by applying special kernel functions on pairs of data vectors. The most demanding challenge of research on the field of SVM is to find suitable shapes and sizes of kernel functions for a given classification problem.

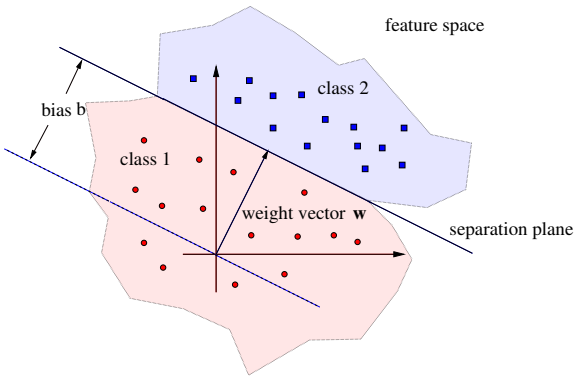


Figure 2.3: Realization of a binary segmentation by the linear separation plane of a linear neuron.

Lets us not go deeper into the theory of classification, but only remark the following comment: If the feature space is chosen in a clever way, the classification problem can be simplified, such that it often can be solved by linear discriminators. So, the complexity of higher level classification tasks can be drastically reduced by an appropriate choice of low level features and segmentation methods in early processing steps.

2.3 Grouping Principles and Pairwise Compatibilities

The previous overview of existing segmentation methods has shown, that there is a variety of implementation possibilities for the different grouping principles. Each grouping algorithm has it's own tuning parameters that have to be adapted to the actual grouping problem. The situation becomes more complex, if there is a problem domain, where several grouping principles can be applied at the same time, like in Fig. 2.4. In this example, a set of objects is presented that could be grouped according to similarity in color or shape or columnwise by the principle of proximity. For the human observer, this problem is solved by the law of Prägnanz, such that probably the principle of similar color dominates over the other principles. For an artifical segmentation system, this behavior has to be specified by a suitable adjustment of the parameters of the different grouping algorithms against each other.
To overcome these problems, in this work an abstract representation of grouping behaviors is used that can be specialized to any concrete grouping problem. In the following, a grouping behavior will be encoded into pairwise compatibilities between the elementary objects. These compatibilities can be positive, which shall express the compatibility of two objects in the same group, or negative, if two objects are incompatible for the same group. Examples of such compatibilities are displayed in figure 2.4, where objects that clearly belong to the same group,

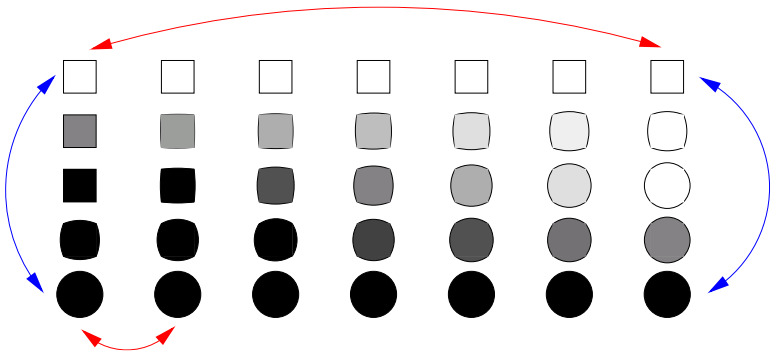


Figure 2.4: Parallel Occurrence of different Gestalt laws. The displayed objects can be divided by different grouping principles of similar color, similar shape and proximity. For a human observer the principle of similar color dominates by the law of Prägnanz. For an artifical system, this behavior shall be described by pairwise compatibilities displayed by red (positive values) and blue (negative values) arrows.

like two white squares or two black circles, have a high positive compatibility (red arrows), while objects that clearly belong to different groups, like a white square and a black circle, have a high negative compatibility (blue arrows). [1]
Not all pairwise compatibilities are specified in figure 2.4. Compatibilities between objects, where it is not directly clear whether they belong to the same group or not, like a light gray and dark gray circle or a black square and a black circle, have to be adjusted to smaller positive or negative values to describe the preference of similarity in color or shape. Obviously, a manual specification of all compatibilities is very extensive. Also an automatic generation of such compatibilities from heuristic models can be become complex, because the respective model parameters have to be adapted suitably. For this reason, a central point of this work is the learning of suitable compatibilities from a low number of grouping examples to adapt properties of a single grouping principle as well as the parametrical balance between several differentiating grouping principles.
The generation of training examples shall be as natural as possible for a human user by dividing datasets into occurring groups, like in the example of Fig. 2.5. In the case of image data, this is done by labeling images.

2.4 Summary

It was shown, that human perception is strongly affected by grouping processes which can be described by principles like the Gestalt laws. The field of image

[1] This color encoding of positive compatibilities by red color and negative compatibilities by blue color will be maintained for different visualizations of grouping behaviors throughout this work.

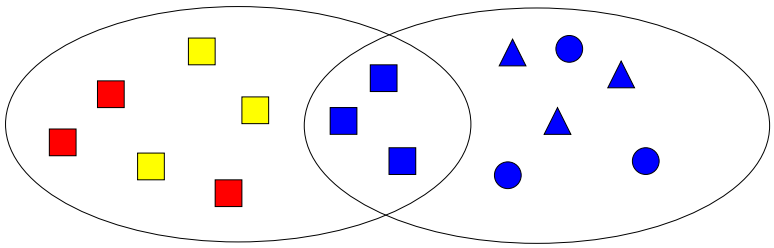


Figure 2.5: Description of a simple grouping problem. A human observer specifies, that the set of displayed objects should be divided into the groups of red and yellow squares, blue squares and blue triangles and circles. A learning process should adapt the pairwise compatibilities between the objects, to describe the application of the principles of similarity in color and shape.

processing provides a variety of segmentation algorithms to implement such principles. The choice of an algorithm for a concrete application requires an adaptation of the respective parameters and is connected to the related fields of feature extraction and pattern classification. In this work, an approach is investigated that describes grouping behaviors by pairwise compatibilities between elementary objects. These compatibilities are learned from hand-labeled data. The next chapter describes, how a grouping process can be implemented with the pairwise compatibilities introducing the central method of this work, the Competitive Layer Model. The concrete method of learning the pairwise compatibilities is treated in the following chapters.

Chapter 3

The Competitive Layer Model

3.1 The Problem Domain

The Competitive Layer Model (*CLM*) [50] is a recurrent neural network that can perform grouping and labeling tasks. Before we start to investigate how the CLM implements the binding process of combining elementary sub patterns to more complex groups or concepts, let give us a formal description of such tasks.

Assume, that each grouping process belongs to a problem specific set, denoted by $\mathcal{F}$, which defines the set of all elementary objects or sub patterns $\mathbf{m}_r$ that can occur as components of a certain class of problems and that are called in the following "features". Therefore, $\mathcal{F}$ is called the "feature-domain". In the case of image segmentation problems, $\mathcal{F}$ is often a discrete set of pixel instances given by quantized pixel coordinates combined with also quantized color values. In general grouping problems, like the example of 2D-clustering in Fig. 3.1, $\mathcal{F} = \mathbb{R}^d$ is given by a space of real valued d-dimensional vectors.

A concrete grouping task is given by a subset $\mathcal{R}^i = \{\mathbf{m}_r \in \mathcal{F} | r = 1, \ldots, N^i\}$ of all possible features and is called a "pattern", where the index r addresses the features in the respective pattern $\mathcal{R}^i$.

Assume, that each pattern consists of a certain structure, given by the organization of the features into several disjunct groups $G^i_\alpha, (\alpha = 1, \ldots, L^i)$, e.g. point-clusters in two-dimensional space, where L^i specifies the number of groups in pattern $\mathcal{R}^i$ and $\bigcup_{\alpha=1}^{L^i} G^i_\alpha = \mathcal{R}^i$.

To reveal this structure, the pattern has to be segmented by means of a labeling function $\hat{\alpha}(\mathbf{m}_r)$, which is denoted shorter by $\hat{\alpha}(r)$, that assigns each feature $\mathbf{m}_r$ to one of L^i possible labels $\alpha \in \{1, \ldots, L^i\}$. This function binds all features that are assigned to the same label to a coherent group and therefore transforms elementary information to a higher order concept.

Let $x_{r\alpha} \geq 0$ describe the certainty of assigning the feature $\mathbf{m}_r$ to the label α and let $f_{rr'}$ describe the certainty for the compatibility of the features $\mathbf{m}_r$ and $\mathbf{m}_{r'}$ in the same group G^i_α, where $f_{rr'} > 0$ expresses that it is preferable that $\mathbf{m}_r$ and $\mathbf{m}_{r'}$ belong to the same group G^i_α and $f_{rr'} < 0$ expresses that it is preferable that $\mathbf{m}_r$

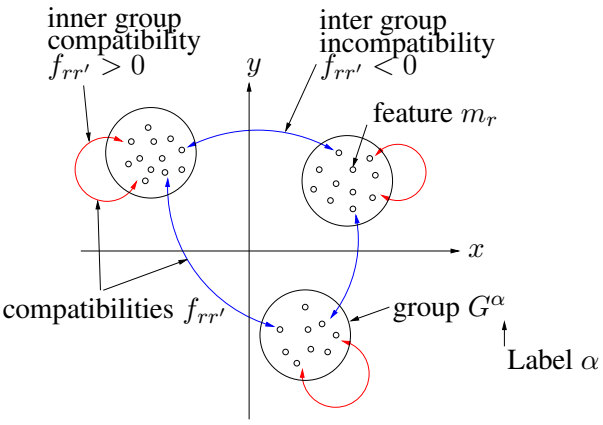


Figure 3.1: Example for a grouping problem: the 2D-Clustering Problem.

and $\mathbf{m}_{r'}$ belong to different groups $G^i_\alpha \neq G^i_\beta$.
The labeling function $\hat{\alpha}(r)$ should assign each feature $\mathbf{m}_r$ uniquely to the label of that group to which it is most compatible:

$$x_{r\hat{\alpha}(r)} > 0, \quad x_{r\beta} = 0, \quad \text{for all} \quad r, \beta \neq \hat{\alpha}(r), \tag{3.1}$$

and

$$\sum_{r'} f_{rr'} x_{r'\hat{\alpha}(r)} > \sum_{r'} f_{rr'} x_{r'\beta}, \quad \text{for all } r, \beta \neq \hat{\alpha}(r), \tag{3.2}$$

under the constraint of a fixed certainty h_r of assigning feature $\mathbf{m}_r$ to a label α

$$\sum_{\beta} x_{r\beta} = h_r. \tag{3.3}$$

The weighted sum $\sum_{r'} f_{rr'} x_{r'\alpha}$ is called the "support" of feature $\mathbf{m}_r$ in group α. Finding $\hat{\alpha}(r)$, the group with the maximal support, depends on a parallel estimation of $x_{r\alpha}$ and a recurrent feedback to the support of the other features. To solve an arbitrary grouping problem, the answers for two important questions have to be found. The first is: if pairwise compatibilities are given, how can the corresponding labeling function be found? This question will be discussed in section 3.2 to 3.5. The second question goes in the opposite direction: if a certain labeling function shall be implemented, how must the pairwise compatibilities be chosen to achieve this goal? This question will be treated in section 3.6 and 3.7.

3.2 The Architecture

The Competitive Layer Model represents the assignment variables $x_{r\alpha}$ as activations of recurrently connected linear threshold neurons. These kind of neurons are

characterized by non-saturating linear transfer functions

$$\theta_j(x) = \begin{cases} 0 & : \quad x < \theta_j \\ k_j(x - \theta_j) & : \quad x \geq \theta_j \end{cases}, \tag{3.4}$$

where θ_j is an activation threshold and $k_j > 0$ is the gain of the transfer function of neuron j. The plausibility of such neurons is supported by observations that cortical neurons rarely operate close to saturation [9], despite strong recurrent excitation. Additionally they provide useful theoretical properties, like convergence conditions for asynchronous update procedures [11] and non-divergent and multistable dynamics in winner-takes-all (WTA) networks [14].
The neurons are organized in layers $\alpha = 1, \ldots, L$ and columns $r = 1, \ldots, N^i$, where each layer contains all neurons belonging to the same label α and each column contains the neurons belonging to the same feature $\mathbf{m}_r$. Obviously the number of columns is determined by N^i, the number of features in the actual pattern, but L, the number of layers, can to be chosen according to the number of maximally expected groups in the pattern.
A binding between two features, represented by columns r and r', is expressed by simultaneous activities $x_{r\hat{\alpha}} > 0$ and $x_{r'\hat{\alpha}} > 0$ that share a common layer $\hat{\alpha}$. All neurons in a column r are equally driven by an external input h_r, which represents the significance of the detection of feature r by a preprocessing step. The afferent input h_r is fed to the activities $x_{r\alpha}$ with a connection weight $J > 0$. Within each layer α the activities are coupled via lateral connections $f_{rr'}$, which characterize the degree of compatibility between features r and r' and which are symmetric under feature exchange, thus $f_{rr'} = f_{r'r}$. The purpose of the layered arrangement in the CLM is to enforce an assignment of the input features to the layers by a dynamics, using the contextual information stored in the lateral interactions. The unique assignment of each feature to a single layer is realized by a columnar WTA circuit, which uses mutual symmetric inhibitory interactions with absolute strength $J > 0$ between neural activities $x_{r\alpha}$ and $x_{r\beta}$ that share a common column r. Due to the WTA coupling, for a stable equilibrium state of the CLM only a neuron from one layer can be active within each column [65]. The number of layers does not predetermine the number of active groups, since for sufficiently many layers only those are active that carry a salient group.
The combination of afferent inputs and lateral and vertical interactions is combined into the standard linear threshold additive activity dynamics

$$\dot{x}_{r\alpha} = -x_{r\alpha} + \sigma\Big(J(h_r - \sum_{\beta} x_{r\beta}) + \sum_{r'} f_{rr'} x_{r'\alpha} + x_{r\alpha}\Big), \tag{3.5}$$

where $\sigma(x) = \max(0, x)$ is the uniform non-saturating linear threshold transfer function. The special form of this function makes it possible to formulate concrete statements about the behavior of the CLM dynamics. The main statements can be summarized in two theorems about the convergence and assignment properties of the CLM [65].

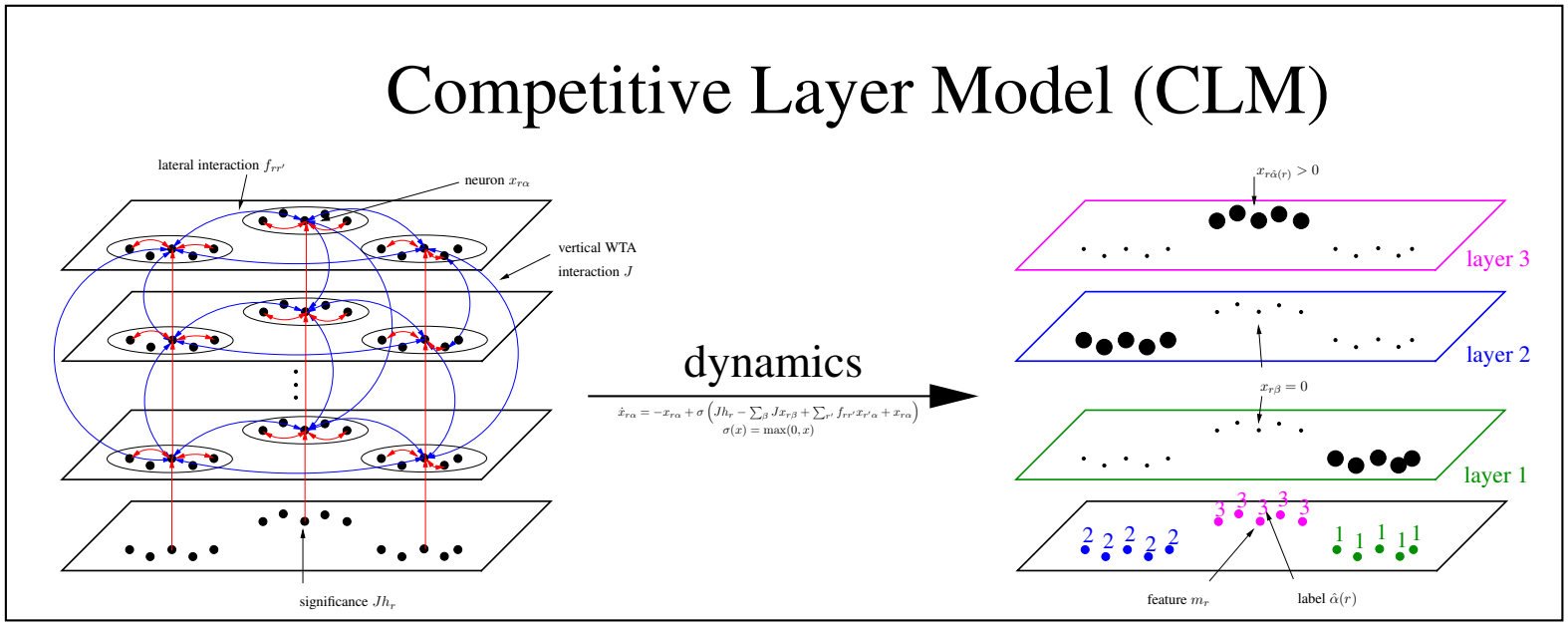


Figure 3.2: The Competitive Layer Model. The CLM is based on a layer wise arrangement of feature representative neurons $x_{r\alpha}$. These neurons are recoupled by inhibitive (blue) connections J between the layers and by inhibitive and excitatory (red) lateral connections $f_{rr'}$ within the layers. In the attractor state of the CLM each feature is uniquely assigned to one of the layers $\hat{\alpha}(r)$ such that this state can be interpreted as segmentation or labeling of the input pattern.

Convergence Theorem: If $J > \lambda_{\max}\{f_{rr'}\}$, where $\lambda_{\max}\{f_{rr'}\}$ is the largest eigenvalue of the lateral interaction matrix F, or $J > \max_r \left(\sum_{r'} \max(0, f_{rr'})\right)$, then the CLM dynamics is bounded and convergent.
Assignment Theorem: If the lateral interaction is self-excitatory, $f_{rr} > 0$ for all r, then an attractor of the CLM has in each column r either
i) at most one positive activity $x_{r\hat{\alpha}(r)}$ with

$$x_{r\hat{\alpha}(r)} = h_r + \frac{\sum_{r'} f_{rr'} x_{r'\hat{\alpha}(r)}}{J}, \quad x_{r\beta} = 0 \quad \text{for all } \beta \neq \hat{\alpha}(r), \tag{3.6}$$

where $\hat{\alpha}(r)$ is the index of the maximally supporting layer characterized by

$$\sum_{r'} f_{rr'} x_{r'\hat{\alpha}(r)} > \sum_{r'} f_{rr'} x_{r'\beta} \quad \text{for all } r, \beta \neq \hat{\alpha}(r) \tag{3.7}$$

or,
ii) all activities $x_{r\alpha}$, $\alpha = 1, \ldots, L$ in a column r vanish and $\sum_{r'} f_{rr'} x_{r'\alpha} \leq -J_r h_r$ for all $\alpha = 1, \ldots, L$.

3.3 Figure-Background-Separation

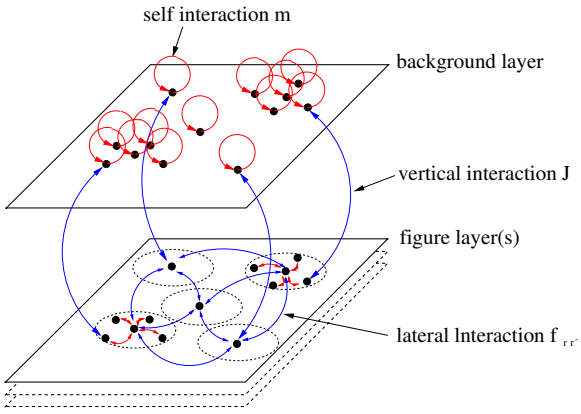


Figure 3.3: Extension of the CLM architecture with a background layer. The lateral connections in the background layer are reduced to mere self excitatory connections.

A special goal in many grouping and segmentation tasks is the separation of the relevant objects in the data from noisy or incoherent parts that form some kind of background for the desired information. This requirement can be easily integrated into the CLM architecture by using an additional layer b, where the lateral interactions are restricted to a mere self interaction, expressed by the product of the self

interaction strength m with the Kronecker Delta $\delta_{rr'}$ between the features $\mathbf{m}_r$ and $\mathbf{m}_{r'}$, such that

$$f^b_{rr'} = m\delta_{rr'} = \begin{cases} m & : \quad r = r' \\ m & : \quad r \neq r' \end{cases} . \tag{3.8}$$

By inserting this expression into the consistency conditions (3.7), it can be seen easily that m defines a threshold of minimal mutual support that is necessary to assign a feature m_r to one of the relevant groups ($\alpha \neq b$). All features whose mutual support lies below this threshold are assigned to the background.

3.4 Properties of the Binding Process and Annealing

The dynamics (3.5) has an energy function of the form

$$E = -J \sum_{r\alpha} h_r x_{r\alpha} + \frac{1}{2} J \sum_{r} \sum_{\alpha\beta} x_{r\alpha} x_{r\beta} - \frac{1}{2} \sum_{\alpha} \sum_{rr'} f_{rr'} x_{r\alpha} x_{r'\alpha}. \tag{3.9}$$

The energy is non-increasing under the dynamics (3.5) [61] :

$$d/dt\ E = -\sum_{r\alpha} E_{r\alpha} \dot{x}_{r\alpha} = -\sum_{r\alpha} E_{r\alpha}(-x_{r\alpha} + \sigma(E_{r\alpha} + x_{r\alpha})) \leq 0, \tag{3.10}$$

where

$$E_{r\alpha} = -\partial E / \partial x_{r\alpha} = J h_r - J \sum_{\beta} x_{r\beta} + \sum_{r'} f_{rr'} x_{r'\alpha}. \tag{3.11}$$

Thus the attractors of the dynamics (3.5) are the local minima of (3.9) under constraints $x_{r\alpha} \geq 0$. Additionally a kind of annealing process can be included in the dynamics by extending the energy function with:

$$E' = E + T \sum_{r\alpha} x^2_{r\alpha}, \tag{3.12}$$

which adds a convex term that biases the local minima towards graded assignments and thus makes the WTA process more soft. Within the dynamics this introduces a new self-inhibitory term

$$\dot{x}_{r\alpha} = -x_{r\alpha} + \sigma\Big(J(h_r - \sum_{\beta} x_{r\beta}) + \sum_{r'} f_{rr'} x_{r'\alpha} + (1 - T) x_{r\alpha}\Big). \tag{3.13}$$

Through gradually lowering the self-inhibition T, (3.12) becomes (3.9) and (3.13) becomes (3.5).
A detailed analysis of the annealing process can be found in [61], where an eigensubspace analysis of the linear part of the CLM-dynamics is made by observing the eigenmodes of the matrix $G \in \mathbb{R}^{N^i \cdot L \times N^i \cdot L}$, which is the matrix of all lateral, vertical and self inhibitory weights:

$$G^{\alpha\beta}_{rr'} = -J\delta_{rr'} + \delta_{\alpha\beta} f_{rr'} - T\delta_{\alpha\beta}\delta_{rr'}. \tag{3.14}$$

The dynamics (3.5) is approximated by the linear system:

$$\dot{\mathbf{x}} = J\mathbf{h} + G\mathbf{x}, \tag{3.15}$$

where the activities and the external inputs are represented as $N^i \times L$ vectors

$$\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_L) \quad \text{with} \quad \mathbf{x}_\alpha = (x_{1\alpha}, \ldots, x_{N^i\alpha}) \quad \text{and} \tag{3.16}$$

$$\mathbf{h} = (\mathbf{h}_0, \ldots, \mathbf{h}_0) \quad \text{with} \quad \mathbf{h}_0 = (h_1, \ldots, h_{N^i}). \tag{3.17}$$

The CLM-dynamics is then characterized by the $N^i \cdot L$ eigenvectors $\mathbf{v}^{k\gamma} \in \mathbb{R}^{N^i \times L}$ and eigenvalues $\Lambda_{k\gamma}$ of G $(k = 1, \ldots, N^i, \gamma = 1, \ldots, L)$.
The result of this analysis is, that the CLM-dynamics is driven by two kind of eigenmodes called the AC- and DC-eigenmodes (sketched in Fig. 3.4) whose eigenvalues and eigenvectors mainly depend on the matrix F of lateral weights $f_{rr'}$.

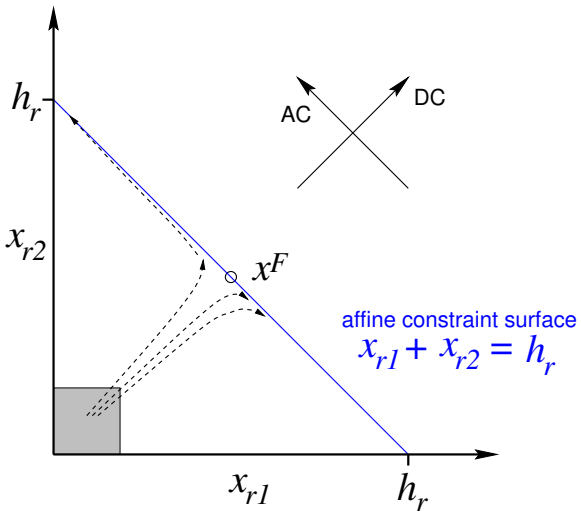


Figure 3.4: Sketch of the linear dynamics for two layers (sketch and caption from [61], p. 51). Shown are the activity trajectories for the two activities x_{r1}, x_{r2} of a single column r. Starting from small initial values (grey square) the activities quickly approach the fixed point $\mathbf{x}^F$, which represents the "totally undecided" state, and the constraint surface $\sum_{\alpha=1,2} x_{r\alpha} = h_r$ in the DC subspace. Then the dynamics in the orthogonal AC subspace drives the WTA process until only one layer is active.

DC-Modes: It is shown, that the eigenmodes in the DC-Subspace have equal components in all layers of the CLM

$$\mathbf{v}^{k1} = 1/\sqrt{L}(\mathbf{b}^k, \ldots, \mathbf{b}^k), \tag{3.18}$$

where $\mathbf{b}^k$ is the eigenvector of the kth greatest eigenvalue λ_k of the matrix of lateral interactions F. The eigenvalues of the DC-eigenvectors depend also on the eigenvalues of F

$$\Lambda_{k1} = \lambda_k - JL, \tag{3.19}$$

which can be assumed to be high negative values, if J is chosen clearly higher than the greatest eigenvalue of F to fulfill the convergence theorem of the CLM. The DC-eigenmodes drive the activities of the CLM to the constraint surface

$$\forall r : \sum_{\alpha} x_{r\alpha} = h_r, \tag{3.20}$$

and, therefore, cause a partition of the external input h_r to the activities of the rth column of the CLM.

AC-Modes: In contrast to the DC-Modes, the eigenvectors of the AC-Modes have different components in the layers of the CLM

$$\mathbf{v}^{k\gamma\neq 1} = (q_1^\gamma \mathbf{b}^k, \ldots, q_L^\gamma \mathbf{b}^k), \tag{3.21}$$

where the coefficients q_α^γ are the components of the γth eigenvector $\mathbf{q}^\gamma$ of $\mathbb{I}^{L\times L}$ which is the $L \times L$ matrix of 1's

$$\mathbf{q}^\gamma = (q_1^\gamma, \ldots, q_L^\gamma)^T \in \mathbb{R}^L. \tag{3.22}$$

Through the special structure of $\mathbb{I}^{L\times L}$ for $\gamma \neq 1$ the components $q_1^\gamma, \ldots, q_L^\gamma$ sum to zero. Therefore, also the layer vectors of the AC-eigenvectors sum to zero. So the AC-eigenvectors change the distribution of activity among the layers and drive the WTA process within the columns of the CLM.

The eigenvalues of the AC-modes correspond to the eigenvalues of F

$$\Lambda_{k\gamma} = \lambda_k \tag{3.23}$$

and can be positive or negative, where only modes with positive eigenvalues contribute to the WTA-process. For high values of J, the absolute values of the AC-eigenvalues are significant smaller than the DC-eigenvalues, such that the AC-modes influence the CLM-dynamics on a slower time scale than the DC-modes.

Influence of annealing of the dynamics: Since the modification of the pseudo-temperature T has only influence to the weights on the main diagonal of G, the annealing process of lowering T can be interpreted as a simple shift of the eigenvalues of G. At $T = \lambda_{max}\{F\}$, which is the greatest eigenvalue of the lateral interaction matrix F, all eigenvalues in the AC-Subspace are negative and the dynamics is mono stable, which means it is driven to the global fixed point $\mathbf{x}^F$:

$$\forall r, \alpha \quad : \quad x_{r\alpha}^F \approx \frac{1}{L} h_r. \tag{3.24}$$

By gradually lowering T the eigenvalues of the AC-Subspace are shifted into the positive quadrant, such that step by step the different AC-Modes are switched on

in the course of the dynamics in the ordering of the strength of the corresponding eigenvalues. The dynamics becomes multi stable, where the CLM can converge to different distributions of the groups to the layers, while the distribution of the features to the groups is the same for all attractor states.

3.5 Algorithm

The CLM dynamics can be simulated efficiently with a Gauß Seidel approach of solving iteratively the fixed point equations of (3.5) for a randomly chosen activity $x_{r\alpha}$, while all other activities are held constant [40], [61]. The algorithm can be implemented in the following way:

1. Initialize all $x_{r\alpha}$ with small random values around $x_{r\alpha}(t=0) \in [h_r/L - \epsilon, h_r/L + \epsilon]$.
   Initialize T with greatest eigenvalue of matrix $\{f_{rr'}\}$.
2. Do $N{\cdot}L$ times: Choose (r,α) randomly and update $x_{r\alpha} = \max(0,\xi)$, where
   $\xi := \frac{J(h_r - \sum_{\beta\neq\alpha} x_{r\beta}) + \sum_{r'\neq r} f_{rr'} x_{r'\alpha}}{J - f_{rr} + T}$
3. Decrease T by $T := \eta T$, with $0 < \eta < 1$. Go to step 2 until convergence.

3.6 Hand-tuned Interaction Weights

The behavior of the CLM grouping process mostly depends on the structure of the lateral weight matrix F, whose components describe the problem specific grouping principle by mapping each pair of features $(\mathbf{m}_r, \mathbf{m}_{r'})$ in a pattern onto it's compatibility $f_{rr'}$. To be able to generate a similar weight matrix for every new pattern from the same problem domain a general interaction function $f : \mathcal{F}^2 \to \mathbb{R}$ has to be specified that maps all possible pairs from the feature-domain onto their compatibilities. In early applications, this was done by hand-tuning parameterized functions. The following recapitulation of examples from the history of the CLM motivates the necessity of an automatic learning method.

3.6.1 Point Clustering

The first introduction and application of the CLM by Ritter [50] treated the grouping problem of point clustering based on the principle of proximity. Each feature $\mathbf{m}_r$ was described as a simple position vector $\mathbf{m}_r = \mathbf{p}_r$ (exemplary applications were made in 2D, such that $\mathbf{p}_r \in \mathbb{R}^2$). The lateral weights $f_{rr'}$ were computed by the simple "on-center-off-surround" function

$$f_{rr'} = \begin{cases} 1 \text{ if } \| \mathbf{p}_r - \mathbf{p}_{r'} \| < R_0 \\ -J_2 \text{ else} \end{cases} , \tag{3.25}$$

which returns positive compatibilities of 1 for feature pairs $(\mathbf{m}_r, \mathbf{m}_{r'})$, whose distance is smaller than a certain radius R_0, and returns negative compatibilities of $-J_2$ otherwise.
Obviously, the size of resulting groups in the output of the CLM depends on the choice of the parameters R_0 and J_2, where R_0 defines the principle shape of the interaction function and J_2 describes some balance between positive and negative interactions. This simple example already describes the main difficulties in the design of the interaction function:

- The main feature properties, which are the basis of the interaction function have to be chosen. Here this property is given trivially by the local distance.
- The principal shape of the interaction function has to be chosen. In the example a step function is chosen.
- The shape parameters of the interaction function, here given by the expected cluster radius R_0, have to be chosen.
- Different parts of the interaction function have to be balanced against each other, e.g., by scaling with the parameter J_2.

The complexity of these degrees of freedom in the design of the interaction function increases with the complexity of the grouping principle, as is shown in the following.

3.6.2 Color Segmentation of Gray Scale Images

For the segmentation of gray scale images in [61], each pixel in the image is represented by a feature vector $\mathbf{m}_r = (\mathbf{p}_r, I_r)$, given by it's position $\mathbf{p}_r = (p_r^x, p_r^y)^T$ and intensity value I_r. The compatibility between two pixels is expressed according to the difference of intensity by

$$f_{rr'} = \begin{cases} 1 - \frac{|I_r - I_{r'}|}{\Theta} - k & : \quad r' \in \mathcal{N}_r \\ -k & : \quad \text{otherwise} \end{cases}, \tag{3.26}$$

where $\mathcal{N}_r$ is the neighborhood of the pixel $\mathbf{m}_r$ defined by the maximum norm and radius R

$$\mathcal{N}_r = \{r' | \max(|p_r^x - p_{r'}^x|, |p_r^y - p_{r'}^y|) \leq R\}, \tag{3.27}$$

Θ is the intrinsic short range similarity given by

$$\Theta = \frac{1}{N} \frac{1}{(2R-1)^2 - 1} \sum_r \sum_{r' \in \mathcal{N}_r} |I_r - I_{r'}| \tag{3.28}$$

and k defines the global inhibition. Here the behavior of the segmentation process is effected by the parameters R and k, while the direct scaling of the differences in intensity by Θ depends on the statistics of intensity differences in the given image.

3.6.3 Contour Grouping

A possibility to express the principle of good continuation by pairwise compatibilities between local edge elements $\mathbf{m}_r = (\mathbf{p}_r, \hat{\mathbf{n}}_r)$, given by position $\mathbf{p}_r = (p_r^x, p_r^y)$ and unit orientation vector $\hat{\mathbf{n}}_r = (\hat{n}_r^x, \hat{n}_r^y)$ ($\| \hat{\mathbf{n}}_r \| = 1$), is applied in [65]. The lateral interaction weights are expressed by

$$f_{rr'} = exp(-(|\hat{\mathbf{n}}\hat{\mathbf{d}}| - |\hat{\mathbf{n}}\hat{\mathbf{d}}|)^2\sigma)exp(-\mathbf{d}^2/R) - k, \tag{3.29}$$

where $\mathbf{d} = \mathbf{p}_r - \mathbf{p}_{r'}$, $\hat{\mathbf{d}} = \mathbf{d}/ \| \mathbf{d} \|$ is the spatial (normalized) difference vector. The first factor of the interaction describes the difference in orientation, while the second factor implements a variant of the law of proximity. The range of these two factors is controlled by the parameters σ and R to balance the two grouping principles against each other, while the global inhibition k, like before, defines the ratio between positive and negative interactions.

3.6.4 Segmentation of Fluorescence Cell Images

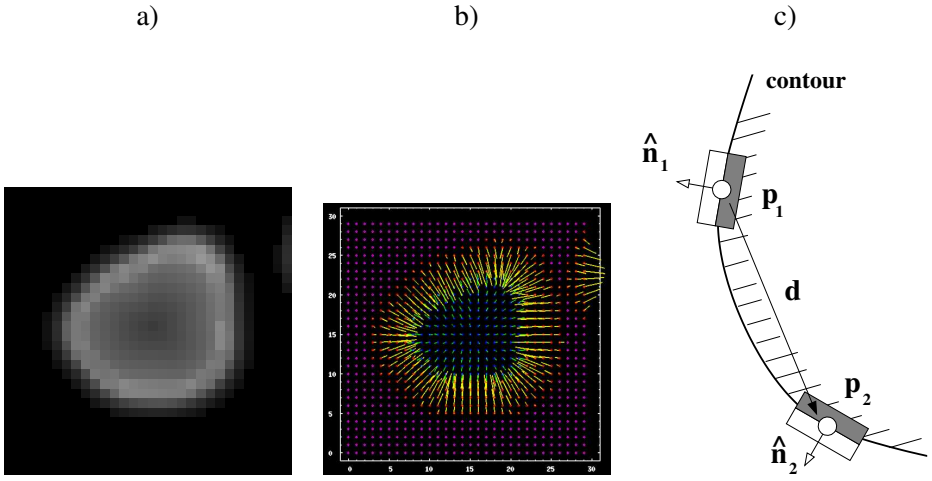


Figure 3.5: Segmentation of fluorescence cell images: a) artifical generated example: cells show a characteristic structure of low intensity at the center and high intensity are the borderline, for details of the pattern generator see [36]; b) plot of intensity gradient directions in a): violet dots are features with gradient strength zero, red/yellow are orientations of features form the background of the cell, blue/green are orientations of features from the cell body; c) principle of convexity after [34]: two features $(\mathbf{m}_1, \mathbf{m}_2)$ belong to the borderline of an cell, if they lie on a circular contour measured by the angles between the two feature orientations $\hat{\mathbf{n}}_1$ and $\hat{\mathbf{n}}_2$ and the connection difference vector $\mathbf{d}$.

In [33], [34], [35], Nattkemper et al. apply the CLM to a problem from medical image processing. A probe of lymphocyte cells is observed that is treated with a

contrast medium that highlights proteins from the membrane of the cells. The cells show a high irregular structure of low intensity at the centers and a high intensity corona at the cell borders, as it is demonstrated in the artifical generated prototype image Fig. 3.5 a). This structure has the characteristics of intensity gradients pointing from inside the cell to outside (see Fig. 3.5 b), while the intensity gradients around the cell point inside.
The task is to separate the regions of different cells from each other and from a diffuse background. Therefore, the interaction function is specified by an advanced version of contour grouping according to continuity, which is extended by additional measures for the convexity of the contours to describe the closed borders of the cells.
Again, the features $\mathbf{m}_r = (\mathbf{p}_r, \hat{\mathbf{n}}_r)$ are represented by position $\mathbf{p}_r = (p_r^x, p_r^y)$ and orientation $\hat{\mathbf{n}}_r = (\hat{n}_r^x, \hat{n}_r^y)$ ($\| \hat{\mathbf{n}}_r \| = 1$), computed by standard Sobel-x- and Sobel-y-operators. The interaction function is chosen as:

$$f_{rr'} = \begin{cases} \xi((\frac{\|\mathbf{d}\|\pi}{2R})^2 + \Psi(\hat{\mathbf{n}}_r, \hat{\mathbf{n}}_{r'})^2) & : \quad \| \mathbf{d} \| < R, \bar{\mathbf{n}}_r \hat{\mathbf{d}} < s \text{ and } \bar{\mathbf{n}}_{r'} \hat{\mathbf{d}} < -s \\ -I_l & : \quad \| \mathbf{d} \| < R, \bar{\mathbf{n}}_r \hat{\mathbf{d}} \geq s \text{ or } \bar{\mathbf{n}}_{r'} \hat{\mathbf{d}} \geq -s \\ -I_g & : \quad \| \mathbf{d} \| \geq R \end{cases} . \tag{3.30}$$

The vectors $\bar{\mathbf{n}}_r = (-\hat{n}_r^y, \hat{n}_r^x)$ are the normal vectors to the feature orientations $\hat{\mathbf{n}}_r$. Positive interaction is returned, only if the angles between the two features $\mathbf{m}_r$ and $\mathbf{m}_{r'}$ and the length of the connecting difference vector $\mathbf{d}$ are relative small and the two orientation normal vectors $\hat{\mathbf{n}}_r$ and $\hat{\mathbf{n}}_{r'}$ point in the same direction of curvature, such that it is probable that both features lie on a circular contour, like in Fig. 3.5 c). The strength of this positive interaction depends on the difference of the two feature orientations measured by

$$\Psi(\hat{\mathbf{n}}_r, \hat{\mathbf{n}}_{r'}) = \frac{\pi}{4}(1 - \hat{\mathbf{n}}_r \hat{\mathbf{n}}_{r'}) \tag{3.31}$$

and the local distance in the term $(\frac{\|\mathbf{d}\|\pi}{2R})^2$.
The area of this positive interactions is determined by the parameters R and s, which control the radius of the segmented cells and the strength of the convexity constraint, and the function

$$\xi(x) = \begin{cases} \cos(x) & : \quad \cos(x) > S \\ 0 & : \quad \cos(x) \leq S \end{cases} . \tag{3.32}$$

The positive interaction is scaled against a local and a global inhibition, described by the parameters I_l and I_g. The complexity of this interaction function is expressed by the case differentiations for feature configurations, scaling of different parts of the interaction function and the adaption of the shape parameters R, s, and S to the experiment settings.

3.6.5 Texture Segmentation

In [40], Ontrup extracts texture information from an image by convolution with a set of Gabor filters. Gabor filters describe a wavelet function with a specified

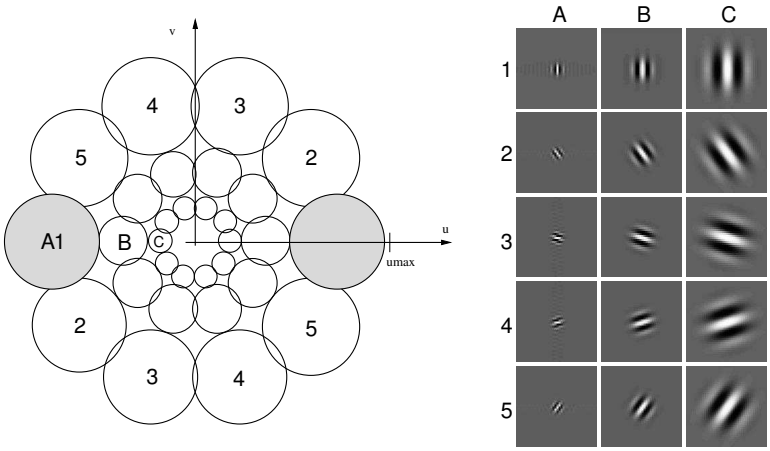


Figure 3.6: The set of 2D Gabor filters used for the feature extraction (Figure and caption from [40]): On the left hand side the daisy-like pattern of the frequency domain is shown. Again, the circles denote the half-peak contours of each 2D Gabor. Their corresponding receptive fields in the spatial domain are depicted on the right. Note, that small receptive fields in the spatial domain have a large counterpart in the frequency domain, and vice versa. This also expresses the uncertainty relation and shows that information content in spatial and frequency domain are inversely related.

frequency and orientation that is overlaid with a Gaussian function. In frequency space Gabor filters look like two Gaussians that are centered at the frequencies (u_0, v_0) and $(-u_0, -v_0)$. Gabor filter responses are complex valued, where the real part describes the components of even symmetric cosine waves, while the imaginary part describes the components of odd symmetric sinus waves. Figure 3.6 shows the set of Gabor-filters and a sketch of their shape in the frequency space, applied by Ontrup. This set consists of fifteen filters at five different orientations and three different frequencies that realize a sparse sampling of the whole frequency space. For an argumentation in [30] and results from own experiments, Ontrup motivates the usage of only the even symmetric components of the Gabor filters. Further, he applies a non-linear function in form of a hyperbolic tangent onto the filter responses, such that texture information of a feature $\mathbf{m}_r$ at position $\mathbf{p}_r = (p_r^x, p_r^y)$ in encoded by the fifteen transformed components of the Gabor filters c_r^{mn}, $m = 1, \ldots 3$, $n = 1, \ldots, 5$.

Since texture is not only a property of a single point, but from a whole image region, these components are substituted by statistical information in form of their mean values μ_r^{mn} and variances σ_r^{mn} ($m = 1, \ldots 3$, $n = 1, \ldots, 5$) in a neighborhood of pixels. This is achieved by convolution of the filter components c_r^{mn} with a Gaussian, the size of which is roughly two to tree times the width of the respective Gabor filter. So the texture information at position $\mathbf{p}_r$ is now characterized by the

30-dimensional vector $\mathbf{z}_r = (\mu_r, \sigma_r)$, where

$$\mu_r = (\mu_r^{11}, \ldots, \mu_r^{35})^T, \quad \sigma_r = (\sigma_r^{11}, \ldots, \sigma_r^{35})^T. \tag{3.33}$$

To reduce the dimensionality of the texture vectors, the standard multidimensional scaling method Principle Component Analysis (PCA) [15], [22] is applied and the texture vectors $\mathbf{z}_r \in \mathbb{R}^{30}$ are replace by their projection onto their first four principal components $\hat{\mathbf{z}}_r \in \mathbb{R}^4$.
A feature $\mathbf{m}_r = (\mathbf{p}_r, \hat{\mathbf{z}}_r)$ is finally represented as aggregation of position and texture information. Consequently the design interaction function is based on differences within these two properties:

$$f_{rr'} = c_{prox} e^{-\|\mathbf{p}_r - \mathbf{p}_{r'}\|^2 / R_{prox}^2} + e^{d_{text}(\hat{\mathbf{z}}_r, \hat{\mathbf{z}}_{r'})/R_{text}^2} - k, \tag{3.34}$$

where the first term describes the Euclidean distance according to feature position and the second term describes the city block distance in the principle components of the texture vectors:

$$d_{text}(\hat{\mathbf{z}}_r, \hat{\mathbf{z}}_{r'}) = \sum_{i=1}^{4} \left(\frac{|(\hat{\mathbf{z}}_r)_i - (\hat{\mathbf{z}}_{r'})_i|}{\sqrt{\alpha((\hat{\mathbf{z}}_r)_i)}} \right). \tag{3.35}$$

$\alpha((\hat{\mathbf{z}}_r)_i)$ is here the standard deviation of the ith principle component.
The range of positive interaction in the two distance measures is controlled with the parameters R_{text} and R_{prox}. They are weighted against each other by the parameter c_{prox} and, furthermore, weighted against the strength of a global inhibition k.

3.7 The Learning Approach

The last section has shown, that the manual design of a suitable interaction function for a grouping problem can result in the construction of complex functions, whose parameters have to be adapted to achieve a multidimensional scaling of different feature properties, like position, orientation, color and texture. To simplify the design process for the user, Wersing introduced a learning approach for the CLM that estimates the parameters of the desired interaction function from a set of hand-labeled training patterns [62]. In this section, the original approach of Wersing is presented and its application on the segmentation of fluorescence cell image (compare section 3.6) is reviewed.

3.7.1 Formulation of the Learning Problem

The learning problem consists of finding a suitable compatibility function $f_{rr'} = f(\mathbf{m}_r, \mathbf{m}_{r'})$, which expresses the preference to bind similar features $\mathbf{m}_r, \mathbf{m}_{r'}$ by positive values or the preference to segregate dissimilar features by negative values, respectively. Assume that a set of M labeled training patterns $\mathcal{P}^i, i = 1, \ldots, M$ is given. For each $\mathcal{P}^i$ a subset $\mathcal{R}^i = \{\mathbf{m}_1, \ldots, \mathbf{m}_{N^i}\}$ of N^i different features

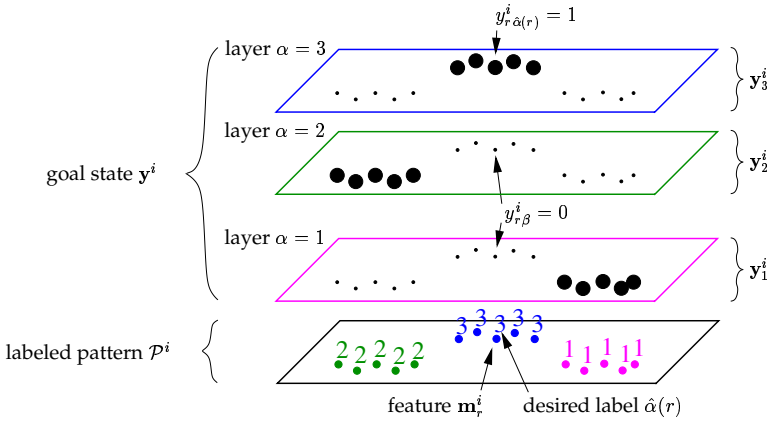


Figure 3.7: Goal state of the CLM: Activations of neurons that describe a correct assignment $y_{r\hat{\alpha}(r)}$ are set to one, while all others are set to zero.

and their corresponding target labels $\hat{\alpha}^i(r) = \hat{\alpha}(\mathbf{m}^i_r) \in \{1, \ldots, L^i\}$ is obtained, where L^i is the number of groups in the pattern $\mathcal{P}^i$. Unless otherwise stated, the convention is used that $r \in \{1, \ldots, N^i\}$ and that $\hat{\alpha}(r)$ denotes the target label for feature $\mathbf{m}_r$ in pattern $\mathcal{P}^i$, while $\beta, \beta' \neq \hat{\alpha}(r)$ denote other possible labels from $\{1, \ldots, L^i\}$.
To obtain a target state $\mathbf{y}^i$ from the desired labels, $\hat{\alpha}(r)$ the activations of neurons that describe a correct assignment $\hat{\alpha}(r)$ are set to one, while all others are set to zero.

$$y^i_{r\hat{\alpha}(r)} = 1; \quad y^i_{r\beta} = 0; \quad \text{for all } r, \beta \neq \hat{\alpha}(r). \tag{3.36}$$

The learning goal is to choose $f_{rr'}$, such that these target states are stable states of the CLM grouping dynamics, which implies that they must be consistent with respect to (3.7). Therefore, the target states (3.36) are substituted into (3.7) to obtain $(L-1)\sum_i N^i$ target consistency conditions

$$\textstyle\sum_{r'=1}^{N^i} f_{rr'} y^i_{r'\beta} < \sum_{r'=1}^{N^i} f_{rr'} y^i_{r'\hat{\alpha}(r)}. \tag{3.37}$$

It is typical for real world problems, that not all conditions (3.37) can be fulfilled simultaneously, because of inconsistencies or ambiguities in natural data. So, learning results in an optimization problem to violate the consistency of the hand-labeling for the training set as little as possible.

3.7.2 Introduction of Basis Functions

Since in natural problems the feature domain $\mathcal{F} \in \mathbb{R}^d$ often is a high dimensional discrete or non-finite set and the training examples cover only a small discrete subset of $\mathcal{F}$, the estimation of $f_{rr'}$ from (3.37) can be impracticable. To achieve generalization from the training data, it is necessary to reduce the number of parameters which have to be adapted during learning. This reduction can be realized

by the decomposition of the sought interaction function into a linear combination

$$f_{rr'} = \sum_{j}^{K} c_j g^j_{rr'} \tag{3.38}$$

of a set of basis functions $g^j_{rr'} = g^j(\mathbf{m}_r, \mathbf{m}_{r'})$, which contribute a priori knowledge about the grouping problem. So, the problem of learning is reduced to the estimation of the K coefficients of the basis functions. The definition of the basis functions (3.38) is substituted into the consistency conditions (3.37) to get a set of dimension-reduced consistency conditions of the form:

$$\sum_{j} c_j Z^k_j < 0 \quad \text{for all } k = (i, r, \beta \mid \mathbf{m}_r \in \mathcal{R}^i, \beta \neq \hat{\alpha}^i(r)), \tag{3.39}$$

where k is a super-index for the consistency conditions running over all combinations of pattern i, feature r, and label $\beta \neq \alpha^i(r)$ and the values Z^k_j describe the information from the training set and the basis functions by

$$Z^k_j = Z^{ir\beta}_j = \sum_{r' \mid \hat{\alpha}^i(r') = \beta} g^j_{rr'} - \sum_{r' \mid \hat{\alpha}^i(r') = \hat{\alpha}^i(r)} g^j_{rr'}. \tag{3.40}$$

The values Z^k_j are written as vectors $\mathbf{Z}^k = (Z^k_1, \ldots, Z^k_K)$. Each vector $\mathbf{Z}^k$ is associated exactly with one consistency condition in (3.37) and, therefore, is called a "consistency vector".

The observation of the consistency vectors in the weight space of learning parameters c_j gives a simple geometrical interpretation of the learning problem (see Figure 3.8). To fulfill the kth consistency condition $\mathbf{c} = (c_1, \ldots, c_K)^T$ has to be chosen from the opposite half space of $\mathbf{Z}^k$. Therefore, each new consistency condition restricts the area of suitable interaction weights $\mathbf{c}$. In order to achieve a better generalization to further yet unseen patterns, it can be tried to restrict this area further by the introduction of a positive margin variable $\kappa > 0$ in (3.39).

$$\sum_{j} c_j Z^k_j + \kappa < 0 \quad \text{for all } k = (i, r \mid \mathbf{m}_r \in \mathcal{R}^i, \beta \neq \hat{\alpha}^i(r)). \tag{3.41}$$

Obviously, this approach can only achieve good results, if the basis functions describe relevant aspects of the desired interaction function. An example for the difficulties in the manual design is given in the following, which describes the application of this learning method to fluorescence cell images. A strategy to generate suitable basis functions more automatically is presented in chapter 5 about the simplification of the learning approach.

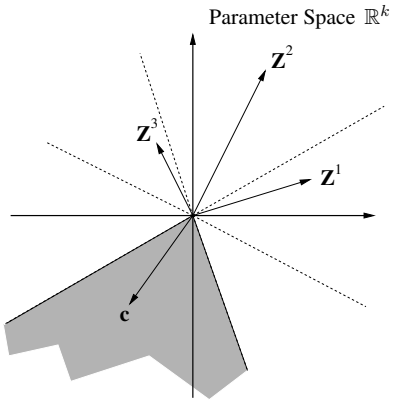


Figure 3.8: Geometry representation of the learning problem of QCO. The consistency vectors $\mathbf{Z}^k$ describe the learning problem. To fulfill the k-th consistency condition, the vector of learning parameters $\mathbf{c}$ has to be chosen from the opposite half space of the weight state space. Therefore, each new consistency condition restricts the area of suitable interaction weights further (shaded area).

3.7.3 Optimization of Consistency Conditions

In [59, 62], the optimization problem to satisfy (3.41) is solved by searching the minimum of the quadratic error

$$E_{QCO} = \sum_k \left(\sum_j c_j Z_j^k + \kappa \right)^2 . \tag{3.42}$$

This approach is more restrictive than the original learning problem (3.41), because it demands, that all consistency conditions are fulfilled in the same manner. However, applications of this approach for designing BSB associative memories have shown, that it is competitive to more sophisticated optimization methods [43]. The minimum of (3.42) is searched by gradient descent under additional constraints $|c_j| \leq 1$. This property should keep single interaction coefficients from obtaining large values, which might disturb the group formation process. E.g., if one basis function describes a pure self interaction $g_{rr'}^i = \delta_{rr'}$, it follows, that all Z_i^k are equal to one, such that the minimum of (3.42) can be found trivially by setting c_i to κ and all other parameters $c_j, j \neq i$ to zero. The result would be $f_{rr'} = \kappa\delta_{rr'}$, which, obviously, can not produce a reasonable grouping, since all lateral interactions a set to zero. However, if $|c_j| \leq 1$, it is claimed that this trivial solution is not allowed for $\kappa > 1$, such that all c_j have to be adapted to find a minimum of (3.42) and a suitable interaction function can be learned.
The qualitative performance of the QCO-approach is good [60], [62]. However, the computational performance suffers from the fact that all consistency conditions (3.37) have to be transformed by equations (3.40) and (3.41) to the space of learn-

ing parameters c_j, before the optimization step is performed according to the error function (3.42).

3.7.4 Application on Fluorescence Cell Images

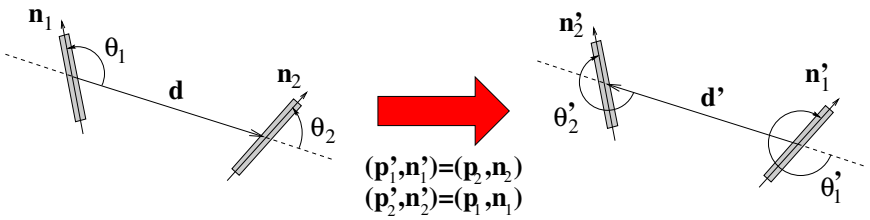


Figure 3.9: Relative orientations and symmetry under feature exchange (Figure and caption from [62]).

For a practical implementation of the learning method, the previous description still lacks an important ingredient: the exact definition of the basis functions $g^j_{rr'}$, which introduces problem specific knowledge about the relevant grouping principles into the interaction function. In [62], Wersing showed, how suitable basis functions can be designed by hand for the segmentation of fluorescence cell images, presented in section 3.6.

As already stated, the features $\mathbf{m}_r = (\mathbf{p}_r, \hat{\mathbf{n}}_r)$ are represented by position $\mathbf{p}_r = (p^x_r, p^y_r)$ and orientation $\hat{\mathbf{n}}_r = (\hat{n}^x_r, \hat{n}^y_r)$ ($\| \hat{\mathbf{n}}_r \| = 1$). Further, the interaction function depends on the relative angles between the two orientation vectors $\hat{\mathbf{n}}_r$ and $\hat{\mathbf{n}}_{r'}$ and the difference vector $\mathbf{d}$ as well as on the length of $\mathbf{d}$, where the two angles, called θ_1 and θ_2, always start in the direction of $\mathbf{d}$ and turn to the left, until they reach the two orientation vectors (see Fig. 3.9).

The designed basis functions describe a disjunct partitioning of the range of the parameters θ_1, θ_2 and $\| \mathbf{d} \|$, plus an additional basis function for the self-interaction in the background layer. The local distance is discretized into three intervals $\| \mathbf{d} \| < \frac{R}{2}$, $\frac{R}{2} < \| \mathbf{d} \| < R$, and $R < \| \mathbf{d} \|$, where R is the expected radius of the lymphocyte cells in the image. The angle θ_1 and θ_2 are discretized into eight orientation intervals by

$$s(\theta) = floor(8\theta/(2\pi)) \in \{0, \ldots, 7\}, \tag{3.43}$$

where $floor(x)$ is the largest integer number n for which $n < x$. There are 64 combination of the orientation intervals of θ_1 and θ_2. As can be seen in Fig. 3.9, some combinations are identical, if the ordering of the features is changed. These combinations have to be mapped on the same response of the basis functions $g^j_{rr'}$ to ensure the symmetry of the interaction $f_{rr'} = f_{r'r}$ under feature exchange. The

matrix

$$Q = \begin{pmatrix} 1 & 2 & 3 & 4 & 27 & 28 & 29 & 30 \\ 5 & 6 & 7 & 8 & 28 & 31 & 32 & 33 \\ 9 & 10 & 11 & 12 & 29 & 32 & 34 & 35 \\ 13 & 14 & 15 & 16 & 30 & 33 & 35 & 36 \\ 17 & 18 & 19 & 20 & 1 & 5 & 9 & 13 \\ 18 & 21 & 22 & 23 & 2 & 6 & 10 & 14 \\ 19 & 22 & 24 & 25 & 3 & 7 & 11 & 15 \\ 20 & 23 & 25 & 26 & 4 & 8 & 12 & 16 \end{pmatrix} \tag{3.44}$$

describes this symmetry and shows, that there actually exist only 36 different combinations of the orientation intervals of θ_1 and θ_2. The basis functions are now defined as membership functions of the different quantization intervals of θ_1, θ_2 and $\| \mathbf{d} \|$:

$$\begin{aligned}
g^1_{rr'} &= g^{1\alpha}_{rr'} = \delta_{\alpha 1}\delta_{rr'} \\
g^2_{rr'} &= g^{2\alpha}_{rr'} = \begin{cases} 1 & : \quad \alpha > 1 \text{ and } \| \mathbf{d} \| > R \\ 0 & : \quad \text{else} \end{cases} \\
g^{3\leq j\leq 38}_{rr'} &= g^{j\alpha}_{rr'} = \begin{cases} 1 & : \quad \alpha > 1, \| \mathbf{d} \| < \frac{R}{2} \text{ and } j = 2 + q(\theta_1, \theta_2) \\ 0 & : \quad \text{else} \end{cases} \\
g^{39\leq j\leq 74}_{rr'} &= g^{j\alpha}_{rr'} = \begin{cases} 1 & : \quad \alpha > 1, \frac{R}{2} < \| \mathbf{d} \| < R \text{ and } j = 2 + q(\theta_1, \theta_2) \\ 0 & : \quad \text{else} \end{cases}
\end{aligned} \tag{3.45}$$

where

$$q(\theta_1, \theta_2) = Q_{s(\theta_1)s(\theta_2)} \tag{3.46}$$

returns the index of the discretization of θ_1 and θ_2 in Q and $\alpha = 1$ belongs to the interaction in the background layer, while $\alpha > 1$ belongs to the other layers.
The first basis function of (3.45) defines the self-interaction in the background-layer. The second basis function expresses a uniform interaction, if the distance between the features is bigger than the expected cell radius R. The other 72 basis functions are the combinations of the 2×36 remaining quantization intervals of θ_1, θ_2 and $\| \mathbf{d} \|$. This gives a total number of 74 basis functions.
Wersing showed that suitable interaction coefficients c_j of the defined basis functions can be estimated with the presented QCO learning approach to solve the cell segmentation problem, which constitutes an enormous simplification of the design process of the problem specific interaction function. Nevertheless, this approach still has some difficulties. The case differentiation in (3.45) shows, that there is still a high demand on the user, who must decide for quantization levels and combinations of feature properties. Further, the presented basis functions have to be evaluated for a high number of times in the learning process: equation (3.40) has to be computed for each combination of a feature in the training set with a basis function and a possible label, where each computation of (3.40) demands, that a basis function has to be evaluated for each feature in the actual training pattern. If the CLM is applied on image data, where each pixel represents a feature, this means an unreasonable computational effort.

3.8 Related Architectures and Algorithms

The presented CLM architecture can be extended to general labeling problems by connecting the neurons with symmetric inter layer weights $f_{rr'}^{\alpha\beta}$, expressing the compatibility of the assignment of feature $\mathbf{m}_r$ to label α with the assignment of feature $\mathbf{m}_{r'}$ to label β. This section refers to several alternative approaches that use lateral interaction weights to implement feature binding and segmentation processes and either suggest own methods of learning or would profit from the development of a learning algorithm.
Presented are the binary recurrent neural networks Competitive Hopfield Neural Network (CHNN) [6] and Contextual-Content-Based Hopfield Neural Cube (CCBHNC) [4], since they are based on the same layer-wise architecture as the CLM, the method of Relaxation Labeling (RL) [49], which treats labeling processes in a probabilistic framework, the Energy-based Cluster Update (ECU) as representative of spin models, and the Locally Excitatory Globally Inhibitory Oscillatory Network (LEGION) [52], where binding is implemented by the principle of temporal correlation of oscillating units, as it is proposed in [55], [56], [57].

3.8.1 Competitive Hopfield Neural Network (CHNN) and Contextual-Context-Based Hopfield Neural Cube (CCBHNC)

The CHNN [6] and the CCBHNC [4] use the same layer-wise architecture of neurons $x_{r\alpha}$ as the CLM, but consist of binary neurons instead of linear threshold neurons, where the vertical WTA loop in the CLM is replace by an explicit WTA update rule

$$x_{r\alpha} = \begin{cases} 1 & : \quad F_{r\alpha} = \max_\beta\{F_{r\beta}\} \\ 0 & : \quad \text{otherwise} \end{cases} . \tag{3.47}$$

$F_{r\alpha}$ is the support the neuron $x_{r\alpha}$ receives form all other neurons in the network. Thereby the CHNN and CCBHNC differ in the kind of lateral connections $f_{rr'}^{\alpha\beta}$, which enables them to solve different types of labeling and assignment problems.
In [6], the CHNN is used for polygon approximation, where a closed polygon contour given by a set of L points $\mathbf{m}_\alpha = \mathbf{p}_\alpha = (x_\alpha, y_\alpha)^T$ with cyclic indices $\alpha + L = \alpha$, which should by approximated by a sequence of line segments connecting a set of breakpoints $\mathbf{m}_r$, $r = 1, \ldots, N$, $r + N = r$, $N < L$ on the given contour. Therefore, the given points $\mathbf{m}_\alpha$ are associated with the layers of the CHNN and the indices r of the breakpoints are associated with the columns. Each neuron receives only input from all neurons in the previous and subsequent column (breakpoint).

$$F_{r\alpha} = \sum_{\beta=1}^{L} f^{\alpha\beta} x_{(r-1)\beta} + f^{\alpha\beta} x_{(r+1)\beta}. \tag{3.48}$$

Two points on the contour interact with

$$f^{\alpha\beta} = \begin{cases} -\infty & : \quad \mathbf{p}_\alpha = \mathbf{p}_\beta \\ -\max_{\mathbf{p} \in \widetilde{\mathbf{p}_\alpha \mathbf{p}_\beta}} \{d(\mathbf{p}, \overline{\mathbf{p}_\alpha \mathbf{p}_\beta})\} & : \quad \mathbf{p}_\alpha \neq \mathbf{p}_\beta \end{cases} , \tag{3.49}$$

where $d(\mathbf{p}, \overline{\mathbf{p}_\alpha \mathbf{p}_\beta})$ is the distance of a point $\mathbf{p}$ to the line connecting $\mathbf{p}_\alpha$ and $\mathbf{p}_\beta$ and $\widetilde{\mathbf{p}_\alpha \mathbf{p}_\beta}$ is the clockwise pass on the given contour starting in $\mathbf{p}_\alpha$ and ending in $\mathbf{p}_\beta$.
The WTA-rule (3.47) guarantees the unique assignment of the breakpoints r to the contour points $\mathbf{m}_\alpha$ and enables the CHNN to deal with strictly negative $f^{\alpha\beta} < 0$ and non-symmetric $f^{\alpha\beta} \neq f^{\beta\alpha}$ lateral weights, while the infinite negative self-interaction weights in the first line of (3.49) ensure the unique assignment of the contour points to the break points.
In [4], the CCBHNC is applied to the segmentation of Computed Tomography (CT), Magnet Resonance (MR) and Single Photon Emission Computed Tomography (SPECT) images using a similar approach like the CLM color segmentation example in section 3.6.2, based on the local intensity I_r and position $\mathbf{p}_r = (x_r, y_r)$ of image pixels $\mathbf{m}_r = (\mathbf{p}_r, I_r)$.

$$F_{r\alpha} = \frac{1}{2} \sum_{r'} \sum_{\beta} f_{rr'}^{\alpha\beta} x_{r\beta}. \tag{3.50}$$

$$f_{rr'}^{\alpha\beta} = -(A \frac{|I_r - I_{r'}|}{\max_r\{I_r\}} + B\Phi_{rr'}(1 - \delta_{\alpha\beta}), \tag{3.51}$$

where $\Phi_{rr'}$ is the membership function of a certain neighborhood $\mathcal{N}_r$ of the pixel $\mathbf{m}_r$

$$\Phi_{rr'} = \begin{cases} 1 & : \quad \mathbf{m}_r \in \mathcal{N}_r \\ 0 & : \quad \mathbf{m}_r \notin \mathcal{N}_r \end{cases}, \tag{3.52}$$

$\delta_{\alpha\beta}$ is the Kronecker Delta between label α and β, and A and B are scaling factors to weight the influence of intensity distance and uniform labeling in the neighborhood $\mathcal{N}_r$. Again, the lateral interactions are strictly negative $f_{rr'}^{\alpha\beta} < 0$. Each neuron interacts with all neurons in the same layer according to the distance in intensity and with all neurons in neighboring columns, depending whether they are assigned to the same label or not.

3.8.2 Relaxation Labeling (RL)

In the framework of RL, introduced by Rosenfeld, Hummel and Zucker [49], the assignment variables $x_{r\alpha}$ are interpreted directly as probabilities for assigning a feature $\mathbf{m}_r$ to the labels $\alpha = 1, \ldots, L$, which means they are constrained by

$$\sum_{\alpha} x_{r\alpha}(t) = 1. \tag{3.53}$$

Starting from an initial estimation $x_{r\alpha}(0)$ for the labeling probabilities, the algorithms relaxes inconsistencies in the labeling expressed by compatibilities $f_{rr'}^{\alpha\beta}$ with an iterative update rule:

$$x_{r\alpha} = \frac{F_{r\alpha} x_{r\alpha}(t)}{\sum_\beta F_{r\beta} x_{r\beta}(t)}. \tag{3.54}$$

The support $F_{r\alpha}$ should return positive values to guarantee, that the probabilities $x_{r\alpha}$ stay in $[0,1]$, which is usually ensured by shifting the lateral interactions $f_{rr'}^{\alpha\beta} \in [-1,1]$ into the positive quadrant $f'^{\alpha\beta}_{rr'} = f_{rr'}^{\alpha\beta} + 1 > 0$.

$$F_{r\alpha}(t) = \frac{1}{N} \sum_{r'\beta} (1 + f_{rr'}^{\alpha\beta}) x_{r'\beta}(t) \tag{3.55}$$

The probabilities $x_{r\alpha}$ converge under the iterative update (3.54) to a consistent labeling depending on the initial guess, which, however, does not necessarily means, that the output labeling describes a unique assignment with

$$\forall r: \quad x_{r\hat{\alpha}(r)} = 1 \wedge \quad \forall \beta \neq \hat{\alpha}(r): \quad x_{r\beta} = 0. \tag{3.56}$$

Rosenfeld, Hummel and Zucker suggest to specify the lateral interactions $f_{rr'}^{\alpha\beta}$ by the correlation of labels α and β, computed from the probabilities $P(\alpha)$ and $P(\beta)$ for feature $\mathbf{m}_r$ and $\mathbf{m}_{r'}$ and the joint probability $P(\alpha,\beta)$ of observing both labels in parallel at $\mathbf{m}_r$ and $\mathbf{m}_{r'}$:

$$f_{rr'}^{\alpha\beta} = \frac{P(\alpha,\beta) - P(\alpha)P(\beta)}{[(P(\alpha) - P^2(\alpha))(P(\beta) - P^2(\beta))]^{\frac{1}{2}}} \in [-1,1]. \tag{3.57}$$

Other approaches interprete the lateral interactions weights as mutual information between labeling feature $\mathbf{m}_r$ with α and $\mathbf{m}_{r'}$ with β [44] or suggest supervised learning methods [45]. Kittler and Illingworth [26] refer to a wide range of applications based on RL, covering fields of feature (e.g. edge) enhancement, figure-background separation and shape and stereo matching.

3.8.3 Energy-Based Cluster Update (ECU)

In [42], the ECU algorithm is applied to color segmentation problems given by pixels $\mathbf{m}_r = (\mathbf{p}_r, I_r)$. The assignment variables $x_{r\alpha}$ are replaced by a set of spin variables $x_1, \ldots, x_N \in \{1, \ldots, L\}$, which can take one of L different spin states. The output of ECU is given by a total spin state $\mathbf{x} = (x1, \ldots, x_N)$ which minimizes the energy

$$E(\mathbf{x}) = \sum_{r=1}^{N} \left[J_r - \sum_{r' \in \mathcal{N}_r} f_{rr'} \delta_{x_r x_{r'}} \right]. \tag{3.58}$$

J_r is a global inhibition and the lateral interactions are chosen by

$$f_{rr'} = 1 - \frac{|I_r - I_{r'}|}{\Theta}. \tag{3.59}$$

$\mathcal{N}_r$ is the neighborhood of feature $\mathbf{m}_r$, Θ is the intrinsic short range similarity (compare equation (3.28) in section 3.6.2), and $\delta_{x_r x_{r'}}$ is the Kronecker Delta between the spin states of x_r and $x_{r'}$. To find the minimum of (3.58), ECU starts

with a random initialization of the spin variables and divides the spin variables into coherent clusters of uniform spin states ($\delta_{x_r x_{r'}} = 1$). Therefore, each spin x_r is assigned initially to its own cluster C_r and two clusters C_r and $C_{r'}$ of two neighboring features are merged with the probability

$$P(\text{merging } C_r \text{ and } C_{r'}) = 1 - exp\left(-\frac{1}{2T} f_{rr'} \delta_{x_r x_{r'}}\right), \tag{3.60}$$

where T is a annealing temperature which controls the size of the merged clusters: large values of T allow only small clusters, while small values of T allow large clusters.
After the clusters are build, the amount of energy is computed for changing the state of all spin variables in cluster C_i to spin α.

$$E(C_i \rightarrow \alpha) = \sum_{r \in C_i} (J_r - \sum_{r' \in \mathcal{N}_r, C_{r'} \neq C_i} f_{rr'} \delta_{x_r x_{r'}}), \tag{3.61}$$

which allows to compute the probability of switching the whole cluster C_i to spin state α:

$$P(C_i \rightarrow \alpha) = \frac{E(C_i \rightarrow \alpha)/T}{\sum_\beta E(C_i \rightarrow \beta)/T}. \tag{3.62}$$

The minimum of (3.58) is found by performing alternating clustering (3.60) and cluster update (3.62) steps under gradually lowering of the annealing temperature T. An comparison of ECU and the CLM approach in section 3.6.2 showed [61], that both algorithms achieve qualitative similar results. However, ECU showed a severe faster rate of convergence, especially, when the range of lateral interactions is restricted to a relative small pixel neighborhood $\mathcal{N}_r$, such that that CLM dynamics requires very slow annealing speed to reveal the desired segmentation.

3.8.4 Locally Excitatory Globally Inhibitory Oscillator Networks (LEGION)

While the CLM describes a binding or grouping of features by the principle of spatial coactivation in the same layer, the LEGION approach [52] follows the alternative principle of temporal correlation [55], [56], [57]. Each feature $\mathbf{m}_r$ in the input is associated with an oscillatory unit, specified by an excitatory and inhibitory subunit x_r and y_r.

$$\dot{x}_r = 3x_r - x_r^3 + 2 - y_r + \rho + h_r + F_r, \tag{3.63}$$

$$\dot{y}_r = \epsilon(\gamma(1 + tanh(x_r/\beta)) - y_r), \tag{3.64}$$

where ρ is an amplitude of Gaussian noise, h_r is an external input and F_r is the input from other units. A single unit with positive input $h_r > 0$ and without connection to other units ($F_r = 0$) describes a stable periodic orbit with different time phases of increasing and decreasing values of x_r and y_r, where the adaptation

rate ϵ and the parameters β and γ determine the exact shape of the orbit and the time the unit spends in the respective phases of oscillation. For $h_r < 0$, a single unit converges to a stable fixed point and stops oscillating.
Connecting several of such oscillator units with excitatory weights $f_{rr'}$ results in a temporal synchronization of the units firing rates. In contrast, inhibitive connections $f_{rr'} < 0$ result in a temporal desynchronization of the respective units. The synchronal oscillation of different units expresses the binding of the associated features to the same group.
As denoted by the name LEGION, neighboring units are connected by positive interactions $f_{rr'} > 0$, while simultaneously they are connected through an inhibitive weight J with a global inhibitor z.

$$F_r = \sum_{r' \in \mathcal{N}_r} f_{rr'} \sigma_\infty(x_{r'}, \theta_x) - J\sigma_\infty(z, \theta_{xz}). \tag{3.65}$$

where $\sigma(x, \theta)$ is the Fermi function

$$\sigma_\infty(x, \theta) = \frac{1}{1 + exp\{-\kappa(x - \theta)\}}. \tag{3.66}$$

The activation of the global inhibitor z is defined by

$$\dot{z} = \Phi(\xi - z), \tag{3.67}$$

where Φ is the reaction rate of the global inhibitor and the global inhibitor is excited ($\xi = 1$), if the activation of at least one oscillator exceeds of threshold x_{zx}, and it is suppressed ($\xi = 0$), if all oscillator are inactive.

$$\xi = \begin{cases} 1 & : \quad \exists r : x_r \geq \theta_{xz} \\ 0 & : \quad \forall r : x_r < \theta_{xz} \end{cases}. \tag{3.68}$$

In [52], LEGION is used to segment coherent regions in binary images ($\mathbf{m}_r = (\mathbf{p}_r, I_r), I_r \in \{0, 1\}$), where the lateral interaction are simply set to one for all oscillators in the direct neighborhood $\mathcal{N}_r$ of $\mathbf{m}_r$:

$$f_{rr'} = \begin{cases} 1 & : \quad \mathbf{m}_{r'} \in \mathcal{N}_r \\ 0 & : \quad \mathbf{m}_{r'} \notin \mathcal{N}_r \end{cases}, \tag{3.69}$$

and the external input of the oscillators is set to

$$h_r \begin{cases} > 0 & : \quad I_r = 1 \\ < 0 & : \quad I_r = 0 \end{cases}. \tag{3.70}$$

Thereby connected regions of $I_r = 1$ are synchronized through the lateral interactions, while the global inhibitor causes desynchronization of disconnected regions of $I_r = 1$, and the regions of $I_r = 0$ stay permanent inactive.
In [58], LEGION is demonstrated on the segmentation of real world satellite images with pixels ($\mathbf{m}_r = (\mathbf{p}_r, I_r), I_r \in \{0, \ldots, 255\}$), where the lateral interactions are set to

$$f_{rr'} = \frac{255}{1 + |I_r - I_{r'}|}. \tag{3.71}$$

3.9 Summary

This chapter has described the grouping process implemented by the dynamics of the Competitive Layer Model. Cited results have shown that the dynamics converges to consistent attractor states in the sense of grouping principles encoded by pairwise compatibility weights between feature representative neurons. An efficient simulation algorithm is available that includes an annealing process implemented by self-inhibitory weights of the CLM neurons.
Several examples of hand-tuned interaction functions have shown the applicability of the CLM on a wide range of segmentation tasks. However, they have also shown the necessity of an automatic learning method to simplify the design of such functions.
The original learning approach of Wersing was recapitulated. The learning problem is formulated by constructing a set of CLM target states from hand-labeled training patterns. These states are transformed into a set of linear constraints on the lateral interaction weights that characterize the attractors of the CLM-dynamics. To make the optimization problem of estimating feasible interaction weights practicable, the number of learning parameters is reduced by the projection of the linear constraints to a set of basis functions. The new learning parameters can then be estimated by gradient descent according to an error function that punishes the violation of the projected constraints.
As will be shown in the application examples of chapter six, this learning approach reaches a reasonable level of quality, but can be computational expensive, because all consistency conditions have to be projected onto the basis functions. Further, it demands some experience and skills from the user in designing suitable basis functions. To handle these drawbacks the learning algorithm is improved in chapter five.

Chapter 4

Lateral Interactions & the Grouping Process

The last chapter has shown that the grouping result of the CLM dynamics is mainly determined by the lateral interaction weights $f_{rr'}$. Especially, section 3.4 has stressed the importance of the eigenvalues and eigenvectors of the interaction matrix F during the self inhibitory annealing process. As preparation for the design of an effective and efficient learning algorithm the present chapter gives abstract examples of ideal grouping problems to discuss properties like the automatic adaption to the number of groups in the input or the robustness of the grouping process against noise. Thereby it introduces a quality measure to estimate the correctness of different output groupings in relation to a specified target labeling. This measure will be used in the later chapters to evaluate the learning success in adapting specified grouping behaviors.
Further, this chapter gives a more detailed insight into the annealing process, especially for high dimensional data, like image data. This process is characterized by a sequence of prototypic attractor states. Based on the ordering of these states the concept of saliency of presented groups is introduced which opens new view points to the problem of figure-background separation and motivates an attention-based control of the dynamics presented in chapter 8.

4.1 Degrees of Freedom and Cardinality of Grouping

To observe the influence of the lateral interactions on the grouping process independently from the influence of other parameters of the CLM the experiments of this chapter follow the conventions:

1. All neuron inputs are equal and constant:

$$\forall r: \quad h_r := 1. \tag{4.1}$$

2. The strength of the vertical inhibitory weights J is specified uniquely by the matrix of lateral interactions:

$$J := 2 \max_r \sum_{r \neq r'} \max(0, f_{rr'}). \tag{4.2}$$

3. The self-interaction weights are set to zero:

$$\forall r : \quad f_{rr} := 0. \tag{4.3}$$

Equation (4.1) provides that the basic amount of activity that can be assigned to the neurons in a column of the CLM is the same for all columns. Equation (4.2) guarantees that a scaling of the lateral interaction Matrix $F' := aF$ is accompanied by an equivalent scaling of the vertical competition weights $J' := aJ$. Both scalings indicate a scaling of the whole CLM energy function:

$$\begin{aligned} E' &= -J' \textstyle\sum_{r\alpha} h_r x_{r\alpha} + \frac{1}{2} J' \sum_r \sum_{\alpha\beta} x_{r\alpha} x_{r\beta} - \frac{1}{2} \sum_\alpha \sum_{rr'} f'_{rr'} x_{r\alpha} x_{r'\alpha} \\ &= -aJ \textstyle\sum_{r\alpha} h_r x_{r\alpha} + \frac{1}{2} aJ \sum_r \sum_{\alpha\beta} x_{r\alpha} x_{r\beta} - \frac{1}{2} \sum_\alpha \sum_{rr'} a f_{rr'} x_{r\alpha} x_{r'\alpha} \\ &= -a \left(J \textstyle\sum_{r\alpha} h_r x_{r\alpha} + \frac{1}{2} J \sum_r \sum_{\alpha\beta} x_{r\alpha} x_{r\beta} - \frac{1}{2} \sum_\alpha \sum_{rr'} f_{rr'} x_{r\alpha} x_{r'\alpha} \right) \\ E' &= aE \end{aligned} \tag{4.4}$$

Since the CLM dynamics minimizes E, as any other E', the choice of the scaling a only effects the concrete step size of the dynamics in direction of the gradient of E but not the location of the attracted minima, as long as the dynamics is not trapped in local minima of E as a result of the changed step size.
To prevent such suboptimal attractor states self-inhibitory annealing is employed which is controlled by the pseudo temperature T. Since there is a contradiction between the inhibitory pseudo temperature and possible self-excitatory lateral connections f_{rr}, equation (4.3) sets all diagonal elements of the lateral interaction matrix F to zero. This convention could violate the CLM assignment theorem which requires self-excitatory weights $f_{rr} > 0$ to guarantee the unique assignment of all features to the layers. To fulfill the assignment theorem anyway, annealing can be performed to a pseudo temperature $T = -\varepsilon$ which lies slightly beneath zero. This strategy is equivalent to shifting all eigenvalues of F into the positive quadrant by ε. Thereby equation (4.2) guarantees that the CLM convergence theorem is fulfilled as long $\varepsilon < \max_r \sum_{r \neq r'} \max(0, f_{rr'})$.
The interaction matrix F of a grouping problem with N features consists of N^2 weights $f_{rr'}$, but since F is symmetric and (4.3) erased all diagonal elements of F, there exist only

$$M = \sum_{i=0}^{N} i - 1 = \frac{1}{2} N(N-1) \tag{4.5}$$

different interaction weights $f_{rr'}$, $r, r' \in \{1, \ldots, N\}$, $r > r'$. Thus choosing an interaction matrix F corresponds to choosing a point from $\mathbb{R}^M$. This situation is

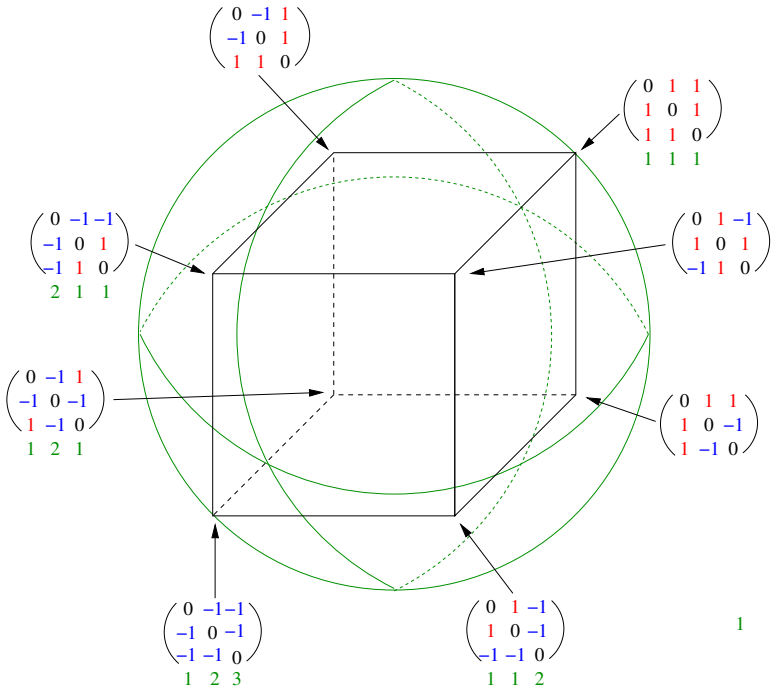


Figure 4.1: Weight space for grouping problems with three features. The green sphere shows the sphere of interaction weights that describe different grouping processes for three features. The corners of the black cube correspond to the eight binary matrices that lie on this sphere. Only five of these eight matrices are consistent with one of the five possible groupings (indicated by green labels) for the three features.

visualized by the $3D$ cube in Fig. 4.1 for the case of $N = 3$ features. Since the grouping behavior is invariant (besides a modified gradient step size) to a scaling of F, the space of possible interaction matrices can be restricted further to a hyper sphere in $\mathbb{R}^M$ with a fixed radius. E.g., for $R = \sqrt{M}$ the hyper sphere contains all 2^M symmetric matrices of binary off-diagonal elements $f_{rr'} \in \{-1, 1\}, r \neq r'$ which in the following are simply called "binary matrices".
Each point on this hyper sphere in $\mathbb{R}^M$ is mapped by the application of the CLM dynamics on the respective interaction matrix F onto an output labeling. There exists only a limited number of possible output labelings. Figure 4.2 sketches a recursive method to construct all possible labelings for a grouping problem. Two labelings are assumed to be different, if they can not be mapped onto each other by permutation of the labels. The trivial origin for $N = 1$ is the case that only one feature exists that is assigned to the label one. In the following recursive steps each grouping on the level N splits into $L + 1$ new groupings on the level $N + 1$, where L is the maximal label in the original grouping and the new feature can be either assigned to any of the existing labels or to a newly introduced label $L + 1$.

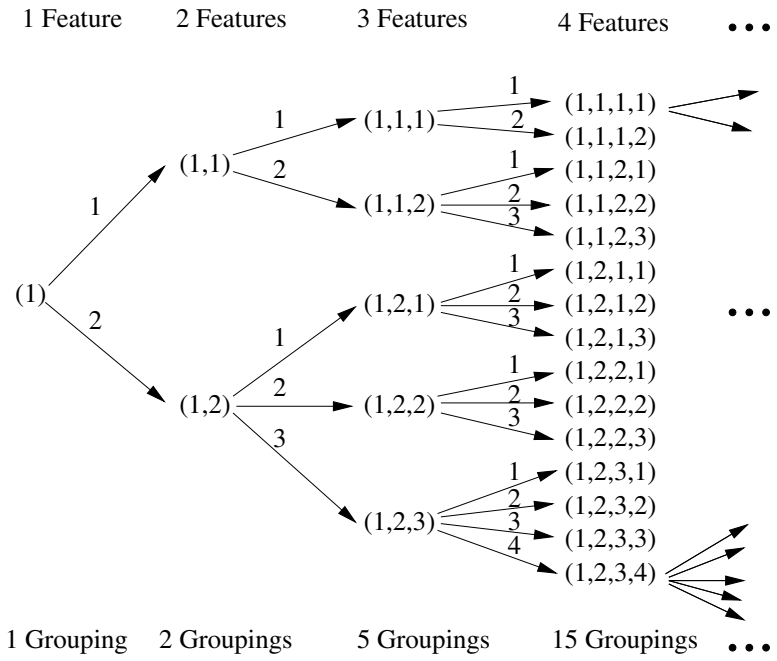


Figure 4.2: Sketch for the recursive construction of all possible groupings for a pattern with N features. Starting from the trivial case of a single feature, which is assigned to the one and only group, each labeling is split into L+1 new labelings on the next level, where L is the maximal index of the labels at the actual level. The number of possible labelings explodes, but stays clearly below N!.

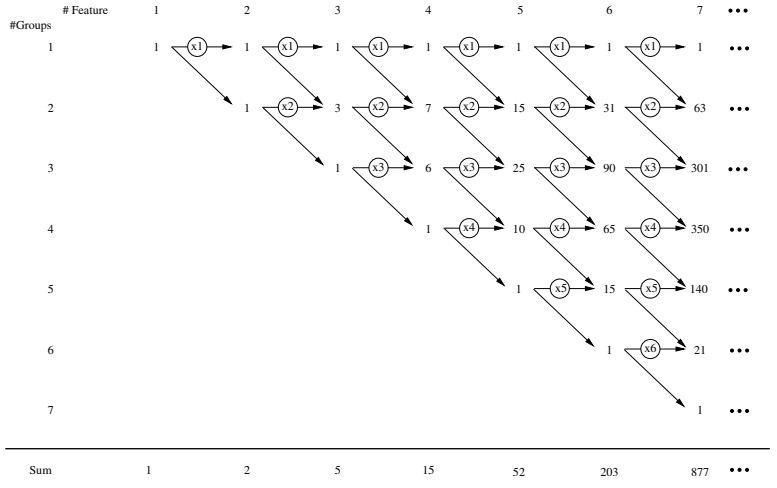


Figure 4.3: Sketch for the iterative increase of the number of groupings.

# features	# groupings	# weights	# binary matrices
2	2	1	2
3	5	3	8
4	15	6	64
5	52	10	1024
6	203	15	32768
7	877	21	$\approx 2,1 \cdot 10^6$
8	4149	28	$\approx 2,7 \cdot 10^8$
9	21147	36	$\approx 6,9 \cdot 10^{10}$
10	115975	45	$\approx 3,5 \cdot 10^{13}$

Figure 4.4: Table of the number of groupings and the number of binary matrices in the weight space for $N = 2$ to $N = 10$.

Figure 4.3 shows that the number of possible groupings explodes super exponentially with the number of features N. However the number of possible groupings increases slower than $N! = \Pi_{i=1}^{N} i$.
Each grouping can be associated with an binary interaction matrix on the hyper sphere in $\mathbb{R}^M$, with

$$f_{rr'} = \begin{cases} 0 & : \quad r = r' \\ 1 & : \quad r \neq r' \wedge \hat{\alpha}(r) = \hat{\alpha}(r') \\ -1 & : \quad r \neq r' \wedge \hat{\alpha}(r) \neq \hat{\alpha}(r') \end{cases} , \tag{4.6}$$

where all features that are assigned to the same label are connected by $f_{rr'} = 1$, while all features that are assigned to different labels are connected by $f_{rr'} = -1$. This structure is called in the following an "ideal block diagonal structure" independent of a concrete ordering of the features, because if the features are ordered according to their assigned labels ($r > r' \Leftrightarrow \hat{\alpha}(r) > \hat{\alpha}(r')$) the 1s appear in blocks along the main diagonal, while the -1s appear in all off-diagonal blocks. Compared to the set of all binary matrices which increases according to N with $2^{\frac{1}{2}N(N-1)}$ faster than $N!$ the set of ideal block diagonal matrices of possible groupings is distributed sparsely for high number of features.
Figure 4.4 shows how the numbers of possible groupings and possible binary matrices increases from $N = 2$ to $N = 10$. The matrix only consisting of -1 corresponds to the highest segmentation level, where each feature is assigned to its own group, while the matrix only consisting of 1s corresponds to the lowest segmentation level, where all features are assigned to the same group. A trajectory between these two extreme cases along the surface of the weight hyper sphere corresponds to a decrease or increase of the segmentation level by splitting or merging existing groups.
The following experiments shall demonstrate the robustness of the grouping result against errors in the interaction matrix. Therefore a quality measure is defined

which estimates the overlap between a desired target labeling and an existing CLM output labeling.

4.2 Evaluation of the Grouping Success

To be able to evaluate the success of learning a specified grouping behavior, a suitable criterion for the achieved grouping quality has to be defined. This is done by comparing the labels $\hat{\alpha}^C(r)$ resulting from the CLM-dynamics with a set of target labels $\hat{\alpha}^T(r)$, which, in the case of natural input patterns, are given by a human user. In the case of artifical input patterns, they are provided directly from the respective pattern generator.
The grouping quality measure Q computes the percentage of features that have been labeled correctly by the CLM-dynamics compared to the target labeling.
Assuming that there exist L^T target labels $\alpha^T = 1, \ldots, L^T$ and L^C layers $\alpha^C = 1, \ldots, L^C$ in the CLM, the entries of an $L^T \times L^C$ matrix $O^{(0)}$ are initialized by the number of features that are assigned in the target labeling to α^T and in the CLM-answer to α^C.

$$O^{(0)}_{\alpha^T \alpha^C} = \#\mathbf{m}_r \text{ with } \hat{\alpha}^T(r) = \alpha^T \wedge \hat{\alpha}^C(r) = \alpha^C. \tag{4.7}$$

The maximal entry of $O^{(0)}$ is selected, which describes the largest overlap between a group in the target labeling and the CLM-answer. All other entries in the same column and row are erased. From the resulting matrix $O^{(1)}$ the second largest entry is selected and this procedure is iterated for the number of target labels L^T. At the end of this procedure the sum of remaining entries in $O^{(L^G)}$ gives the number of correctly labeled features. Divided by the total number of features this number Q lies within $[0, 1]$ and can be interpreted as the percentage of overlap between the target labeling and the CLM-output (Fig. 4.5 sketches the process for an example with $N = 12$ features and $L^T = L^C = 4$ groups).

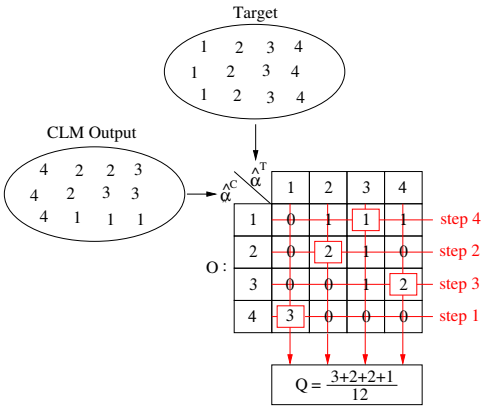


Figure 4.5: Sketch for the evaluation of the grouping quality.

The maximal quality $Q = 1$ is reached, when the target and the CLM labeling are equivalent, while values close to 1 indicate minor errors in the labeling. The minimal Quality corresponds to $Q = \frac{1}{N}$, in the case where all features in the target labeling are assigned to the same label, while in the CLM labeling each feature is assigned to its own label, or vice versa.

Disadvantages of the quality measure are that many labelings are mapped onto the same quality value, especially for medium values (see examples in Fig. 4.6), and that suboptimal quality values give no information about the type of errors in the CLM labeling, e.g., whether they were caused by oversegmented, merged or overlapping groups.

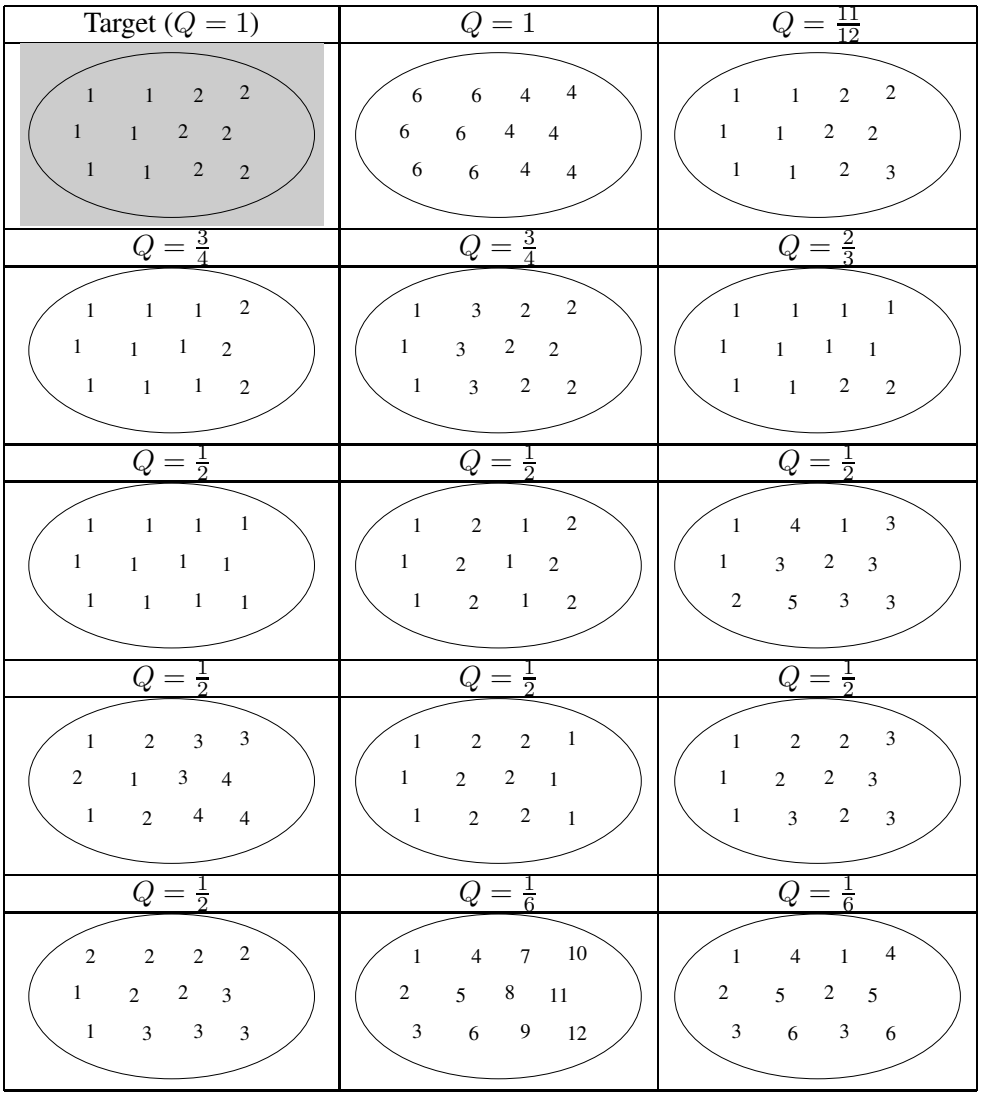


Figure 4.6: Quality values for different labelings of a 12-feature pattern compared to the shaded target labeling.

4.3 The Trajectory of the Annealing Process

The qualitative evaluation of the achieved grouping result is an important indicator to investigate the influence of the lateral interaction matrix on the dynamics (3.13). However, another important indicator is the trajectory of the network states that lead to the output grouping. To show this, the following experiment investigates the annealing process specified by an *ideal block diagonal matrix* from an abstract image segmentation problem.

The observed pattern (see Fig. 4.7) consists of $N = 900$ features which are arranged in a 30×30 image and shall be divided into five disjoint groups of 390, 236, 188, 81 and 5 features. The segmentation is abstracted from a concrete application, like color or texture segmentation, such that the features $\mathbf{m}_r = (x_r, y_r)$ are assumed in first place to be ordered according to the target labels to visualize the *ideal block diagonal* structure that is specified by (4.6).

If the diagonal elements of this matrix were set to one ($\forall r : f_{rr} = 1$), this matrix would have only five linear independent columns respectively rows which would mean that only five eigenvalues $\lambda_1, \ldots, \lambda_5$ would be unequal to zero, with $\sum_{i=1}^{N} \lambda_i = N$. Erasing the diagonal elements corresponds to a shift of all eigenvalues by -1 which results in the eigenvalue spectrum on the right hand side of Fig. 4.7, where $\sum_{i=1}^{N} \lambda_i = 0$ holds. The eigenvectors corresponding to the four largest eigenvalues describe mainly the separation of one of the five groups from the rest of the pattern, as can be seen in the first row of Fig. 4.8.

Up to now, the pattern shows no clear interpretation of the groups, besides their size. Therefore it is transformed into a more reasonable structure by permuting the features such that the groups describe several interlaced circular regions. To construct the equivalent interaction matrix with (4.6), the columns and rows of the original block diagonal matrix have to be permuted in the same way as the features. The permutation of the features has no influence on the size of the groups or the eigenvalues of the interaction matrix. Only the components of the eigenvectors of F are permuted in the same way as the features.

According to Wersing's eigensubspace analysis, the eigenvalues of F mark characteristic thresholds for the pseudo temperature T which activate corresponding AC eigenmodes and change the attractor space of the CLM dynamics during the self-inhibitory annealing process (see [61] or section 3.4).

To visualize this process in Fig. 4.9, annealing is started with $T > \lambda_1(F)$ and the neuron activations of an $L = 20$-layered CLM with $L \cdot N = 18000$ neurons are observed under gradually lowering T to zero.

Therefore, the maximal activity in a present CLM state is normalized to an intensity value of 255. Thus each layer of the CLM can be displayed by a 30×30 activity image whose pixel values lie in $[0, 255]$. A complete CLM state is visualized by a concatenating of the $L = 20$ layer activity images to a 2×10 matrix of such images.

To highlight the information from neurons with low activation a second visualization shows the same activity images in a logarithmic scale. A third visualization

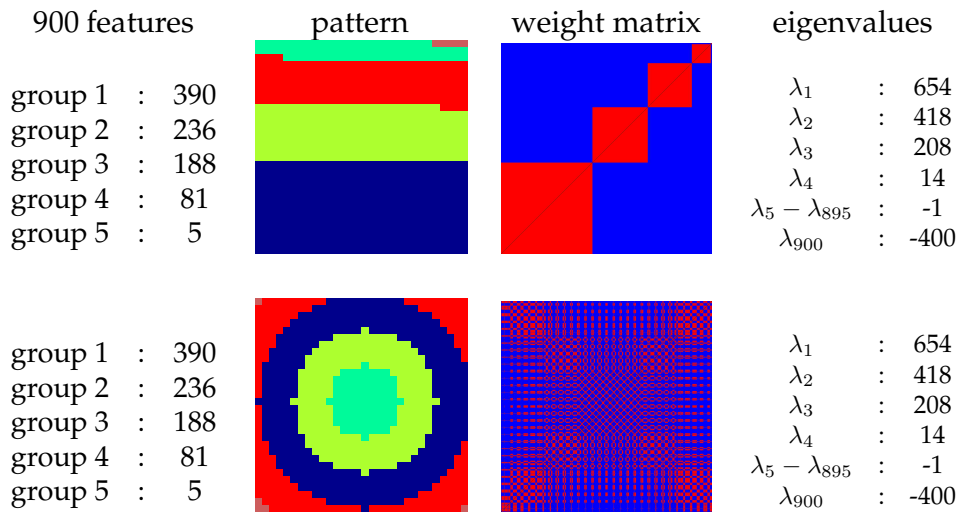


Figure 4.7: Example pattern with 900 features. The table an the left shows the distribution of the 900 features to five groups. The features are arranged in a 30×30 image. In the upper case the features are ordered according to their target labels. The corresponding interaction matrix shows an ideal block diagonal structure of 1's (red) and -1's (blue) lateral interaction weights $f_{rr'}$, where the self-interaction-weights f_{rr} are set to zero (black). It's eigenvalues are shown on the right. The lower case shows the same pattern and it's corresponding interaction matrix after permutation of the features which has no effect on the size of the groups nor the eigenvalues of the interaction matrix.

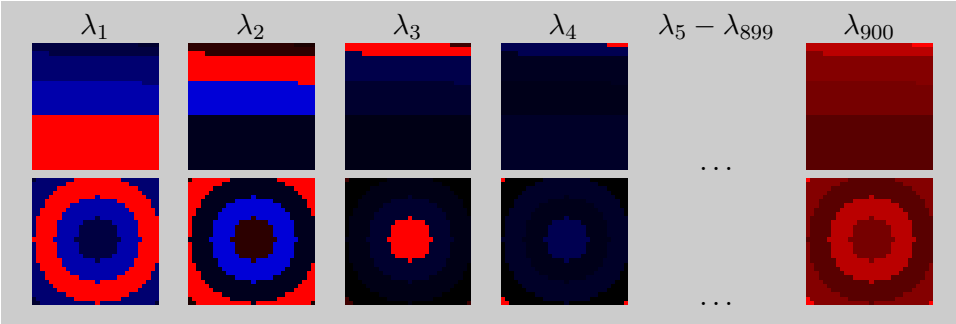


Figure 4.8: Eigenvectors of the interaction matrices shown in Fig. 4.7. The eigenvectors are scaled, such that their components cover the interval [-255,255]. The positive components are displayed by intensities in red color, while the negative components are displayed by intensities in blue color. The top row shows the eigenvalues, if the features are ordered according to the labeling. The lower row shows the eigenvectors, if the features are permuted. In this case the components of the eigenvectors have to be permuted in the same way as the features.

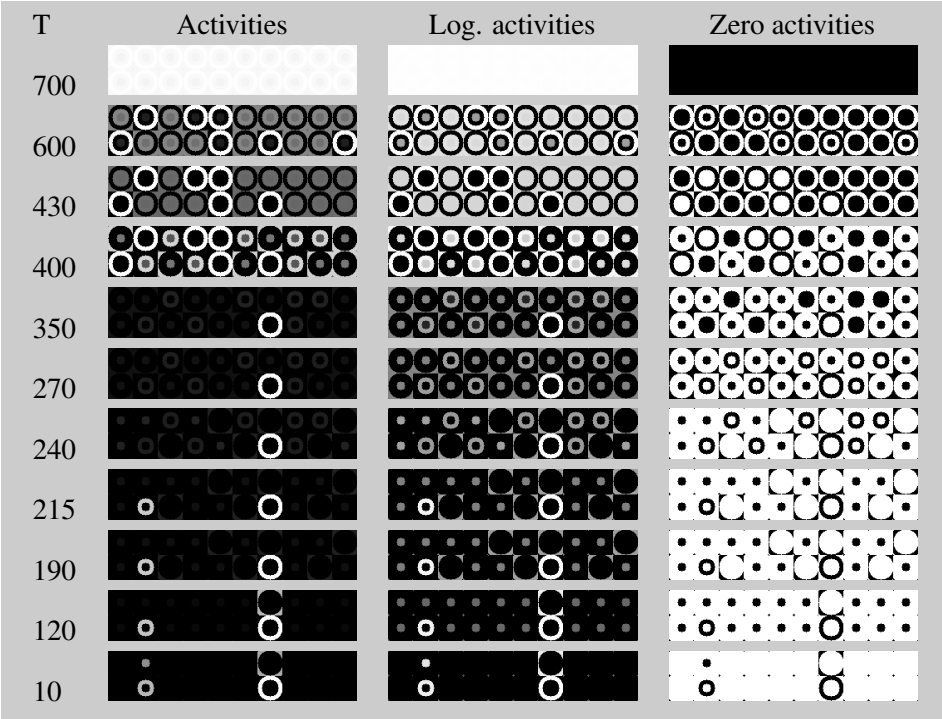


Figure 4.9: Attractor states of the CLM during the annealing process of the dynamics. Each row shows an attractor state of a 20-layered CLM, specified by the interaction matrix F in Fig. 4.7, at a certain pseudo-temperature T, if T is decreased slowly. In the first column the activities in the CLM are displayed as intensity values, where the maximal activity is always normalized to the intensity value 255. The activities in the 20 layers are arranged in a 2×10 matrix of 30×30 images. The second column shows the logarithms of the activities to give a better resolution of low activities. Again, the maximal value is normalized to the intensity value 255. The third column shows binary values of the activities. White pixels show activity equal to zero, while black pixels show positive activity.

highlights the state of the WTA processes by showing binary images for neurons with activity equal to zero, where neurons with $x_{r\alpha} > 0$ are mapped to intensity 0, while neurons with $x_{r\alpha} = 0$ are mapped to intensity 255.
If the CLM dynamics has reached a stable attractor state the three visualizations stay constant. To decide for a given temperature T whether the CLM has reached a stable state or not, the activity histogram, as is shown for prototypic attractor states in Fig. 4.10, is observed manually. If the activity histogram stays constant for several iterations of the CLM dynamics, it is assumed that the CLM has reached an attractor state and T is lowered for the next iteration. Otherwise the CLM dynamics is iterated further at the present temperature.
As can be seen in the simulation algorithm of the dynamics in section 3.5, T occurs only in the denominator of the neuron activities. Thus each modification of T in the

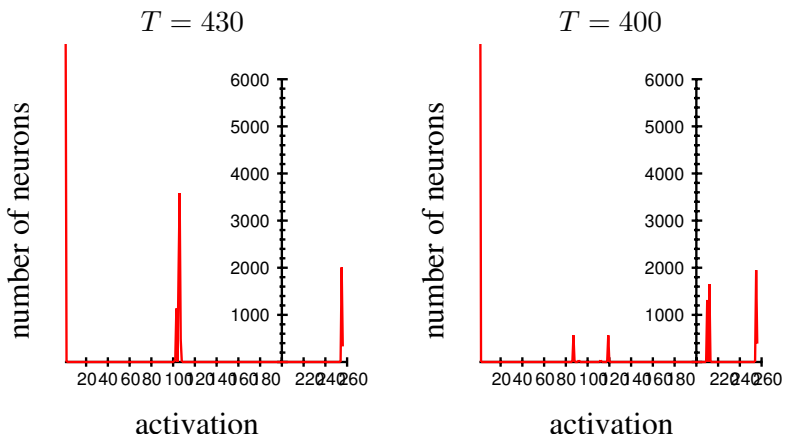


Figure 4.10: Activity histograms of the CLM-states in Fig. 4.9 at $T = 430$ and $T = 400$.

first neuron updates simply rescales the activity values. Together with the random selection of neurons for update this process disturbs the activity histogram. A modification of T is called "not significant", if the activity histogram stabilizes to a similar distribution as before the change of T, which means the position of peaks in the histogram can move a little bit, affected by the rescaling of the activities, but the number of peaks and the height of the peaks is restored. An "not significant" change of T does not change the principal activity patterns in the visualization and changes the attractors continously. In contrast a "significant" change of T destabilizes the current activity pattern and the CLM dynamics converges to a new attractor state with a different histogram, leading to different numbers and heights of existing peaks.

A set of critical temperatures which mark significant changes of the attractor states is approximated by slow annealing and backtracking reheat of T. The observed attractor states are visualized for specified temperatures in Fig.4.9, while the critical temperature thresholds which switch between these attractors are listed in Fig. 4.11.

The temperature thresholds activate three types of phenomena in CLM states, which in the following are called "splitting the layer directions", "orthogonalization", and "WTA activation". Thereby the temperature thresholds for splitting the layer directions partially correspond to eigenvalues of F and the thresholds for the WTA activations correspond to the inner group support, indicated by the size of the corresponding group (minus one for the erasement of the self-interaction weights). Each temperature threshold in Fig. 4.11 is attached with a label for hysteresis effects. "No hysteresis" effects means, that the observed phenomena is directly "reversible" by a reheat of the temperature above the corresponding threshold. "Reversible" means, that the CLM returns to the same activation patterns in the layers as before the temperature had fallen beneath the threshold. However, the patterns can be permuted according to the layer indices. At a temperature threshold "with hysteresis"

threshold	meaning	phenomena	hysteresis
$T = 654$	$\lambda_1(F)$	splitting	no
$T \approx 450$	-	orthogonalization	yes
$T = 418$	$\lambda_2(F)$	splitting	no
$T = 389$	support(group 1)	WTA group 1	yes
$T \approx 300$	-	orthogonalization	yes
$T \approx 250$	-	splitting	no
$T = 235$	support(group 2)	WTA group 2	yes
$T = 188$	support(group 3)	WTA group 3	yes
$T = 80$	support(group 4)	WTA group 4	yes
$T = 4$	support(group 5)	WTA group 5	yes

Figure 4.11: Temperature thresholds for the self-inhibitory annealing process.

effect, the CLM state can only be restored (up to permutation of the layers) after a severe reheat clearly above the corresponding activation threshold.
The annealing process starts for $T > \lambda_1 = 654$ with a single global attractor, where all features show the same activation in each layer, indicating that the DC eigenmodes dominate against the AC eigenmodes (see section 3.4). By lowering T below $T = \lambda_1 = 654$, the first AC eigenmode is activated and causes a differentiation of the layer activation patterns in the direction of the corresponding eigenvector of F. However, the CLM state is still driven by the DC eigenmodes to the constraint surface $\forall r : \sum_\alpha x_{r\alpha} = h_r$, which causes an equalization of all layer vectors pointing into the same direction. Thereby the attractor state shows the typical distribution of seven to thirteen layers, which arises in each simulation run, while the permutation of the layers can change from run to run. As can be seen in the third column, second row of Fig. 4.9 there exists a small but significant overlap between the activity patterns in the two types of layers, where the features of the inner circle are activated in both types of layers.
When T falls below a certain temperature (approximately at $T \approx 450$), this overlap vanishes abruptly by moving all activation of the inner circle form the layers containing the features of the outer ring to the layers containing the other features. Parallel to this process one of the layers containing the features of the outer ring changes its attracted activation pattern and equalizes with the layers of the other activation pattern to compensate the perturbation of the activation equilibrium. After this process the two types of activation patterns are orthogonal to each other.
The process of splitting the layer direction repeats at $T = \lambda_2 \approx 408$, where T falls below the second eigenvalue of F, and, more surprisingly, approximately at $T \approx 250$ which does not correspond to any of the eigenvalues of F. A further splitting which separates the five features of group 5 from the rest of the pattern must exist, but is hard to detect in the activity histogram, such that it is neglected in Figs. 4.9 and 4.11. The other eigenvalues of F beside λ_1 and λ_2 mark no significant temperatures thresholds of the annealing process.

As third type of phenomena, the activation of the WTA behavior, which causes to the unique assignment a group to a single layer, can be observed at $T = 389$, $T = 235$, $T = 188$, and $T = 80$. These thresholds correspond to the inner group support through the lateral interaction weights, given by the number of features in the respective group minus one for the erasement of the diagonal self-interaction weights f_{rr}. The WTA process starts by differentiating the strength of the activation pattern in the layers showing the same group. Then the activation of the layers with weakest activation is suppressed by the layers with strongest activation, until only one layer shows activation for the respective group. The other layers whose activities reach zero are reused by equalization to the activity patterns of layers containing other groups.The WTA process for group 5 is not visualized in Fig. 4.9, because of the small size of the group, but it is activated at $T = 4$.
In summary Fig. 4.9 shows, how the output grouping results from a sequence of splitting and WTA processes among the layers, triggered by the AC and DC eigenmodes of the CLM dynamics. Thereby the recycling of unused layers in the course of the WTA processes provides the detection of the correct number of groups, as long as the number of layers is larger than the number of groups. Otherwise the groups with lowest inter-group inhibition are merged into the same layer, because no further splitting processes can be performed. Slow annealing prevents suboptimal grouping states, in the sense of an oversegmentation of the input, because the WTA processes of the groups with highest inner-group support are activated before the WTA processes of subgroups with smaller support or subsequent splitting processes.
Compared to the two dimensional sketch of the AC/DC eigenmodes in Fig. 3.4 this experiment stresses different aspects of the grouping process. It puts higher attention onto the WTA thresholds, here indicated by the size of the groups, than on the splitting thresholds, indicated by the eigenvalues of the lateral interaction matrix F. Thereby the groups with the highest inner-group support are first switched into the WTA process, such that the ordering of the WTA processes mirrors some kind of importance or saliency of the groups in the input. This aspect is investigated in detail in chapter 8, where it is tried to change the ordering of the WTA processes by modifying F with a kind of attention map. Further, the experiment opens a new interpretation of figure-background separation presented in section 4.5.
The demonstrated recycling of idle layers from the WTA process shows that neurons with activation zero can be reactivated to detect further groups. This stresses the importance of continuing the simulation of the dynamics for these neurons. The equalization of inactive layers with other layers in the course of the splitting and WTA processes suggests a dynamical adaptation of the number of layers in the course of the CLM dynamics. The simulation could be started with a low number of layers and succeeded as long as at least two layers converge to the same activation pattern. Otherwise, the CLM could be extended by free layers which automatically converge to an activation pattern of undecided groups.
Surely, the observed clear activation thresholds for the different phenomena are a result of the *ideal block diagonal* structure of the interaction matrix, assumed

by (4.6), which provides a simple eigenvalue spectrum and a constant support of all features belonging to the same group. More natural interaction matrices with real valued and partial inconsistent interaction weights initiate a richer and more unpredictable set of phenomena. However, the following experiments show the resistance against random distortions of the lateral interaction weights.

4.4 Robustness against Noise

For the example in the last section a regular interaction matrix with a *ideal block diagonal* structure was assumed. Therefore, clear thresholds of T for the three phenomena of splitting of layer directions, orthogonalization of layer directions and activation of the WTA-process could be observed. In general grouping problems much more irregular matrices occur for various reasons: The applied grouping behavior might not be described completely consistently by positive and negative values. The CLM must work on noisy our incomplete patterns with absent features and therefore also absent columns and rows of the interaction matrix and, finally, the user of the CLM might purposely decide to neglect weights $f_{rr'}$ of the interaction matrix to reduce the computation time and memory requirements for the interaction matrix. Consequently, real interaction matrices show much more eigenvectors with different eigenvalues and also the support that a feature gets from the other features in its group may vary within the groups. For these reasons it must be expected that there exist much more thresholds for the splitting of layer directions and that the WTA process is no longer activated at a unique threshold for the complete groups, but more gradually for the different part of the groups, such that the activation of the WTA-process extends over the whole intervals of T.

Below, it is investigated how random perturbations of the *ideal block diagonal* matrix in Fig. 4.7 influence the output grouping and the course of the annealing process. Therefore, weights in the ideal block-diagonal interaction matrix in Fig. 4.7 are erased or randomly resetted.

Since a much more complex set of switching thresholds is expected, the strategy of determining all of them by manual exploration of T is changed. Instead only a fixed set of predefined values of T, derived from the sample values in Fig. 4.9 and the maximal eigenvalues of the distorted interaction matrices, are inspected. First the relations $T/\lambda_1(F)$ between the eleven sample thresholds in Fig. 4.9 and the maximal eigenvalue $\lambda_1(F) = 654$ of the ideal block-diagonal matrix are computed. For a new distorted interaction matrix $\tilde{F}$, its biggest eigenvalue $\lambda_1(\tilde{F})$ is computed and the CLM-dynamics is simulated only at values of T that show roughly the same relation $T/\lambda_1(\tilde{F})$ to the biggest eigenvalue as the sample values to the maximal eigenvalue of the *ideal block diagonal* matrix.

The resulting attractor states can be observed for four different types of distortions in Figs. 4.13, 4.15, 4.17 and 4.19.

4.4.1 Reduced Connection Strength

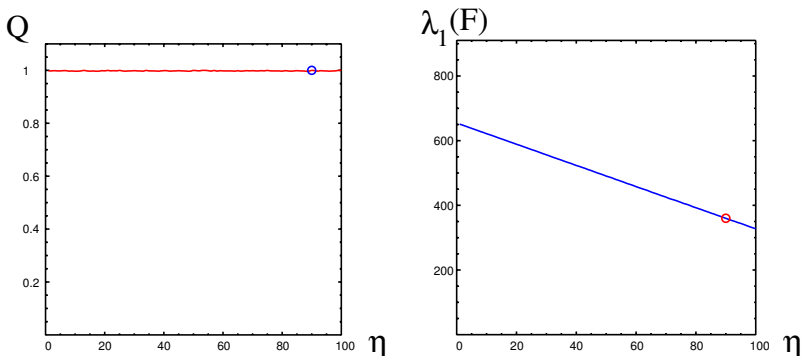


Figure 4.12: Influence of reducing the interaction strength. On the left hand side the quality of the CLM-output grouping is plotted against the percentage η of weights with changed interaction strength. The right hand side shows the maximal eigenvalues of the corresponding interaction matrices.

The experiment is started with a very weak distortion, where the block-diagonal structure of positive and negative weights is left intact and only the strength of the interactions weights is changed. Here, a certain percentage η of all interaction weights is chosen whose connections strength is reduced by multiplying them with random values from the interval $[0, 1]$. The remaining interaction weights are left unchanged. Figure 4.12 plots the effect of this perturbation on the grouping quality and the largest eigenvalues of the perturbed interaction matrices for $\eta = 0\%$ to 100%. The changed connection strength has only small effects on the grouping process. It can be seen that the output grouping stays correct for all values of η, while the largest eigenvalues decrease linearly with η. The impact on the annealing process is visualized exemplary in Fig. 4.13 for $\eta = 90\%$.
The main difference to the original annealing process in Fig. 4.9 is, that the maximal eigenvalue $\lambda_1 = 360$ is clearly smaller than before, which corresponds roughly to a scaling of factor $\frac{1}{2}$. The structure of the activity patterns in the first two columns of Fig. 4.13 show hardly visible differences to the original attractor states. The splitting of the layer directions, the orthogonalization and the WTA-processes are activated at the same thresholds relative to the largest eigenvalue of F. The only exception can be found in the eighth row, where the third split of layer directions is activated a little bit later, such that it's activation time lies closer to the activation of the WTA-process for the inner ring than before.
Beside this, slight differences in the other attractor states can be detected, if the third column of Fig. 4.13 is inspected. It can be seen that the neurons forming a coherent group do not meet zero activity at exactly the same time any more. The reason for this can be found in the small deviation of the inter-group support through the random connection strength. The distortion of the interaction shows

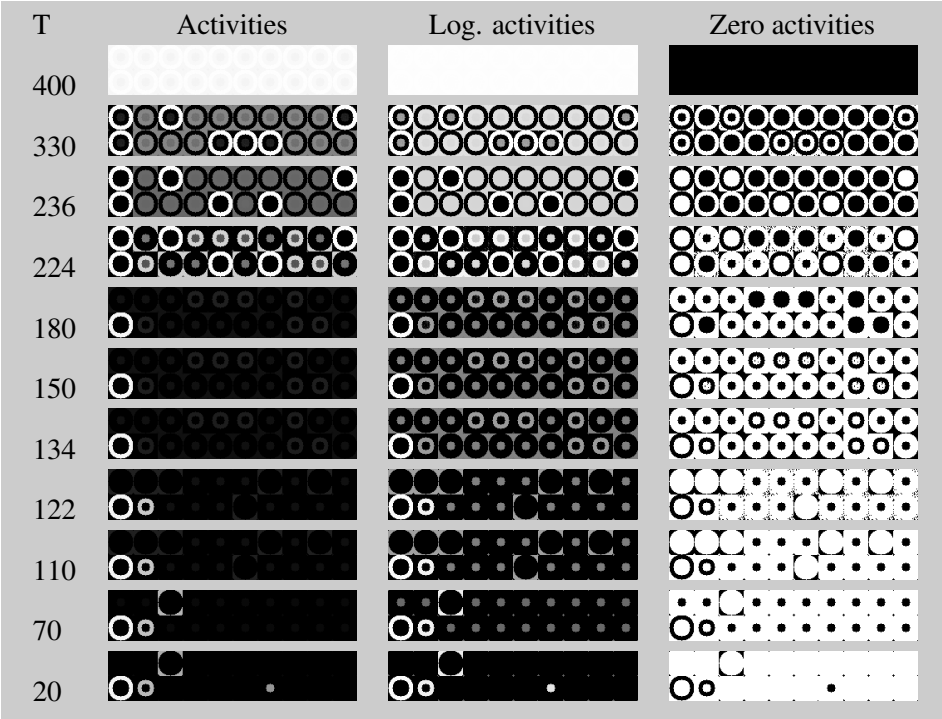


Figure 4.13: Attractor states of the CLM, if the original block-diagonal interaction matrix is blurred by scaling 90% of the weights with random values from [0,1].

mainly the same effects as a scaling of the original interaction matrix with the factor $\frac{1}{2}$. Since J is chosen depending on F, this can also be interpreted as a direct scaling of the whole CLM-dynamics, which means, that the dynamics shows only half the speed of convergence compared to before. So the concrete interaction strength within the interaction matrix has no effect on the grouping result, but on the speed of it's computation.

4.4.2 Erased Interaction Weights

As the exact interaction strength in the interaction matrix does not seem to play an important role for the grouping result, in the following the degree of distortion is enforced by completely removing η percent of the original interaction weights, where the erased interaction weights are chosen randomly. Fig. 4.12 shows the influence of η on the grouping quality and the largest eigenvalues of F. While the largest eigenvalue decreases linearly until it reaches zero, the grouping quality stays unaffected up to $\eta \approx 95\%$ and then decreases abruptly. The course of the resulting attractor states for $\eta = 90$ can be seen in Fig. 4.14, where a more distinct deviation from the original attractor states can be observed.
At the first glance on the first column of Fig. 4.15, the attractor state seem to be the same as before. However, differences become clear, if the number of layers

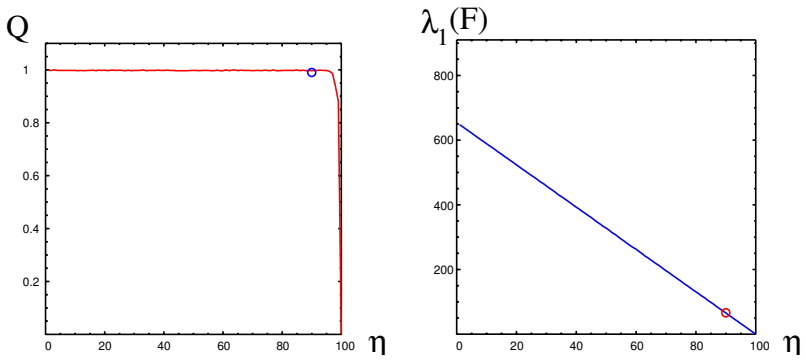


Figure 4.14: Influence of erasing entries in the interaction matrix. On the left hand side the quality of the CLM-output grouping is plotted against the percentage of erased interaction weights. The right hand side shows the maximal eigenvalues of the corresponding interaction matrices.

in the different layer groups is counted. After the first split of layer directions the CLM shows a new attractor state at $T = 270$ with a slightly modified relation of eight to twelve layers in the two layer groups. With the decrease of T this state becomes unstable and switches to the typical relation of seven to thirteen layers, as in the case of the original interaction matrix. With the further decrease of T a second layer changes its direction during the orthogonalization process, such that the relation between the layer groups passes over to six against fourteen layers. This observation shows that the CLM passes more attractor states than before, where gradually single layers change their layer direction from one layer group to the other.

Further, a distortion of the different layer directions can be seen in the small activities of the second column of Fig. 4.15 . Also the pattern of neurons that reach activity zero in the third column of Fig. 4.15 shows a higher distortion compared to the reduced strength of interaction weights. This indicates that the inner-group support is more diverse, because the columns of the interaction matrix show different numbers of non negative weights due to the random selection of connections in the erasure step. Actually the phenomena of orthogonalization does not occur any more: the effect that the activation of the inner circle in the layer group that contains the outer ring vanishes with the decrease of T can still be observed, but this process happens gradually during the complete annealing process, such that at most temperature values a small but significant overlap between the activation patterns in the different layer groups exists. Orthogonalization occurs only according to certain feature activities in the layer groups, but no more for the whole layer vectors.

The value of the largest eigenvalue is reduced further to $\lambda_1 = 65$ which indicates a much slower speed of convergence. This shows, that a sparse interaction matrix on the one hand reduces the computation time and memory demand for

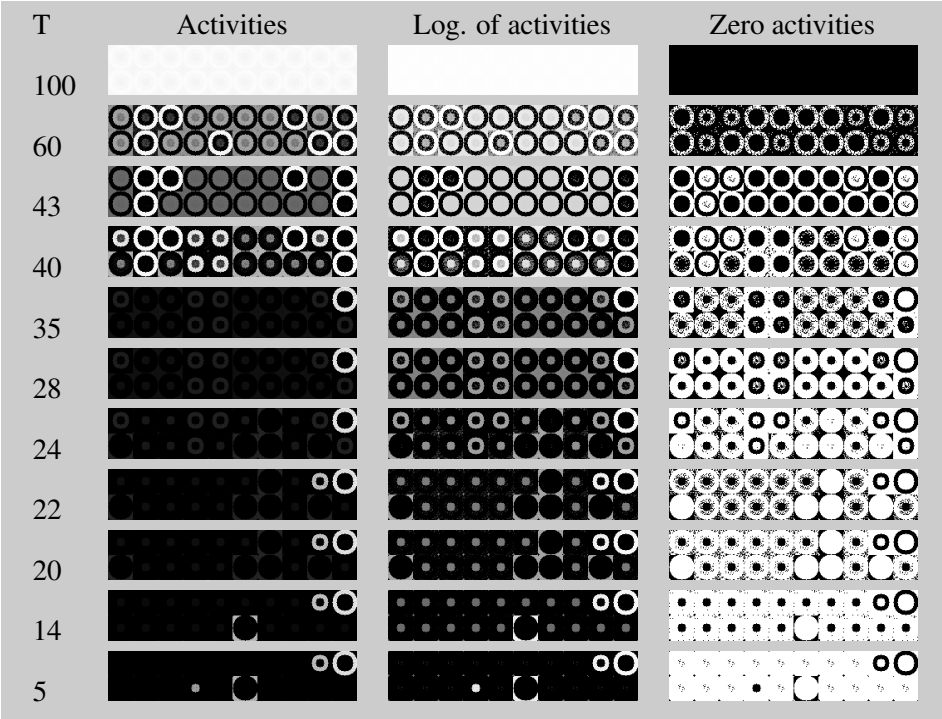


Figure 4.15: Attractor states of the CLM, where 90% of the weights in the original interaction matrix a set to zero to build a sparse interaction matrix.

the lateral interactions, but on the other hand increases the simulation time of the CLM-dynamics, because of a smaller speed of convergence.

4.4.3 Random Reset of Interaction Weights

Up to now the distortion of the interaction matrix leaves the block-diagonal structure intact and only modifies the connection strength. In the following, inconsistent weights are introduced to the interaction matrix by randomly selecting η percent of the interaction weights and reseting them to random values from the interval [-1,1]. The effect of this modification on the grouping quality and the largest eigenvalue of F can be observed in Fig. 4.16. Similar to the erasement of weights the largest eigenvalue decreases linearly towards zero. However, its value stays constant for $\eta = 95\%$ to 100%. For $\eta = 0\%$ to 90%, the grouping quality stays nearly correct. Only sporadic errors in the smaller groups of the pattern disturb the optimal quality for $\eta = 50\%$ to 90%. For $\eta > 95\%$ the grouping quality breaks down.
The grouping process is investigated for $\eta = 90\%$ in more detail in Fig. 4.17. Since there is a probability of 50% that a weight changes it's sign, if it is resetted, it can be assumed that 45% of all interaction weights disturb the block diagonal structure of the interaction matrix. However, the unchanged 10% of the interaction weights still show a higher connection strength of +1 and -1 than the modified weights,

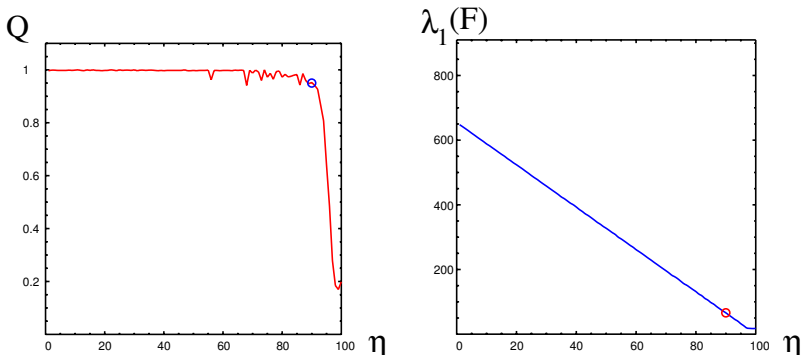


Figure 4.16: Influence of resetting entries in the interaction matrix. On the left hand side the quality of the CLM-output grouping is plotted against the percentage of reset interaction weights. The right hand side shows the maximal eigenvalues of the corresponding interaction matrices.

such that it can be assumed that the block diagonal structure still dominates the grouping process.
Through the change of sign the inner-group support differentiates stronger than in the case of the earlier distortions. Therefore, the activation thresholds for the WTA-process and the splitting of layer directions become fuzzy, such that the different processes proceed in parallel and can not be divided clearly from each other. For example, the WTA-process of the outer ring no longer happens at a single threshold, but on multiple thresholds, where layers of the corresponding layer group change their direction. This can be observed in the third, fourth and fifth row of Fig. 4.17, where the number of layers that contain activities corresponding to the outer ring decreases from seven over four to one. In the meanwhile the remaining layers split into two new groups with different activation patterns.
Also, the activity patterns of the different layer groups are disturbed. Indeed this is not obvious from the direct observation of the activities, but the logarithms of the activities show serious noise in the activation patterns. The noise covers mainly the regions of smaller groups which show the smallest inner-group support, which can be seen in the last row of Fig. 4.17, where relevant parts of the inner circle are not assigned uniquely to a single layer. The speed of convergence is further decreased, however, the size of the largest eigenvalue $\lambda_1 = 65$ did not decrease in comparison to the sparse but consistent interaction matrix.

4.4.4 Switching Signs of Interaction Weights

In the last kind of distortion the inconsistent weights in the matrix showed the same strength as the remaining weights originating from the original block-diagonal structure. In the following experiment, η percent of the weights are randomly chosen from the original interaction matrix whose signs are switched from 1 to -1, or

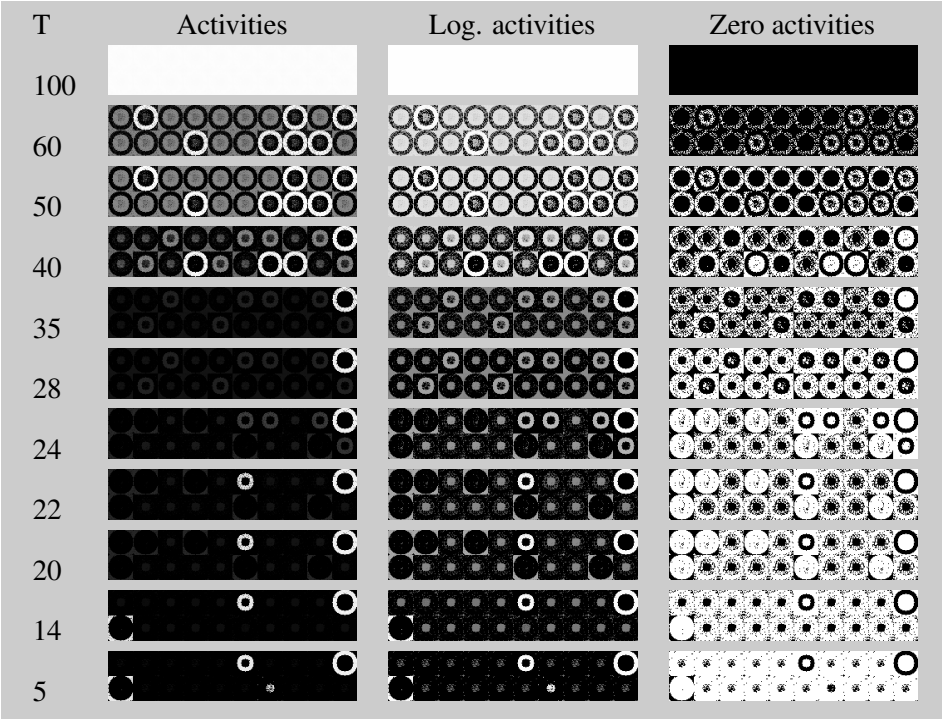


Figure 4.17: Attractor states of the CLM, where 90% of the original interaction matrix are reseted randomly in the interval $[0, 1]$.

from -1 to 1, respectively. The influence of the distortion on the grouping quality and the largest eigenvalue can be seen in Fig. 4.18.
The grouping quality remains high for $\eta = 0\%$ to 40%, which indicates that the separation of the bigger groups is more robust against random noise, because of the higher redundancy in the inner-group support, while smaller groups can be disturbed more easily. If the percentage of switched interaction weights is increased above 40% the bigger groups become also disturbed stronger than before. At a percentage of changed weights near 50%, the distortion becomes so strong that the original groups are totally decomposed and the output grouping of the CLM results in a random distribution of the features to the 20 layers. If the percentage of switched interaction weights is increased above 50% the modified interaction matrix is more similar to the negative of the original interaction matrix as to the original interaction matrix itself. Since the interaction matrix is dominated by positive interaction weights between all parts of the patterns, all features are assigned to the same layer.
The annealing process is investigated for $\eta = 45\%$ in Fig. 4.19. The maximal eigenvalue of the interaction matrix lies roughly at $\lambda_1 = 65$, which is the same value as in the case of the sparse and noisy interaction matrices from the last two experiments. However, the speed of convergence is seriously reduced in comparison to the other experiments. This can be explained by the fact that there exist

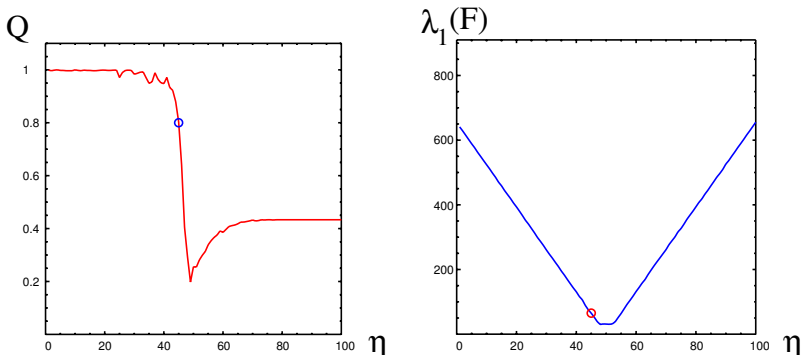


Figure 4.18: Influence of binary switching of the entries of the interaction matrix. On the left hand side, the quality of the CLM-output grouping is plotted against the percentage of switched interaction weights. The right hand side shows the maximal eigenvalues of the corresponding interaction matrices.

much more different eigenmodes than before, such that it is more likely that T is chosen close to a critical threshold, where a new eigenmode is activated. In this case, the just activated eigenmode drives the dynamics on a slow timescale.
For this reason, the dynamics is exceptionally interrupted before the dynamics has completely converged in this experiment, as can be seen in the top left of row five in Fig. 4.19, where one layer is not equalized to the directions of the other layers. Nevertheless, the attractor states of the dynamics can be anticipated and they can be compared to the attractors in the previous experiments. It becomes difficult to judge whether phenomena, like the activation of the WTA-process or the splitting of layer directions, have been activated at a certain value of T or not. The noise in the activity patterns of the attractor states is increased. However, they still show a high similarity to the original attractor states. If the attractor states of low values of T are inspected, it can be seen, that the grouping of the first three groups is almost correct, but in each group a small subset of features exists that show activation in other layers. This indicates that these features are either spuriously assigned to other layers or at least need very long time to be assigned to the correct layer. This effect can be observed more clearly for the fourth group of the inner circle. Indeed, the logarithms of the activities show a preference of this group to the layer on the lower right, but this preference is very hard to see in the original activities, where it seems that the inner circle is separated into several random sets.

4.5 Interpretation of the Background Layer

The observations of the annealing process for the *ideal block-diagonal* matrix have demonstrated that the grouping result of the CLM arises from gradually splitting layer directions and activation of the WTA-processes for the different groups, or-

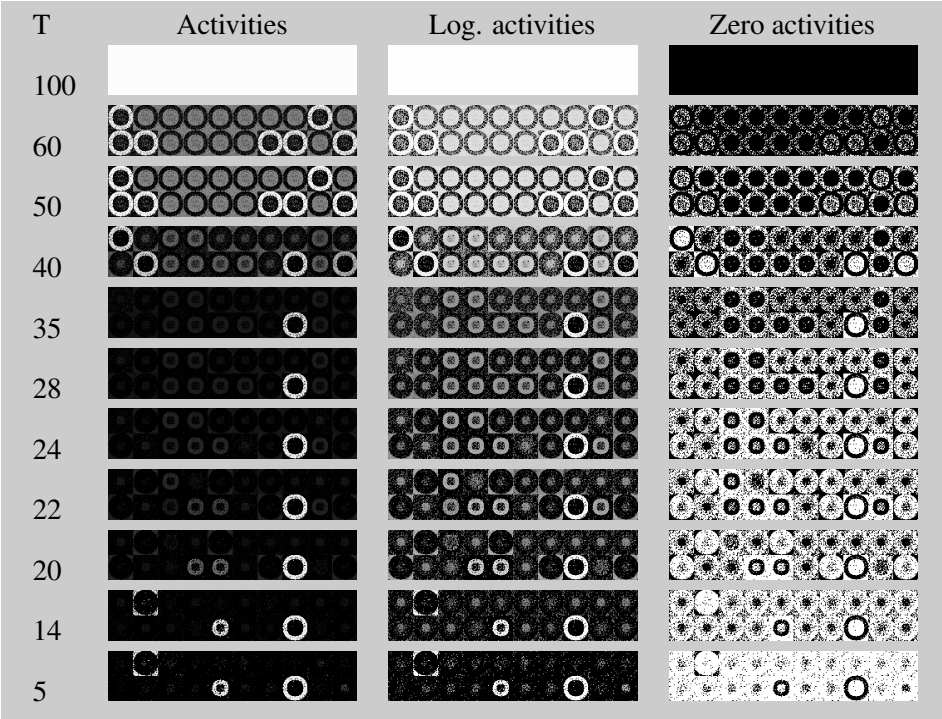


Figure 4.19: Attractor state of the CLM, where the sign of 45% of the weights in the original interaction matrix is switched.

dered by the strength of the inner-group support. In this section these observations are connected to the use of the background layer in section 3.3. Therefore, a new example pattern is investigated that consist of four relevant groups and a fifth group with no coherent structure that forms some kind of background for the other groups. This pattern is shown in Fig. 4.20, where four squirrel-shaped regions can be seen that are assigned to different labels, while the rest of the features are assigned randomly to ten further labels. According to (4.6) a consistent interaction matrix is constructed, which again shows a characteristic (after permutation) *ideal block diagonal* structure. Since the region of the background is split into several labels, the blocks of the four relevant groups and therefore also their inner-group supports are bigger than the blocks of the subsets in the background.
One interpretation of the background layer is that it collects all features whose support is smaller than the self-interaction strength m. But if the annealing process is inspected, like it is displayed in Fig. 4.21 at four typical values of T, a second interpretation of m arises. At $T = 300$, it can be seen that the dynamics converges to the global fixed point, where all layers show the same activation for all features in the pattern. At $T = 180$ the layers are split into five different layer directions. Four of them show activation for one of the four relevant groups, while the fifths shows activation for all remaining features in the background. At $T = 100$, the WTA-process is activated for the four relevant groups, such that each of them is assigned

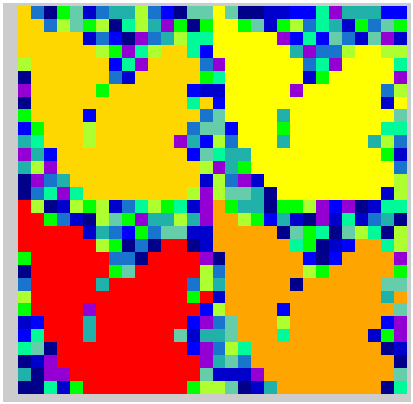


Figure 4.20: Label vector of a pattern with four main groups against an “over segmented background”.

to a single layer, while the features in the background show a uniform activation in the remaining 20 layers. Finally at $T = 0$, the WTA-process is activated for the 10 random subsets of the background, and each of these subsets is assigned to a single layer, while the remaining six layers are empty.
Now, assume the usage of an additional background layer with the interaction $f_{rr'} = m\delta_{rr'}$. The WTA-process for this layer is activated at $T = m$. Therefore, the background layer collects all features whose activations describe an undecided state at temperature $T = m$ of the annealing process. Finding a good self-interaction strength m corresponds to finding a good temperature to stop the annealing process. All features that - up to this temperature - are not assigned uniquely to one of the figure layers can be treated as background features. For the present example pattern a suitable choice for the figure-background separation is $m = 180$. Thus a possible strategy to estimate the self-interaction strength m in the background layer is to observe the annealing process of representative test patterns and to adjust m between temperature thresholds which trigger the WTA-processes for relevant foreground groups and spurious background groups.

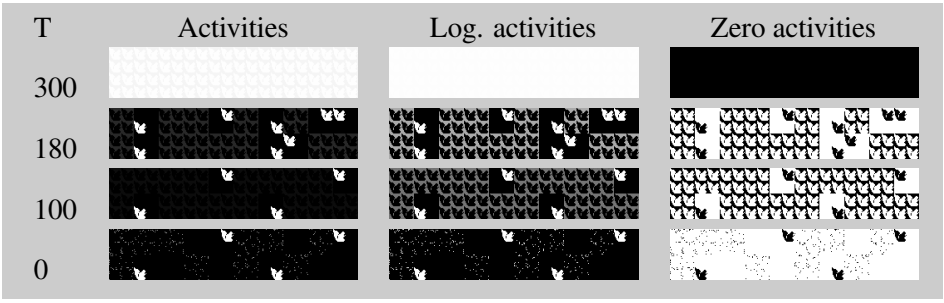


Figure 4.21: Course of the CLM-dynamics during the annealing process for a pattern with an “over segmented background“.

While in most image processing tasks big and homogeneous regions are interpreted as background for more diversified image regions of salient objects, the use of a background layer can result in an switching roles of fore- and background. Large and coherent regions often show a high inner-group support and therefore are switched earlier into the WTA behavior than smaller, more structured groups. Background in the sense of the CLM background layer means a set of non coherent subgroups. To achieve a figure background segmentation more similar to the classical interpretation, either the coherence of the background has to be destroyed by artifical inhibitory connections or the interactions of relevant groups have to be highlighted as will be demonstrated in chapter 8.

4.6 Summary

This chapter has shown on an example of an *ideal block-diagonal* interaction matrix, how the CLM attractor states change in the course of the annealing process. The output grouping of the CLM arises from a gradually splitting of the layer activation patterns and the activation of the WTA processes for the different groups in the order of the inner group support. However, to characterize the complete grouping behavior of the CLM it is not enough to inspect only the output grouping of the CLM. Other aspects, like the ordering of the manifestations of the different groups and the speed of convergence towards the different attractor states during the annealing process, must also be considered.
Thereby, the attractors states are very robust against random noise in the interaction matrix, which can be explained by the high redundancy in the interaction weights, such that the set of possible groupings becomes sparsely distributed in the weight space of the CLM, at least for grouping problems with a high number of features. Bigger groups are more robust than smaller ones, because their inner group support shows a higher redundancy. The grouping result of the CLM stays stable as long as the corresponding block-diagonal structure dominates over the influence of noise. A high number of interaction weights can be neglected or randomly resetted without seriously effecting the correct separation of the relevant groups. However, each distortion of the block diagonal structure results in a slower speed of convergence of the dynamics.
With regard to the implementation of a learning algorithm, the practical sample simulations leave the impression that a coarse coding of the interaction weights might be sufficient to describe a desired grouping behavior, because of the robustness of the CLM attractor states. This suggests a very simple learning approach by binary classification of the lateral interaction weights, as it is presented in the next chapter. In terms of a possible hardware implementation of the CLM dynamics the experiments suggest that a simple binary encoding of the lateral interaction weights might be sufficient to describe the grouping process.

Chapter 5

Learning of Grouping Behaviors

The last chapter has shown, how the CLM computes an output grouping, if a suitable interaction matrix is given. In this chapter, the opposite question is investigated: How must the interaction weights be adapted to achieve a specified grouping with the CLM? To answer this question a new learning approach is developed, which is based on binary classification in a pairwise proximity space and turns out to be computationally much simpler than the earlier learning approach of quadratic consistency optimization presented in section 3.7.

5.1 Hebbian Learning

Our starting point is a reformulation of the learning problem presented in section 3.7.1, which shows that the intuitive choice of a block-diagonal matrix as interaction matrix fulfills the consistency conditions of the attractor states of the CLM. It will be shown, that this approach can be interpreted as a variant of Hebbian learning.
For the moment, the training set is restricted to the case, that it consists only of a single pattern $\mathcal{P}$, such that the consistency conditions (3.37) simplify to:

$$\sum_{r'=1}^{N} f_{rr'} y_{r'\beta} < \sum_{r'=1}^{N} f_{rr'} y_{r'\hat{\alpha}(r)}, \quad r = 1, \ldots, N, \quad \beta \neq \hat{\alpha}(r). \tag{5.1}$$

To fulfill (5.1), the matrix $\hat{F}$ of lateral interactions can be set to the correlation matrix of all differential vectors $(\mathbf{y}_\gamma - \mathbf{y}_\mu)$, where the vectors $\mathbf{y}_\gamma$ and $\mathbf{y}_\mu$ represent the γ-th and μ-th layer vector of the goal state $\mathbf{y}$ (see Fig. 3.7).

$$\hat{F} = \sum_{\gamma} \sum_{\mu \neq \gamma} (\mathbf{y}_\gamma - \mathbf{y}_\mu)(\mathbf{y}_\gamma - \mathbf{y}_\mu)^T. \tag{5.2}$$

Because of the special structure of the vectors $(\mathbf{y}_\gamma - \mathbf{y}_\mu)$, $\hat{F}$ can be computed without an explicit evaluation of (5.2): Without restriction the features and target labels can be considered to be ordered, such that $r \leq r' \Leftrightarrow \hat{\alpha}(r) \leq \hat{\alpha}(r')$. By

this permutation new target state vectors ($\mathbf{y}_\gamma^{ord}$ and $\mathbf{y}_\nu^{ord}$) can be derived. Since the vectors $(\mathbf{y}_\gamma^{ord} - \mathbf{y}_\nu^{ord})$ have only entries 1, -1, and 0, and due to the ordering of the labels, the block-diagonal matrix

$$\frac{\hat{F}}{2} = \begin{pmatrix} \big[L-1\big]_{N_1\times N_1} & \big[\ -1\big]_{N_1\times N_2} & \cdots & \big[\ -1\big]_{N_1\times N_L} \\ \big[\ -1\big]_{N_2\times N_1} & \big[L-1\big]_{N_2\times N_2} & \ddots & \big[\ -1\big]_{N_2\times N_L} \\ \vdots & \ddots & \ddots & \vdots \\ \big[\ -1\big]_{N_L\times N_1} & \big[\ -1\big]_{N_L\times N_2} & \cdots & \big[L-1\big]_{N_L\times N_L} \end{pmatrix} \tag{5.3}$$

is obtained from (5.2), where $[\cdot]_{N_i\times N_j}$ is a constant matrix of size $N_i \times N_j$. A more descriptive derivation of (5.3) is presented in the appendix A for the concrete example of 3 groups with 3, 2, and 1 features.

Remembering, that L is the number of groups in the target labeling, it becomes obvious, that the negative entries are scaled against the positive ones with a scalar factor $\Lambda = \frac{1}{L-1}$, which only depends on the number of groups in the pattern. This reflects, that there exist for each feature r a total number of $L-1$ consistency inequalities (5.1). The block-diagonal structure of (5.3) fulfills the consistency inequalities (5.1), because all positive values $y_{r'\beta}$ on the left hand side are multiplied with negative weights $f_{rr'} < 0$, while all positive values $y_{r'\hat{\alpha}(r)}$ on the right hand side are multiplied with positive weights $f_{rr'} > 0$. This property still holds for all simultaneous permutations of the columns and rows of (5.3) that describe an arbitrary ordering of the features in a concrete pattern.

Obviously, for any positive value of Λ the interaction matrix (5.3) makes the target labeling consistent for the CLM according to (5.1), because it preserves the block-diagonal structure of positive and negative values in the interaction matrix $\hat{F}$. However, in the following argumentation the choice of $\Lambda = \frac{1}{L-1}$ is maintained to show the similarity of this approach with storing a set of attractor states in a Hopfield Network by Hebbian Learning. The later chapters will show, how the modification of Λ controls the segmentation level of the output labeling of the CLM.

To apply well-known arguments from Hebbian learning describing the storage of attractor states in a Hopfield network [19], the consistency inequalities (5.1) are made more restrictive by introducing further assumptions about the stable states of the CLM. It is required, that the support of all features m_r belonging to a group $\hat{\alpha}(r)$ has equal strength in all other layers $\beta, \beta' \neq \hat{\alpha}(r)$, which can be expressed by the equations:

$$\textstyle\sum_{r'=1}^{N} f_{rr'} y_{r'\beta} = \sum_{r'=1}^{N} f_{rr'} y_{r'\beta'}. \tag{5.4}$$

On the other hand, it is assumed that the inequalities in (5.1) for the target label hold with a strict margin of one:

$$1 + \textstyle\sum_{r'=1}^{N} f_{rr'} y_{r'\beta} = \sum_{r'=1}^{N} f_{rr'} y_{r'\alpha(r)} \tag{5.5}$$

Collecting all lateral interaction weights $f_{rr'}$ corresponding to the feature $\mathbf{m}_r$ as a row vector $\mathbf{f}_r = (f_{r1}, \ldots, f_{rN})$ and all components $y_{r\mu}$ in the layer μ, $\mu = 1, \ldots, L$, of the target state $\mathbf{y}$ as $\mathbf{y}_\mu = (y_{1\mu}, \ldots, y_{N\mu})^T$, equation (5.4) and (5.5) are rewritten as scalar products.

$$\mathbf{f}_r(\mathbf{y}_\beta - \mathbf{y}_{\beta'}) = 0 \Leftrightarrow \mathbf{f}_r(\mathbf{y}_{\beta'} - \mathbf{y}_\beta) = 0, \tag{5.6}$$

$$\mathbf{f}_r(\mathbf{y}_{\hat{\alpha}(r)} - \mathbf{y}_\beta) = 1 \Leftrightarrow \mathbf{f}_r(\mathbf{y}_\beta - \mathbf{y}_{\hat{\alpha}(r)}) = -1. \tag{5.7}$$

Now, two arbitrary labels $\gamma, \mu \in \{1, \ldots, L\}$ are chosen and all equations from (5.6), (5.7) which contain the vector $(\mathbf{y}_\gamma - \mathbf{y}_\mu)$ are collected, then for all r holds:

$$\begin{aligned}
\mathbf{f}_r(\mathbf{y}_\gamma - \mathbf{y}_\mu) &= \quad 1, \quad \alpha(r) = \gamma && (\Leftrightarrow (\mathbf{y}_\gamma - \mathbf{y}_\mu)_r = 1)\,, \\
\mathbf{f}_r(\mathbf{y}_\gamma - \mathbf{y}_\mu) &= -1, \quad \alpha(r) = \mu && (\Leftrightarrow (\mathbf{y}_\gamma - \mathbf{y}_\mu)_r = -1)\,, \\
\mathbf{f}_r(\mathbf{y}_\gamma - \mathbf{y}_\mu) &= \quad 0, \quad \gamma \neq \alpha(r) \neq \mu && (\Leftrightarrow (\mathbf{y}_\gamma - \mathbf{y}_\mu)_r = 0)\,.
\end{aligned}$$

Stacking the vectors $\mathbf{f}_r$ to obtain the desired interaction matrix $F = (\mathbf{f}_1^T, \ldots, \mathbf{f}_N^T)^T$ yields the matrix-vector product

$$(F(\mathbf{y}_\gamma - \mathbf{y}_\mu))_r = \begin{cases} 1 & : \quad (\mathbf{y}_\gamma - \mathbf{y}_\mu)_r = \quad 1 \\ -1 & : \quad (\mathbf{y}_\gamma - \mathbf{y}_\mu)_r = -1 \\ 0 & : \quad (\mathbf{y}_\gamma - \mathbf{y}_\mu)_r = \quad 0 \end{cases}. \tag{5.8}$$

Assuming, that the learning method shall not find the lateral connections for the linear threshold neurons of the CLM, but instead shall adapt the matrix F as a weight matrix of a Hopfield network with binary neurons, the learning problem shows high similarity to the problem of storing the $L(L-1)$ pattern vectors $(\mathbf{y}_\gamma - \mathbf{y}_\mu)$ in the weight matrix of a Hopfield network. According to standard Hebbian learning, $\hat{F}$ can be chosen as the correlation matrix of all vectors $(\mathbf{y}_\gamma - \mathbf{y}_\mu)$ as in (5.2).

The argument of Hebbian Learning holds exactly, if all pattern vectors $(\mathbf{y}_\gamma - \mathbf{y}_\mu)$ are orthogonal. In the present case, every two vectors $(\mathbf{y}_\alpha - \mathbf{y}_\beta)$ and $(\mathbf{y}_\alpha - \mathbf{y}_{\beta'})$ have an overlap proportional to the size of the group α. Nevertheless, (5.2) is a reasonable choice, because essentially the CLM shall not converge to an exact reconstruction of the pattern vectors $(\mathbf{y}_\alpha - \mathbf{y}_\beta)$, but shall reconstruct the correct grouping, indicated by the vectors $\mathbf{y}_\alpha$. In this respect, the overlap causes a self-reinforcement of the groups proportional to their size rather than to distort the grouping. Consequently, the CLM dynamics becomes biased towards finding larger groups faster than smaller ones.

5.2 Special Properties of the Learning Problem

Up to now, the lateral interaction weights are computed by constructing the target activity vector of the different layers of the CLM from the desired labeling in the training set and setting F according to (5.2) proportional to the correlation matrix of the difference vectors between the layer wise target activity vectors.

To be able to segment a new pattern with the CLM, the learning method must find a way to construct a similar correlation matrix for a set of features, where the desired labels are unknown. To achieve this, an interaction function is constructed, which approximates the positive and negative interaction weights by mapping typical intervals of pairwise feature relations onto their average interaction.

At the first sight the learning methods simply has to solve the two class problem of dividing the space of feature-pairs $(\mathbf{m}_r, \mathbf{m}_{r'})$ into regions of excitatory and inhibitory weights. From this point of view a wide range of standard classification methods that are available from the field of neural networks, e.g. the MLP, the SVM or several types of SOMs, can be applied on the learning problem.

But before concrete candidates are inspected this section emphasizes some fundamental properties of learning a grouping behavior that are different to general learning problems in pattern classification and recognition.

Supervised Learning Method: The first important property of grouping problems is, that they can often be ambiguous, because there exist many possible ways to divide a data set. E.g. the data set in Fig. 5.1 can be divided according to the principle of proximity into two point clusters, according to the principle of similarity into two symbol classes or by a mixture of both principles. The correct grouping behavior depends on the significance of the different grouping laws to the specific problem domain and the adequacy of differentiation. Therefore, it is assumed, that the learning process always is based on supervised methods, where a teacher specifies the desired grouping in form of a set of pre-labeled reference patterns $\mathcal{P}^i = (\mathcal{R}^i, \mathcal{L}^i)$, where $\mathcal{L}^i$ contains the desired labels $\hat{\alpha}(r)$ of the features in $\mathcal{R}^i$.

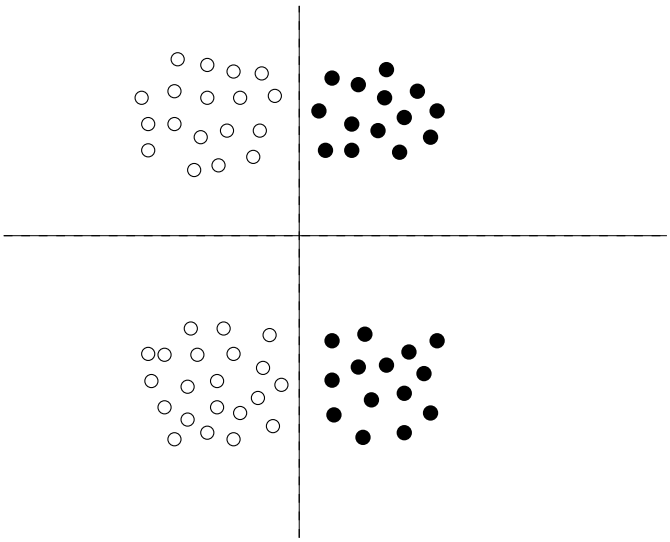


Figure 5.1: Combination of grouping principles.

Consideration of relational information:

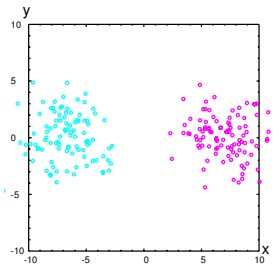


Figure 5.2: An easy two cluster problem

If concrete reference patterns $\mathcal{P}^i$ are inspected, only information about interactions $f_{rr'}$ between features in these patterns is obtained. To be able to segment new patterns using the same grouping principles, this information has to be generalized to interactions $f_{rr'}$ in the whole space of all possible pairs of features $\mathbf{m}_r$ from the feature domain $\mathcal{F}$.
This aspect is clarified for the sample problem of point clustering. Figure 5.2 shows two point clusters of features $\mathbf{m}_r = (x_r, y_r)^T$ that are clearly separated from each other. To reproduce this grouping, $f_{rr'}$ can be trivially be set to

$$f_{rr'} = \begin{cases} 1 & : \quad x_r x_{r'} > 0 \\ -1 & : \quad x_r x_{r'} < 0 \end{cases} . \tag{5.9}$$

This solution works perfectly on the training pattern, but what is desired is a grouping principle that is invariant against simple transformations, like translation and rotation. Also, the learned grouping principle should be robust against varying inter-cluster and intra-cluster distances. Further more, a general grouping principle should abstract from the number of point-clusters given in the pattern, and perhaps it even should be possible to segment an pattern, if it is scaled along the dimensions of $\mathcal{F}$.
The only way to achieve such a generalization is to consider relational information between the features. It would cost a lot of training work to adapt a classification-algorithm that works directly on the feature-pairs $(\mathbf{m}_r, \mathbf{m}_{r'}) = (x_r, y_r, x_{r'}, y_{r'})^T$, to achieve such a generalization ability. Therefore, this external knowledge about the learning problem is provided by applying the learning method to relational functions $d(\mathbf{m}_r, \mathbf{m}_{r'})$ which are symmetric under feature exchange $d(\mathbf{m}_r, \mathbf{m}_{r'}) = d(\mathbf{m}_{r'}, \mathbf{m}_r)$. Note, that these functions need not be metrics in a strict mathematical sense, because it is their only purpose to map excitatory and inhibitory features pairs to different regions of the relational space. The relational functions $d(\mathbf{m}_r, \mathbf{m}_{r'})$ can return negative values, violate the triangle inequality and $d(\mathbf{m}_r, \mathbf{m}_r)$ can be unequal zero. Only symmetry is demanded to guarantee the symmetry of the interaction matrix F, which is necessary for the convergence properties of the CLM.

Overlapping and fuzzy interaction classes:

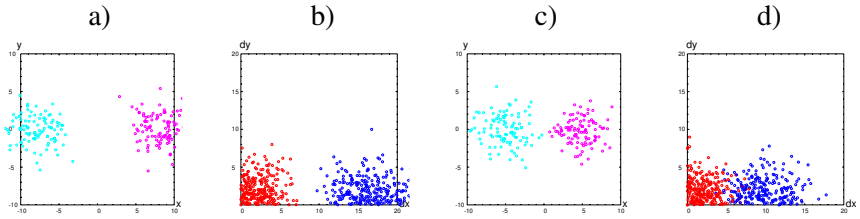


Figure 5.3: Clustering problem with and without perfect separable space of feature pairs

Following the above arguments, a classifier has to be trained on the relational information between features to distinguish between feature-pairs with mutual excitation from features-pairs with mutual inhibition. If the given grouping problem is unambiguous, like the example in Fig. 5.3 a), where two clusters two clusters can be identified whose distance is bigger than the respective cluster radii, this task can be solved perfectly. In Fig. 5.3 b), the distances along the x- and y-axis between features within the same clusters (red) and from both clusters (blue) are plotted. These two classes of feature-pairs can be modeled by the interaction weights

$$f_{ss'} = \begin{cases} 1 & : \quad |x_s - x_{s'}| \leq 9 \\ -1 & : \quad |x_s - x_{s'}| > 9 \end{cases} . \tag{5.10}$$

Unfortunately, this situation does not mirror real world problems, where the borders between the regions of attraction and repulsion of features can not be resolved so easily. If the grouping task is complicated by moving the two clusters closer together, like in Figure 5.3 c), the two classes of interaction weights can not be divided strictly any more (Fig. 5.3 d)). In the area of $3 < |x_r - x_{r'}| < 7$ the two classes intersect. A crisp classifier that tries to discriminate the two classes will have problems in approximating the separation manifold between the classes. Instead, it would be reasonable to use a fuzzy classification function that describes the membership of pairs in the intersection region to one of the two classes by a continuous value, where the sign reflects the dominance of either excitatory or inhibitive pairs and the absolute value shows the degree of this dominance.

5.3 Generalization to New Patterns

In general, the feature domain $\mathcal{F} \in \mathbb{R}^d$ is a high-dimensional discrete or a non finite set and the training examples cover only a small and discrete subset of $\mathcal{F}$. Therefore, the discrete interaction matrix $\hat{F}$ obtained by the Hebbian learning, as it was described above, has to be generalized to an interaction function defined on the full feature domain $f : \mathcal{F}^2 \rightarrow \mathbb{R}$. As proposed already in the context of the quadratic optimization approach to CLM learning [62], generalization can be

obtained by decomposing the interaction function into a linear combination of a set of K arbitrary symmetric basis interaction functions $g^j_{rr'} = g^j(\mathbf{m}_r, \mathbf{m}_{r'}) : \mathcal{F}^2 \rightarrow \mathbb{R}, j = 1, \ldots K$, which are defined on the whole feature space, such that

$$f_{rr'} = \sum_j c_j g^j_{rr'}. \tag{5.11}$$

The analysis above provides an elegant way to choose suitable coefficients c_j by projecting the Hebbian correlation matrix $\hat{F}$ in (5.2) onto the basis functions $\mathbf{g}^j$

$$c_j = \sum_{r,r'} \hat{f}_{rr'} g^j_{rr'} / \parallel \mathbf{g}^j \parallel . \tag{5.12}$$

The basis functions embody contextual knowledge that is used to reduce the dimension of the learning problem. Equation (5.11) is only a good approximation of $f_{rr'}$, if there is a high overlap between $\hat{f}_{rr'}$ and the shape of $g^j_{rr'}$, such that the coefficients c_j describe the relevant components of $\hat{f}_{rr'}$. Therefore, the manual definition and adjustment of suitable basis interactions requires both problem-specific knowledge and detailed knowledge about the functionality of the CLM and results in much training and testing work for the user. This raises the problem, how suitable basis functions can be generated automatically without making assumptions about the shape of the interaction function [59]. The main focus lies here on a high evaluation speed of the learned interaction function. This point is essential for the practical application of the CLM, because the number of lateral interaction weights that have to be computed is quadratic in the number of features in the input pattern, so already the computation of the matrix F can be very resource demanding, if the definition of basis functions is too complex.
Assume the basis functions as binary step functions ($g^j_{rr'} \in \{0, 1\}$) that describe a disjunct partitioning of the space $\mathcal{F}^2$ ($g^j_{rr'} g^i_{rr'} = \delta_{ij}$). This partitioning should be symmetric according to feature exchange ($g^j_{rr'} = g^j_{r'r}$) to ensure symmetry of the matrix F. Now, the projection of the theoretical interaction onto the basis functions (5.12) can be simplified to the average interaction value within the respective region of $\mathcal{F}^2$:

$$c_j = \sum_{r,r'} \hat{f}_{rr'} g^j_{rr'} / \sum_{rr'} g^j_{rr'}. \tag{5.13}$$

This average can be approximated by random sampling of feature pairs $(\mathbf{m}_r, \mathbf{m}_{r'})$ from the training set, evaluation of the basis functions and using $\hat{f}_{rr'} = 1$, if the features belong to the same group, and $\hat{f}_{rr'} = -1/(L-1) = -\Lambda$ (remember section 5.1), if they belong to different groups, respectively.
(5.11) can only be a good approximation of $f_{rr'}$, if the partitioning described by the basis functions defines regions which consist either mainly of feature pairs with $\hat{f}_{rr'} > 0$ or mainly of feature-pairs with $\hat{f}_{rr'} < 0$, such that the interaction coefficients can adopt high absolute values. Therefore, the borders of the basis functions should describe the borders between positive and negative values $\hat{f}_{rr'}$ in $\mathcal{F}^2$ as good as possible.

There are two principal approaches to this problem: The first is to discretize $\mathcal{F}^2$ to a lattice structure. However, through the curse of dimensionality in $\mathcal{F}^2$ for a fine discretization results in an impracticable high number of basis functions.
The second approach is to train a standard classifier in $\mathcal{F}^2$ to separate the regions of $\hat{f}_{rr'} > 0$ and $\hat{f}_{rr'} < 0$ from each other. This approach was tested in [60] by application of the Multi Layer Perceptron (MLP) and the Support Vector Machine (SVM) on the problem of fluorescence cell image segmentation. The results will be presented in section 6.3.1, where they are compared to the application of the original QCO learning of Wersing and the new learning algorithm derived here.
The following section presents an alternative to these two approaches for finding basis functions based on an unsupervised learning step from a variant of Vector Quantization (VQ) that results in a very simple learning algorithm and high evaluation speed.

5.4 Data Driven Generation of Basis Functions

As $f_{rr'}$ and thus as well $g^j_{rr'}$ expresses some degree of mutual compatibility or proximity of a feature pair $(\mathbf{m}_r, \mathbf{m}_{r'})$ these values are transformed into a generalized proximity space $\mathcal{D}$ by a vector function

$$\mathbf{d}_{rr'} = [a_1 d_1(\mathbf{m}_r, \mathbf{m}_{r'}), \ldots, a_P d_P(\mathbf{m}_r, \mathbf{m}_{r'})]^T, \tag{5.14}$$

where each component $d_p(\mathbf{m}_r, \mathbf{m}_{r'})$ defines a proximity function according to some properties of the features $\mathbf{m}_r$ and $\mathbf{m}_{r'}$. These components are normalized by their variance in the pattern or left unchanged, if the variance is zero:

$$a_p = \begin{cases} \frac{1}{\sigma^2(d_p(\mathbf{m}_r, \mathbf{m}_{r'}))} & : \quad \sigma^2(d_p(\mathbf{m}_r, \mathbf{m}_{r'})) > 0 \\ 1 & : \quad \sigma^2(d_p(\mathbf{m}_r, \mathbf{m}_{r'})) = 0 \end{cases} . \tag{5.15}$$

It has to be emphasized, that the proximity functions $d_p(\mathbf{m}_r, \mathbf{m}_{r'})$ need not define a metric in a strict sense, e.g. proximity between local edge feature $\mathbf{m}_r = (\mathbf{p}_r, \mathbf{o}_r)$, given by position $\mathbf{p}_r$ and orientation $\mathbf{o}_r$, can be defined by local distance $\parallel \mathbf{p}_r - \mathbf{p}_{r'} \parallel$ as well as the scalar product $\mathbf{o}_r^T \mathbf{o}_{r'}$ describing the angle between the edges. The only constraint is, that they are symmetric under feature-exchange $d_p(\mathbf{m}_r, \mathbf{m}_{r'}) = d_p(\mathbf{m}_{r'}, \mathbf{m}_r)$, such that there is a high freedom in defining grouping principles for the interaction function. It is possible to introduce additional properties, like translation, rotation and scale invariance of the proximity functions, in order to achieve a generalization of the desired grouping behavior according to object properties like position, size and orientation.
In the next step, a variation of Self-Organizing Map, the activity equilibration vector quantization AEV [16], is used to reduce the proximity vectors $\mathbf{d}_{rr'}$ to a set of K prototypes $\tilde{\mathbf{d}}_j$.
Now, the multidimensional Voronoi cells

$$V_j = \{(\mathbf{m}_r, \mathbf{m}_{r'}) \mid \forall i \neq j : \parallel \mathbf{d}_{rr'} - \tilde{\mathbf{d}}_j \parallel \leq \parallel \mathbf{d}_{rr'} - \tilde{\mathbf{d}}_i \parallel\}. \tag{5.16}$$

defined by the proximity prototypes can be use to define the basis functions[59]:

$$g^j_{rr'} = \begin{cases} 1 & : \quad (\mathbf{m}_r, \mathbf{m}_{r'}) \in V_j \\ 0 & : \quad (\mathbf{m}_r, \mathbf{m}_{r'}) \notin V_j \end{cases} . \tag{5.17}$$

Since clustering techniques based on SOMs are dependent on the initialization of prototypes and, therefore, might result in states with "unused" prototypes, which means low activation of prototypes $\tilde{\mathbf{d}}_j$ measured by low values of $A_j = |V_j|$, the AEV uses a heuristics to reinitialize prototypes with low activation close to prototypes with a high activation that causes an activity equalization of the prototypes, such that in the following the normalization term in (5.13) can be neglected.
The representation of the basis functions as Voronoi cells in the proximity space simplifies the computation of the interaction weights $f_{rr'}$. For a given feature pair we only have to compute the proximity vector $\mathbf{d}_{rr'}$, evaluate it's nearest neighbor from the set of K prototypes $\tilde{\mathbf{d}}_j$ and return the interaction coefficient c_j of this prototype. Figure 5.6 shows a sketch of this procedure.

5.5 Aspects of Implementation

The special structure of the theoretically derived interaction matrix $\hat{F}$ in section 5.1 and the basis functions in the last section motivate an efficient straight-forward learning algorithm. This algorithm gives also the possibility to control the segmentation level of the grouping result, as the following shows.
Since activation equalization is a part of the clustering algorithm for the design of basis functions, the normalization term in equation (5.13) can be neglected for the computation of the interaction coefficients c_j. Further, section 5.1 has shown, that the theoretical interaction weights $\hat{f}_{rr'}$ can adopt only two different values, namely one positive and one negative, where the negative value is scaled against the positive one by the factor $\Lambda = \frac{1}{L-1}$. Therefore, (5.13) is rewritten as

$$c_j = \sum_{r,r'|\hat{\alpha}(r)=\hat{\alpha}(r')} g^j_{rr'} - \sum_{r,r'|\hat{\alpha}(r)\neq\hat{\alpha}(r')} \Lambda g^j_{rr'}. \tag{5.18}$$

This term can be computed by separately counting the feature-pairs $(\mathbf{m}_r, \mathbf{m}_{r'})$ with positive and negative values $\hat{f}_{rr'}$ within the Voronoi cells of the corresponding basis functions.

$$c^+_j = \sum_{(\mathbf{m}_r,\mathbf{m}_{r'})\in V_j|\hat{\alpha}(r)=\hat{\alpha}(r')} 1; \quad c^-_j = \sum_{(\mathbf{m}_r,\mathbf{m}_{r'})\in V_j|\hat{\alpha}(r)\neq\hat{\alpha}(r')} 1, \tag{5.19}$$

Afterwards the two parts can be combined to the desired learning parameters by

$$\mathbf{c} = \mathbf{c}^+ - \Lambda \mathbf{c}^-. \tag{5.20}$$

where $\mathbf{c}$, $\mathbf{c}^+$ and $\mathbf{c}^-$ are the vectors of the values c_j, c^+_j and c^-_j, $j = 1, \ldots, K$ respectively.

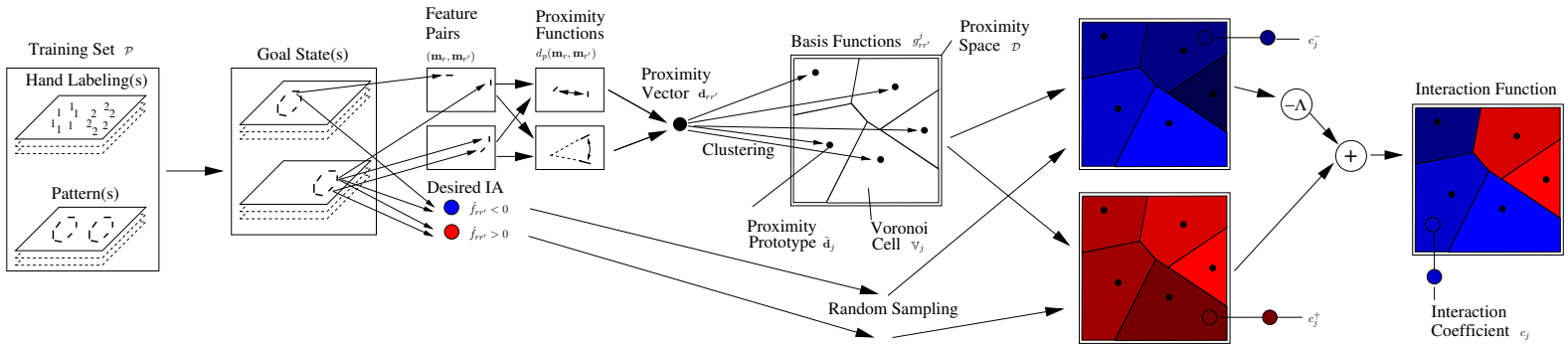


Figure 5.4: Sketch of the Approximated Hebbian Learning (AHL). Starting from a set of hand-labeled patterns, which can be interpreted as target states of the CLM-dynamics, the learning algorithm extracts two aspects of information. On the one hand, each pair of features $(\mathbf{m}_r, \mathbf{m}_{r'})$ can be mapped onto a proximity vector $\mathbf{d}_{rr'}$ by applying a set of predefined proximity functions $d_p(\mathbf{m}_r, \mathbf{m}_{r'}), p = 1, \ldots, P$. On the other hand, each feature-pair can be mapped onto a desired positive (red) or negative (negative) interaction value. In the first learning phase, a set of randomly chosen proximity vectors $\mathbf{d}_{rr'}$ is clustered to achieve a Voronoi partitioning of the proximity space. In the second learning phase, the distribution of positive and negative interactions within the proximity space is measured by counting feature pairs with positive and negative desired interaction within the Voronoi cells (dark color shows low concentration of feature-pairs and light color shows high concentration). These two density distributions are added to a resulting interaction function by multiplication of the distribution of negative interaction with the factor $-\Lambda$ and adding both distributions.

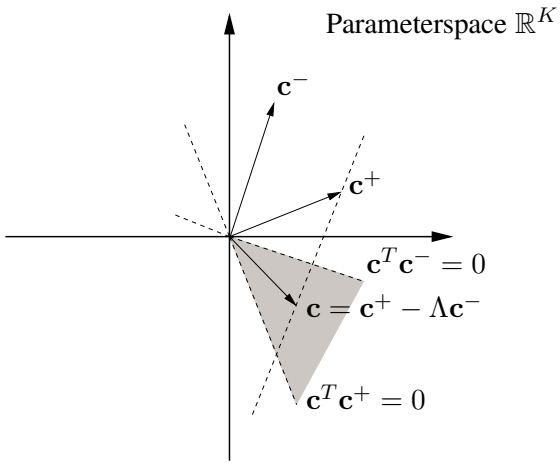


Figure 5.5: Choice of Λ: The vector $\mathbf{c}^+$ describes the projection of the positive components of F to the basis functions $g^j_{rr'}$, while $\mathbf{c}^-$ shows the projection of the negative components of F to the basis functions. The vector of interaction coefficients $\mathbf{c}$ is computed by $\mathbf{c} = \mathbf{c}^+ - \Lambda\mathbf{c}^-$. Fulfillment of the consistency conditions is implied, if Λ is chosen, such that c lies between the lines $\mathbf{c}^T\mathbf{c}^- = 0$ and $\mathbf{c}^T\mathbf{c}^+ = 0$. The right choice of Λ depends essentially on the angle between $\mathbf{c}^+$ and $\mathbf{c}^-$. If both vectors are linear independent, Λ can be chosen almost arbitrarily, while the feasible interval for Λ shrinks as smaller the angle between $\mathbf{c}^+$ and $\mathbf{c}^-$ becomes. Therefore, the qualitative performance of the learning approach depends on a good choice of basis functions, which are ideally orthogonal for feature pairs within and between groups.

Obviously, the quality of the so-constructed interaction matrix depends on the angle between the vectors $\mathbf{c}^+$ and $\mathbf{c}^-$, because clear positive or negative interaction weights $f_{rr'}$ can only arise, if for each basis function $g^j_{rr'}$ either the value of c^+_j dominates against c^-_j or c^-_j against c^+_j respectively. Otherwise, the values of the c^+_j and c^-_j annihilate each other, such that the resulting c_j coefficients are close to zero. In this sense, the learned interaction is optimal, if $(\mathbf{c}^+)^T\mathbf{c}^- = 0$ holds, while in the other cases it depends on the strength of the factor Λ and the relation between c^+_j and c^-_j whether the corresponding coefficient c_j becomes positive or negative. Thus the angle between $\mathbf{c}^+$ and $\mathbf{c}^-$ is a good measure for the evaluation of learning success. It yields information whether the basis functions have been chosen appropriately and whether the labeled training pattern is compatible with this choice.

Now regard the scaling factor Λ not as a fixed value, but as a variable parameter, then equation (5.20) defines a line within the learning parameter space $\mathbb{R}^K$ (see Fig. 5.5), where the choice of Λ effects the grouping behavior of the CLM: an increase of Λ biases the interaction function towards a higher attraction, which results in

fewer, but bigger groups in the segmentation of new patterns; a decrease biases the interaction function towards a higher repulsion, which results in more, but smaller groups in the output of the CLM. Thus after training of $\mathbf{c}^+$ and $\mathbf{c}^-$, Λ can be used to adjust the CLM to a finer or rougher segmentation of the input pattern.
To constrain the range of Λ, once more the consistency conditions (5.1) are inspected, which are fulfilled, if the left hand side of equation (5.1) is always < 0 and the right hand side is always > 0. By separate summation of all left hand sides and all right hand sides, the insertion of the superposition of basis functions (5.11) and the comparison of the results with the definitions of the components c_j^+ and c_j^- in (5.19), two inequality constraints for $\mathbf{c}$ can be derived (for details see appendix B):

$$\mathbf{c}^T\mathbf{c}^+ > 0 \quad \text{and} \quad \mathbf{c}^T\mathbf{c}^- < 0. \tag{5.21}$$

From (5.20) and (5.21) an upper and lower bound for Λ can be derived (see Fig. 5.5) :

$$\frac{(\mathbf{c}^+)^T\mathbf{c}^+}{(\mathbf{c}^-)^T\mathbf{c}^+} > \Lambda > \frac{(\mathbf{c}^+)^T\mathbf{c}^-}{(\mathbf{c}^-)^T\mathbf{c}^-}. \tag{5.22}$$

A linear search between the upper and lower bound can be performed to adjust Λ to a desired segmentation level. In the following, the convention is used that $\mathbf{c}^+$ and $\mathbf{c}^-$ will be normed by $\frac{1}{\|\mathbf{c}^+\|}$ and $\frac{1}{\|\mathbf{c}^-\|}$, where $\| \cdot \|$ means the sum of all (per definition nonnegative) values c_j^+ respectively c_j^- to uniquely specify the value of Λ.
Since the generation of basis functions as well as the estimation of the interaction coefficients depends on the statistical distributions of the feature-pairs within the proximity space, both learning phases can be applied on a representative sample set of feature-pairs for the training pattern. This can be realized by a fixed number of clustering respectively sampling steps, where the computational complexity of these two types of steps depends of the complexity of the proximity functions and the nearest neighbor search in the prototype vectors of the basis functions. The number of learning steps in each learning phase has to be chosen large enough to get an adequate number of feature-pairs for each basis function, but, because of the simplicity of each learning step even large numbers of learning steps result in a high learning speed of the whole algorithm, such that the number of learning steps is not a critical parameter of the algorithm.
In this framework, the only choice left to the user is the selection of appropriate feature properties, like color, position, orientation, an adequate distance function $d_p(\mathbf{m}_r, \mathbf{m}_{r'})$ and the number of basis functions K. From this choice, the learning is fully self-contained with Λ as the only free parameter left for setting a bias on larger or smaller groups.

5.6 Training Sets with several Training Patterns.

Up to now, it was assumed that the training set consists only of a single pattern. This approach can be sufficient to learn a grouping behavior, because the learn-

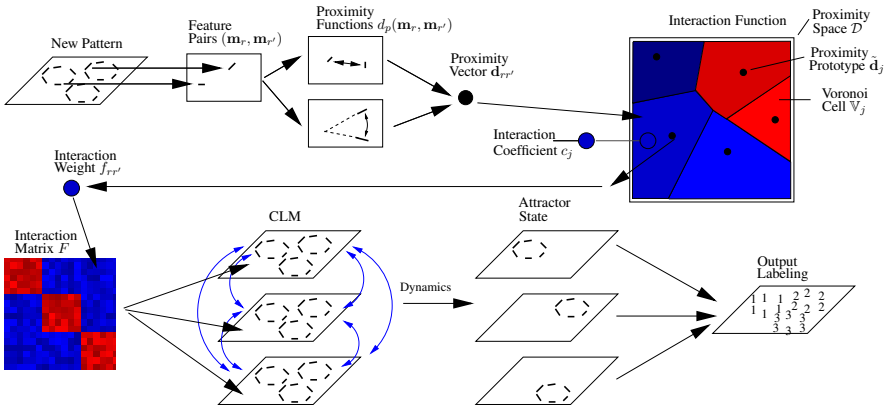


Figure 5.6: Sketch of the application process. An interaction matrix F is constructed for a new pattern by mapping each pair of features $(\mathbf{m}_r, \mathbf{m}_{r'})$ onto their proximity vector $\mathbf{d}_{rr'}$ and taking the interaction coefficient c_j of the nearest proximity prototype $\tilde{\mathbf{d}}_j$ as their pairwise interaction. Then F is used as the weight-matrix for the lateral connections within the layers of the CLM and the dynamics of the CLM is simulated, until it converges to an attractor state which defines the output labeling.

ing algorithm mainly uses statistics on pairwise feature relations and their correlation according to the hand labeling. A single patterns can comprise a huge set of pairwise feature combinations and, therefore, might be sufficient to represent the statistics of the desired grouping behavior.
The algorithm can be applied to a training set with several patterns without major modifications. The scaling of proximity functions (5.15), the clustering step of AEV and the estimation of $\mathbf{c}^+$ and $\mathbf{c}^-$ (5.19) can be simply performed on a representative sample of feature-pairs, which is uniformly selected from all patterns. The only thing that has to be provided is, that the two features forming a pair are drawn from the same pattern.
An implementation of the whole learning algorithm, which in the following is called "Approximated Hebbian Learning" (*AHL*), can be found in the appendix C. A summary of the complete approach is sketched in Fig. 5.4. An additional sketch for the application process of the CLM is given in Fig. 5.6.

5.7 Estimation of Background Layer Strength

If the CLM contains a special background layer g, the strength of the self-interaction m in the background layer has to be weighted against the lateral interactions $f_{rr'}$. A feature $\mathbf{m}_r$is assigned to the background, if m is larger than the support in all other layers, while it is assigned to the relevant groups, if the support in at least one layer is larger than m. Therefore, m is constrained by additional consistency conditions, where m has to be bigger or smaller than the maximal lateral support

$\max_\alpha \sum_{r'} f_{rr'} x_{r\alpha}$ of the feature $\mathbf{m}_r$ in the other layers $\alpha \neq g$ whether the feature $\mathbf{m}_r$ should be assigned to the background or not.
A suitable value for m can be estimated by assuming the average maximal support of all features in the background as lower bound m_{low}

$$m_{low} = \frac{\sum_{\mathbf{m}_r|\hat{\alpha}(r)=g} \max_\alpha \left(\sum_{r'} f_{rr'} y_{r'\hat{\alpha}}\right)}{\#\mathbf{m}_r|\alpha(r) = g}, \tag{5.23}$$

and the average maximum support of all features in the figure groups as upper bound m_{up} for m.

$$m_{up} = \frac{\sum_{\mathbf{m}_r|\hat{\alpha}(r)\neq g} \max_\alpha \left(\sum_{r'} f_{rr'} y_{r'\hat{\alpha}}\right)}{\#\mathbf{m}_r|\alpha(r) \neq g}. \tag{5.24}$$

A simple heuristics, which proved to work, is to set m between m_{low} and m_{up}, e.g.

$$m = (m_{low} + 3m_{up})/4. \tag{5.25}$$

5.8 Summary

This chapter has introduced the new AHL method for the CLM learning problem. It started with general and theoretical assumptions about the learning problem. Showing parallels to the Hopfield network, it was argued, that learning can be realized in principle by binary classification on pairwise feature relations. Thereby, some problems were expected because of the possible overlap between the two classes of excitatory and inhibitory feature-pairs. Further, the necessity of parameter tuning should be prevented and the trained classifier should have a high application speed, because during the computation of a lateral interaction matrix F the classifier has to be applied for each pair features in the input pattern.
Due to the high redundancy of the CLM, it is not required to find the optimal classifier (compare experiments in section 4.4), such that in this work the AHL-algorithm was derived. It transforms feature-pairs to a predefined, problem-specific proximity space, performs vector quantization to segment this space into a set of basis functions, and estimates the partitions of inhibitory and excitatory feature-pairs according to these basis functions.
These two partitions are combined to an interaction function that computes the lateral interaction weights of the CLM for new input patterns, where the weighting between the two partitions controls the segmentation level of the CLM output grouping.
The AHL algorithm can be implemented very efficiently by a number of clustering and sampling steps within the feature proximity space. Thereby, the main learning parameter is the number of basis functions within the proximity space, which dominates the accuracy and the evaluation speed of the learned interaction function. The other parameters can be specified naturally for the user by presenting a set of hand labeled example patterns.

To be able to learn a demonstrated grouping behavior, the AHL algorithms requires knowledge from the application designer about the feature structure and the proximity functions between the features that characterize the relevant grouping principles. A clever choice of proximity functions is essential to guarantee, that AHL can separate excitatory and inhibitory feature-pairs in the proximity space, but also gives the possibility to determine some a priori properties of the learned grouping behavior, like translation and rotation invariance.
Surely, the construction of suitable proximity functions for the AHL algorithm faces the same problems as designing heuristics for complete interaction functions. However, these problems occur on a much simpler level, where grouping principles can be described by the underlying distance functions of elementary features, e.g. smoothness and continuity by angles between edges, similarity of color by distances in color spaces and similarity of texture by distance of Gabor jet responses, while the further parameters, like the range and strength of excitatory and inhibitory connections and the weighting of different proximity principles, are extracted from the target labeling.

Chapter 6

Application

So far, the AHL algorithm has been motivated and it's theoretical properties have been discussed. This chapter now shows, how AHL performs practically on the problems of point clustering, fluorescence cell image segmentation, texture segmentation and contour grouping.

6.1 Point Clustering

As a simple example, the clustering of 2d-points $\mathbf{m}_r = \mathbf{p}_r = (x_r, y_r)^T$, which is implemented in [50] by an on-center-off-surround function (compare section 3.6 Point Clustering), is inspected. The top row of Fig. 6.1 shows four example patterns from this problem domain. Pattern 1 shows 160 points which are equally distributed on two clusters at the positions (8,0) and (-8,0), where each cluster has a radius of 8, such that the two clusters show slight contact. In pattern 2, the left cluster is divided into two subclusters at position (-8,4) and (-8,-4) with radius 4 and a size of 40 points per cluster, such that the two subclusters again show slight contact. In pattern 3 and 4, this splitting procedure is iterated for one of the smallest clusters. The four patterns describe borderline cases of separating neighboring, non-overlapping clusters, where the conflict between forming large clusters and separating small clusters increases from pattern 1 to 4.
To be able to apply AHL, the necessary problem knowledge has to be introduced by specifying a set of proximity functions $d_p(\mathbf{m}_r, \mathbf{m}_{r'}), p = 1, \ldots, P$. Since the features are described only by their position $\mathbf{p}_r$, a single proximity function, defined by the Euclidean distance, is chosen:

$$d_1(\mathbf{m}_r, \mathbf{m}_{r'}) = \| \mathbf{p}_r - \mathbf{p}_{r'} \| . \tag{6.1}$$

For clusters with a different shape, e.g. oval, an alternative would be to use separate distance functions for the different feature components, whose scaling factors then can be adapted to the variance of the clusters in different orientations. But since in the example circular clusters are inspected, one function is adequate, such that the proximity prototypes $\tilde{\mathbf{d}}_j, j = 1, \ldots, K$ are one dimensional.

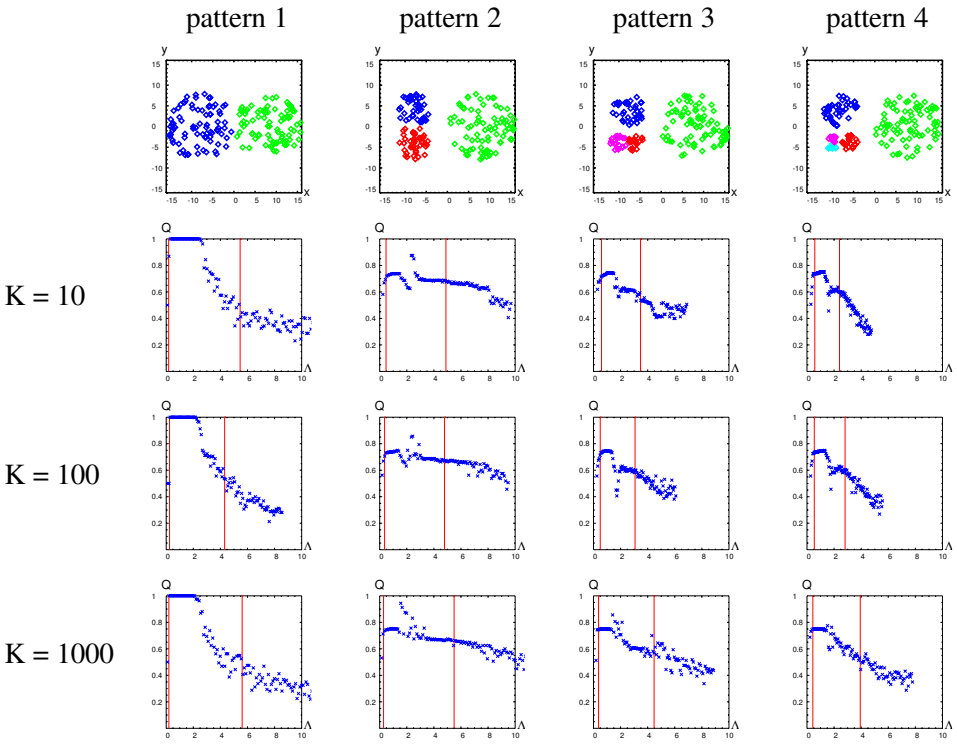


Figure 6.1: Influence of the control parameter Λ on the grouping quality. The top row shows four training patterns from the problem domain of point clustering. The plots in the rows two to four show, how the grouping quality on the training pattern, achieved from the CLM with an AHL-learned interaction function of $K = 10$, $K = 100$ and $K = 1000$ basis functions, develops with the modification of the control parameter Λ. The bounds Λ_{min} and Λ_{max} derived from (5.22) are marked by vertical red lines within the plots.

As a next step, the number of basis functions has to be chosen and the AHL algorithm has to be applied. For each of the four patterns, AHL is trained with $K = 10$, $K = 100$, and $K = 1000$ basis functions to estimate the positive and negative interaction coefficients c_j^+ and $c_j^-, j = 1, \ldots, K$. Finally, a value for Λ has to be chosen to adjust the segmentation level of the constructed interaction function.
Figure 6.2 shows the effect of Λ on the shape of the interaction function resulting from training with pattern 1 and $K = 10$ basis functions. The values of the resulting interaction coefficients c_j at specified Λ values are plotted against the one dimensional prototypes $\tilde{\mathbf{d}}_j$. The color of the data points (red = positive, blue = negative) shows the sign of the corresponding interaction coefficients. It can be seen, that the range and the strength of positive interactions decreases with the increase of Λ, such that the grouping behavior changes from forming few large clusters to forming many smaller clusters.

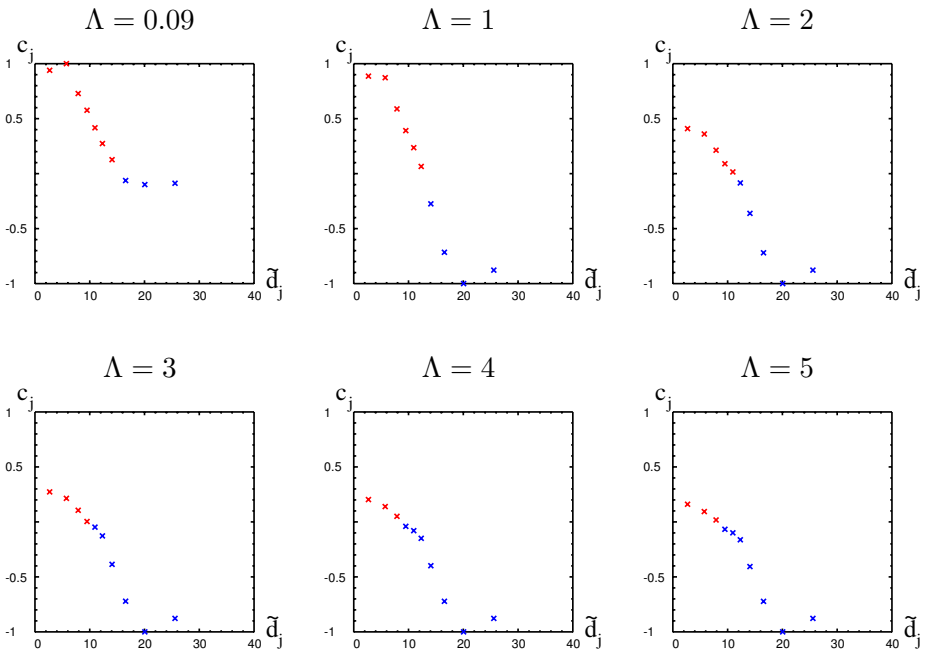


Figure 6.2: Influence of the control parameter Λ onto the shape of the interaction function. The plots shown the interaction coefficients of the ten prototypes resulting from training with pattern 1 and $K = 10$ basis functions for the specified values of Λ. The interaction coefficients are normalized, such that their maximal absolute value is equal to one. Positive coefficients have red color, and negative coefficients have blue color. The position of the prototype according to the local distance of two features in shown on the x-axis, while the interaction value of the interaction coefficients is shown on the y-axis.

The question is: How must Λ be chosen to achieve an optimal segmentation of the training pattern according to the target labeling? And, is it always possible to reconstruct the target labeling 100 % correctly? To examine these questions, generalization aspects of the training are neglected and only the grouping performance of the learned interaction functions on the training patterns themself is inspected.

Figure 6.1 shows, how the grouping quality changes with the choice of Λ. The plots in a column show results from training with the corresponding pattern, the rows 2, 3 and 4 show the results from training with $K = 10$, $K = 100$ and $K = 1000$ basis functions, where each learning phase (see AHL implementation in appendix C) consists of 10000 steps. Each plot shows the values of Λ on the x-axis and the quality values Q resulting from the application of the learned interaction function on the training pattern on the y-axis. The vertical, red lines mark the interval of Λ_{min} and Λ_{max} estimated from training. The value of Λ runs with a constant step size from $0.5\Lambda_{min}$ to $2\Lambda_{max}$. At each step, the resulting interaction

function is applied within an $L = 20$-layered CLM on the training pattern. During the annealing process, T is decreased from $T = 100$ to $T = 0$ in 1000 steps of constant step size. At $T = 0$, 100 additional steps are performed. Each annealing step consists of $L \cdot N$ neuron updates, where $N = 160$ in each pattern. The state of the CLM at the end of the annealing process is assumed as attractor state, where the maximal activated neuron in each column specifies the output label. These output labels are compared with the target labeling according to the the algorithm in section 4.2. The resulting quality values Q specify the y-values of each data point.

The interval of Λ_{min} and Λ_{max}, where the optimal Λ value is expected, shrinks from pattern 1 to 4. This mirrors, that the grouping problem becomes more difficult and inconsistent to the proximity function, such that the overlap between positive and negative interaction increases. The optimal value lies always between Λ_{min} and Λ_{max}. However, the optimal quality reaches only 100% for pattern 1. For the other patterns, each choice of Λ causes errors by splitting the bigger clusters or merging the smaller clusters, which is represented by an optimal quality around 95%, 85% and 80%. For pattern 1, the optimal value is reached in a wide interval which, however, covers only the half of the interval $[\Lambda_{min}, \Lambda_{max}]$. For the other patterns, the optimal value lies on a sharp spike in the quality plot.

The increase of K brings for most values of Λ no gain in the grouping quality. However, for patterns 2 and 3 the spike of maximal quality is more distinct in the quality plot of $K = 1000$, but this increase of quality means also a significant increase of the evaluation time of the next neighbor criterion during the computation of the lateral interaction weights.

It can be concluded, that pattern 1, which shows an extreme case of two neighboring clusters, can be reconstructed correctly by the AHL algorithm. The exact grouping can not be restored for the different scale levels in the pattern 2, 3, and 4, where either small groups are merged of large groups are split. An automatic estimation of Λ seems to be challenging, because the process of merging and splitting of small and large groups can have contradictory effects on the quality, such that there can be local minima in the graph of quality along Λ. Further, the evaluation of the quality demands the simulation of the CLM dynamics, which can be very time consuming, especially, at a slow annealing speed. For this reason in the following Λ is chosen by hand.

6.2 Spiral Problem

In the last example, circular clusters were inspected which show a compact shape. In the next example, the principle of proximity is applied to extreme cases of clusters in form of two spiral arms. The first column of Fig. 6.3 shows four example patterns from this problem domain. Each pattern shows a set of 40 two-dimensional points $\mathbf{m}_r = \mathbf{p}_r = (x_r, y_r)^T$. These points are divided into two groups. Each group is specified by a two-dimensional spiral function $\mathbf{s}_\alpha(R)$ with a predefined

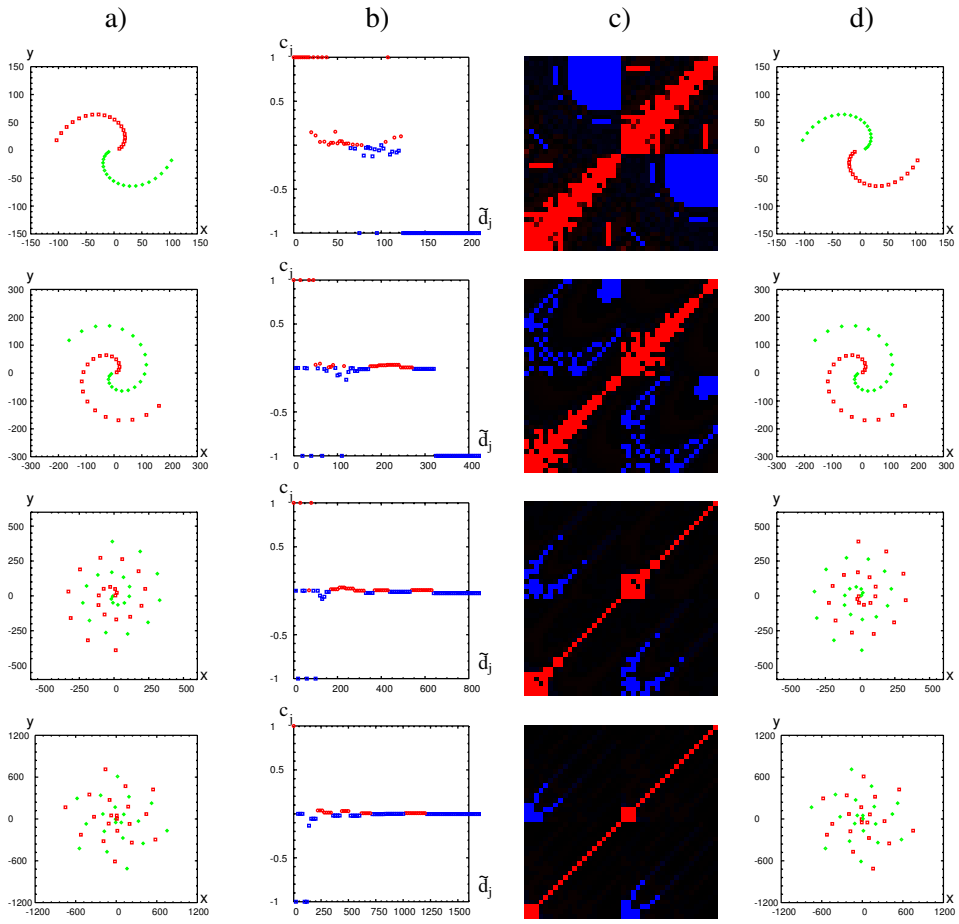


Figure 6.3: Spiral Problem: Two spiral shaped sets of 2d-points shall be separated from each other. a) Pattern and target labeling for training with AHL ($K = 100$). b) Visualization of the learned interaction coefficients c_j (y-axis) according to the 1d-proximity prototypes $\tilde{d}_j$ (x-axis) represented by the local distance. c) Visualization of the reconstructed interaction matrix F. The features along the rows and columns of F are ordered according to their target labeling and their appearance on the two spiral arms. d) Output labeling of an $L = 2$-layered CLM resulting from the interactions in c).

curvature, where R is the distance of a point on the spiral to the origin of the two spirals. The two spirals are rotated around each other by $180°$, such that they are contorted into each other. The points on each spiral arm are generated by evaluating the two spiral functions for 20 points, whose distances to the central point increase with a constant step size from R_{min} to R_{max}. R_{min} is set to 10 to ensure, that there is a gap between the start points of the two spirals, while for R_{max} the values 100, 200, 400 and 800 are chosen in pattern 1, 2, 3 and 4, such that the

degree of contortion between the two spiral arms increases from pattern 1 to 4.
The proximity between two features is specified, like in the previous example, by the local distance

$$d_1(\mathbf{m}_r, \mathbf{m}_{r'}) = \| \mathbf{p}_r - \mathbf{p}_{r'} \| . \tag{6.2}$$

The number of basis functions K is set to 100, where each learning phase consists of 10000 learning steps. The control parameter Λ is set to the value 0.7.
The second column of Fig. 6.3 shows the interaction functions that result from training with the four example patterns. The x-axis describes the one-dimensional proximity prototypes, the y-axis the interaction coefficients and the color the sign of the interaction coefficients.
Training with pattern 1 results in clear positive interactions for small distances and clear negative interactions for high distances. In the intermediate range, positive and negative interaction annihilate each other, but the regions where positive and negative interactions dominate alternate one time, which mirrors the curvature of the spiral functions.
For pattern 2 to 4, the learning problem becomes more difficult, such that there is a higher overlap between positive and negative interaction, which results in more interaction coefficients close to zero. The stronger curvature of the spiral arms in the pattern 2, 3 and 4 results in a higher alternation of positive and negative interactions according to the local distance.
The third column of Fig. 6.3 shows the interaction matrices that result from the application of the learned interaction functions onto the training patterns. Positive entries are visualized by red color, negative entries are visualized by blue color, and the absolute values are described by the color intensities. The features are ordered along the x- and y-axis according to their target labeling, such that in the ideal case a block-diagonal structure can be expected.
For pattern 1, the reconstructed interaction matrix comes close to this block-diagonal structure. For pattern 2 to 4, this structure degenerates increasingly, because most of the interaction coefficients are close to zero. For pattern 3 and 4, the positive sub-blocks become more and more similar to the identity matrix. Consequently, the annealing process has to be performed very slowly to prevent an over segmentation of the two spiral arms. For the same reason the interaction matrices are only applied only in a $L = 2$-layered CLM.
The fourth column of Fig. 6.3 shows the labeling result from this two-layered CLM. The annealing process is performed in 200 steps of constant step size for $T = 50$ to $T = 0$. Each annealing step consist of $L \cdot N = 2 \cdot 40 = 80$ neuron updates. The input labeling can be reconstructed for pattern 1 to 3. However, at least for pattern 3 it seems that the correct grouping is only achieved, because the number of layers is restricted to the correct number of groups. For pattern 4, the deformation of the interaction matrix is too strong, such that errors occur in the output labeling.
Conclusion: From this example application it can be seen, that the global structure of patterns, like the membership to the two spiral arms, which usually requires

information about the relative positions of the features to the spiral origins, can be reconstructed by local feature relations, like in this case the local distance, as long as positive and negative interactions can be clearly separated according to these relations. Further, the increase of inconsistency affects the learning and grouping result. It should be clear, that the grouping performance can be enhanced for the spirals arms in pattern 3 and 4 by inserting more features, such that a higher sampling rate to the two groups is achieved. This shifts the interaction coefficients for smaller distances stronger into the positive values, such that a higher inner group support arises, like in the case of pattern 1 and 2.

6.3 Segmentation of Fluorescence Images

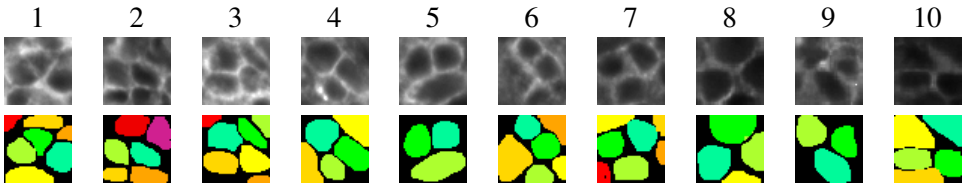


Figure 6.4: Dataset of fluorescence cell images. The ten images in the upper row show areas of fluorescence cell probes that shall be divided into the regions of the occurring cells, origin [33] and [35]. The lower row shows a human hand labeling of the images.

In this section, AHL is applied to a real-world problem, namely the segmentation of fluorescence cell images as introduced in section 3.6 and investigated also by Wersing [61] (see section 3.7.4).
The first row of Fig. 6.4 shows the dataset that represents this problem domain. It consists of ten 45×45 images, such that each pattern consists of 2025 features. In each image the relevant groups representing cells are indicated by a dark cell body and a light corona. These images are a part of the data source in [33] and [35]. The second row of Fig. 6.4 shows a human hand labeling that is applied as target labeling for the training.
In this domain four experiments are performed. The first shows that learning can be implemented in principle by standard binary classifiers in the proximity space. The second experiment describes the influence of the parameter Λ in the AHL algorithm on the segmentation results. The third experiment compares and discusses several ways of describing the grouping behavior in different proximity spaces. And the last experiment shows the influence of different labeling strategies.

6.3.1 Application of Binary Classifiers

The AHL algorithm can be interpreted as a kind of binary fuzzy-k-nearest-neighbor classifier. Therefore, it is compared with the application of standard classifiers from

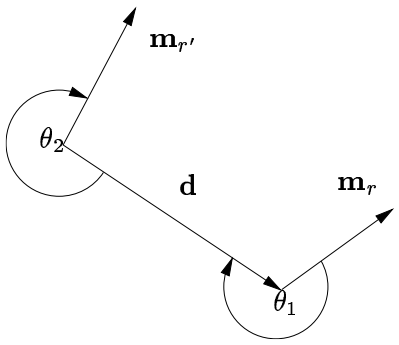


Figure 6.5: Edge proximity measures for the cell segmentation problem: Proximity is expressed by the local distance $\| \mathbf{d} \|$ between two edges and the relative angles $\theta_1, \theta_2 \in [0, 2\pi]$ between the edge directions and the connecting vector $\mathbf{d}$.

the field of artifical neural networks.
The feature vectors are represented as $\mathbf{m}_r = (\mathbf{p}_r, \varphi_r)$, where $\mathbf{p}_r$ is the pixel position and φ_r is the orientation of the intensity gradient described by Sobel-x and Sobel-y filters. Features with gradient strength equal to zero are removed from the pattern, such that the real number of features N can be slightly smaller than 2025. Proximity between the features is defined, similar to Wersing's approach, by the local distance

$$d_1(\mathbf{m}_r, \mathbf{m}_{r'}) = \| \mathbf{p}_r - \mathbf{p}_{r'} \| \tag{6.3}$$

and the relative angles θ_1 and θ_2, see Fig. 6.5. To ensure the uniqueness and symmetry under feature exchange, $\mathbf{m}_r$ and $\mathbf{m}_{r'}$ are swapped, such that the vector $\mathbf{d} = \mathbf{p}_r - \mathbf{p}_{r'}$ always points from the left to the right or from the top to the bottom, if both features have the same x-coordinate. The angles θ_1 and θ_2 always start in direction of the vector $\mathbf{d}$ and turn to the right till φ_r respectively $\varphi_{r'}$ is reached.
Training is always performed on one of the ten images in Fig. 6.4 and the resulting interaction function is tested on all ten images (including the training image) by comparing the output labeling of the CLM with the target labeling.
The algorithms QCO ($\kappa = 100$), AHL ($\Lambda = \frac{1}{L^i - 1}$) and Average Consistency Condition (ACC) are applied for learning. ACC is a variant of QCO, where the optimization step (3.42) is avoided by setting the interaction coefficients into the opposite direction of the average of the consistency vectors $\mathbf{Z}^k = \mathbf{Z}^{i\beta r}$:

$$\mathbf{c} = -\frac{1}{\sum_k 1} \sum_k \mathbf{Z}^k. \tag{6.4}$$

For the generation of the basis functions, $K = 30$ is set and the clustering step is performed. A slight difference to section 5.4 is that the scaling factors a_1, a_2 and a_3 of the three proximity dimensions are not generated automatically, but set by hand to the values $a_1 = 1$, $a_2 = 5$ and $a_3 = 5$, such all three proximity functions are normalized to the same range of values.

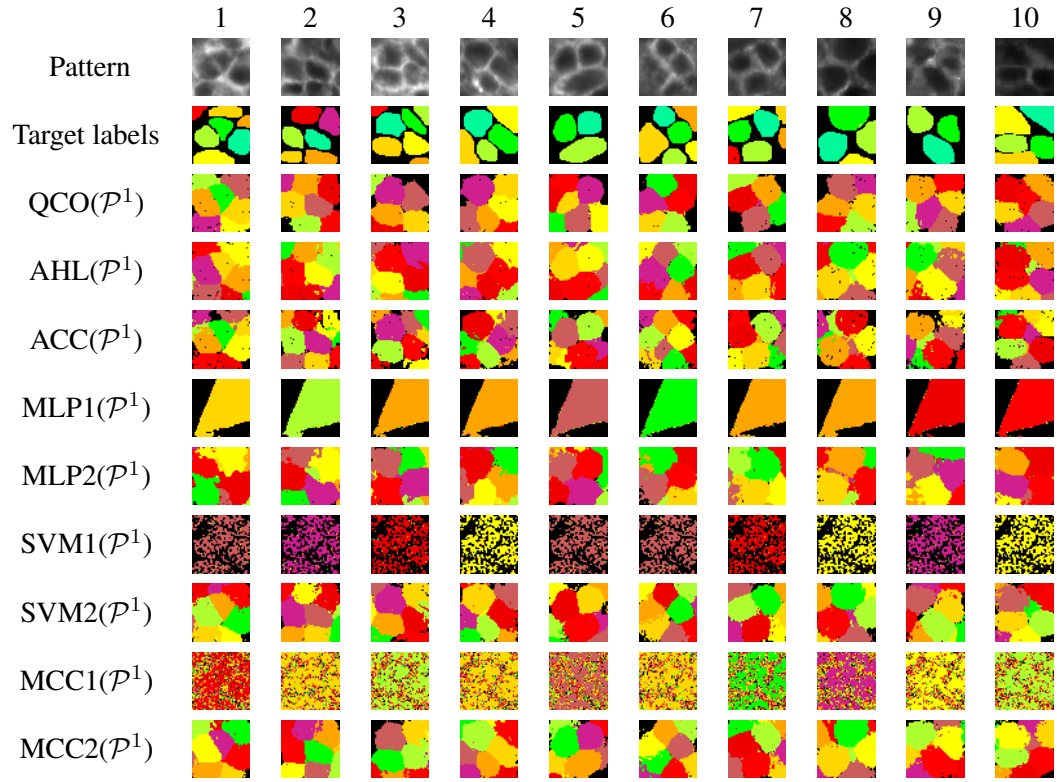


Figure 6.6: Comparison of segmentation results from training with different learning algorithms. The algorithms QCO, AHL, ACC, MLP, SVM and MCC are applied on pattern one. For MLP, SVM and MCC the feature-pairs are presented in variant 1 by a simple concatenation of the feature vectors and in variant 2 in the same way as for QCO, ACC and AHL by the proximity function in Fig. 6.5. Each row shows the output labeling on all ten test patterns resulting from the respective learning method.

The results of these three algorithms are compared to that of three standard classifiers: the Multi Layer Perceptron (MLP), the Support Vector Machine (SVM) and the Maximum Contrast Classifier (MCC) [32], which is a kernel based classifier similar to the SVM. It models the conditional density of each class by a weighted sum of Gaussian kernel functions centered at given training points. Therefore, the weights of the kernel functions are optimized under additional constraints that maximize the contrast, which is the difference in the answers of the class conditional densities.

Each classifier is trained with 1000 positive feature pairs (target value 1) and 1000 negative feature pairs (target value -1). Thereby, the feature-pairs are presented in two variants. In the first variant, the feature vectors are simply concatenated, such that the classifier has to adapt suitable proximity functions on its own. In the

second variant, the feature pairs are presented, like in the case of QCO, ACC and AHL, by the proximity vectors $\mathbf{d}_{rr'}$.
The parameters of the MLP, SVM and MCC were explored by hand. These are for the MLP the architecture, a 6-6-6-1 network of Fermi neurons, where the output neuron has a linear threshold function, and the learning step size $\epsilon = 0.01$ for the standard on-line back propagation learning algorithm. For the SVM, the parameters are the width of Gaussian kernel functions $\sigma = 2$, a uniform margin variable $\epsilon = 0.01$ for all constraints and $C = 30$, a scaling weight that punishes the violation of the constraints by slack variables. The MCC uses similar parameters as the SVM. These are also the width of Gaussian kernel functions $\sigma = 2$, $\lambda_{MCC} = 0.2$ the maximal contrast between the two class densities and $\delta = 0.2$, a scaling weight between the optimization term and the constraints.
The interaction functions resulting from the training of each classifier are tested within an $L = 10$-layered (one background layer with $m = 0$ and nine figure layers) on all ten patterns. This time, the CLM-dynamics is simulated without annealing ($T = 0$), until it reaches an attractor state. The output labeling of the attractor state is compared with the target labeling of the test pattern and displayed statistically in Fig. 6.7. For the training of each learning algorithms, the average, standard deviation, minimal and maximal quality, achieved on the ten test patterns, is displayed. To make these statistics more descriptive, Fig. 6.6 shows explicitly the labeling results that are achieved on all ten test patterns after training the different learning methods with pattern 1.
In the quality plots, it can be seen, that there is a relative high deviation in the quality results from training with one of the images and testing on all ten images. Further, the maximal quality reached lies only around 80 $\%$ of the images. Before the results of the different algorithms are compared, these observations shall be discussed.
First, the self-interaction strength $m = 0$ is too weak to assign smaller regions to the background. Consequently, the background is split into several subgroups, which reduces the quality. Second, the target labeling is controversial. Even for a human expert it is not sure, how many cells should be actually marked in the target labeling, such that for each region it has to be decided individually whether it should be assigned to the background or not. Further, the exact contour of the cell borderline may be controversial between different human observers [37], where already a pixel wide difference in two labelings results in a noticeable difference according to our quality measure. Consequently, there is some kind of uncertainty in the target labeling, such that it is improbable that two labelings agree at 100%.
As third point, it should be kept in mind that there is some variation along the example patterns according to the size of the cells, i.e. pattern 1 and 2 show relative small cells, while pattern 5 and 8 show relative large cells. If an adaptation of the learning methods to the cell size is assumed, an over segmentation of large cells, like it can be observed in Fig. 6.6 for patterns 5, 6, 8 and 9 after training with small cells, and a merging of small cells after training with large cells, can be expected. This behavior explains the high deviation in the quality plots.

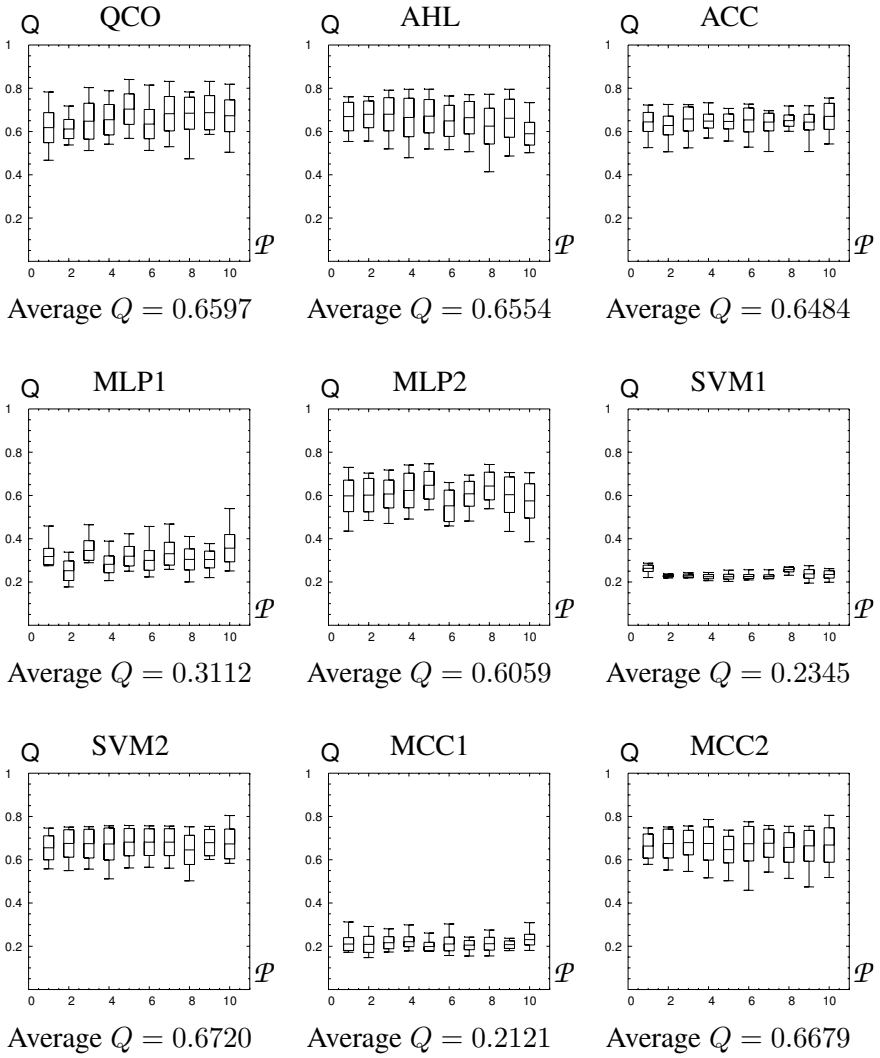


Figure 6.7: Comparison of the statistical grouping quality Q achieved on all ten patterns after training the different learning methods on one pattern $\mathcal{P}^i$.

Besides these three points, the following results can be observed in the experiment. The best grouping performance is achieved by training with the algorithms SVM and MCC in variant two, where the feature pairs are represented in form of pre-processed proximity vectors. In contrast to this, the presentation of the feature pairs by concatenation of the feature vectors leads to a total failure in the grouping performance for MLP, SVM and MCC.

This failure can be expected from the fact, that only a single pattern is presented for learning. The interaction function shall learn, that features from the same cell interact positively, while features from different cells interact negatively, or more formally:

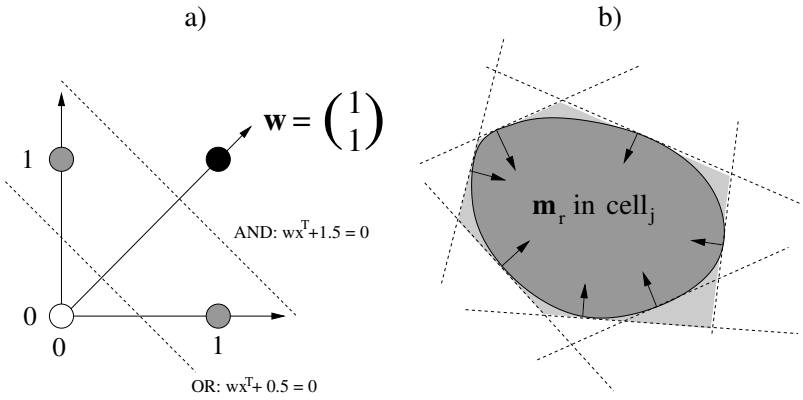


Figure 6.8: Construction of a trivial solution for the interaction function in equation (6.5) that overfits to the cell positions in a single training pattern. a) Logical And- and Or-operators can be implemented by simple linear perceptrons. b) The membership of a pixel to a cell region can be approximated by an And-combination of linear discriminators that work on pixel positions and are tangential to the cell borderline.

$$f_{rr'} = \begin{cases} 1 & : \bigvee_j (\mathbf{m}_r \in \text{cell}_j \wedge \mathbf{m}_{r'} \in \text{cell}_j) \\ -1 & : \text{otherwise} \end{cases} . \tag{6.5}$$

The logical And- and Or-operations in (6.5) can be implemented by simple linear perceptrons (see Fig. 6.8), while the membership $\mathbf{m}_r \in \text{cell}_j$ of a feature $\mathbf{m}_r$ belonging to a cell region can be discriminated on the pixel positions in the input pattern (see Fig. 6.8). Thus there exists a solution for the interaction function that perfectly fits to the pixel positions in the target labeling of the training pattern. However, this solution cannot generalize to other patterns, since it ignores the orientation information of the features. It would return for all patterns the same interaction matrix and output grouping, since the pixel positions in all patterns are the same.

The results in Fig. 6.6 show, that the MLP, SVM and MCC were not even able to find this overfitting solution, not to mention a more general solution, indicating, that it is a non trivial problem to choose a suitable training set and learning parameters without external knowledge. The choice of problem specific proximity functions constraints the degrees of freedom in the classifier function and enforces the generalization of the learned grouping principle from a single training pattern. On the other side, it simplifies the learning problem, such that the simple quantization approach of the algorithms AHL, QCO and ACC becomes competitive to the standard classifiers.

AHL and QCO perform slightly worse than SVM and MCC in variant two. The algorithm ACC shows, again, a small loss of quality compared to AHL and QCO and the application of MLP in variant two shows a further significant decrease in

grouping quality, but the resulting output labeling still shows some similarity to the target labeling.
The experiment shows, that the learning problem can be solved in principle by several binary classifiers. This result provokes the question: Why should the learning process not be implemented by the SVM, since it shows a better grouping performance than AHL. For the present choice of learning parameters of the different learning methods, the answer to this question lies in the application speed of the learned interaction functions. For the proximity functions in Fig. 6.5, the classes of feature pairs with positive and negative target interactions have a high overlap. At the same time, the parameters of the SVM define relative short-ranged kernel functions and relative hard constraints (high value of C). For this reason, the SVM takes more than the half of the 2000 training examples as support vectors (the same argumentation holds for MCC). This means, that for each application of the interaction function the kernel function has to be evaluated at more than 1000 support vectors, which is a relatively high computational effort compared to the next neighbor criterion according to the $K = 30$ proximity prototypes of AHL, QCO and ACC. Since the interaction function has to be evaluated a quadratic number of times according to the number of features in a pattern, which in this sample application is around $2025^2 = 4100625$ times, the higher application time relativizes the small gain in grouping quality.
Surely, there exist methods to reduce the number of support vectors either by sophisticated methods of parameter tuning, like in [5], and even the classification rate of the MLP can be enhanced by well known techniques, like weight decay or an learning rule with an adaptive size of learning steps, e.g. RPROP etc. [2],[15]. However, this means, that the problem of parameter tuning has to be solved during the learning process, which can be for problems with high overlap between positive and negative interaction a highly non trivial task. In contrast to this, the number of basis functions K can be directly controlled for AHL, ACC and QCO.
For small values of K, AHL, ACC and QCO show a slightly worse binary classification rate than SVM and MCC, but as it was observed in chapter 4, the high redundancy within the CLM can compensate this, such that a relative weak, but efficient classifier, like AHL, shall be preferred against a stronger, but also more (computational) demanding algorithm, like SVM or MCC.
From the comparison of AHL, ACC and QCO, AHL is preferred, because it has the simplest and fastest learning algorithm of these three approaches, since it prevents the explicit computation of all consistency conditions (3.39). Further, the classification manifold of AHL can be shifted by modification of Λ towards higher attraction or higher repulsion between the features without repeating the learning process, as is shown in the next experiment.

6.3.2 Influence of the Control Parameter Λ

The last experiment stated, that in this work the AHL algorithm is preferred against other binary classifiers, even if it's grouping performance is slightly worse than

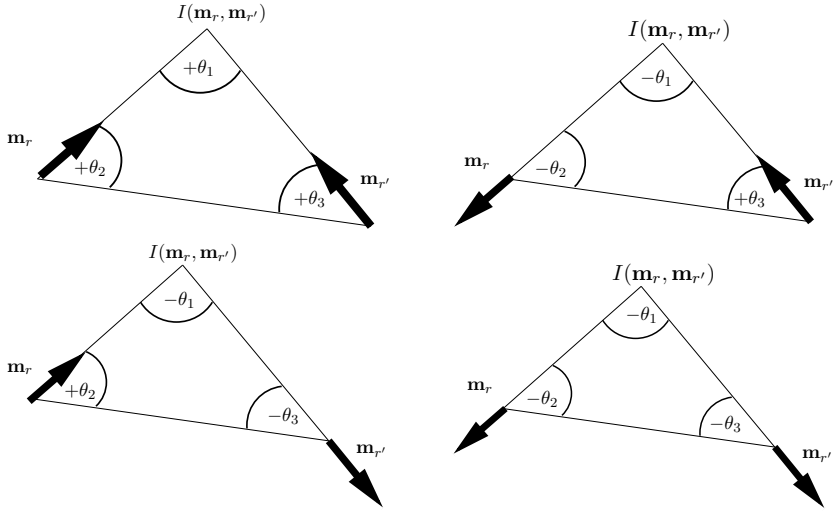


Figure 6.9: Proximity functions for directed edge features: The proximity functions for oriented line segments have to be enhanced by additional case differentiations for the directions of the features. The angles θ_1, θ_2, and θ_3 are counted positive, if both $\mathbf{m}_r$ and $\mathbf{m}_{r'}$ point towards their intersection point $I(\mathbf{m}_r, \mathbf{m}_{r'})$. They are counted negative, if both feature point away from $I(\mathbf{m}_r, \mathbf{m}_{r'})$. In the case, that only one feature points towards $I(\mathbf{m}_r, \mathbf{m}_{r'})$, the angle at this feature is counted positive, while the others are counted negative. To preserve symmetry under feature exchange θ_2 and θ_3 are swapped, if $\theta_3 > \theta_2$.

other approaches, like SVM, because of it's fast learning and application rate. This experiment highlights the effect of the control parameter Λ on the grouping result of the cell segmentation problem .

In contrast to the last experiment, an alternative feature representation and definition of the proximity space is used. The features in the input images $\mathbf{m}_r = (\mathbf{p}_r, \mathbf{o}_r)$ are encoded by their position $\mathbf{p}_r$ and orientation vector $\mathbf{o}_r = (S_r^x, S_r^y)$, which is given by the Sobel-x and Sobel-y response of position $\mathbf{p}_r$, such that this approach can also deal with pixels with intensity gradient magnitude equal to zero.

The proximity functions are an enhancement of an approach derived for oriented line segments that will be discussed on another application later in this chapter (see section 6.5). Two features $\mathbf{m}_r$ and $\mathbf{m}_{r'}$ define a triangle (see Fig. 6.9) with vertices at the two feature positions $\mathbf{p}_r$ and $\mathbf{p}_{r'}$ and the intersection point $\mathbf{I}(\mathbf{m}_r, \mathbf{m}_{r'})$ of the two features. This triangle is described by the local distance between the features and the three angles within the triangle. Further, configurations shall be distinguished, where the intensity gradients point towards each other or away from each other. Therefore the angles θ_1, θ_2 and θ_3 are counted positive or negative according to whether $\mathbf{m}_r$ and $\mathbf{m}_{r'}$ point towards the intersection point $\mathbf{I}(\mathbf{m}_r, \mathbf{m}_{r'})$ or not, like it is shown in the four configurations in Fig. 6.9. As last property of the proxim-

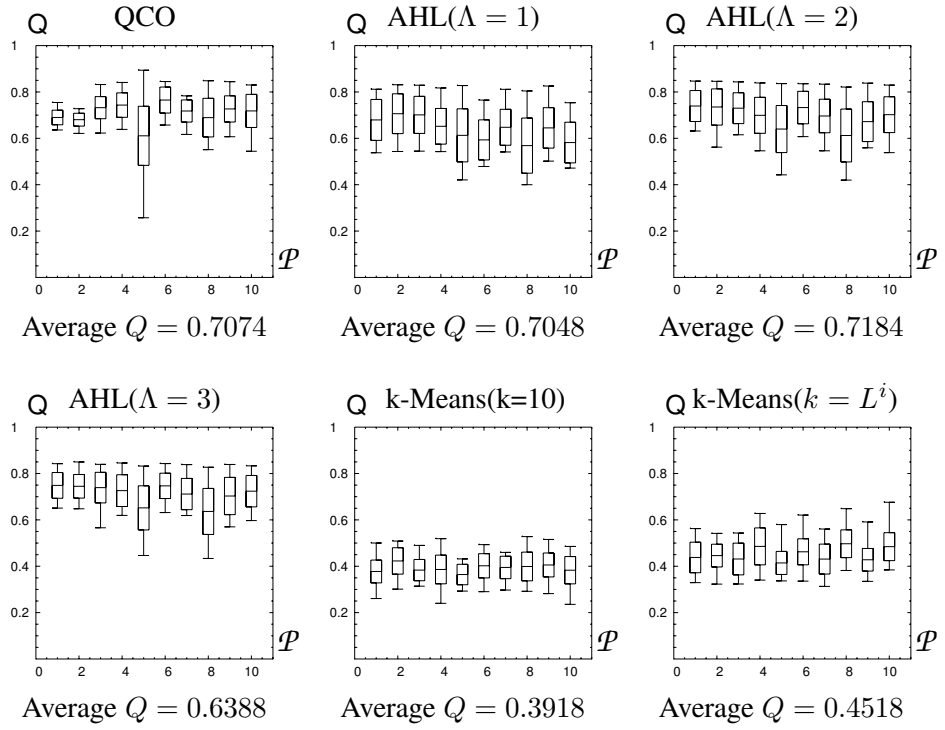


Figure 6.10: Comparison of the statistical grouping quality for different values of the control parameter Λ of AHL.

ity functions the symmetry under feature exchange must be ensured. Therefore, the components θ_2 and θ_3 are swapped in the four dimensional proximity space, if $\theta_3 > \theta_2$.

Learning is performed by QCO ($\kappa = 100$) and AHL, where K is set to 100 and 10000 learning steps are performed per learning phase. After learning with AHL, Λ is set to the values $\Lambda = 1$, 2 and 3 and the grouping performance is compared to QCO and a naive approach of directly clustering the feature vectors $\mathbf{m}_r$.

The interaction functions resulting from learning with QCO and AHL are applied within an $L = 20$-layered (19 figure layers, one background layer) CLM and the self-interaction strength m is estimated, like it is sketched in section 5.7. Annealing is performed in 1000 steps from $T = 1000$ to $T = 0$. $L \cdot N = 20 \cdot 2025 = 40500$ neurons are updated in each annealing step. The results are compared to a naive application of the k-means algorithm onto the feature vectors $\mathbf{m}_r$. The feature vectors $\mathbf{m}_r$ in a training pattern are reduced by k-means algorithm in variant 1 to $k = 10$ prototypes and in variant 2 to $k = L^i$ prototypes, where L^i is the number of labels in the target labeling of the training pattern. After that, all patterns are segmented by mapping each feature vector onto the index of it's nearest feature prototype.

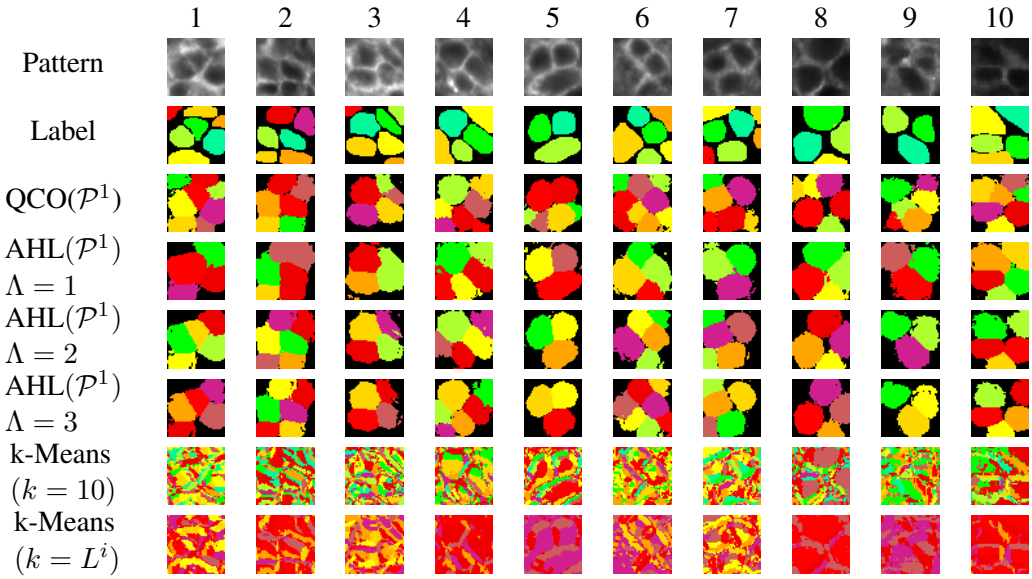


Figure 6.11: Comparison of segmentation results for different values for the control parameter Λ of AHL.

The statistical grouping qualities that arise from testing on all ten input patterns after training with one of them, are displayed for the three approaches QCO, AHL, and k-means in Fig. 6.10. The exemplary grouping results on all ten input patterns, after training with pattern 1, are shown in Fig. 6.11.

It can be seen, that the naive approach of k-means clustering fails in separating the cells from the background, which is indicated by an average quality of 39 respectively 45 %. The application of the QCO trained interaction function shows with an average grouping quality around 70 % an adequate reproduction of the grouping behavior. However, it shows in Fig. 6.11 the tendency of oversegmentation, which might be explained by the fact that the training pattern 1 shows relative small cells. The application of AHL with $\Lambda = 1$ shows slightly worse average grouping quality results than QCO. In Fig. 6.11, the main errors in the output segmentation arise from merging smaller cells in the patterns 1, 2 and 3, which indicates that the influence of positive interactions is too strong against the negative ones. This effect can be compensated by increasing Λ, such that the influence of the negative interactions is increased. The average grouping quality is improved for AHL with $\Lambda = 2$. It can be seen in Fig. 6.11, that smaller cells are no longer merged and that the areas of salient cells become smaller. If Λ is increased to $\Lambda = 3$, the average grouping quality decreases again. The size of the salient cell regions in Fig. 6.11 is reduced and, in the case of pattern 6, over segmentation starts. It can be concluded, that a further increase of Λ enforces over segmentation, such that the average grouping quality decreases further.

This experiment has shown, that, even if the interaction function adapts to a dominant cell size from the training pattern, the parameter Λ can be used to adjust the interaction function to cell sizes in new patterns. It showed, that there are several ways of feature representations and corresponding proximity functions that are sufficient to construct an interaction function to solve the cell segmentation problem. In the next experiment, it is tried to evaluate, how suitable a set of given proximity functions is for the adaptation of a target labeling. Further, it will inspect the effect of spurious proximity functions on the grouping performance of the learned interaction function.

6.3.3 Comparison of Proximity Functions

In the last two experiments, two different proximity functions were used to solve the cell segmentation task. Both of them were suitable to solve the problem. This implies the question: Can different sets of proximity functions be compared to judge whether one is more appropriate for a problem than the other? And what happens, if the set of proximity functions is extended by spurious proximity functions?
In the following, the feature representation and proximity functions from the binary classifier experiment are called "Proximity 1"and the feature representation and proximity function from the experiment showing the influence of Λ are called "Proximity 2". "Proximity 3" is defined on the same features $\mathbf{m}_r = (\mathbf{p}_r, \mathbf{o}_r)$ as in the last experiment by the local distance

$$d_1(\mathbf{m}_r, \mathbf{m}_{r'}) = \| \mathbf{p}_r - \mathbf{p}_{r'} \|, \tag{6.6}$$

the orientation distance

$$d_2(\mathbf{m}_r, \mathbf{m}_{r'}) = \frac{\mathbf{o}_r^T}{\| \mathbf{o}_r \|} \frac{\mathbf{o}_{r'}}{\| \mathbf{o}_{r'} \|}, \tag{6.7}$$

and a third function that indicates with high negative values, that both features point away from each other, and with high positive values, that they point towards each other, while small positive and negative values indicate, that both features point into the same direction:

$$d_3(\mathbf{m}_r, \mathbf{m}_{r'}) = \frac{\mathbf{o}_r^T}{\| \mathbf{o}_r \|} \frac{\frac{1}{2}(\mathbf{p}_r + \mathbf{p}_{r'}) - \mathbf{p}_r}{\| \frac{1}{2}(\mathbf{p}_r + \mathbf{p}_{r'}) - \mathbf{p}_r \|} + \frac{\mathbf{o}_{r'}^T}{\| \mathbf{o}_{r'} \|} \frac{\frac{1}{2}(\mathbf{p}_r + \mathbf{p}_{r'}) - \mathbf{p}_{r'}}{\| \frac{1}{2}(\mathbf{p}_r + \mathbf{p}_{r'}) - \mathbf{p}_{r'} \|}. \tag{6.8}$$

"Proximity 4" is given by a spurious set of proximity functions, which is based on the principle of similarity in color, where each feature $\mathbf{m}_r = (\mathbf{p}_r, I_r)$ is presented by it's position and intensity value in the input image. Proximity 4 is expressed by local distance

$$d_1(\mathbf{m}_r, \mathbf{m}_{r'}) = \| \mathbf{p}_r - \mathbf{p}_{r'} \|, \tag{6.9}$$

and distance in intensity

$$d_2(\mathbf{m}_r, \mathbf{m}_{r'}) = |I_r - I_{r'}|. \tag{6.10}$$

This proximity set is spurious, because it ignores the relevant information of the intensity gradient directions and only uses the intensity information, which is only capable to segment dark cell centers from the light cell corona, but not to find the exact borders of each cell.
Additionally, Proximity 1 and 4 are combined to "Proximity 5", Proximity 2 and 4 are combined to "Proximity 6", and Proximity 3 and 4 are combined to "Proximity 7" by appending the intensity distance (6.10) to the proximity vectors of Proximity 1, 2 and 3.
For each of the seven proximity functions, AHL is applied with $K = 30$, $\Lambda = 2$ by performing 10000 learning steps per learning phase. The resulting interaction functions are tested in an $L = 20$-layered (19 figure layers and one background layer) CLM. The self-interaction strength m is estimated deviant from section 5.7. The upper and lower bounds m_{low} and m_{up} are estimated by taking the sum of a certain number n_m of randomly selected feature pairs, where both features are labeled as background for m_{low} and both features are labeled to the same figure group for m_{up}. Then m is set heuristically, like in section 5.7, to

$$m = (m_{low} + 3m_{up})/4. \tag{6.11}$$

n_m can be interpreted as the expected group size, such that m_{low} represents the expected support in the background and m_{up} represents the expected support in a figure layer. n_m was set to 100, which seems to be too small, as can be seen in the segmentation results in Fig. 6.13, because the self-interaction is too weak to assign the smaller regions in the pattern to the background.
The annealing process is performed in 1000 steps of uniform step size from $T = \lambda_{max}(F)$ (maximal eigenvalue of the lateral interaction matrix) to $T = 0$. 20000 neuron updates are performed in each annealing step .
The quality plots in Fig. 6.12 show similar results for the application of Proximity 1, 2 and 3 with an average grouping quality around 70 $\%$. Also, the exemplary grouping results from training with pattern 1 in Fig. 6.13 show almost now difference between the output labelings. The only striking point is the tendency of Proximity 3 to over-segment the patterns 5, 8, 9 and 10, which indicates that Proximity 3 shows a higher adaption rate to the typical cell size in the training pattern 1. But as was shown in the last experiment, this cell size can be influenced by the adjustment of Λ.
For Proximity 4, which is assumed to be spurious to the target labeling, a clear decrease in group quality is observable compared to Proximity 1, 2 and 3. This impression is confirmed in the exemplary output labelings of Fig. 6.13, where the output labelings show no similarity to the target labeling for most of the test patterns.
When the spurious Proximity 4 is combined with Proximity 1, 2 and 3 to construct Proximity 5, 6 and 7, the average grouping quality rises again and comes close to the results of Proximity 1, 2 and 3. It can be seen in the exemplary grouping results in Fig. 6.13, that the original behavior can be reconstructed, besides the tendency of merging cells in pattern 1, 2 and 3, where Λ can be adjusted to reduce this effect.

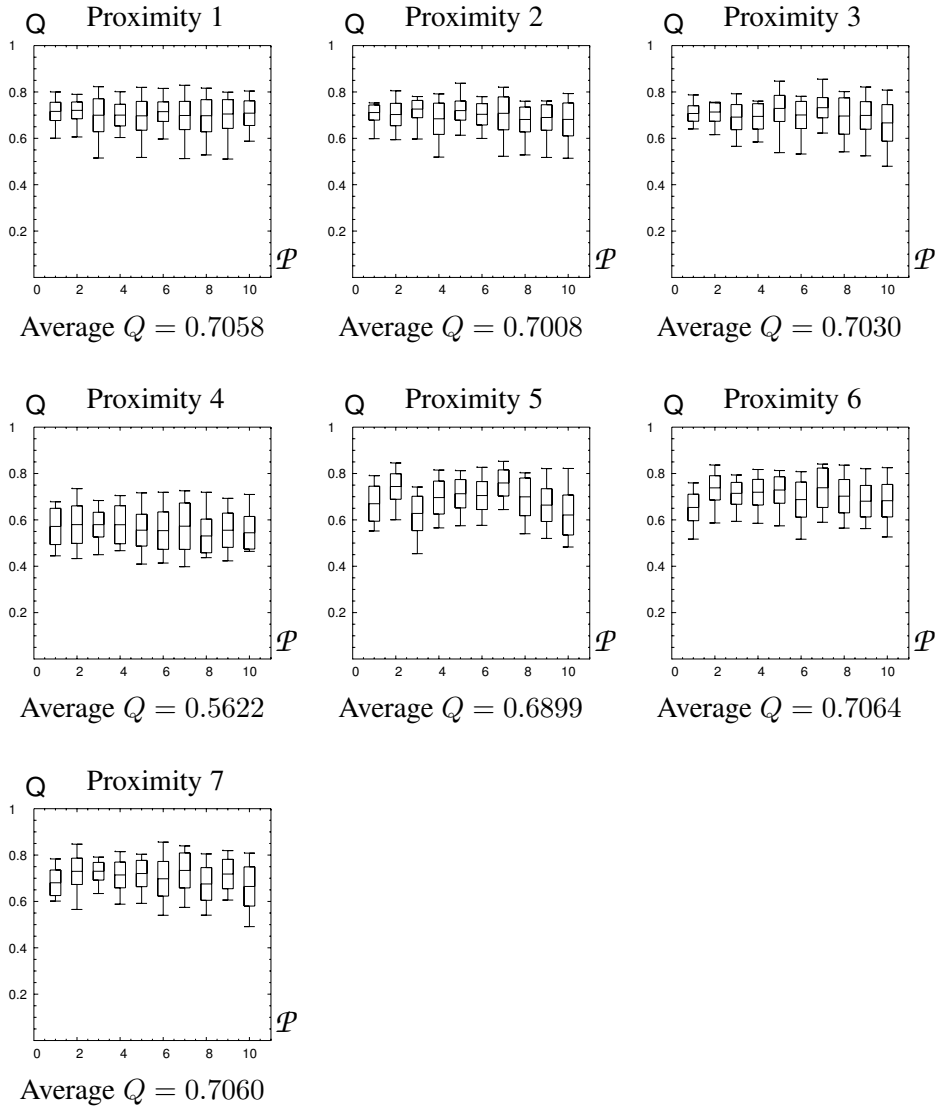


Figure 6.12: Comparison of the statistical grouping quality for the application of AHL with different proximity functions.

It can be concluded, that the grouping performance of Proximity 1, 2, 3, 5, 6 and 7 are hardly to distinguish. Only Proximity 4 fails in solving the grouping problem. The AHL learning algorithm showed the ability to select the relevant proximity criteria to reconstruct the target labeling. But the question is: Are there other arguments to prefer one of the proximity spaces?

One indicator might be the overlap between positive and negative interaction, described by the bounds Λ_{max} and Λ_{min} (see section 5.5), because a small overlap between $\mathbf{c}^+$ and $\mathbf{c}^-$ results in higher absolute values of the interaction coefficients c_j, which define the block-diagonal structure of the interaction matrix F. As chap-

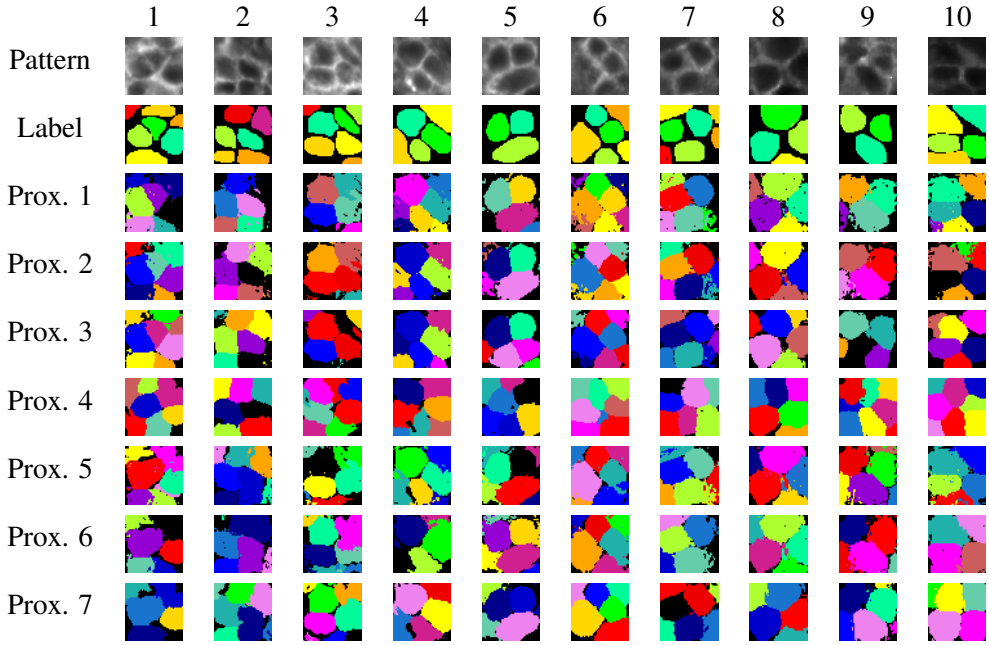


Figure 6.13: Comparison of segmentation results from the application of AHL with different proximity functions.

ter 4 has shown, the strength of the block-diagonal structure has influence on the eigenvalues of F and defines the speed of convergence of the CLM dynamics.
Another criterion of the proximity functions is the application time of the learned interaction function, which on the one-hand side depends on the complexity of the proximity functions, such that spurious or useless proximity functions should be preferably removed, and on the other hand depends on the number of used basis functions K.
To investigate these two criteria pattern, seven is chosen as fixed training pattern for all proximity functions, AHL is applied for all values of $K = 1$ to $K = 200$ (with $K \cdot 100$ learning steps per learning phase) and the graphs of Λ_{max} and Λ_{min} that result from these training runs are plotted in Fig. 6.14, where the red lines show the upper bounds Λ_{max} and the blue lines show the lower bounds Λ_{min}.
In all plots a relatively high fluctuation of Λ_{max} against K can be observed, which mirrors the stochastic components of the AHL learning approach, where different samples of feature pairs in the clustering and interaction sampling phase can result in different vectors $\mathbf{c}^+$ and $\mathbf{c}^-$ with a varying overlap from run to run. However, in average the overlap between $\mathbf{c}^+$ and $\mathbf{c}^-$ decreases with the number of basis functions K, which means that the classification rate of the learned interaction function increases.
For the spurious Proximity 4, the graph course of Λ_{max} converges to the relative poor upper bound $\Lambda_{max} \approx 6$, while for the other proximity functions it increases further. No significant differences are observable between the graphs of Proximity

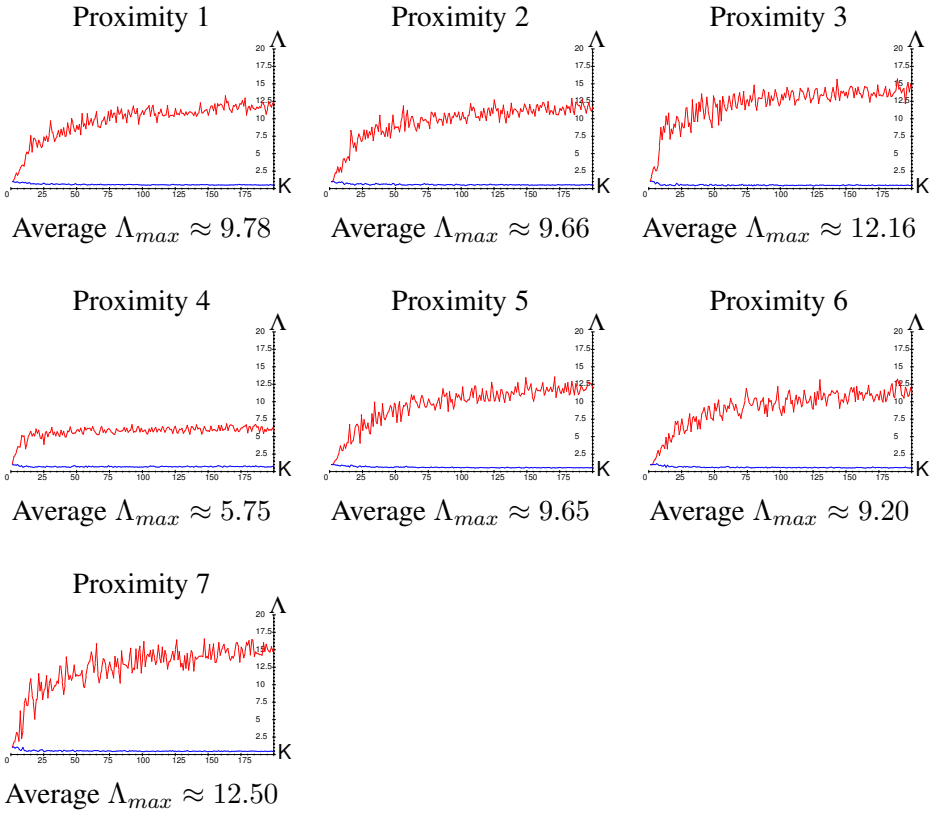


Figure 6.14: Plot of the bounds Λ_{min} (blue) and Λ_{max} (red) against the increase of the number of basis functions K for different proximity functions.

1 and 2. Both runs end up with a maximal value of Λ_{max} around 11 to 12, such that it is hard to say, which proximity should be preferred. However, if these proximity functions are compared to Proximity 3, it becomes clear, that Proximity 3 ends up in a higher maximal value of Λ_{max} around 13 to 15. Additionally, the value of Λ_{max} increases faster for small values of K, such that the interaction function of Proximity 3 needs less basis functions to reach the same classification rate than Proximity 1 and 2.

If the proximity functions 1, 2 and 3 are compared to their aggregations with Proximity 4, no differences in the graphs of Λ_{max} are visible. Consequently, the simpler variants should be preferred, since the distance in intensity values brings no increase in the classification rate.

The result of the above analysis is, that Proximity 3 gives the best results in terms of grouping quality and convergence speed of the CLM-dynamics at a predefined evaluation time of the interaction function. However, this benchmark has to be regarded with caution, since in the sample results in Fig. 6.13 have also shown, that Proximity 3 shows a higher adaption to the cell size in the training pattern than

the other proximity sets. The overlap between $\mathbf{c}^+$ and $\mathbf{c}^-$ gives only information about the consistency of the training pattern with the target labeling, but gives no information about the generalization properties of the learned interaction function. Therefore, it is very important to choose a representative training pattern and to give a suitable target-labeling.
In the last experiment on the cell segmentation problem, it is investigated, how different ways of labeling influence the learning process.

6.3.4 Influence of the Target Labeling

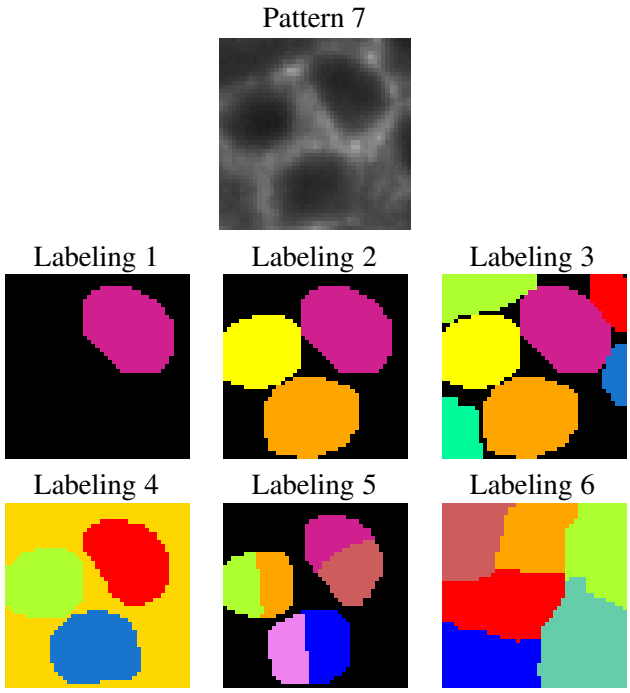


Figure 6.15: Six different ways to label pattern 7. Black labeled features are marked as background, while the other labels describe figure groups.

It was stated before, that the correctness of the target labeling of the cell images is controversial. Figure 6.15 shows six different ways of labeling pattern seven.
In Labeling 1, only a single cell is marked as salient group, while the rest of the image is assigned to the background. This kind of labeling is conformable for a human teacher, since he can choose the most salient cell region and must not consider whether the rest of the image shows a cell or not. But this labeling might be confusing for the learning algorithm. Since there exists only one figure label, the only source for negative interaction are feature pairs from the background. Consequently, feature pairs from the unmarked cell regions are treated as negative examples instead of positive ones.

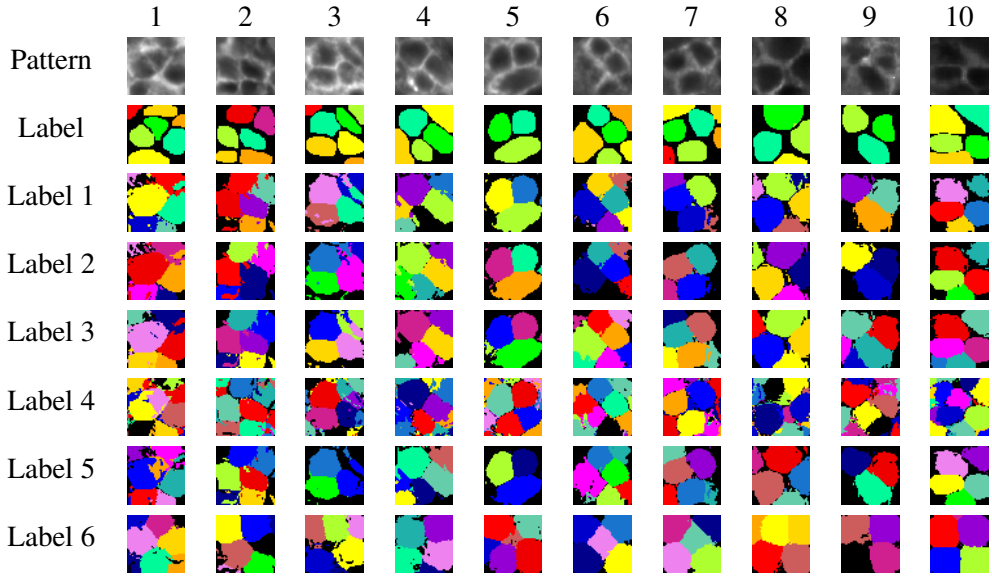


Figure 6.16: Comparison of segmentation results from the application of AHL on pattern seven with the different target labelings in Fig. 6.15.

In Labeling 2, this problem is avoided by marking the three most salient regions, which actually describe complete cells. And in Labeling 3, additionally the smaller regions at the border of the image, which might be parts of neighboring cells, are marked.

Labeling 4 is a spurious labeling similar to Labeling 2, but this time the background is marked as salient group, such that the learning algorithm might be confused by spurious positive interactions.

Labeling 5 shows another spurious variant of Labeling 2, where each of the three cell regions is divided into two labels, and, finally, Labeling 6 shows a total spurious labeling of pattern seven into six random, but coherent regions.

This experiment shall inspect, how the AHL algorithm can deal with these six different labelings of pattern seven. AHL is applied with $K = 30$ basis functions (10000 learning steps per learning phase) on each of the labelings, where Λ is set to $\Lambda = 2$. The resulting interaction functions are tested on an $L = 20$-layered (19 figure layers, one background layer) CLM, where the self-interaction strength m is estimated like in the last experiment. The annealing process is performed in 1000 steps of uniform step size from the maximal eigenvalue $T = \lambda_{max}\{F\}$ of the matrix of lateral interactions F to $T = 0$, and for 200 additional steps at $T = 0$. 20000 neurons are updated in each annealing step. The grouping results from testing the six interaction functions on all ten input patterns are shown in Fig. 6.16. The quality statistics of these results are plotted for each labeling in Fig. 6.17 and the bounds Λ_{max} and Λ_{min} are plotted in Fig. 6.18.

It can be seen in the quality statistics, that AHL is able reconstruct a suitable in-

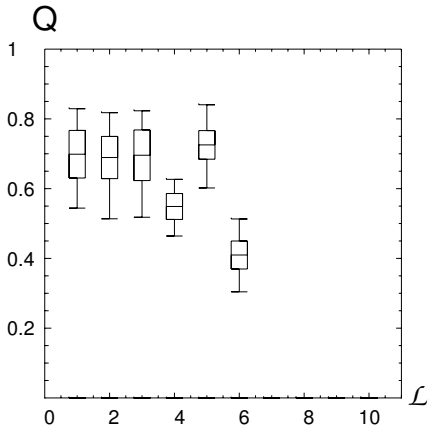


Figure 6.17: Quality results for different types of labeling.

teraction function from Labeling 1, 2, 3 and 5. Additionally, Fig. 6.18 shows, that these four labelings result in a relative high gap between Λ_{min} and Λ_{max}, which indicates a low overlap of positive and negative interactions within the basis functions. The value of Λ_{max} is reduced only for Labeling 1, since the unmarked cell regions cause additional overlap between positive and negative interactions. The grouping results in Fig. 6.16 show the correct grouping behavior, besides merging of small neighboring cells, which can be reduced by the adjustment of Λ.

Errors in the target labeling that can not be compensated are shown in Labeling 4 and 6. Features that form no group should not be labeled uniformly, but be assigned as unknown to the background, and an arbitrary labeling leads to no acceptable interaction function, which shows the importance of the target labeling respectively the knowledge of the human teacher for the learning process.

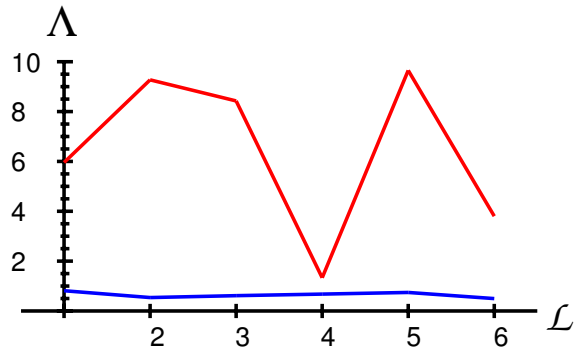


Figure 6.18: Λ-bounds Λ_{max} (red) and Λ_{min} (blue) resulting from training with the different labelings in Fig. 6.15.

It can be concluded, that the AHL learning algorithms does not need a perfect target labeling, which for some problems, like the cell segmentation problem, might not exist. A relative rough approximation of relevant parts of the salient groups is sufficient, such that a suitable statistics of positive and negative interactions can be achieved during the learning process.

6.3.5 Summary

The grouping behaviors realized by the hand-tuned, respectively learned, interaction functions reviewed in sections 3.6 and 3.7.4 from [34] and [61] can be reproduced with the AHL algorithm and in principle with several other binary classifiers. In this work, AHL is preferred over other approaches, because of it's simple and fast learning algorithm, the management of the evaluation speed of the interaction function with the number of basis function K and the control abilities of the segmentation level with the parameter Λ.
Further, it was shown that different possible ways of encoding proximity criteria between feature pairs can be compared by the observation of the classification rate of the AHL learned interaction function. Finally, the robustness of the AHL approach against uncertainties in the target labeling was shown.

6.4 Texture Segmentation

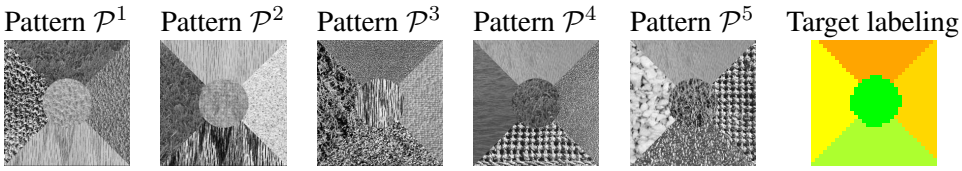


Figure 6.19: Dataset for texture segmentation.

As second application of the AHL algorithm on real world problems, it is tried to reproduce results Ontrup [40] on the problem of texture segmentation, achieved with the hand tuned interaction function reviewed in section 3.6.
Figure 6.19 shows five patterns that are applied as test data set for this problem domain. Each pattern shows five different texture regions constructed from the Bordatz album [1], such that the same unique target labeling can be used for all five patterns.
The feature extraction is performed according to Ontrup's approach by the following steps (compare section 3.6):

1. Computation of the even symmetric responses of a set of Gabor filters in m different scales and n different orientation.

2. Applying a non linearity in form of the hyperbolic tangents on the filter responses.

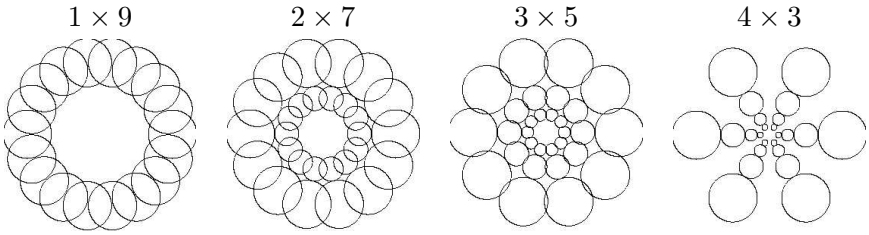


Figure 6.20: Four different sets of Gabor filters.

3. Computing the average and standard deviation for each Gabor filter response by convolution with a Gaussian filter. The widths of the Gaussian filter $\sigma_{Gaussian}(m,n)$ in frequency space that is used for the convolution with the filter at scale m and orientation n is taken by dividing the width of the Gabor filter $\sigma_{Gabor}(m,n)$ with a factor k:

$$\sigma_{Gaussian}(m,n) := \sigma_{Gabor}(m,n)/k. \tag{6.12}$$

4. The results are concatenated to the $2 \cdot m \cdot n$ dimensional texture vector $\mathbf{z}_r = (\nu_r, \sigma_r)$, where ν_r consists of the $m \cdot n$ mean values and σ_r of the $m \cdot n$ standard deviation values of the Gabor filter responses at position $\mathbf{p}_r = (x_r, y_r)$. A feature $\mathbf{m}_r = (\mathbf{p}_r, \mathbf{z}_r)$ is described by it's position and texture information.

The size of the input patterns is 256×256 pixels. If each pixel defines a feature, this would require $256^4 \approx 2.3 \cdot 10^9$ lateral interaction weights. Therefore, the features are subsampled by the factor 8 in each dimension of the images, resulting in $32 \times 32 = 1024$ features and $32^4 = 1048576$ lateral interaction weights per pattern. The feature extraction depends on the two parameters m and n for the Gabor filters and the scaling k of the width of the Gaussian filters against the width of the Gabor filters in frequency space. Four different sets of $m \times n$ Gabor filters, sketched in Fig.6.20: 1×9, 2×7, 3×5 and 4×3, and five scaling factors: $k \in \{2, 4, 8, 16, 32\}$ are tested as feature parameters.
The dimension reduction step, applied by Ontrup on the texture vectors $\mathbf{z}_r$, is omitted and in a first approach proximity is defined by the local distance

$$d_1(\mathbf{m}_r, \mathbf{m}_{r'}) = \| \mathbf{p}_r - \mathbf{p}_{r'} \| \tag{6.13}$$

and texture distance

$$d_2(\mathbf{m}_r, \mathbf{m}_{r'}) = \| \mathbf{z}_r - \mathbf{z}_{r'} \| . \tag{6.14}$$

Training is performed with $K = 100$ basis functions and 10000 learning steps per learning phase, Λ is set to 2.
The grouping process for the learned interaction functions is simulated within an $L = 10$-layered CLM, where annealing is performed in 1000 steps of uniform step

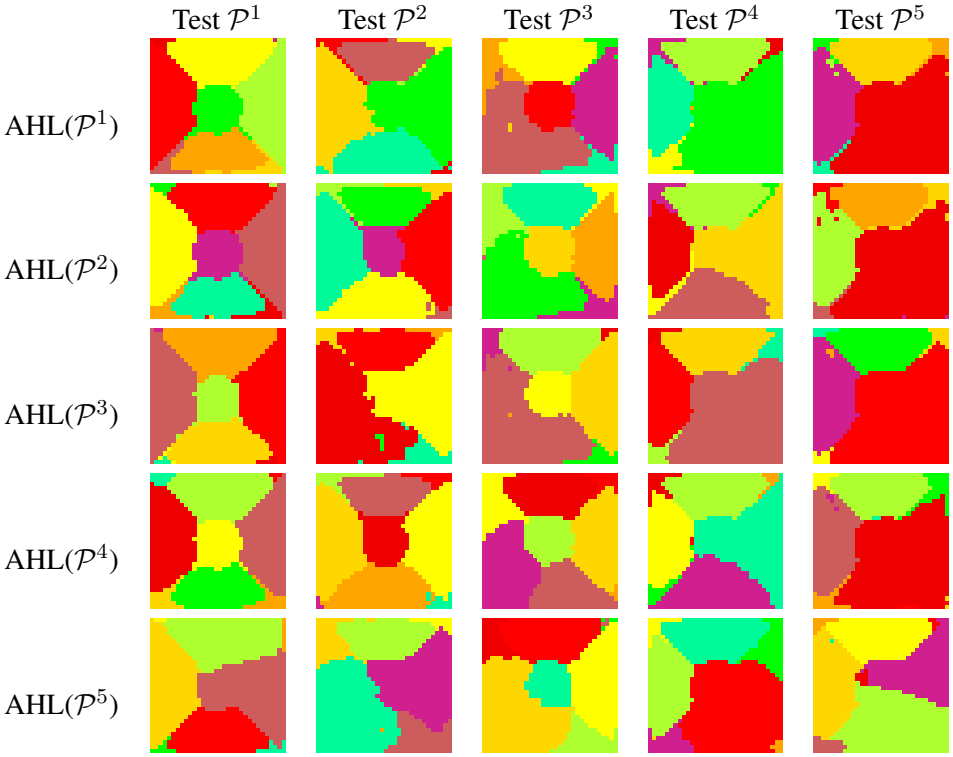


Figure 6.21: Segmentation results for 3x5 Gabor filters, k = 8 and a single texture proximity function. Each row shows the grouping results from an $L = 10$-layered CLM on the five test patterns after training is performed on one of them.

size from $T = 1000$ to $T = 0$, and each annealing steps consists of 10000 neuron updates.

Figure 6.21 shows results for the feature extraction parameters $m = 3$, $n = 5$ and $k = 8$. Each row shows the segmentation results on all five patterns, while training is performed only on one of these patterns. The quality values for these results are shown in Fig. 6.22, where each column of the plot corresponds to the quality values of a row in Fig. 6.21.

The segmentation results show differences in the grouping performances for the five patterns. The best result can be achieved for pattern 1. This result is independent of the training pattern (besides training with pattern five), where in each of these training runs a grouping quality around 90% is realized. For the other patterns, the grouping performance is worser, where mostly two (grouping quality around 80%) or three texture regions (grouping quality around 60%) are assigned to the same label. Partially, this result is plausible for a human observer, e.g. the left and bottom textures in pattern 3 look very similar, but the textures in pattern 2, 4 and 5 are clearly separable for the observer. This time, these grouping errors cannot easily be prevented by a simple adjustment of the parameter Λ, because at

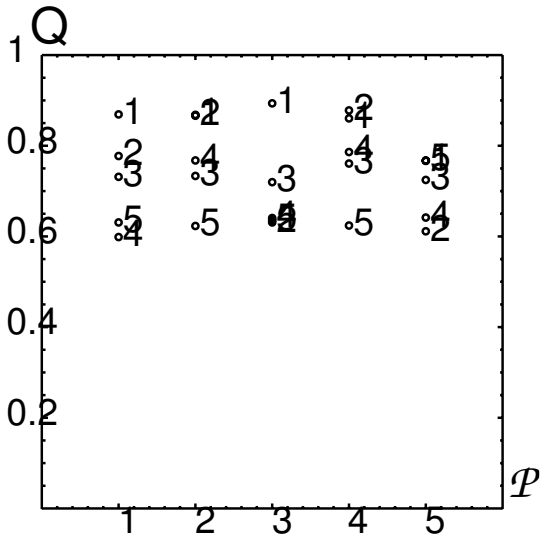


Figure 6.22: Qualities for the segmentation results in Fig. 6.21 based one a 3×5 Gabor jet and a smoothing factor of $k = 8$. The x-axis shows the number of the respective training pattern. The y-axis shows the achieved grouping quality on the numerated test patterns.

a higher segmentation level the new borders between the groups do not arise at the expected borders between the textures, but form spurious groups, like the results achieved by training with pattern five in Fig. 6.21. Thus the only way to enhance the grouping performance is to change the feature representation and the proximity functions.

To investigate, if the AHL algorithm is capable to reproduce the correct grouping of the different texture types, the learning success is measured by two values: the average grouping quality achieved one the test patterns, in the following called adaptation rate A:

$$A := \frac{1}{5} \sum_{i=1}^{5} Q(AHL(\mathcal{P}^i), \mathcal{P}^i), \tag{6.15}$$

and the average quality over all combinations of training and test patterns, in the following called generalization rate G:

$$G := \frac{1}{25} \sum_{i=1}^{5} \sum_{j=1}^{5} Q(AHL(\mathcal{P}^i), \mathcal{P}^j). \tag{6.16}$$

Figure 6.23 lists the observed adaptation and generalization rates for the explored parameters m, n and k. The best generalization rates around $G \approx 0.73$ and $G \approx 0.74$ are achieved for $k = 8$ and $k = 4$. An exact segmentation of pattern 3, 4, and 5 is not achieved, not even, if they are used as training pattern. The best adaptation

A G		$k = 32$	$k = 16$	$k = 8$	$k = 4$	$k = 2$
	Average	0.659 0.633	0.732 0.707	0.774 0.735	0.752 0.735	0.700 0.706
1×9	0.726 0.727	0.710 0.700	0.738 0.732	0.759 0.743	0.733 0.739	0.692 0.722
2×7	0.731 0.717	0.716 0.673	0.733 0.735	0.745 0.738	0.744 0.722	0.717 0.717
3×5	0.728 0.688	0.621 0.593	0.759 0.695	0.802 0.736	0.760 0.730	0.697 0.689
4×3	0.708 0.681	0.587 0.566	0.698 0.666	0.789 0.721	0.772 0.749	0.693 0.701

Figure 6.23: Table of the adaptation (red) and generalization (black) rate of different types of $m \times n$ Gabor jets and smoothing parameters k for a single texture proximity function. The shaded values are investigated in detail in Fig. 6.21 and 6.22.

A G		$k = 32$	$k = 16$	$k = 8$	$k = 4$	$k = 2$
	Average	0.829 0.711	0.853 0.716	0.825 0.693	0.777 0.662	0.686 0.626
1×9	0.773 0.700	0.829 0.728	0.855 0.731	0.775 0.704	0.735 0.683	0.670 0.656
2×7	0.800 0.686	0.844 0.715	0.867 0.721	0.837 0.694	0.756 0.656	0.696 0.646
3×5	0.800 0.661	0.810 0.685	0.846 0.691	0.867 0.698	0.806 0.647	0.664 0.586
4×3	0.804 0.677	0.831 0.715	0.845 0.719	0.822 0.675	0.811 0.663	0.712 0.615

Figure 6.24: Table of the adaptation (red) and generalization (black) rate of different types of $m \times n$ Gabor jets and smoothing parameters k for individual proximity functions for each texture dimension.

rate of $A = 0.802$ is achieved for the initial parameter setting of $(m, n, k) = (3, 5, 8)$. The only systematic results are, that segmentation gets worse, if the shape of the Gaussian filters that are used to compute the average and standard deviation values is too sharp ($k = 2$) or too wide ($k = 32$), and, that the segmentation of pattern 1 is more difficult for 3×5 and 4×3 Gabor filters than for 1×9 and 2×7. The last way to enhance the grouping performance is the modification of the proximity functions. Therefore, the single texture distance (6.14) is split into $2 \cdot m \cdot n$

A G		$k = 32$	$k = 16$	$k = 8$	$k = 4$	$k = 2$
	Average	0.843 0.730	0.853 0.723	0.848 0.709	0.796 0.694	0.718 0.662
1×9	0.793 0.723	0.835 0.756	0.822 0.723	0.807 0.710	0.775 0.716	0.728 0.709
2×7	0.804 0.705	0.861 0.748	0.837 0.738	0.850 0.720	0.769 0.676	0.703 0.643
3×5	0.813 0.683	0.837 0.694	0.856 0.705	0.876 0.703	0.805 0.686	0.690 0.629
4×3	0.836 0.702	0.840 0.721	0.898 0.724	0.857 0.702	0.835 0.696	0.750 0.667

Figure 6.25: Table of the adaptation (red) and generalization (black) rate of different types of $m \times n$ Gabor jets and smoothing parameter k for individual proximity function for each texture dimension plus an additional proximity function for the intensity distance. The shaded values are investigated in detail in Fig. 6.27 and 6.26.

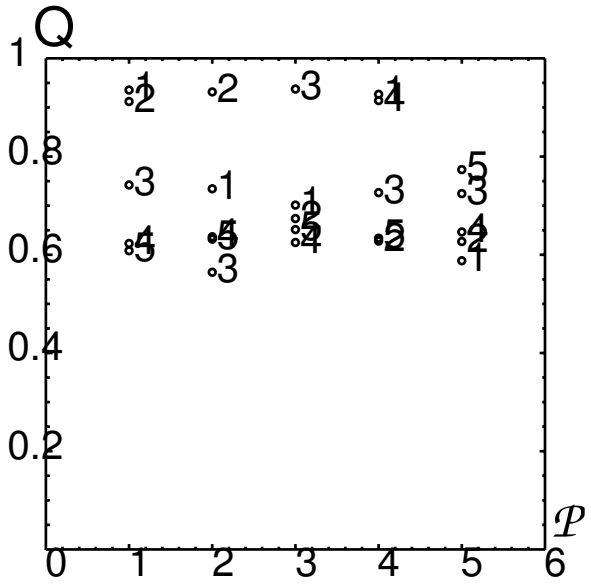


Figure 6.26: Qualities for the segmentation results in Fig. 6.27 based one a 4×3 Gabor jet and a smoothing factor of $k = 16$. The x-axis shows the number of the respective training pattern. The y-axis shows the achieved grouping quality on the numerated test patterns.

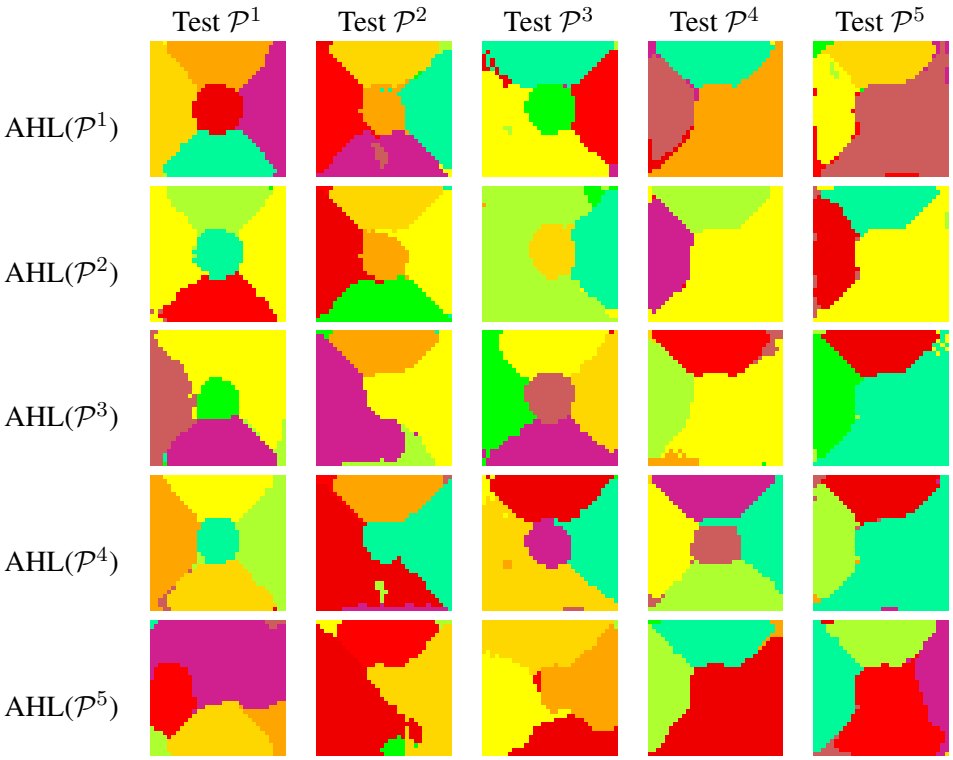


Figure 6.27: Segmentation results for 4x3 and k = 16 and individual texture proximity function for each texture dimension, plus an additional proximity function for the intensity distance.

independent proximity functions (one for each component of the texture vectors $\mathbf{z}_r$):

$$d_{1+l}(\mathbf{m}_r, \mathbf{m}_{r'}) = |(\mathbf{z}_r)_l - (\mathbf{z}_{r'})_l|, \quad l = 1, \ldots, 2 \cdot m \cdot n, \tag{6.17}$$

with the intention that the AHL learning algorithm selects the relevant proximity components to segment the patterns. Figure 6.24 lists the adaptation and generalization rates of the corresponding exploration of (m, n, k). The results show, that the split texture proximity can achieve a better adaptation (about 90%) for the pattern 1, 2, 3 and 4. For pattern 5, the best performance lies only around 80%, which means, that roughly one of the five texture regions is labeled incorrectly, such that the maximal adaptation rate lies at $A \approx 0.867$ for $(m, n, k) = (2, 7, 16)$ and $(3, 5, 8)$. However, there is lower generalization between the patterns indicated by a maximal value of $G \approx 0.731$ for $(m, n, k) = (1, 9, 16)$. The interaction function mostly adapts to the training pattern and shows high segmentation errors on the other patterns.

As further extention, the proximity set (6.13) and (6.17) is extended by additional color (intensity) information. Let I_r be the average intensity in 64×64 subblocks

of the original 256×256 images that corresponds the subsampled feature $\mathbf{m}_r$. Then the distance of average intensity

$$d_{2+2 \cdot m \cdot n}(\mathbf{m}_r, \mathbf{m}_{r'}) = |I_r - I_{r'}| \tag{6.18}$$

is appended to the set of proximity functions. The adaptation and generalization rate for this experiment are shown in Fig. 6.25. The gray shaded parameter set of $m = 4$, $n = 3$, and $k = 16$ is selected for detailed display of the results in Fig. 6.27. Compare them with those of Fig. 6.21.
The new grouping results show a higher adaption $A \approx 0.898$ to the respective training pattern, while the segmentation results of the other patterns, not applied for training, are equal (mainly for pattern 1) or worse ($G \approx 0.724$) than in the case, when the texture proximity is represented by a single function.
The example shows that the AHL algorithm can deal with (relative) high (around 30) dimensional proximity spaces. Further it once more stressed the importance of feature representation and proximity criteria. It has to considered, that it is difficult to predict the generalization abilities of the learned interaction functions.

	single texture proximity fct.	individual proximity for each texture dimension	individual texture proximity fcts. plus distance in intensity
avg. adaptation	0.723	0.794	0.815
avg. generalization	0.7032	0.681	0.703

Figure 6.28: Average adaptation and generalization rates for the three types of texture proximity functions.

6.5 Contour Grouping

As last example for the application of the AHL and CLM algorithm, artifically generated data sets from the problem domain of contour grouping are inspected. As stated in section 3.6, Wersing [61] specified a parameterized interaction function for the grouping of continuous and smooth contours in images. In contrast to this application, the following experiments try to adapt characteristic object specific shape parameters by constructing corresponding interaction functions. In the next chapter, this approach will be elaborated to a shape classifier by using several of such object specific interaction functions in mutual competition within the CLM.
The investigation starts on simple and highly regular polygons, like lines, triangles, squares and circles. The task is to combine local line segments to regular geometrical objects of predefined size and shape.
A polygon is described by the parameters $(x, y, R, \theta_{start}, S)$, where x, y, and R define the center and radius of a circle. Starting on the contour of this circle at the angle $\theta = \theta_{start}$, a set of surrounding lines of a regular geometrical object arises

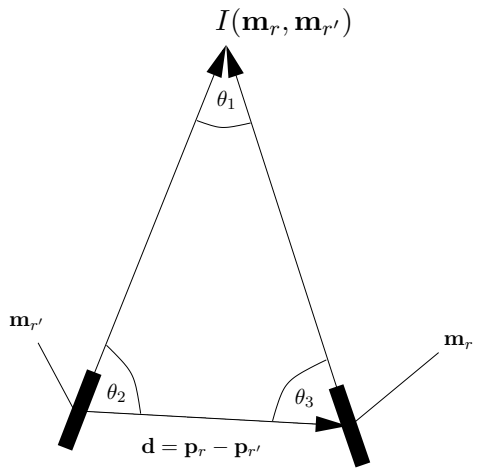


Figure 6.29: Proximity measures for oriented line segments: Two line segments $\mathbf{m}_r$ and $\mathbf{m}_{r'}$ define a triangle $(\mathbf{p}_r, \mathbf{p}_{r'}, I(\mathbf{m}_r, \mathbf{m}_{r'}))$, where $I(\mathbf{m}_r, \mathbf{m}_{r'})$ is the intersection point of the two line segments. The proximity of $\mathbf{m}_r$ and $\mathbf{m}_{r'}$ is expressed by the local distance $||\mathbf{d}||$ and the three angles within the triangle, where θ_2 and θ_3 are swapped, if $\theta_3 > \theta_2$ to ensure the symmetry of $\mathbf{d}_{rr'}$. So $\mathbf{d}_{rr'} = (a_1||\mathbf{d}||, a_2\theta_1, a_3\theta_2, a_4\theta_3)^T$. If $I(\mathbf{m}_r, \mathbf{m}_{r'})$ does not exist, because both features have parallel orientation, θ_1 is set to zero, describing an infinitely sharp angle. θ_2 and θ_3 describe the angles between the orientation of the two features and the vector $\mathbf{d}$ under the constraint $\theta_1 + \theta_2 + \theta_3 = \pi$.

from successively increasing θ with $\frac{2\pi}{S}$ and stepping to the next contour point of the circle at the angle θ. So S generates a polygon shaped as point, line, triangle, square, etc.

These lines are divided in into small line segments of equal length, where each oriented segment defines a feature vector $\mathbf{m}_r = (\mathbf{p}_r, \varphi_r)^T$ by it's position $\mathbf{p}_r = (x_r, y_r)$ and orientation $\varphi_r \in [0, \pi]$. A pattern is given by one to five objects of the same shape S and similar size R, where the remaining object parameters are chosen randomly as $x, y \in [R, 3R]$ and $\theta_{start} \in [0, 2\pi]$.

The CLM has to learn to segment patterns which contain objects of the same shape S and size R into the constituting polygons. It has to adapt the compatibilities for typical angles and distances within the observed polygons. To model this task with basis functions, it is assumed, that each two features define a triangle (see Fig. 6.29). The proximity between two features is described by their local distance and the three angles within the triangle, which defines a four-dimensional proximity space $\mathcal{D}$.

The proximity vectors $d_{rr'}$ occurring in one typical training pattern are clustered to 100 prototype vectors $\tilde{\mathbf{d}}_j$, whose Voronoi cells define the basis functions according to (5.17). Then the algorithms QCO ($\kappa = 100$) and AHL ($\Lambda = 0.5$) are applied to estimate the interaction coefficients c_j of these basis functions. Some examples for

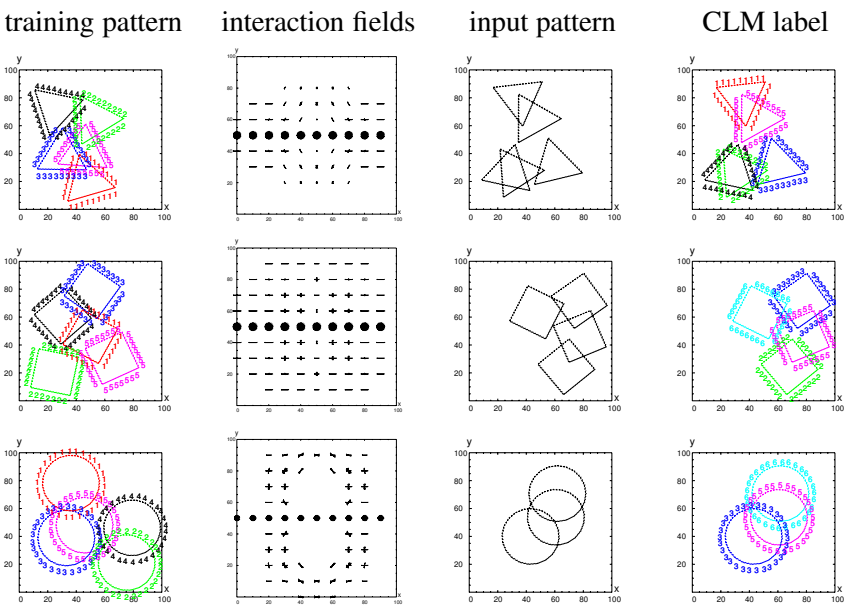


Figure 6.30: Training results for contour grouping on artifical data: the first column shows three training patterns, with five ideal triangles, squares and circles of size $R = 20$. The label of the features are displayed by color and number. The second column displays parts of interaction functions resulting from the application of AHL ($\Lambda = 0.5$) on the training patterns. In each diagram a discretization (according to position and orientation) of line segments is plotted, which have a positive interaction with a reference segment at position (50,50) with horizontal orientation. The length of the line segments reflects the strength of positive interaction. The third column shows unlabeled test patterns. The last column shows results of a ten-layered (nine figure-layer, one background layer) CLM which applies the learned interaction function of the test image. The labels are displayed by color and number.

training patterns, learned interaction functions, input patterns and CLM-Outputs are shown in Fig. 6.30 for datasets containing triangles (S=3), squares (S=4) and approximated circles (S=20) of size R = 20.

Figure 6.31 and 6.32 demonstrate the effectiveness of the learned interaction function in more detail on the example of the triangle interaction function. Equation (5.22) suggests an upper and lower bound for Λ at $\Lambda_{min} \approx 0.44$ and $\Lambda_{max} \approx 8.4$. Two patterns are presented to an $L = 10$-layered CLM which uses the learned triangle interaction function and Λ is varied from 0 to 10.

In the first pattern, two overlapping triangle contours in random, uncorrelated orientation are presented. The second pattern presents a configuration, where two triangles show two neighboring lines with parallel orientation. Figure 6.32 plots the grouping quality achieved for the two pattern against Λ, where the red graph

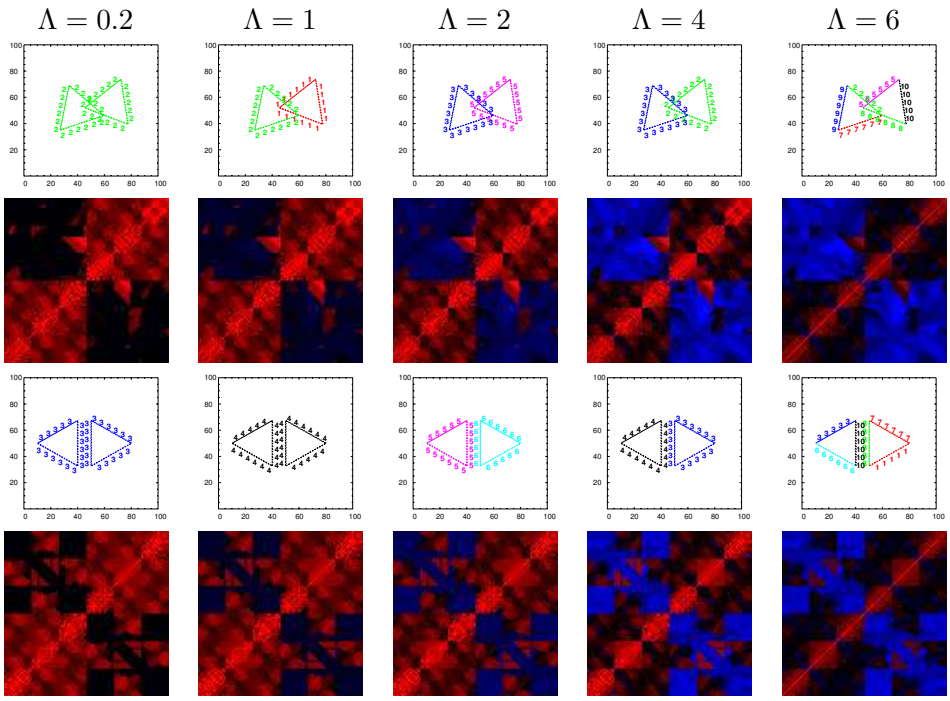


Figure 6.31: Output labelings and corresponding interaction matrices resulting from a triangle interaction function. The first and third row show output labelings of a ten-layered CLM for two triangles in different configurations resulting from the triangle interaction function at different values of Λ. The second and fourth row show the corresponding interaction matrices, where red color represents positive interaction and blue color negative interaction. The interaction weights of each matrix are normalized to $\max_{r,r'} |f_{rr'}| = 1$. The normalized connection strength $|f_{rr'}|$ is mapped to the color intensities.

corresponds to the pattern with uncorrelated orientations and the blue graph corresponds to the pattern with parallel orientations. Additionally, Fig. 6.31 shows typical output groupings and interaction matrices of the two patterns for characteristic values $\Lambda \in \{0.2, 1, 2, 4, 6\}$.

Each displayed interaction matrix is normalized to $\max_{rr'} |f_{rr'}| = 1$, positive interactions are mapped to red color, negative interactions are mapped to blue color, and the normalized interaction strength $|f_{rr'}|$ is mapped to the color intensities. The features are ordered according to their membership to the two present triangles and, on a finer level, according to their occurrence on the different lines that form the triangles to clarify the visualization. This ordering reveals the observed block structures, which for some values of Λ come close to an *ideal block diagonal* interaction matrix.

For small values of Λ and for both patterns, the positive interactions dominate, such

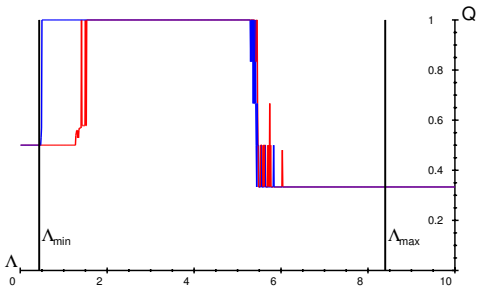


Figure 6.32: Plot of the quality achieved on the two triangle stimuli in Fig. 6.31 with values of $\Lambda \in [0, 10]$ for a triangle interaction function. The red graph corresponds to the pattern with triangles in random orientation, while the blue graph describes the situation when both triangles show a symmetric orientation.

that both triangles are assigned to the same layer reaching a quality of $Q = 0.5$. However, it can be seen, that the positive cross interactions between the triangles is higher for the pattern with parallel orientations, where the two parallel lines attract each other as also attract the two remaining lines of the respective other triangle, than for the pattern with uncorrelated orientation, where positive cross interaction only occurs in the small area of contour overlap.

The increase of Λ enhances the influence of negative interactions by shifting the components of the interaction matrices into the negative quadrant. Within the output labeling this process is accompanied by a splitting of the labeling into more, smaller groups.

In the case of uncorrelated orientations, the negative cross interactions between the triangles become early ($\Lambda \approx 0.5$) strong enough to produce the desired segmentation of the two triangles, indicated by $Q = 1$. In the more complicated situation of parallel orientation, Λ has to be increased to a higher level ($\Lambda \approx 1.5$), until the negative interactions between the far away parallel lines of the triangles are strong enough to compensate the positive cross interactions. From this point on, both input patterns behave in the same way and result in the correct grouping, until ($\Lambda \approx 4.8$) inner-triangle interactions are shifted into the negative quadrant and the triangles are decomposed into subgroups. Thereby, the high positive inner-line subblocks of the interaction matrices still result in a reasonable segmentation of the triangles into three lines indicated by $Q = 0.33$.

The interval of feasible Λ values is larger for the pattern with uncorrelated triangle orientations than for the pattern with parallel triangle orientations, but in both cases it covers a relative large subinterval of $[\Lambda_{min}, \Lambda_{max}]$. If Λ is chosen from this interval and the pattern is extended by further triangle contours in random position and orientation, the corresponding interaction matrix is simply extended by corresponding positive inner-triangle interaction blocks along the main diagonal of the interaction matrix and negative off-diagonal inter-triangle interaction blocks,

a) input pattern b) interaction matrix

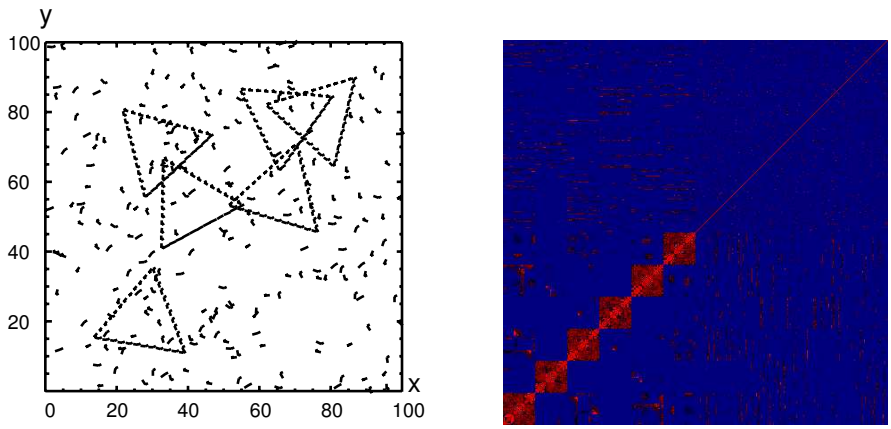


Figure 6.33: Example of an input pattern with six triangle contours and a background of random line segments.

as can be seen in Fig. 6.33. Spurious features with random orientation can be separated easily from the relevant triangle contours by application of a background layer, because background features receive an evident smaller support from other features than features on a triangle contour.
The following four experiments address the evaluation of the learning method according to the adaption of object size and shape, the influence of spurious features, the influence of errors in feature position and the influence of errors in feature orientation:

6.5.1 Adaptation to Object Size and Shape

In this experiment, it is investigated whether the algorithms QCO and AHL can adapt an interaction function to noise-free datasets with $S \in \{3, 4, 20\}$ and $R \in \{15, 20, 25, 30, 35, 40, 45, 50, 55, 60\}$. For each combination of (algorithm, S, R) random patterns with five groups are obtained from a pattern generator to train an interaction function $f_{rr'}$.
Each of these functions is tested with a ten-layered (one ground layer, nine figure layers) CLM on ten new patterns with the same object properties S and R, but with the number of objects varying from one to five. This gives a number of 100 CLM responses for each triple (algorithm, S and R) for which the average grouping quality is plotted in Fig. 6.34.
The results show, that AHL is always able to adapt the correct compatibilities for the relevant feature combinations in the patterns. It has to be remarked, that the desired grouping behavior is learned from the representation of a single pattern,

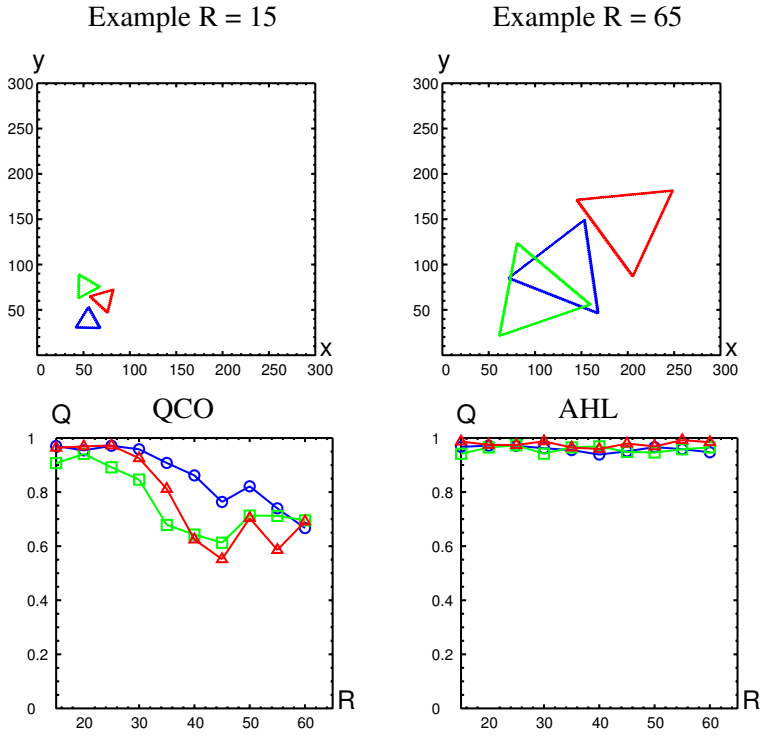


Figure 6.34: Average grouping quality for objects of different size: Plots of grouping quality against object size R (15 to 65) for $S = 3$ (triangles), 4 (squares) and 20 (circles). Top row shows stimulus examples.

where the property of rotation invariance is guaranteed from the design of the proximity functions. With the learned interaction function the CLM is able to reveal the correct number of groups, as long as the number of groups does not exceed the number of figure layers in the CLM.

QCO also performs quite good, at least when R ranges from 15 to 30. For higher values of R, a serious decrease can be observed in the grouping quality for QCO. The reason for this lies in the different number of features per pattern. For a pattern with low R, this number is smaller than for a pattern with high R, because the length of the surrounding contour grows, while the length of each local edge feature is fixed. In QCO, a set of $L-1$ consistency conditions has to be computed for each feature, which are extended by a uniform margin κ. It seems, that the choice of $\kappa = 100$ is not suitable for varying R. Actually, the qualitative performance of QCO for patterns with a higher value of R can be improved, if the value of κ is increased with the number of features in the patterns. A second disadvantage of QCO against AHL lies in the higher computation time, which becomes significant for patterns which consist of many features.

6.5.2 Influence of Spurious Features

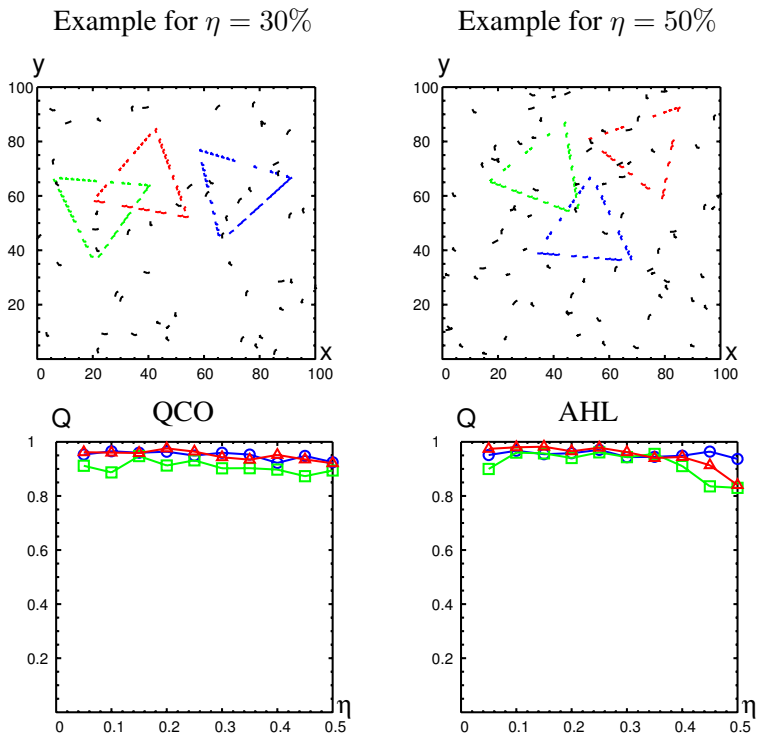


Figure 6.35: Statistic grouping quality for noisy patterns: Plots of grouping quality against percentage of spurious features η (5% to 50%) for S = 3 (triangles), 4 (squares) and 20 (circles). Top row shows stimulus examples.

In the other three experiments, it is investigated, how different types of errors in the datasets affect the grouping performance of the CLM. Again, the learning algorithms QCO and AHL are tested on objects $S \in \{3, 4, 20\}$, but this time $R = 20$ is kept fixed. Instead $\eta \in \{\ 5\%, 10\%, 15\%,\ 20\%, 25\%,\ 30\%, 35\%, 40\%, 45\%, 50\%\}$ varies, where η indicates the percentage of features that are removed from the pattern and that are replaced by randomly generated features $\mathbf{m}_r$ with $x_r, y_r \in [0, 5R], \varphi_r \in [0, \pi]$ that are labeled as background features. The average grouping quality over 100 CLM responses (10 tests for each of 10 training runs) is plotted in Fig. 6.35 for each triple (algorithm, S, η).

A high grouping quality for AHL and QCO is obtained in all test runs. These results highlight both, the robustness of the two learning methods and the robustness of the CLM against fragmentary data, which can be explained by the high redundancy in the matrix of lateral interactions and the redundant character of the CLM. The results also show, that the CLM is able to separate the relevant groups from a noisy background using the background layer.

6.5.3 Influence of Errors in Feature Position and Feature Orientation

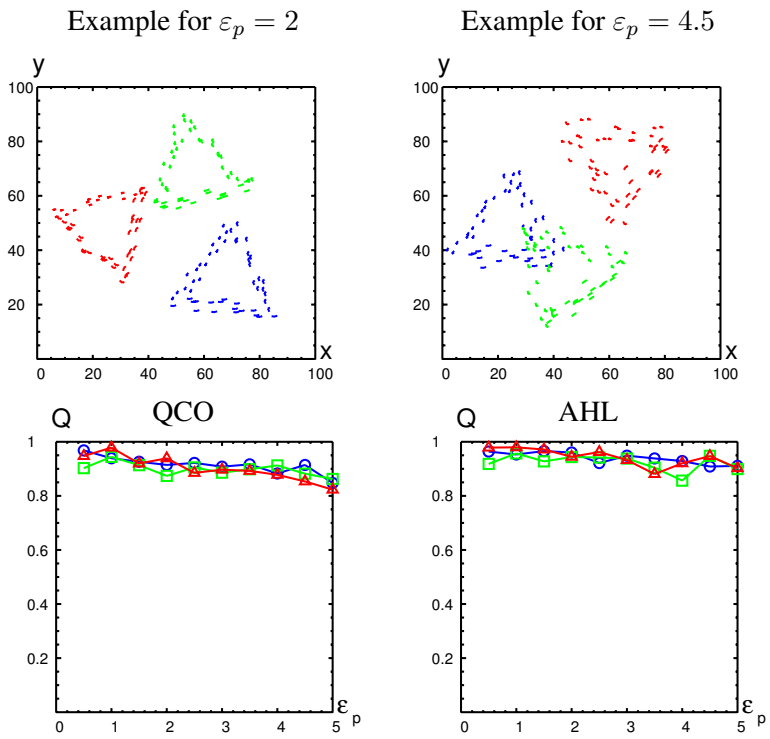


Figure 6.36: Statistic grouping quality for translation errors: Plots of grouping quality against translation error ε_p (0.5 to 5) for S = 3 (triangles), 4 (squares) and 20 (circles). Top row shows stimulus examples.

In the third and fourth experiment, the second experiment is repeated, where the error η is substituted by two other errors $\varepsilon_p \in \{0.5,1, 1.5, 2, 2.5,3, 3.5, 4,4.5, 5\}$ and $\varepsilon_o \in \{0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35 , 0.4, 0, 45, 0.5\}$. ε_p defines an error in feature position, where all features in the dataset are randomly shifted in x- and y-coordinate by $dx, dy \in [-\varepsilon_p, \varepsilon_p]$. ε_o defines an error in feature orientation, where all features in the dataset are randomly rotated by an angle $d\varphi \in [-\varepsilon_o, \varepsilon_o]$. The average grouping qualities of all triples (algorithm, S, ε_p) are plotted in Fig. 6.36, while the average grouping qualities of all triples (algorithms, S, ε_o) are plotted in Figure 6.37.

The results show, that the learned grouping principles are relatively robust against errors in feature extraction. With respect to errors in feature position both AHL and QCO show high grouping quality, which can be explained by the interaction fields in Fig. 6.30. Each feature gets positive feedback not only from features with the correct orientation, which are in a small interval of local distance, but also from features which are shifted parallel to the relevant orientation.

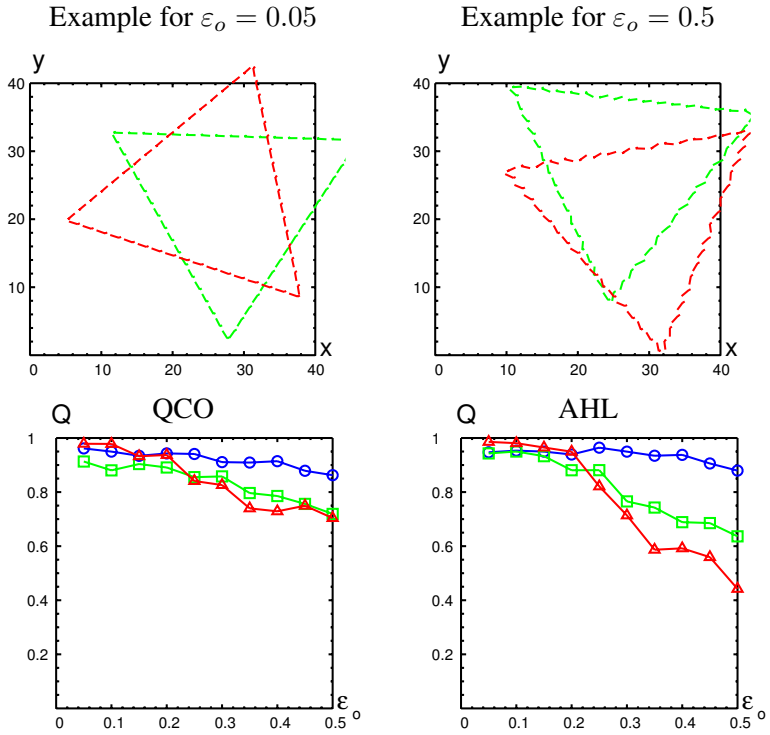


Figure 6.37: Statistic grouping quality for rotation errors: Plots of grouping quality against rotation error ε_o (0.05 to 0.5) for S = 3 (triangles), 4 (squares) and 20 (circles). Top row shows stimulus examples.

In the case of errors in feature orientation, there is a noticeable decrease in grouping quality, which is serious for AHL. This difference between AHL and QCO becomes clear, if the kind of errors are inspected that are made by the AHL-learned CLM. In patterns of triangles, a triangle is often divided into three separate lines, which causes a typical quality of 0.33. Similarly, in patterns of squares, a square in often divided into two perpendicular groups of two parallel lines, which causes a typical quality of 0.5. Consequently, the interactions of feature pairs with the characteristic inner angles of 60 respectively 90 degrees are not excitatory enough. These errors can be reduced by decreasing the parameter Λ to achieve a coarser segmentation of the data into bigger groups, as was is demonstrated in Fig. 6.31 and Fig. 6.32. The patterns of circles are more robust to this kind of errors, because they are more regular.

6.5.4 Adaptation to more Complex Contours.

The previous experiments have shown, that the AHL approach can achieve a good amount of generalization and robustness, if the representation of the feature space

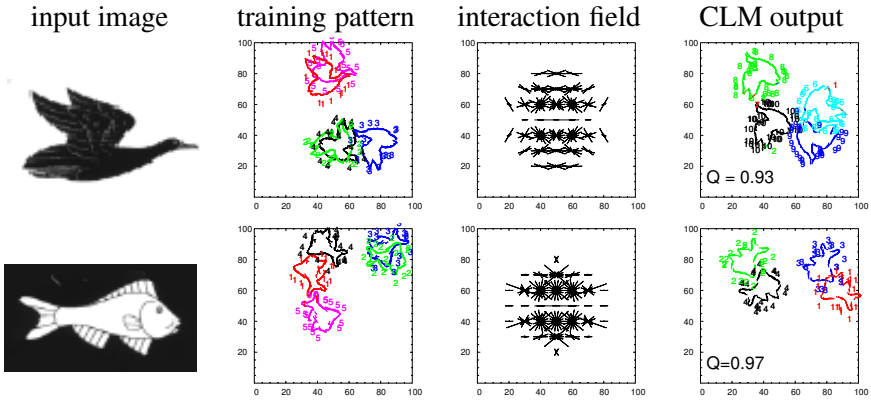


Figure 6.38: Contour grouping for more complex objects like a fish and a bird.

$\mathcal{F}$ and the pairwise proximity space $\mathcal{D}$ are consistent with the grouping behavior specified by the target labeling. For the artifical polygon contours, this consistency is a result of the high self-similarity of the polygon shapes, where the rotation invariant interaction function matches at each point of the contour.

In Fig. 6.38, the application of the contour grouping approach is demonstrated on the more complex and non-symmetric contours of a bird and a fish. Therefore, Sobel-x and Sobel-y filters are applied on the two images in column one of Fig. 6.38 to extract salient edge features $\mathbf{m}_r = (\mathbf{p}_r, \varphi_r)$. The Sobel responses are thinned out to one pixel wide contours by non maximum suppression according to the intensity gradient strength and orientation, followed by a thresholding with 50% of the maximal intensity gradient strength and a subsampling of the resulting edge set by the factor 2. The orientation angles $\varphi \in [0, \pi]$ are extracted from the Sobel-responses $\mathbf{o}_r = (S_r^x, S_r^y)$ at the remaining edge positions $\mathbf{p}_r = (x_r, y_r)$ neglecting the unique direction information, such that the edge sets are invariant to a black/white switching of the foreground and background color. Training and test patterns are constructed by copying randomly rotated and translated versions of the extracted edge sets into a 2D coordinate system, where the maximal distance of an edge to the center of its group is normalized to a fixed radius $R = 30$ (see the second column of Figure 6.38).

Since the contour of the bird and the fish show varying shape along the border line, the learning problem of extracting relevant inter-edge angles and distances is less consistent than before. The higher overlap between positive and negative interactions within the pairwise proximity space can be interpreted as loss of higher order relation information in the AHL-learned ($\Lambda = 2$) interaction functions in column three of Figure 6.38. Both interactions integrate several edge combinations typical for different points on the object contours, which makes them similar to an ordinary edge clustering interaction expressed by excitatory connections for short ranged local distances at all feature orientations. Thereby, the interaction functions

concentrate the excitatory connections to different types of object specific edge configurations. The similarity to the edge clustering behavior weakens the ability of the CLM to separate overlapping and close-by contours, which is reflected by suboptimal grouping quality values of $Q = 0.93$ and $Q = 0.97$ for the CLM outputs in Fig. 6.38. Also, the ability of figure-background separation are weaker, because both type of interactions generate an extensive support for features in random edge clusters. However, the next chapter will show, that these degenerated interaction functions still contain enough information to differentiate the contours of the respective objects in a CLM architecture with layer specific interaction functions.

6.6 Summary

Several examples on artifical and real world problems have shown, how the learning can be practically realized to solve a wide range of grouping problems.
The example of point clustering has shown, how the control parameter Λ influences the shape of the interaction function by scaling the strength of inhibitory feature-pairs against that of the positive ones. Thereby, the segmentation level of the CLM output labeling can be adapted to the target labeling. Further, this example showed, that the problem of finding a suitable value for Λ complicates with the degree of inconsistency in the training pattern.
The effect of inconsistent data was visualized in detail on the spiral problem examples, where the increasing degree of contortion of two spiral arms resulted in an increasing annihilation of positive and negative interaction.
The cell image segmentation problem compared the AHL-algorithm with standard binary classification approaches. Qualitatively, the results of AHL were slightly outperformed by the application of the SVM. But it was argued, that, nevertheless, AHL has advantages against the other approaches from its simple algorithm, which reduces parameter tuning to a minimum and results as well in a very fast learning speed as a fast evaluation speed of the learned interaction function. Further, it was shown, that the control parameter Λ can be used to adapt the grouping behavior to different cell sizes and that the grouping behavior can be implemented by different types of feature proximity functions, where spurious proximity functions are ignored by the AHL algorithms. Finally, the robustness of the AHL approach against errors in the target labeling was shown.
The problem domain of texture segmentation was characterized by a relative high dimensional proximity space, where the AHL algorithm was able to extract the relevant information from this space.
Finally, an example from the domain of contour grouping showed that the AHL algorithm can extract simple shape parameters from edge sets of dynamical length. This behavior were very robust against lacking or noisy features and showed the interesting properties of translation and rotation invariance, which were specified a priori by the choice of the feature proximity functions.

Surely, this high adaptation bases on the high regularity within the polygon patterns, but it motivates the question: Can several such shape descriptive interaction functions be applied in competition within the CLM to develop a shape classifier that can work on object contours? This question is subject of the next chapter.

Chapter 7

Classification Abilities of the CLM

Up to this point, the CLM was only applied on pure grouping and segmentation tasks, where in all layers of the CLM the same interaction matrix was used.
Wersing [61] has already shown, that the convergence and assignment conditions (see section 3.2) can be generalized to the case of layer specific interaction matrices, where $f^{\alpha}_{rr'}$ is the interaction between feature $\mathbf{m}_r$ and $\mathbf{m}_{r'}$ in the layer α.
This motivates the approach to implement several class specific grouping behaviors by applying the corresponding interaction functions in competition between different types of layers. As a result of this approach, the features can not only be divided into groups, but can also be classified according to the class of layer, they are assigned to.
Inspecting the artifical polygon examples in section 6.5, it becomes clear, that this approach causes problems, if two or more interaction functions describe the same or similar grouping behavior. In the polygon example this is the case, because lines are subgroups of triangles and squares and, therefore, can be grouped by any of the three corresponding interaction functions.
To solve this subsumption problem, additional layer weights $w_c, c = 1, \ldots, C$ are introduced, where C is the number of different object classes. These weights are used to scale the different interaction functions, respectively matrices, against each other by

$$f'^{c}_{rr'} = w_c f^{c}_{rr'}. \tag{7.1}$$

Before the problem of finding suitable layer weights is inspected, it is observed, similar to chapter 4, how the CLM behaves during the annealing process, if several *ideal block-diagonal* interaction matrices are set into competition to each other.

7.1 Competition of Interaction Matrices

Consider the seven binary patterns displayed in Fig. 7.1. Each pattern consists of $N = 900$ features, where each feature has it's own relation between black and

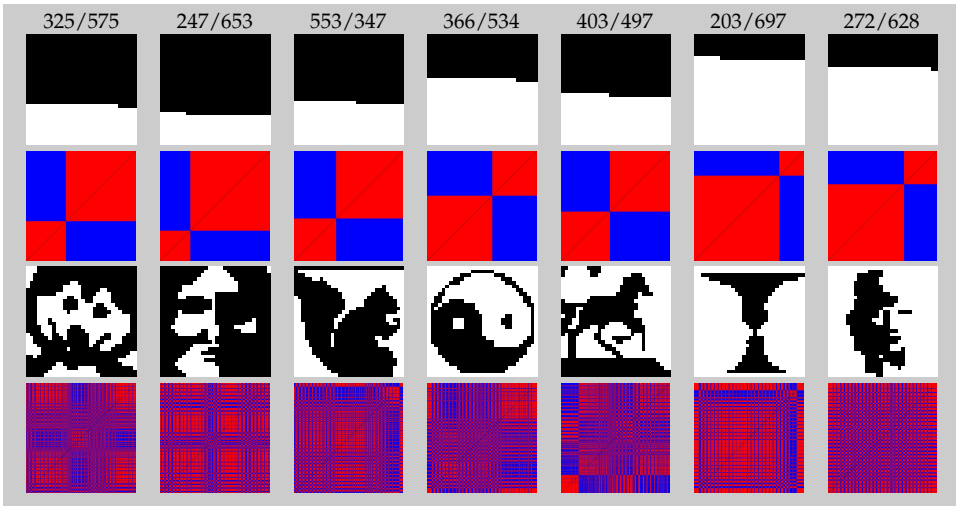


Figure 7.1: Seven patterns and their interaction matrices

white labeled features, e.g. the first pattern consists of 325 white features and 575 black features. An ideal block-diagonal interaction matrix of 1's and -1's is constructed for each pattern, where the diagonal elements are set to zero ($\forall c: \ \forall r: f^c_{rr} = 0$).

These seven interaction matrices have the same eigenvalues $\lambda_1 = 899$, $\lambda_2, \ldots, \lambda_{900} = -1$, but different eigenvectors, where the eigenvector of the largest eigenvalue λ_1 describes always the separation of the corresponding pattern into it's two groups. If only one of these seven interaction matrices is applied within the different layers of the CLM, the WTA behavior of the two groups in the respective pattern are activated when T reaches the corresponding "group size-1" (compare observations of chapter 4). For the first pattern these thresholds are $T = 574$ and $T = 324$.

To give each pattern a meaning, the features are permuted according to the third row of 7.1 and the patterns are called: *ghost*, *man*, *squirrel*, *yin-yang*, *horse*, *vase*, and *woman*.

Consequently, the columns and rows of the corresponding interaction matrices have to be permuted in the same way, which has no effect on the eigenvalues and activation thresholds of the WTA behavior and permutes the components of the eigenvectors correspondingly.

Now the annealing process is simulated for an $L = 70$-layered CLM, where always 10 layers of the CLM share the same interaction matrix and all seven layer classes are scaled identically: $w_1, \ldots, w_7 = 1$. The external input of the neurons is set uniformly to $h_r = 1$, and the strength of vertical WTA connections is set to $J := 2 \max_{r\alpha} \sum_{r'} \max(0, w^\alpha f^\alpha_{rr'})$ to guarantee convergence. The pseudo-temperature is explored by hand and the attractor states at the interesting thresholds of T are visualized in Fig. 7.2 corresponding to the visualizations in chapter 4.

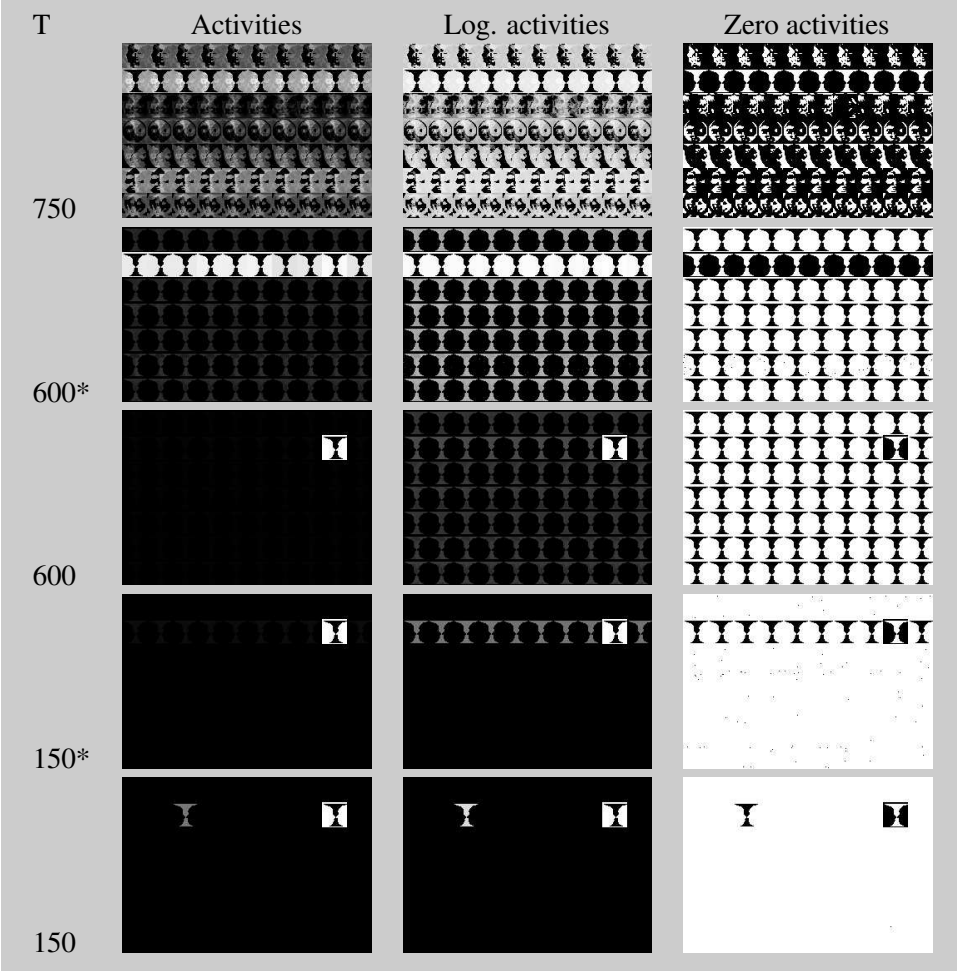


Figure 7.2: Course of the CLM-dynamics under the dominance of pattern *vase*. Each subimage describes the state of an $L = 70$-layered CLM with seven different types of layers. Always ten layers use the same pattern dependent interaction matrix and form a row in the CLM state. The seven different patterns and interaction matrices are shown in Fig. 7.1. The seven types of layers are scaled uniformly by the layer weights $\mathbf{w} = (1, 1, 1, 1, 1, 1, 1)^T$. States marked by "*" show intermediate states, where the dynamics was halted, before it had reached a stable attractor state.

Annealing is started at a temperature, where the WTA behavior is inactive for all groups in the different interaction matrices. Since the pattern *vase* shows the biggest group of all patterns (697 features), T is set to $T = 750$, such that it can be expected that the DC-modes of the CLM dynamics (see section 3.4) dominate for all interaction matrices. The characteristic attractor state in the first row of Fig. 7.2 shows, that all layers which share the same interaction matrix converge to the same activation pattern. Thereby, each layer class shows it's own activation pat-

tern, which tries to point into the direction of the larger group of it's corresponding pattern.
The manifestness of the different activation patterns decreases with the size of the larger group of the corresponding input pattern, such that the structure of pattern *vase* dominates (shows the strongest activation) mostly, while the pattern *horse* is suppressed mostly by the other patterns. Thus already at this early simulation state, the dominance of pattern *vase* in the output grouping becomes visible.
After the dynamics has converged, the pseudo-temperature is switched to the value $T = 600$, which activates the WTA process for the larger groups in the patterns *vase*, *man* and *woman*. Since the pattern *vase* already dominates the attractor state at $T = 750$, it also dominates the WTA process. However, this happens in two phases. In the first phase, the CLM is driven towards an intermediate state, indicated by the notation $T = 600^*$ in Fig. 7.2, where all layers of pattern *vase* show the same activation pattern, which shows the larger group of pattern *vase*. All other layers show the same activation pattern for the smaller group of pattern *vase*, but this state is not stable. Instead, in the second phase, the WTA behavior is activated between the layers of pattern *vase*. The activation of the larger group starts to differentiate, where the activation of the most active layers grows most, while the activation in the less activated layers decreases. If the activation in a layer reaches zero, it changes it's activation pattern towards the smaller group of pattern *vase*. At the end of the WTA process at $T = 600$, one layer of pattern *vase* shows activation for the bigger group of this pattern, while all other layers show activation of the smaller group in pattern *vase*. The activation of the smaller group is slightly higher (see logarithm of activities) in the layers of pattern *vase* than in the other layers, which can be explained by the fact that the remaining features exchange the highest support in the layers of pattern *vase*.
In a second annealing step, the pseudo-temperature is reduced to $T = 150$, which activates the WTA behavior for all groups within the seven patterns. But since the WTA-behavior was already activated at $T = 600$ for the bigger group of pattern *vase*, and since the CLM had time to converge to its attractor state, the *vase* layers are also dominant for the second group of the pattern.
Again, the WTA process is divided into two phases. In the first phase, the activation in all layers not belonging to the pattern *vase* is suppressed and the CLM is driven to the state marked by $T = 150^*$. This state is not stable. In the second phase, the WTA process forces the total activation of the smaller group into a single layer of pattern *vase*. A further decrease has no more effect on the structure of the CLM attractor state (besides a scaling of the activities). Finally, there are only two active layers for the two groups of pattern *vase* in the corresponding layer class.
From this sample simulation two results shall be marked: the first is, that it depends on the ordering of activation thresholds in the different interaction matrices which layer class is dominant for a group. The second is, that the WTA behavior is divided into two phases. The first acts between the dominant and not dominant layer classes and the second acts only within the dominant layer class. Annealing should be performed very slow around the critical T-values, especially, if layers

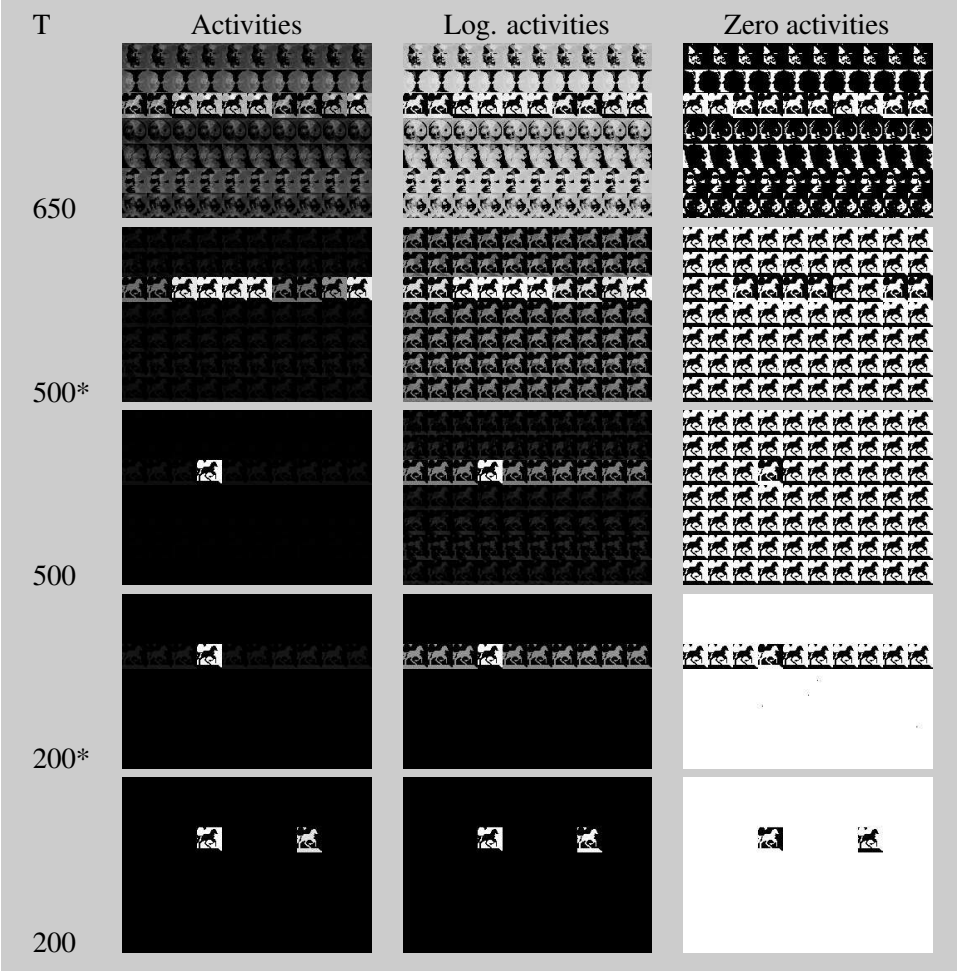


Figure 7.3: Course of the CLM-dynamics under dominance of pattern horse. The seven types of layers are scaled by the layer weights ($\mathbf{w} = (0.65, 0.65, 1, 0.65, 0.65, 0.65, 0.65)^T$). Again, states marked with "*"show intermediate states, where the dynamics was halted before it had reached a stable attractor state.

of the same layer class shall be reused for further groups, and, if the gap between the activation thresholds in the dominant and non-dominant interaction matrices is very small.

Layer weights can be used to rescale the interaction matrices and, thereby, their WTA activation thresholds. Obviously, there exist weights that make any of the seven patterns encoded in the interaction matrices dominant. For example the weights that leave the interaction of pattern *horse* unchanged $w_3 = 1$ and scale all other interactions with the factor $w_1 = w_2 = w_4 = w_5 = w_6 = w_7 = 0.65$ make the pattern *horse* dominant (see Fig. 7.3). The only difference to the simulation in Fig. 7.2 is an early-state split of the layers of the pattern *horse*.

For the example, this predictable behavior is a result of the high regularity of the ideal block diagonal interaction matrices and it becomes more complex for the competition of more approximative interaction matrices, e.g. like results from the AHL algorithm. However, the results on the contour grouping of regular polygons in section 6.5 have shown, that at least for simple shapes the reconstructed interaction matrices come close to the desired block diagonal structure, such that it might be reasonable to classify simple shapes with the CLM. The next section presents an approach to estimate suitable layer weights that make such a classification process possible.

7.2 Training the Layer Weights

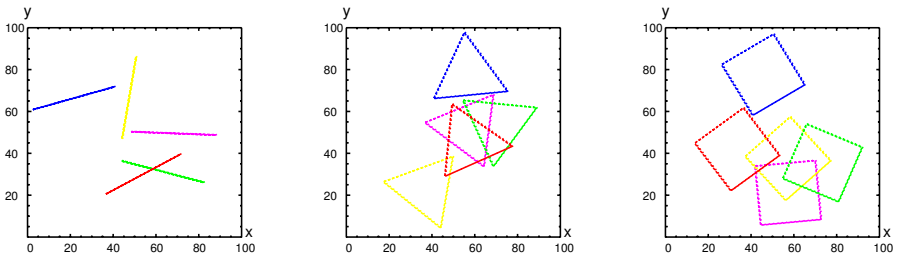


Figure 7.4: Examples of training patterns for learning of class-specific interaction functions with the AHL algorithm.

The goal is to build a simple shape classifier that is able to distinguish lines, triangles and squares of a certain size. In a first step, three different interaction functions $f^c_{rr'}, c = 1, \ldots, 3$ can learned independently, e.g. by applying AHL on the three patterns in Fig. 7.4. Each of these functions is parameterized by its own segmentation control parameter Λ. For simplicity, it is assumed, that a uniform value of Λ is chosen for all functions. Since the adjustment of Λ to an optimal value depends on the number of groups in training pattern, which can differ from the number of labels in the target labeling, and, since the decision about the correct segmentation level demands much problem-specific knowledge, Λ is chosen by hand.
What is left is an automatic estimation of the layer weights $w_c, c = 1, \ldots, 3$ $(c = 1, \ldots, C)$ that induces a correct assignment of the three shapes to the corresponding layers. Therefore, three additional input patterns for a single line, triangle and square, like in Fig. 7.5, are presented to each of the interaction functions to construct an interaction matrix, such that a $3 \times 3(C \times C)$ matrix of interaction matrices is obtained.
Let the index $k, k = 1, \ldots, 3$ $(k = 1, \ldots, C)$ enumerate the interaction functions, and, let the index $l, l = 1, \ldots, 3$ $(l = 1, \ldots, C)$ enumerate the input patterns, such that $F^k(\mathcal{P}^l)$ is the interaction matrix computed with interaction function $f^k_{rr'}$ on pattern $\mathcal{P}^l$.

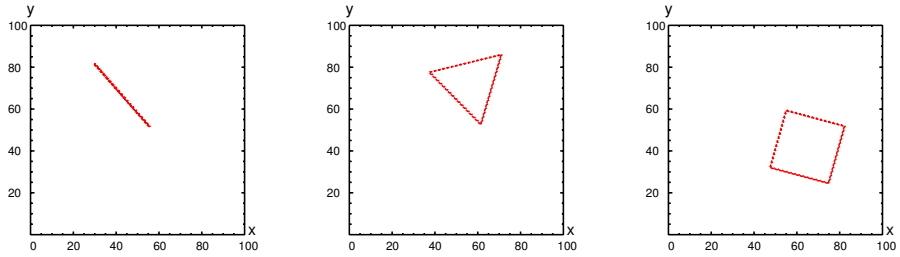


Figure 7.5: Presentation of single object patterns to estimate the layer weights.

For dimension reduction, each matrix is represented only by its largest eigenvalue. Then $e_{kl} = \lambda_{max}\{F^k(\mathcal{P}^l)\}$ is the largest eigenvalue of the interaction matrix constructed from pattern l with the interaction function k, and E is the $3 \times 3(C \times C)$ matrix of these eigenvalues. Since each pattern shows only one figure group, the eigenvalues approximate the inner-group support and, thereby, also the activation thresholds of the WTA behavior.
Each interaction function and, therefore, each column of E is scaled with it's layer weight $w_k, k = 1, \ldots, 3(k = 1, \ldots, C)$.
If only two interaction functions with indices i and j have to be scaled against each other, only the submatrix

$$\begin{bmatrix} w_i e_{ii} & w_j e_{ji} \\ w_i e_{ij} & w_j e_{jj} \end{bmatrix}. \tag{7.2}$$

has to be considered. If on the one hand object i is presented

$$w_i e_{ii} > w_j e_{ji} \tag{7.3}$$

must hold, such that class i dominates, because it is switched firstly into the WTA behavior. If object j is presented on the other hand

$$w_j e_{jj} > w_i e_{ij} \tag{7.4}$$

must hold, such that layer class j dominates, because it is switched firstly into the WTA behavior. Consequently, $\frac{w_i}{w_j}$ is bound by (7.3) and (7.4) to

$$\frac{e_{jj}}{e_{ij}} > \frac{w_i}{w_j} > \frac{e_{ji}}{e_{ii}} \quad \text{respectively} \quad \frac{e_{ii}}{e_{ji}} > \frac{w_j}{w_i} > \frac{e_{ij}}{e_{jj}}. \tag{7.5}$$

Figure 7.6 a) sketches these bounds, where the layer weights w_i and w_j are given along the x- and y-axis and the proportions $\frac{e_{jj}}{e_{ij}}$ and $\frac{e_{ji}}{e_{ii}}$ are drawn as slope triangles. Adding the constraint $\max_k w_k \leq 1$, it becomes clear, that the shaded area describes the base of a volume unit perpendicular to the axes of w_i and w_j, where the size of this volume decreases with the similarity of the two interaction functions $f^i_{rr'}$ and $f^j_{rr'}$, respectively the eigenvalues e_{ii}, e_{ij}, e_{ji} and e_{jj}.
Suitable layer weights for scaling two interaction functions can be found by

$$\frac{w_i}{w_j} = \frac{1}{2}\left(\frac{e_{jj}}{e_{ij}} + \frac{e_{ji}}{e_{ii}}\right) \quad \text{or} \quad \frac{w_j}{w_i} = \frac{1}{2}\left(\frac{e_{ii}}{e_{ji}} + \frac{e_{ij}}{e_{jj}}\right). \tag{7.6}$$

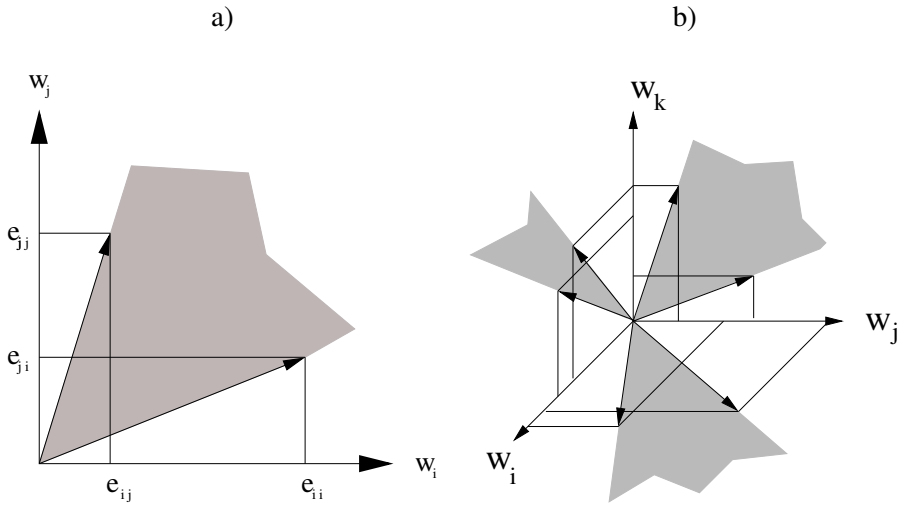


Figure 7.6: a) Relation between two layer weights. The two layer weights w_i and w_j can be chosen from the gray area defined by the eigenvalues e_{ii}, e_{ij}, e_{ji} and e_{jj}. b) Relation between three layer weights results from the intersection of the three pairwise weightings.

However, the exemplary classification problem consists of 3 (C) classes. Therefore, the 6 ($C(C-1)$) inequalities for all pairwise comparisons of the classes have to be fulfilled:

$$\forall i: \quad \forall j \neq i: \quad w_i e_{ii} > w_j e_{ij}. \tag{7.7}$$

This means, that $w_1, \ldots, w_3$ ($w_1, \ldots, w_C$) have to be chosen from the intersection of all constraint volumes perpendicular to the pairs of weight axes (see Fig. 7.6 b). Obviously, this intersection volume of feasible layer weights is restricted more and more by any new constraint. If many or strongly overlapping interaction functions are applied in parallel, this intersection can be empty, such that no feasible layer weights can be found. In this case, a simultaneous classification of all classes is impossible. Therefore, a set of slack variables ξ_{ij} is introduced into equation (7.7) and it is formulated as a constrained optimization problem:

$$\begin{array}{lllll} \text{minimize}: & -\sum_i \sum_{j \neq i} (w_i e_{ii} - w_j e_{ij}) & & & \\ \text{subject to}: & \forall i: & \forall j \neq i: & w_i e_{ii} - w_j e_{ij} + \xi_{ij} > 0 & \\ & \forall i: & \forall j \neq i: & \xi_{ij} > 0 & \\ & \forall i: & w_i > 0 & & \end{array} . \tag{7.8}$$

This problem can be solved with a standard Linear Matrix Inequality (LMI) solver from the mathematical environment SCILAB [12]. Figure 7.7 summaries the complete learning and application process of the classification approach.

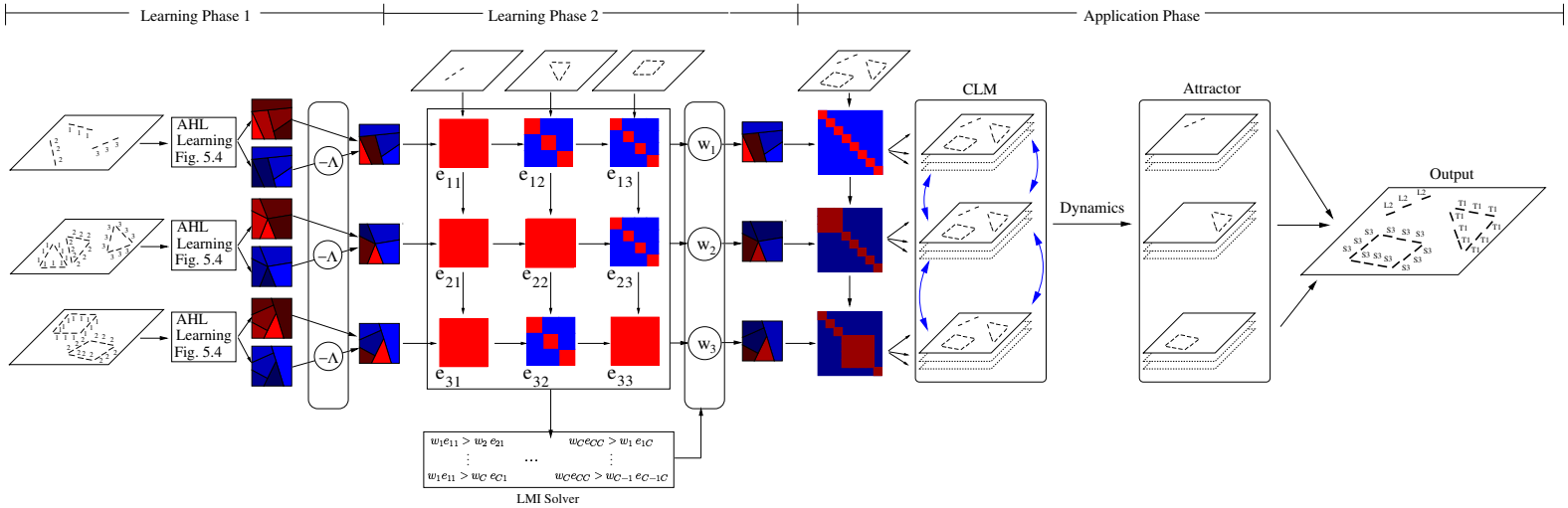


Figure 7.7: Sketch for classification process. In the first learning phase, individual class-specific interaction functions $f^c_{rr'}, c = 1, \ldots, C$ are learned separately with the AHL algorithm. Then a uniform value of Λ is specified for all these interaction functions. In the second learning phase, suitable layer weights are estimated by computing the maximal eigenvalues of the interaction matrices resulting from the application of each interaction function on each object and solving the constrained optimization problem for the layer weights (7.8). In the application phase, each interaction function $f^c_{rr'}$ is scaled with it's layer weight w_c and is used to compute an interaction matrix for the new input pattern. These different interaction matrices compete within the CLM and drive the CLM dynamics to an attractor state, where the layer label can be extended by a class label according to the layer class a group of features is assigned to.

It seems reasonable, that this classification approach works for the three class classification of lines, triangles and squares as sketched in Fig. 7.7, because of the low number of classes and the high regularity of the three shapes. In the next section, it's qualification for more complicated problems is tested.

7.3 Application

7.3.1 Classification of Artifical Letter Contours

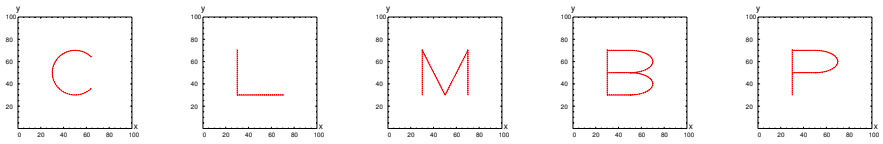


Figure 7.8: Examples of artificial letter patterns for the letters C, L, M, B and P in standard orientation. Each pattern consists of 40 to 100 line segments. The shape of the letters C, L and M is clearly distinguishable, while the letters B and P describe a subsumption problem.

In the first application of the shape classification approach, artifical contours are investigated, where the number and complexity of the contours is increased compared to the regular polygons example in section 6.5. The set of classes is defined by the 26 letters of the alphabet. Each letter is defined by a set of lines and circular or ellipsoid arcs that are divided into small line segments $\mathbf{m}_r = (\mathbf{p}_r, \mathbf{o}_r)$, specified by position $\mathbf{p}_r$ and orientation $\mathbf{o}_r$. Examples are shown for the letters C, L, M, B and P in Fig. 7.8. Each letter consists of 40 to 100 line segments.

The dataset induces several subsumption and high similarity problems between the classes, e.g. between the letters C, G, O, and Q, which all show the same circular arc of line segments, or the letter I, L, T and X, which all consist of single or perpendicular lines.

The interaction weights are adapted to detect each letter in an arbitrary orientation, but fixed size. Therefore, the same proximity functions as visualized in Fig. 6.29 are applied.

In the first step, the 26 interaction functions are learned separately with AHL ($K = 100$, 10000 learning steps per learning phase) on a pattern consisting of five objects of the respective class in random orientation and position. Figure 7.9 shows examples for the training patterns. Since the letters are less regular than the polygons, the interaction functions have to adapt more than one typical object-specific angle and distance. To prevent the AHL algorithm form learning the trivial grouping behavior of edge clustering, each training pattern shows several overlapping contours of the respective letter in arbitrary orientation. Actually, the interaction functions adapt several angle and distance configuration within the letters.

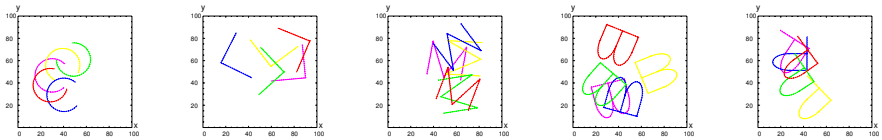


Figure 7.9: Examples of training patterns for AHL in the first phase of the learning approach. Each pattern shows five overlapping instances of the respective letter in arbitrary orientations.

Since the letters have different levels of complexity, the learned interaction functions show different overlaps between the positive and negative interactions. These differences are shown in Fig. 7.10, where the lower bound Λ_{min} (blue) and upper bound Λ_{max} (red) of the control parameter Λ is plotted for each interaction function. For the simple shaped letters, like I, O and T, the gap between both Λ-bounds is relative large, while it is smaller for more complex letters, like B, P and R. The value of Λ is set with $\Lambda = 0.9$ close to the lower bound of all interaction functions. This prevents the split of the letters into several subgroups, but also increases the probability of merging the letters with spurious features from the background.
In the second learning phase, the 26×26 matrix E of largest eigenvalues is computed by representing randomly rotated versions of the patterns in Fig. 7.8 to each of the interaction functions to build corresponding interaction matrices. The entries of the resulting E-Matrix are visualized in Fig. 7.11. This matrix mirrors the similarity between the letters, e.g. the interaction function of the letter C shows high responses for the letters G, O and Q.

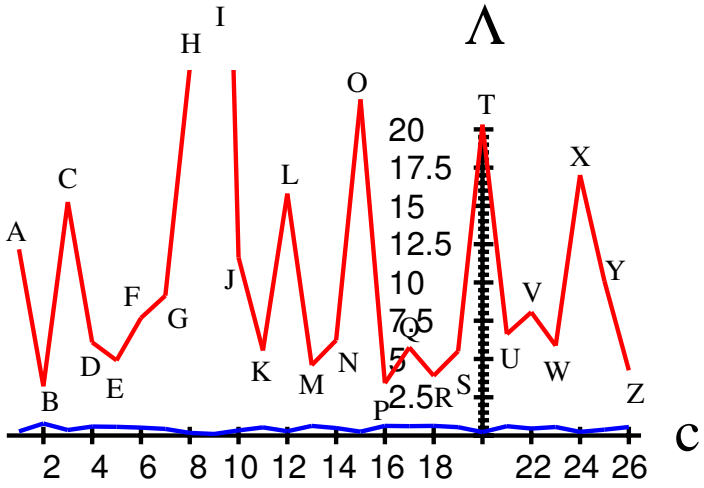


Figure 7.10: Plot of the upper bounds Λ_{max} (red) and lower bounds Λ_{min} of the control parameter Λ for all letter patterns.

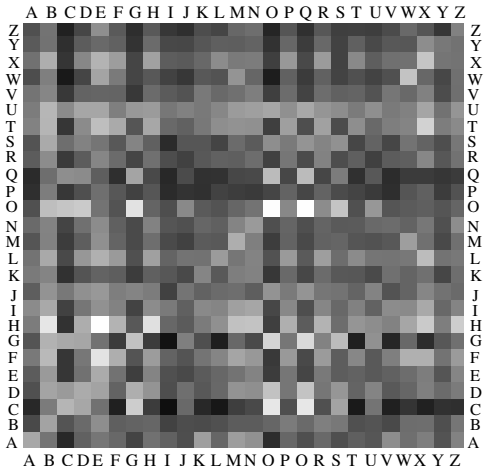


Figure 7.11: Visualization of the 26×26 E-matrix of the artifical letter problem. The maximal entry is normalized to the intensity 255. Each row describes the response (maximal eigenvalue of a resulting interaction matrix) of an interaction function, while each column shows the responses on a single object input pattern.

Binary Classification: Before it is tried to distinguish all letters at the same time, it shall be tested whether at least a correct binary classification between all possible pairs of letters can be achieved. Therefore, the interaction functions of two letters and the corresponding submatrix of E

$$\begin{bmatrix} e_{ii} & e_{ji} \\ e_{ij} & e_{jj} \end{bmatrix}, \tag{7.9}$$

are chosen. The layer weights w_i and w_j are estimated with (7.6) and each of the two scaled interaction functions is applied in three layers of an $L = 6$-layered CLM. Since the eigenvalues in the E-matrix between two classes can lie close together, the annealing process has to be performed very slowly. This is done in 10000 steps of equal stepsize from $T = 30$ to $T = 0$, where in each annealing step 4000 neurons are updated.

The classification rate is tested by presenting each of the two classes five times by an arbitrary rotation of a single object pattern without background, like the examples shown in Fig. 7.8. With very few exceptions all features are assigned to the same layer, such that the class where the majority of features is assigned to is returned as output class. The number of 0 to 10 correct classifications is counted for each test run. This classification rate is tested for ten subsampling rates of the test pattern, where 0% to 90% of the features in the patterns are erased.

Figure 7.12 shows the classification rate for this binary classification. Each column and row corresponds to one of the 26 letters. The squares show the number of correct classifications of 100 test patterns on 10 different pattern subsampling rates.

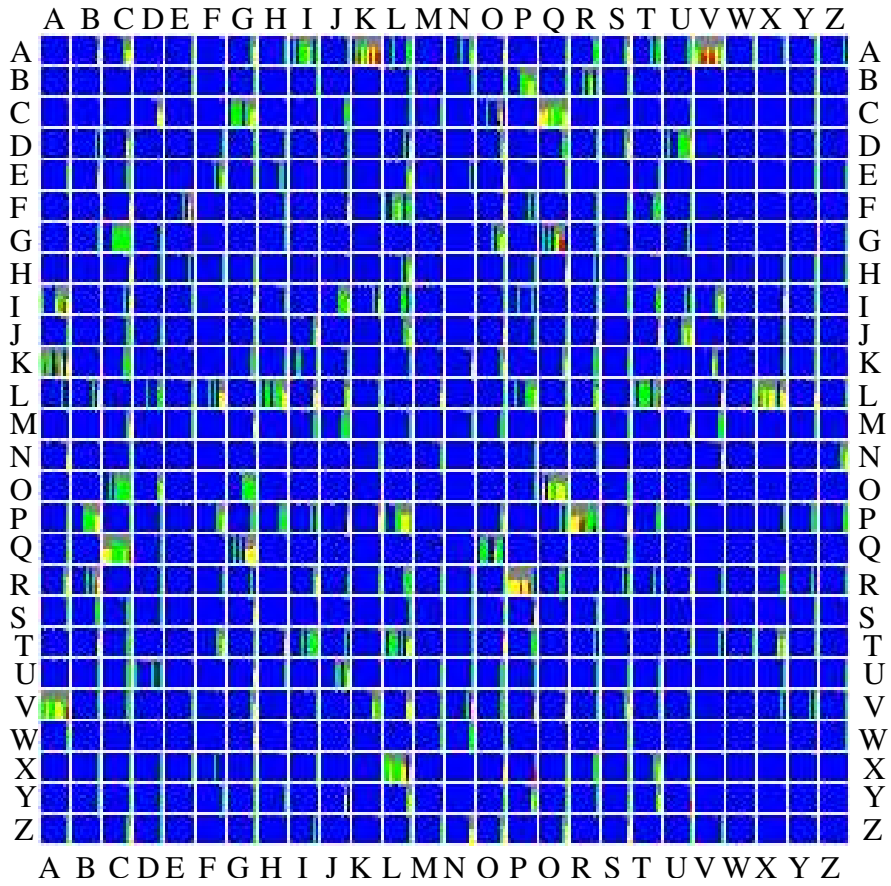


Figure 7.12: Binary classification rates for the artifical letter problem. Each square in the diagram shows the classification rate for the classification of two classes i and j. The complete diagram shows the results of the 26×26 binary classifiers. Each column of the squares corresponds to a subsampling rate of the pattern from 0 to 90%. The height of each column shows the number (from 10 test patterns (5 of each class)) of patterns that are classified correctly. The color of the columns corresponds to different classification rates: blue: 9-10 patterns correct, green: 7-8 patterns correct, yellow: 5-6 patterns correct, red 3-4 pattern correct, and black 0-2 patterns correct.

It can be seen, that a binary classification between the majority of patterns can be achieved. Only for extremely (under arbitrary rotation) similar letters, like A and K or A and V, equation (7.6) results in insufficient layer weights. However, a classification between these classes can be achieved by manual tuning of the layer weights using slow annealing speed during the CLM-dynamics or by an adjustment of the control parameter Λ. The classification rate stays stable for high subsampling rates of the features. However, it is not tested systematically in this experiment whether

the classification abilities show some generalization in terms of figure-background separation, multiple detection of the same object class and the disturbance of the patterns by noisy edge features.
Figure-background separation needs an adjustment of the self-interaction strength m according to the subsampling rate of the patterns and multiple detection of objects needs a very slow annealing speed to prevent a certain class from manifesting in the layers of a similar class, when already one or more objects of the target class have reached the WTA behavior (compare to the behavior in section 7.1, Fig. 7.2 at $T = 600^*/T = 600$).
It can be concluded, that the classification approach is at least suitable for the binary classification of the artifical letter contours, which motivates the question: How does it performs, if all 26 interaction functions are applied in parallel?
Parallel Classification of all Classes:

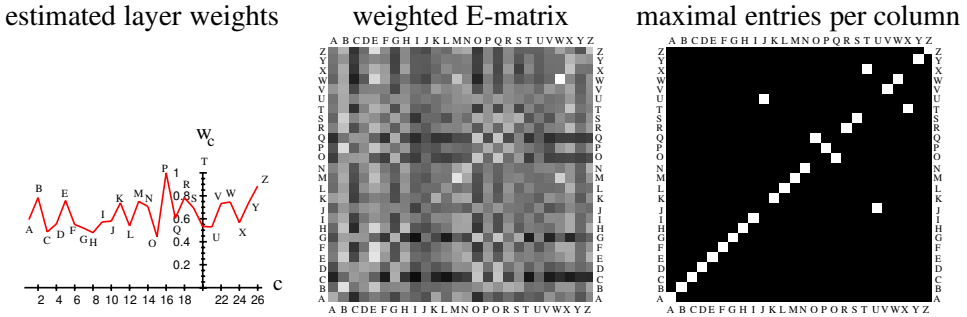


Figure 7.13: Left: layer weights resulting from the optimization of (7.8); middle: rescaled rows of the E-matrix; right: the maximal entries in each column of the weighted E-matrix are highlighted.

The left-hand side of Fig. 7.13 shows the layer weights that result from the SCILAB LMI-solver applied on the E-Matrix in Fig. 7.11. The middle part of Fig. 7.13 shows, how the E-matrix is changed, if its rows are scaled with these layer weights, and on the right-hand side of Fig. 7.13 the maximal entry in each column of the weighted E-matrix is highlighted.
The learned layer weights achieve the domination of most of the diagonal-elements of the E-matrix. However, they also indicate confusion between the classes of the layers J and U, O and Q, and T and X, where the optimization fails. This expectation is tested on an $L = 78$-layered CLM, where each of the 26 interaction functions is applied in 3 layers. Actually, misclassifications arise when letter F is assigned to class P, letter J is assigned to class U, and letter Q is assigned to class G.
This shows, that the LMI-solver did not find a feasible solution of the inequalities (7.7) and therefore violates the conditions for the layer weights of the classes J, O, Q, T, U and X. Further, the criterion of maximal diagonal elements of the E-matrix does not describe perfectly the correct classification criterion for the layer

weights. There exists a deviation between the expected classification errors from the optimization problem and the actual classification errors from the simulation of the CLM. However, these errors occur between very similar classes and for most of the classes the optimization of (7.8) results in the correct layer weights. It seems, that for high number of classes it becomes improbable to find feasible layer weights, while for subsets of these classes feasible solutions are possible. A possible solution might be to find several sets of layer weights that are feasible for different subsets of the classes and combine the results of the different subclassifiers by logical operators to a global classifier. The advantage of such a classifier is, that classes can be simply added or removed by adding or removing corresponding class-specific interaction functions and adjusting the layer weights. The disadvantages are a high memory effort, because an interaction matrix has to be stored for each class, and a high computation time, because annealing has to be performed very slowly, if similar classes shall be discriminated. In the next section, it is investigated whether the classification abilities of the CLM still hold for more realistic patterns.

7.3.2 Example: Classification on COIL20

As second more realistic dataset a subset of the COIL20 [38] data set is observed. This subset is shown in Fig. 7.14. It consists of 20 different objects at one specified view. These patterns are described by the salient edge features $\mathbf{m}_r = (\mathbf{p}_r, \mathbf{o}_r)$ within the images, where $\mathbf{p}_r$ is given by the pixel positions, and $\mathbf{o}_r = (S_r^x, S_r^y)$ by the responses of Sobel-x and Sobel-y filters as approximation of the intensity gradient.
Therefore, the maximal strength of the intensity gradient is computed for each image and only pixels whose intensity gradient strength lies above a certain percentage (here 20%) of the maximal strength are considered as salient features. To reduce the number of features, this set is subsampled with the factor four by randomly erasing 75% of these edge features. As a result of this procedure, each image is represented by a set of around 200 edge features.
The shape classifier shall find a map from arbitrary translations and rotations of these edge sets back to the represented object. Therefore, the same learning approach is applied as for the artifical letter data set. In the first learning phase, AHL ($K = 100$, 10000 learning steps per learning phase) is applied separately on patterns showing five objects of the same class in randomly configuration. An example of a complete training set is shown in Fig. 7.15.
The upper and lower bounds for the control parameter Λ are plotted for each of the 20 resulting interaction functions in Fig. 7.16 a). Compared to the bounds of the artifical letter contours in Fig. 7.10 the gaps between the lower and upper bounds are significant smaller, which mirrors the higher complexity of the contours and the stronger inconsistency of the grouping problems.
As a result of this inconsistency, all interaction functions describe more or less a simple clustering of edges with object specific preferences for some edge combi-

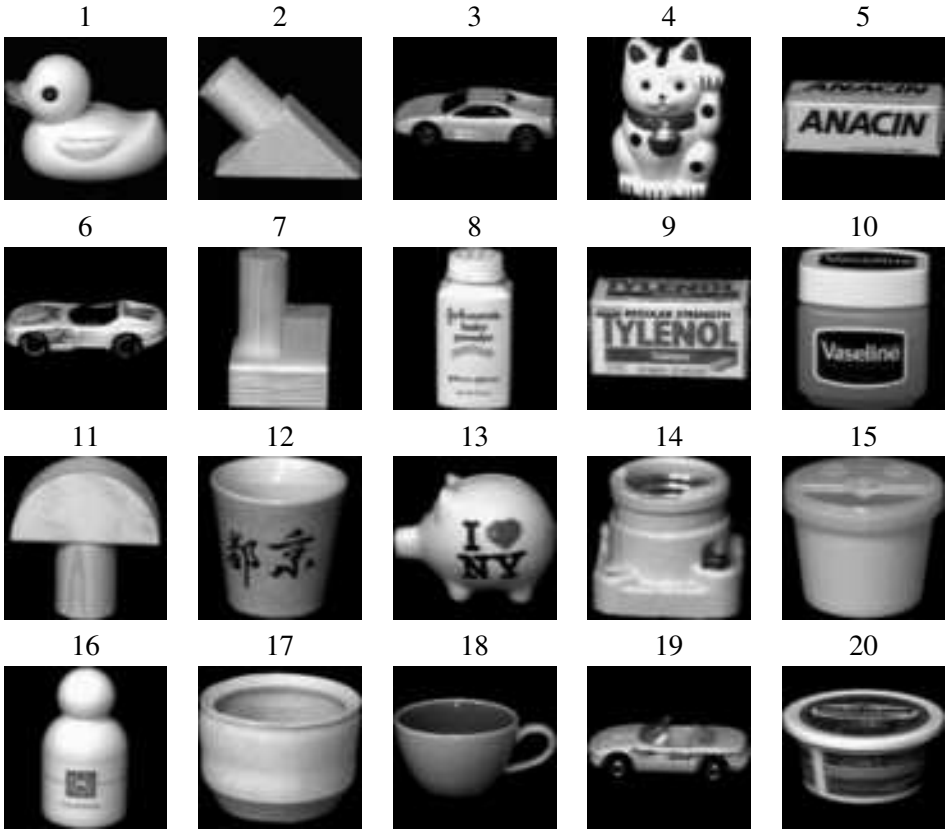


Figure 7.14: Subset of the COIL20 dataset[38].

nations (compare the bird-fish example in Fig. 6.38), e.g. inner angles around $45°$ for pattern 2 and around $90°$ for pattern 5. This effect is enhanced by the uniform choice of $\Lambda = 0.9$ for all interaction functions, which favors positive interactions. So, the figure-background separation abilities and the ability to segment overlapping objects that can be observed for the regular polygons and the artifical letter contours are clearly reduced.

The interaction functions show a high similarity, which requires a slow annealing speed. The different objects are distinguished by the stronger attraction of typical edge configurations to one of the edge clustering interaction functions.

The 20×20 E-matrix is generated by presenting single object patterns to each of the interaction functions and computing the largest eigenvalues of the resulting interaction matrices. An example of E is shown in Fig. 7.16 b).

Binary Classification: Similar to the last experiment on the artifical letter contours, this experiment starts to investigate the binary classification of the data set. Figure 7.18 shows the same benchmark visualization for the 20 class COIL20 classification problem as for the 26 class artifical letter contour problem in Fig. 7.12, with two differences. The first difference is, that the COIL20 patterns contain more

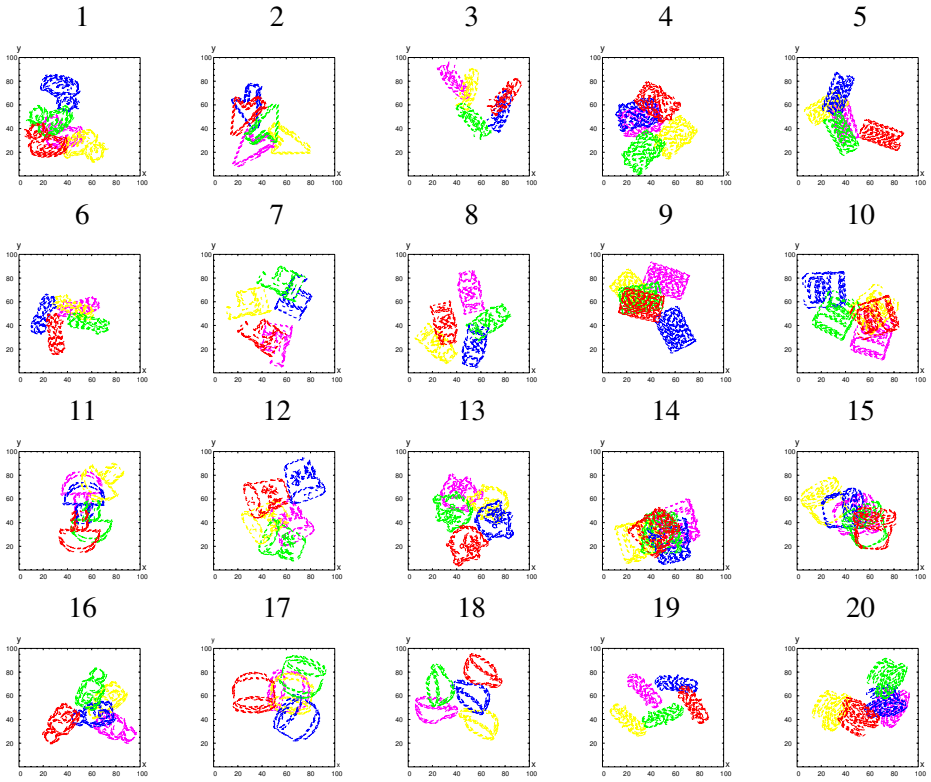


Figure 7.15: Examples of training patterns for the AHL-based learning phase applied on contours of the COIL20 data set.

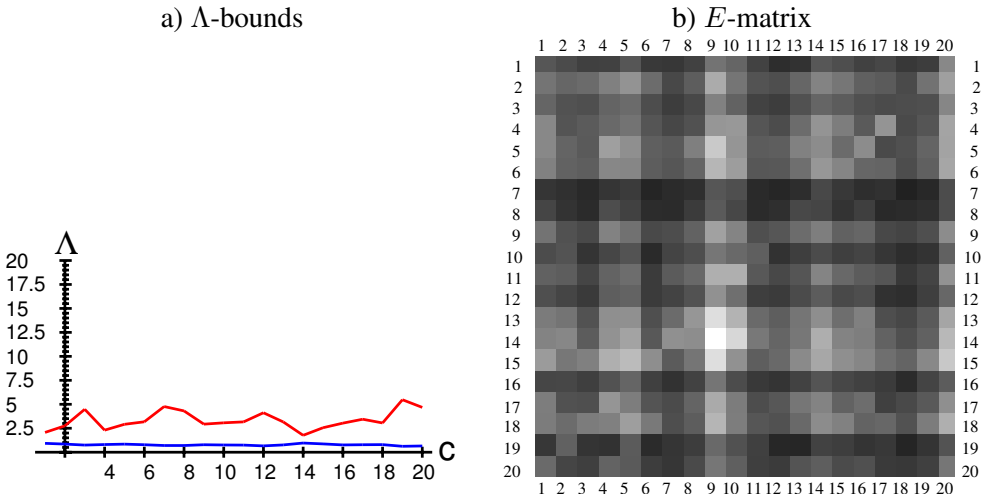


Figure 7.16: Λ-bounds and E matrix for the COIL20 dataset.

features than the artifical letter contours, a higher inner-group support is observed and, therefore, annealing starts at a higher temperature $T = 120$. The second difference is, that the columns of the squares in Fig. 7.18 do not correspond to a simple subsampling of the feature sets, but describe the degree of disturbance of the input patterns. This disturbance is realized by adding a random Gaussian distributed noise in the range of $-\sigma$ to σ intensity values per pixel to the input pattern. The range of σ goes from $\sigma = 0$ for the left most column in each square in steps of 10 to $\sigma = 90$ for the right most column. Figure 7.17 shows the effect of this disturbance on the input images and the set of extracted edges. With the increase of noise the clear edge contours degenerate to random distributions of edges, such that generalization is only possible for low levels of noise.
It is relatively easy to realize a binary classification between objects of strongly differing shape, e.g. between object 2 and 3, where the correct classification holds, even for higher levels of noise, while for similar objects, like pattern 3, 6 and 19, which all show variants of toy cars, the correct classification is only possible for very low noise and under the constraint of very slow annealing.

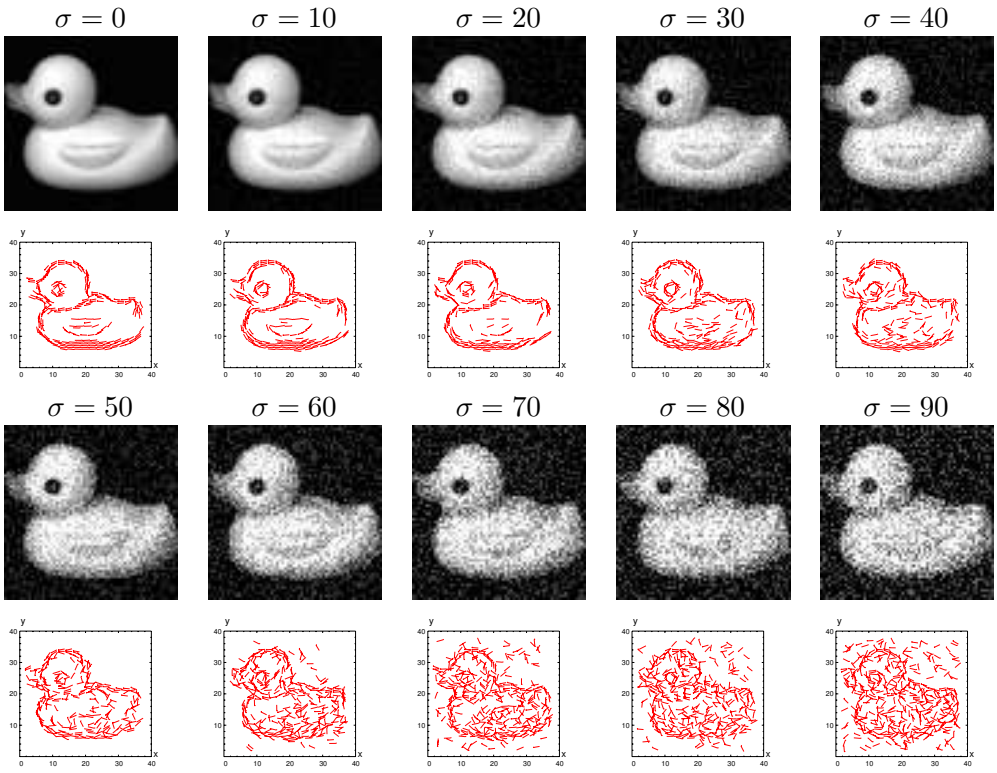


Figure 7.17: Different levels of disturbance for the COIL20 data.

Parallel Classification of all Classes: For the parallel discrimination of all classes at the same time the optimization problem (7.8) has to be solved for the E-matrix. Figure 7.24 a) shows the layer weights resulting form the matrix in Fig. 7.16 b),

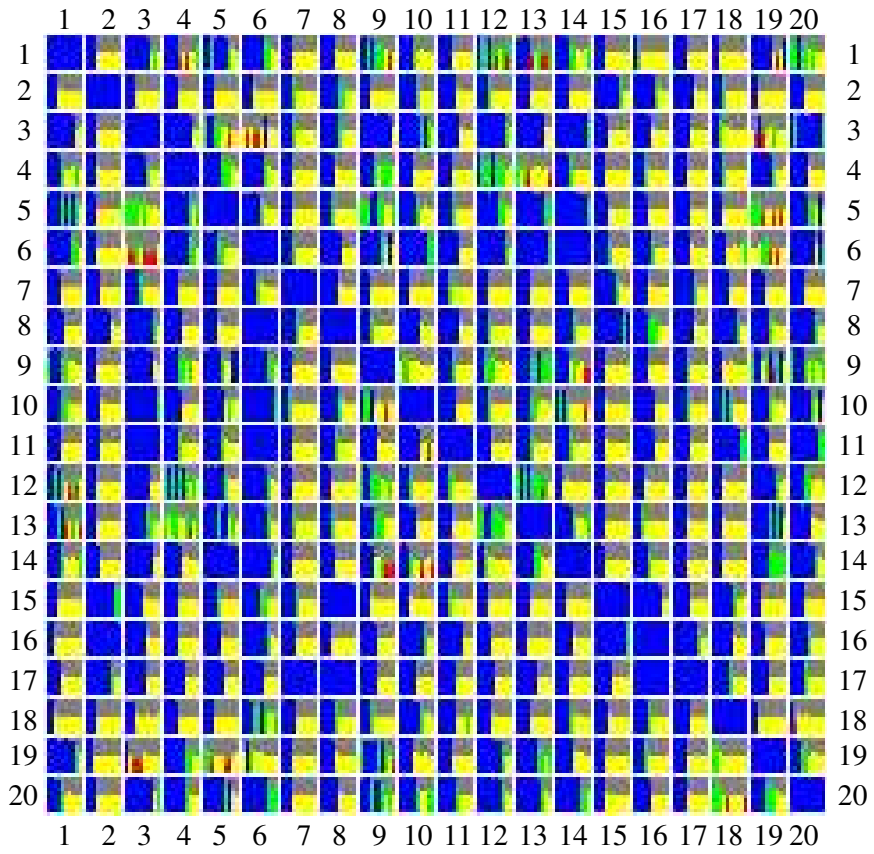


Figure 7.18: Binary classification rates for the COIL20 classification problem. Each square in the diagram shows the classification rate for the classification of two class i and j. The complete diagram shows the results of the 20×20 binary classifiers. Each column of the squares corresponds to a disturbance rate of the pattern from $\sigma = 0$ to $\sigma = 90$ (see Fig.7.17). The height of each column shows the number (from 10 test patterns (5 of each class)) of patterns that are classified correctly. The color of the columns corresponds to different classification rates: blue: 9-10 patterns correct, green: 7-8 patterns correct, yellow: 5-6 patterns correct, red 3-4 pattern correct and black 0-2 patterns correct.

if these weights scale the rows of E, the weighted E-matrix in Fig. 7.16 b) arise, where the maximum entry per column is highlighted in Fig. 7.16c). It seems, that the optimization approach has found a feasible solution for the layer weights that causes a domination of the diagonal elements.

Actually, the practical simulation on an $L = 60$-layered CLM (three layers per class) shows, that they realize the correct classification in the case of no noise and slow annealing. However, generalization is very poor, where even smallest noise causes classification errors.

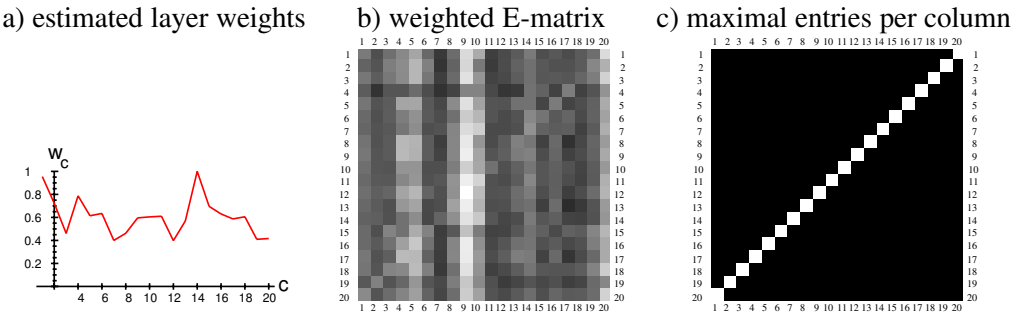


Figure 7.19: Estimation of the layer weights for the COIL20 dataset. a) Layer weights resulting from the optimization (7.8). b) weighted E-matrix. c) the maximal entry in each column of the weighted E-matrix is highlighted.

Generalization according to point of view:
The green graphs in Fig. 7.20 show the classification rate of the $L = 60$ layered CLM: The left hand side of Fig. 7.20 shows the classification rate over all 72 different views of the 20 objects in the COIL20 dataset. The right hand side plots the average classification rate over all views per object. The overall average classification rate lies at $48,2\%$ of all patterns, which describes a relative poor generalization ability compared to a simple Nearest Neighbor (NN) classifier (red graphs), which uses the 20 images in Fig. 7.14 as class prototypes and achieves in average $63,9\%$ correct classifications.
However, the NN classifier computes the distance between all pixels in the input and prototype images to evaluate the index of the nearest prototypes class, which demands an exact alignment of the input and the prototype image according to rotation and translation. In contrast, the CLM-based classifier uses only the salient edge features, concentrates on the shape properties of the objects and is invariant to a rotation or translation of the whole edge set. Thereby, the COIL20 dataset holds some hard problems for a pure shape based classifier, e.g. the objects 3, 6, and 19 all represent toy cars with similar shape, the box (object 5) and the bottle (object 8) have a (according to the rotation invariance of the classifier) simular rectangular shape and some objects, like objects 1, 2, 4, 10 and 11, show a strongly varying shape for different points of view. Additionally, the CLM classifier is confused by inner object edge configurations, like in object 12 and 13, which strongly change their relation to the object outline for different points of view. All these factors provoke, that the CLM-approach, in fact it reaches a comparable classification rate as the NN classifier for the view invariant objects 15, 16, 17, and 18, in average is not competitive to the NN approach.
The generalization abilities of the CLM based classifier can be enhanced by performing a more accurate preprocessing for feature extraction, as it is demonstrated on the right hand side of Fig. 7.21 and by the blue graphs in Fig. 7.20. The input images are preprocessed by a figure-background segmentation from an island

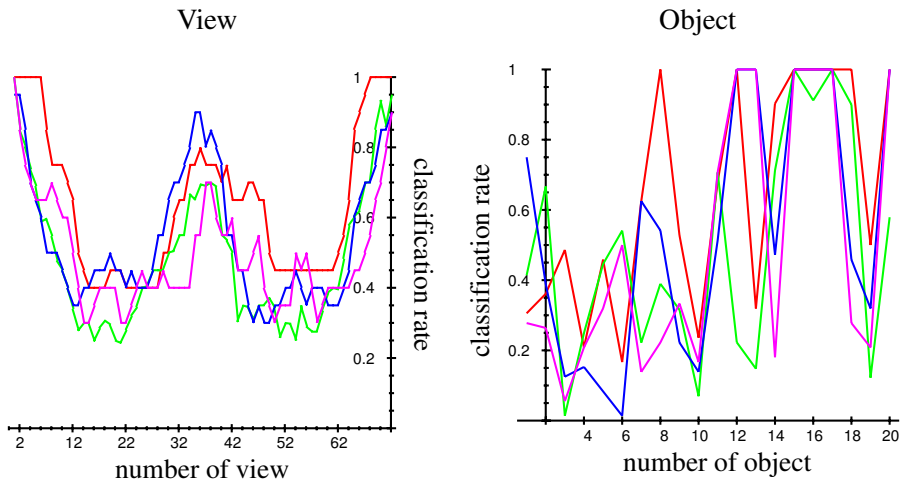


Figure 7.20: Generalization of classification rate according to point of view and type of object. Left: classification rate according to the 72 different views of COIL20. Right: classification rate according to the 20 different objects of COIL20. Red: Nearest Neighbor Classifier on view 1. Green: CLM-approach with rotation invariant basis functions applied on salient edges. Blue: CLM-Approach with rotation invariant basis functions applied on edges from the object boundaries. Magenta: CLM-approach with rotation variant basis functions applied on edges from the object boundaries.

growing method, applied on the black background color at the border of the images. The region of respective object is resized by performing two subsequent morphological erosion steps and edge features are extracted only form the one pixel wide borderline of the remaining object region. The actual learning process is perform with the same AHL-based approach and the same set of parameters as before.

The clearer object contours relax the detection of object 1, 12, 13 and 20, which benefit from the pure outline representation. However, the differentiation of the toy cars (object 3, 6, and 19) is complicated and also the detection of the cup (object 19) is confused by the varying position of the handle. In average, the CLM outline-classifier achieves an overall classification rate of 54%, but it is still performed out by the NN classifier.

The classifier characterized by the magenta graphs in Fig. 7.20 tries to enhance the classification rate of the CLM based classifier further by giving up the rotation invariance of the proximity space. This is done by extending the four dimensional proximity space, sketched in Fig. 6.29, by a fifth dimension describing the angle θ_4 between the connecting vector $\mathbf{d} = (\mathbf{p}_r - \mathbf{p}_{r'})$ and the vertical vector $(1, 0)^T$ (see Fig. 7.22).

This strategy simplifies the differentiation of some objects, like object 5 and 8, which through the different orientation of the contours become easily separable,

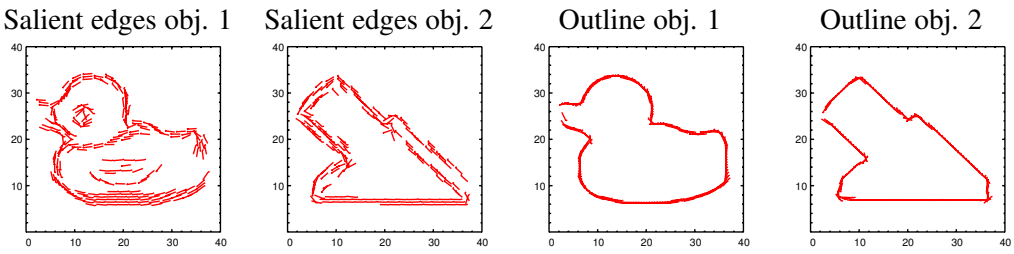


Figure 7.21: Different types of feature extractors. Left: salient edges are extracted based on intensity gradient strength. Right: edges are only extracted from the object outline, estimated by binary segmentation of the image against the black background.

but on the other side enhances the similarity between other objects, e.g. object 8 and 16. Overall, this approach decreases the classification rate to 49.3%.

Although the CLM classifiers seem to be not competitive to standard classification approaches on the COIL20 object recognition task, the experiments show, that the CLM is able to differentiate distinct object contours. A high number or similar sets of object contours makes it hard to find suitable layer weights and leads to low generalization.

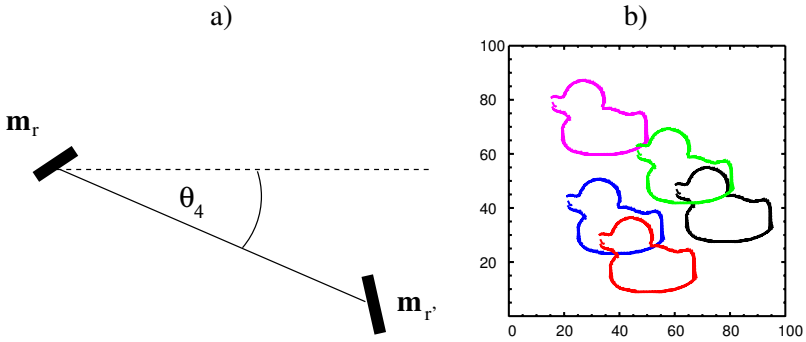


Figure 7.22: a) Rotation invariance is omitted by extending the four dimensional proximity space between the local edge features, sketched in Fig. 6.29, by the angle $\theta_4 \in [0, \pi]$ which specifies the angle between the vertical vector $(1, 0)^T$ and the connecting vector $\mathbf{d} = (\mathbf{p}_r - \mathbf{p}_{r'})$ of the two edges. b) example of a corresponding training pattern.

7.4 On-Line-Learning of the Layer Weights

It was shown, that the optimization problem (7.8) can be solved to find suitable layer weights for the COIL20 classification problem. Through the high number of constraints, solving (7.8) can be very time consuming for high number of different classes C. In this section, an alternative approach is tested that works without computation of the E-matrix and is motivated by an on-line variant of the perceptron learning rule.
At the beginning, all layer weights are set equally to 1: $\forall i = 1, \ldots, C: \quad w_i := 1$. Now, a random sequence of patterns, each showing a single randomly selected object $c^{target} \in \{1, \ldots, C\}$, is presented iteratively to an $L = C$-layered CLM, where each interaction function is applied in one layer. If the classification output c^{CLM} of the CLM corresponds to the index of the presented class c^{target}, the layer weights stay unchanged. Otherwise, the layer weights have to be modified to prevent similar classification errors in following steps, where it is known, that $w_{c^{target}}$ is too small against $w_{c^{CLM}}$.
An naive method, that can be associated as kind of error-correction rule might be the increase of $w_{c^{target}}$ and the decrease of $w_{c^{CLM}}$, e.g. by

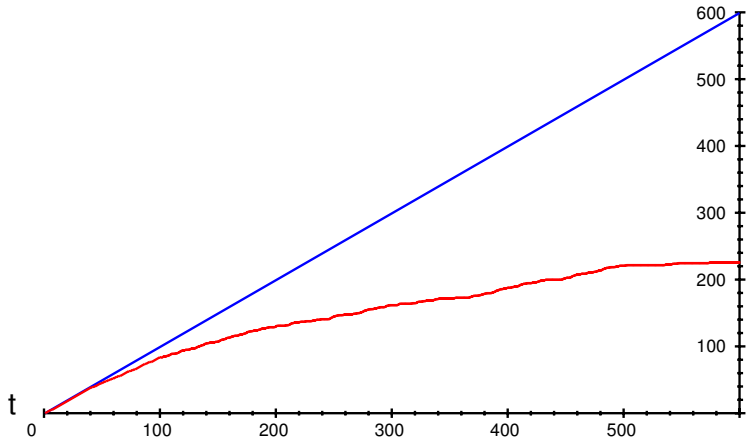
$$w_{c^{target}} := w_{c^{target}} + 1 \tag{7.10}$$

and

$$w_{c^{CLM}} := w_{c^{CLM}} - 1. \tag{7.11}$$

However, the constraint $\forall i: \quad w_i > 0$ has to be fulfilled, such that the learning rule is restricted only to (7.10). If there exists a feasible solution for the layer weights, it should be reached with a finite number of weight modifications (7.10).
Figure 7.23 a) plots, how the number of weight modifications (red) develops against the number of presented objects (blue) for a random sequence of 600 patterns. At the beginning, of the learning process, most of the objects are classified incorrectly, which results in a high proportion of weight changes according to the presented patterns. After the weighting between some of the layer classes is adjusted correctly, the number of weight changes increases slower than the number of presented patterns. If there exists a feasible weighting between all classes, the layer weights reach a stable state, where no further weight changes are necessary. However, if no feasible solution exists there is a characteristic non vanishing increase of the number of weight changes, because the learning rule disturbs correct weightings between some classes that are reached by earlier weight updates.
The disadvantage of this approach is, that it can not be predicted whether it converges to a feasible solution, and, that each pattern presentation needs a simulation of the CLM-dynamics, which means in this case 1000 annealing steps with a uniform step size from $T = 120$ to $T = 0$ and 4000 neuron updates per annealing step.
A faster annealing speed acts like a blurring of the output classes, because objects can be assigned to suboptimal layers, such that the area of feasible layer weights

a) learning with fixed annealing speed



b) learning with variable annealing speed

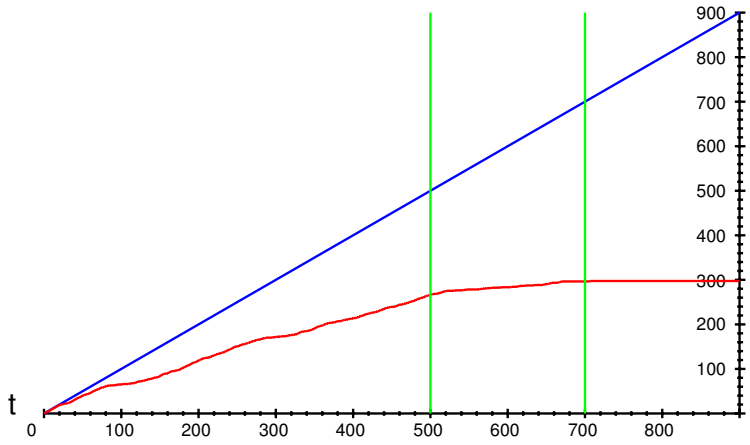


Figure 7.23: Plots of the number of weight modifications (red) against the number of presented patterns (blue). The green lines in run b) mark steps, where the annealing speed is reduced by the factor 10.

is restricted stronger. Nevertheless, a faster annealing speed can be interpreted as faster, but coarser learning rate at the beginning of learning that drives the layer weights close to the feasible region. After the gradient of the weight changes does not decrease any more, the annealing speed can be decreased to achieve a slower, but more exact learning rate.

Figure 7.23 b) shows a number of weight modifications for this strategy. The first 500 pattern presentations are performed with 100 annealing steps from $T = 120$ to $T = 0$. The gradient of weight changes decreases after 70 and 380 pattern presentations. However, after the second decrease, the gradient still describes a

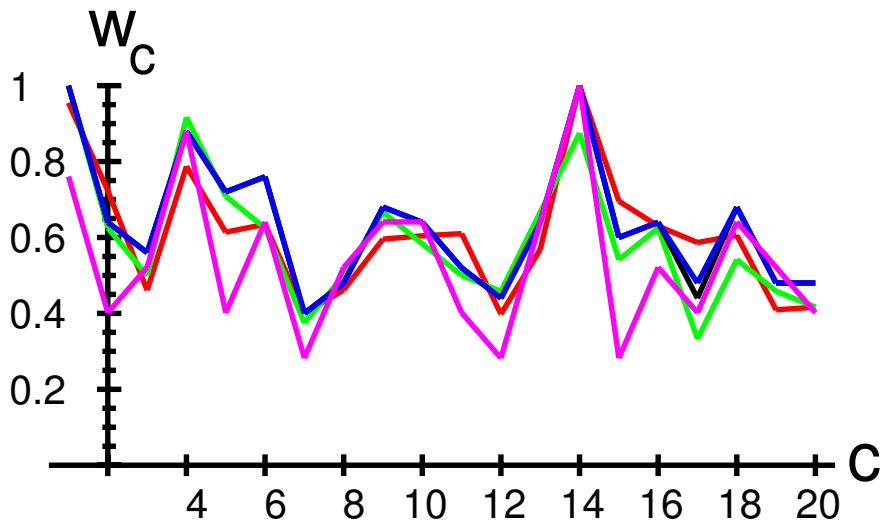


Figure 7.24: Comparison of the layer weights resulting from the optimization of the E-matrix and the on-line learning rule at different time steps. Red line: optimization of the E-matrix; Magenta: on-line learning rule after 600 steps with slow annealing; Green: on-line learning rule after 500 steps with fast annealing; Blue: on-line learning rule after 500 steps plus 200 steps at slow annealing; Black: on-line learning rule after 700 steps plus 200 steps at very slow annealing.

linear increase of the weight modifications, compared to the number of presented patterns. After decreasing of the annealing speed for the next 200 pattern presentations, the gradient of the number of weight updates decreases significantly, but it is hard to say, if it has vanished after the first 700 pattern presentations.

The learning rule is performed for 200 additional steps with an even slower annealing speed of 10000 annealing steps with a uniform step size from $T = 120$ to $T = 0$. Actually, the average gradient of weight updates vanishes and the layer weights reach feasible values for the classification task.

It can be concluded, that the adaptation of the layer weights can be realized by a kind of error-correction on-line learning rule, where the speed of annealing during the simulation of the CLM-dynamics can be interpreted as blurring factor on the output classes.

Figure 7.24 compares the layer weights resulting from the optimization (7.8) of the E-matrix in Fig. 7.16 b) with the layer weights at different stages of the investigated on-line learning rule, where each solution if normed to $\max_i w_i = 1$.

The optimization of the E-matrix (red line) returns high weights for pattern 1, 4 and 14 and low weights for pattern 3, 7, 12, 19 and 20. The magenta line shows the layer weights after the 600 pattern presentations of the first learning run of the on-line learning rule. The structure of these weights is similar to the first result,

e.g. it also shows high weights for pattern 1, 4 and 14 and low weights for pattern 7 and 15, but there are significant deviations between the two solutions. The green line shows the layer weights after the first 500 learning steps of the second run of the on-line learning rule, which was expected to find only a rough approximation of a feasible solution. This result is even more similar to the optimization of the E-matrix (red) than to the first run of the on-line learning rule (magenta).
After decreasing the annealing speed for the next 200 learning steps, the layer weights (blue line) are only fine tuned. During the last 200 learning steps at very slow annealing, only a single adaption of the weight w_{17} is performed (black line). After that, all weights stay constant, which is indicated by the fact that the black line is covered by the blue line.

7.5 Summary

In this chapter, it was investigated how the CLM behaves, if several layer specific interaction functions are applied simultaneously. Layer weights were used to scale the interaction functions against each other. Two methods were presented to estimate these layer weights. One that optimizes the maximal eigenvalues of interaction matrices resulting from the application of the interaction functions on single object patterns, and a second that performs a kind of on-line error-correction rule by classifying a random sequence of single object patterns.
It was shown, that class specific layers can be used to construct simple shape classifiers for artifical letter contours and a subset of the COIL20 data set.
The advantages of such a classifier are, that it is rotation invariant, because of the special properties of the predefined proximity functions, it can deal with incomplete data of data of variable pattern length, and, that interaction functions for different classes can be trained independently, such that they can be added or removed by adding or removing the corresponding layers within the CLM.
The disadvantages are a high memory demand for the computation and storage of the different interaction matrices and a high computation time, because slow annealing speed is acquired to discriminate information form similar interaction matrices. Further, there may be no feasible solution for the layer weights, if many classes have to be discriminated, or, if there is a high similarity between the classes. In this case, the classification network has to be divided into several subclassifiers. As last disadvantage, it has to be recapitulated, that for more complex patterns, like the COIL20 data set, all interaction functions highly react on random clusters of edges, such that a rejection of unknown patterns or parts from noisy background and a segmentation of overlapping objects becomes difficult.
The learning and classification approach is restricted to a uniform segmentation control parameter Λ for all interaction functions. A more sophisticated, but also more complicated learning method could try to optimize class dependent Λ parameters $\Lambda_1, \ldots, \Lambda_C$ and layer weights $w_1, \ldots, w_C$ in parallel, may be in an on-line learning rule with an adaptive learning rate, represented by changing the annealing

speed for the simulation of the CLM-dynamics during learning. The higher number of Λ parameters could be used to enhance the differences between the interaction functions, which allows a faster annealing speed.

Chapter 8

A Model for Attention

In the last chapter, it was shown, that discriminators for global object properties, like shape, can be build from dynamical and highly redundant application of pairwise relations between elementary features. The dynamical character of the CLM binds these elementary features to groups which can be interpreted as higher order features.

In this chapter, it is investigated, if the grouping or classification process of the CLM can be influenced by extending it with a model of attention.

8.1 Experiences from Eye-Tracker Experiments

The approach is motivated by results from eye-tracker experiments made on ambiguous images. Figure 8.1 a) shows an ambiguous image that was investigated by Pomplun in [46]. It is a part of an illustration from the artist M.C. Escher, which can be either interpreted as black devils an a white background or white angels on a black background.

Pomplun traced the gaze trajectories of people, who perceived either the one or the other interpretation. The gaze distributions for the two different interpretations are visualized in Fig. 8.1 b) and c), where the image regions of high gaze frequency are displayed by high intensity, while low gaze frequency is displayed by low intensity.

People, who perceived the angels as foreground, mainly focus the heads of the angels and people, who perceived the devils are foreground, focus mainly the heads of the devils. Pomplun took these results as motivation to construct two derivations of the ambiguous image Fig. 8.1 a) by enhancing the image features at the areas of high attention in the two gaze distributions. He showed, that the two derived images were no longer ambiguous for unbiased observers, but allowed a unique map to the interpretation of the corresponding gaze distribution. This experiment can be summarized by the observation that object recognition is highly dependent on attention, where attention in this case is simply measured by gaze distribution.

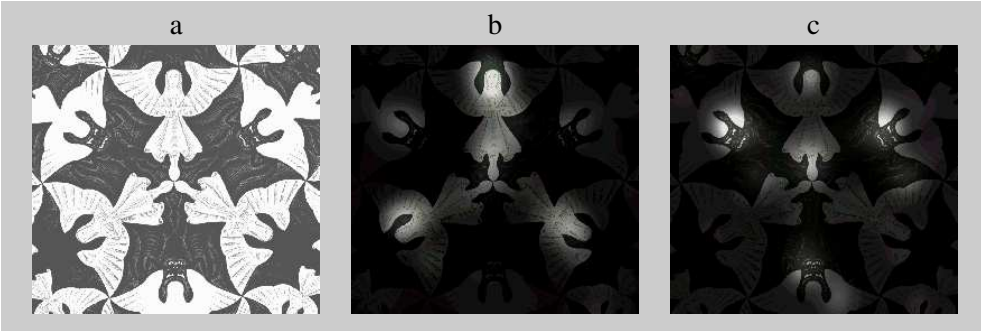


Figure 8.1: Eye-Tracker experiment on the perception of ambiguous images from [46]. a) Illustration of an optical illusion derived by M.C. Escher. It can be interpreted as white angels on a black background or black devils on a white background. b) and c) Distributions of focus points for persons, who see either the angels b) or the devils c). Areas with high gaze rate are shown in high intensity values, while areas of low gaze rate are shown in low intensity values.

8.2 Implementation of Attention

A simple method is introduced to integrate such gaze distributions, here called attention maps, into the interaction matrix of the CLM and to observe their effect on the course of the CLM- dynamics during annealing. Therefore, two results from the previous chapters are used. The first is the observation from chapter 4, that the WTA process for the groups in the input is activated in the ordering of the inner group support within the interaction matrix. The second is equation (5.2), where the desired interaction matrix was formulated as a correlation matrix for the differences between the layer vectors of the target state $\mathbf{y}$:

$$\hat{F} = \sum_{\gamma} \sum_{\nu \neq \gamma} (\mathbf{y}_\gamma - \mathbf{y}_\nu)(\mathbf{y}_\gamma - \mathbf{y}_\nu)^T. \tag{8.1}$$

Suppose a normalized attention map $\mathbf{p} = (p_1, \ldots, p_N)$ has been specified, where the components $p_r, r = 1, \ldots, N$ describe the probability that feature $\mathbf{m}_r$ is focused in the input pattern and $\mathbf{p}$ is normalized to $\max_r p_r = 1$. This attention map is applied on the layer vectors $\mathbf{y}_\alpha, \alpha = 1, \ldots, L$ of the target vector $\mathbf{y}$ by multiplication with a $N \times N$ diagonal matrix $P = \mathrm{diag}(\mathbf{p})$ of the components of $\mathbf{p}$:

$$\forall \alpha = 1, \ldots, L: \quad \mathbf{y}'_\alpha := P\mathbf{y}_\alpha \tag{8.2}$$

If now the target vectors $\mathbf{y}_\alpha$ in (8.1) are substituted by their weighted variant (8.2),

a new weighted interaction matrix arises

$$\hat{F}' = \sum_{\gamma} \sum_{\nu \neq \gamma} (\mathbf{y}'_\gamma - \mathbf{y}'_\gamma)(\mathbf{y}'_\gamma - \mathbf{y}'_\gamma)^T \tag{8.3}$$

$$= \sum_{\gamma} \sum_{\nu \neq \gamma} (P\mathbf{y}_\gamma - P\mathbf{y}_\gamma)(P\mathbf{y}_\gamma - P\mathbf{y}_\gamma)^T \tag{8.4}$$

$$= \sum_{\gamma} \sum_{\nu \neq \gamma} P(\mathbf{y}_\gamma - \mathbf{y}_\gamma)(\mathbf{y}_\gamma - \mathbf{y}_\gamma)^T P \tag{8.5}$$

$$= P\hat{F}P \tag{8.6}$$

by multiplication of $\hat{F}$ from right and left with the attention matrix P. In the following , different attention maps are applied on the pattern in Fig. 4.7 to show the effect of this modification on the CLM-dynamics.

8.3 Influence of Attention on the Annealing Process

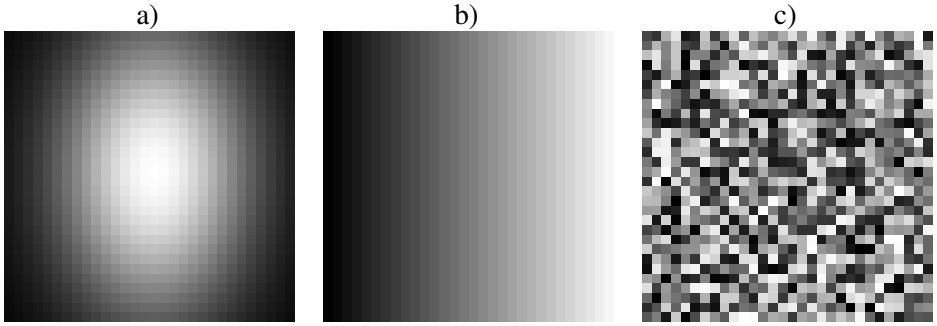


Figure 8.2: Three examples of attention maps. a) Gaussian distributed attention map $\mathbf{p}^{center}$, the focus of attention lies at the center of the input. b) Linearly increasing attention map $\mathbf{p}^{linear}$, the focus of attention lies on the right hand side of the input. c) Diffuse distribution of attention $\mathbf{m}^{random}$.

The first attention map is defined by a Gaussian distribution (see Fig. 8.2 a)) centered at the middle of the input pattern. Figure 8.3 shows the influence on the course of the annealing process, if this attention map is applied on the interaction matrix (4.6) of the pattern in Fig. 4.7.
Since the application of the attention map scales down interactions of features at the periphery of the pattern, it changes the ordering of the inner group support. In the original simulation, the WTA process was activated in the ordering of the group sizes: first the outer ring, then the inner ring, the border region, the inner circle and, finally, the five spurious features in the corners of the image. The new ordering is the inner ring, the inner circle and then the outer ring. The features outside the focus of attention are still distributed equally between the unused layers at $T = 10$ and form the background of the groups within the focus of attention.

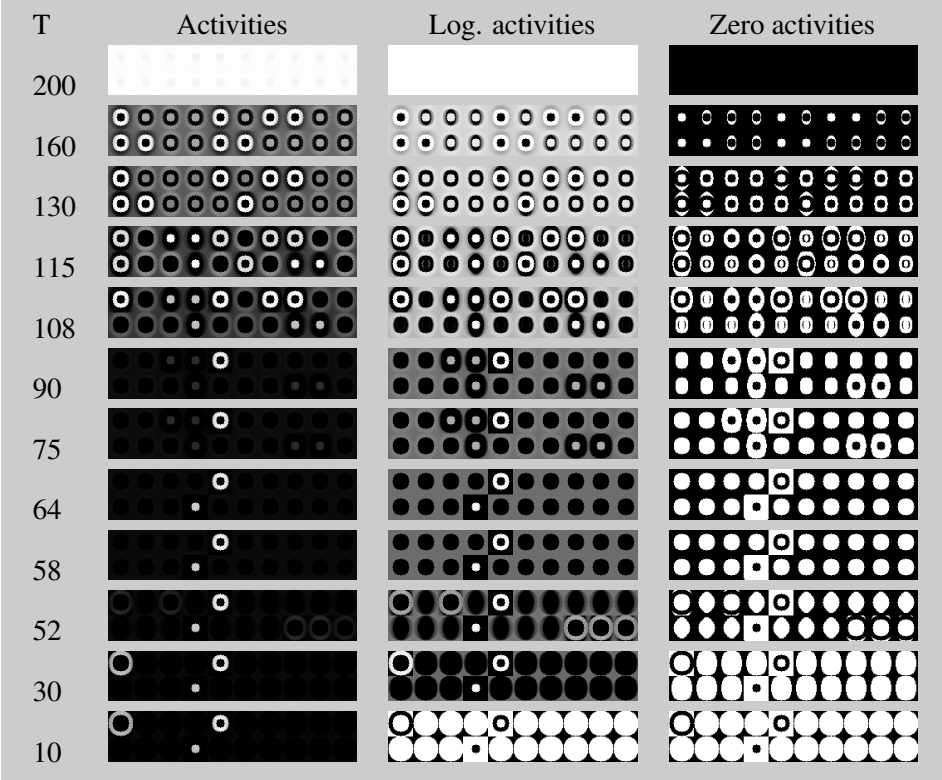


Figure 8.3: Attractor states of the CLM-dynamics during annealing. The interaction matrix is weighted according to the centered attention map in Fig. 8.2 a).

Thus the distribution of attention controls the figure-background separation of the input and defines the saliency of the observed groups according to the annealing thresholds, where they reach the WTA process.

The second attention map in Fig. 8.2 b) shows a gradual increase of attention from left to right. Such kind of attention map might be associated with a verbal instruction "take the object/group on the right". Actually, this kind of instruction makes no sense for the example pattern, because all groups have the same center of mass at the middle of the pattern. But the course of the annealing process in Fig. 8.4 gives a good description of the case, that the WTA process is activated gradually within the groups.

The split of layer directions and their orthogonalization starts at the right hand side of the groups, while the left hand side stays undecided between the layers for a long time period. This effect proceeds to the left with the decrease of T. The exemplary attractor states at $T = 140$, $T = 132$ and $T = 110$ show, that the WTA processes, in the sense of a unique assignment of the features in a group to a single layer, does not happen at a single threshold T any more. Instead, it is a more gradual process, which passes several different minima in the energy landscape of the CLM, where step by step single layers of the corresponding layer class change their attracted

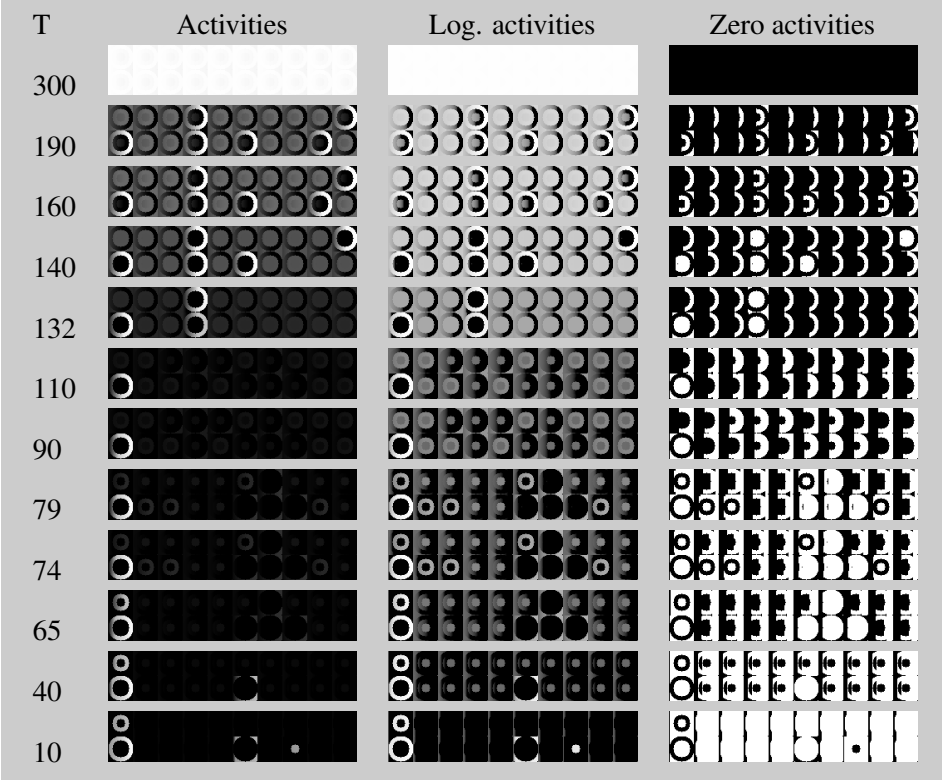


Figure 8.4: Attractor states of the CLM-dynamics during annealing. The interaction matrix is weighted according to the linear attention map in Fig. 8.2 b).

activation patters.
Since the weighting of the attention map does scale the interactions of all groups in the same way, the groups switch into the WTA process in their original ordering.
The last example of an attention map is the random noise shown in Fig. 8.2 c), which actually describes a diffuse distribution of attention. Figure 8.5 shows, that the application of such an attention map has a similar effect as the relatively strong disturbance of the ideal block diagonal interaction matrix by reseting 90% or binary switching 45% of the lateral interaction weights (compare Fig. 4.17 and 4.19). Thus a diffuse attention can also complicate the grouping process by decreasing the rate of convergence and the quality of the grouping result.

8.4 Simulation on an Ambiguous Image

As last simulation in this chapter, it is tried to model the switching effect between the interpretation of an ambiguous image, similar to the results of Pomplun [46]. Figure 8.6 sketches the approach of this experiment: The pattern *vase* from Fig. 7.1 is presented in form of its ideal block diagonal interaction matrix to an $L = 20$-layered CLM. The pattern can be interpreted either as two white faces at both sides

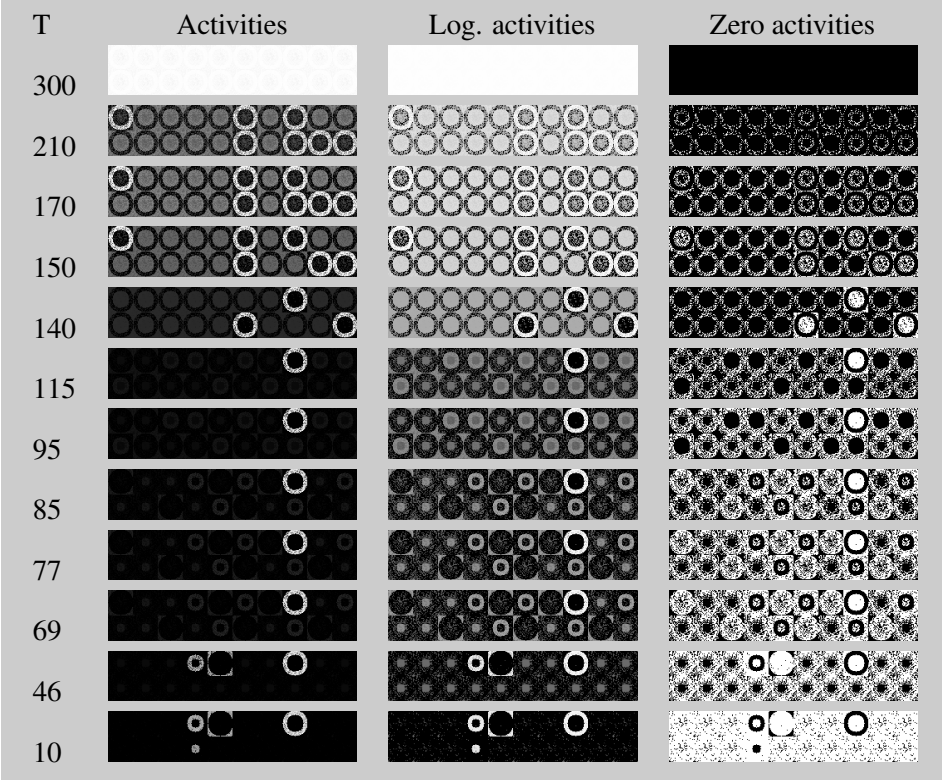


Figure 8.5: Attractor states of the CLM-dynamics during annealing. The interaction matrix is weighted according to the random attention map in Fig. 8.2 c).

of the image on a black background or a black vase on a white background. To simulate these two figure-background separations, two attention maps are applied onto the interaction matrix during the annealing process. The one describes concentration to the center of the pattern, while the other shows concentration to the periphery of the pattern. In the first case, the group of the vase reaches the WTA first and is assigned to a single layer, while the regions of the two faces stay undecided between the rest of the layers. Therefore, this attractor state can be associated with the interpretation, where the vase describes the observed object and the two faces form the background. In the second case, an attractor state is reached, where the faces are the relevant objects and the vase forms the background.
A possibility to switch between these two stages might be to choose a fixed annealing temperature and to modify the gaze distribution of the attention map, instead.

8.5 Summary

In this chapter, attention was implemented as a simple gaze distribution that can be modeled by a relevance weighting of the feature vectors for the patterns. Through the quadratic character of the interaction matrix the model results in an multipli-

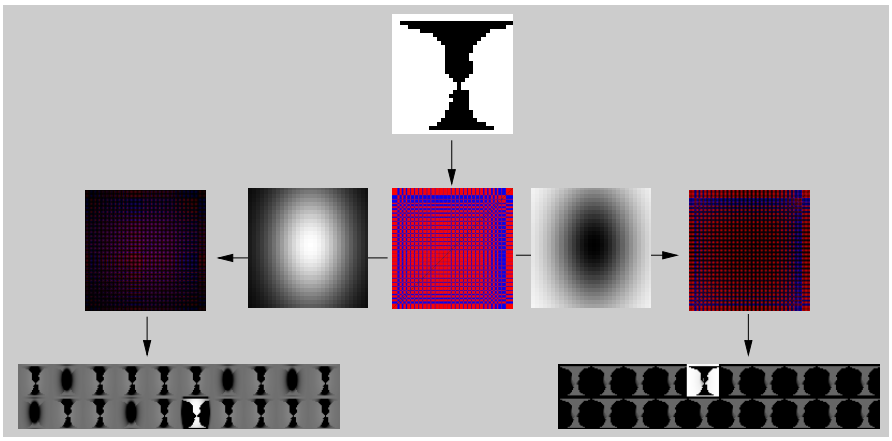


Figure 8.6: Sketch of the switching effect between the two possible figure-background separations of the pattern *vase*.

cation of the attention diagonal matrix P from left and right with the original interaction matrix F. The result of this model is, that the groups are switched into the WTA process according to the ordering of the focus of attention. Together with the interpretation from chapter 4, that groups early switched into the WTA process form relevant groups, while groups switched into the WTA process later form the background, a variable figure-background separation can be implemented dependent on the actual attention map.

Chapter 9

Variants of Implementation

If the CLM is applied to high dimensional patterns that consist of many features, e.g. real world images with several hundred pixels in each dimension, much of the computational resources must be invested into the computation of the lateral interactions $f_{rr'}$, whose number is quadratic in the number of features. Even the simple storage of such a full interaction matrix can become very memory exhausting. Therefore, in this chapter, strategies are tested to run the dynamics with a sparsely filled matrix of lateral interaction weights.

9.1 Random Sparse Support

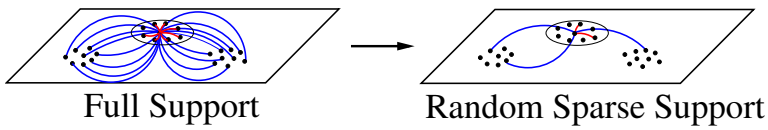


Figure 9.1: Approximation of the exact support from the full interaction matrix by summation over a sparse set of randomly selected interaction weights.

Chapter 4 showed, that the CLM-dynamics can be simulated with a sparse (90% weights erased) interaction matrix. The price to pay for this is a slower speed of convergence, a relatively low level of noise in the labeling result of small groups and a blurring of the activation thresholds of the splitting and WTA processes during self-inhibitory annealing.
In this chapter, the CLM simulation algorithm from section 3.5 is rearranged, such that it operates on random sparse interactions (see Fig. 9.1) and works without storage of fixed connections between the neurons. The idea is, that the support of the features is not computed exactly by the sum over all features at the time of the neuron update $x_{r\alpha}$, but instead each neuron iteratively collects it's support in an individual variable $s_{r\alpha}$. This individual support is collected over time by

iteratively receiving messages from randomly selected neurons in the same layer. These messages consist of the activations $x_{r'\alpha}$ and the associated feature vectors $\mathbf{m}_r$ of the other neurons. The individual support $s_{r\alpha}$ is updated on-line by the computation of the connecting lateral interaction weight $f_{rr'} = f(\mathbf{m}_r, \mathbf{m}_{r'})$ and the summation:

$$s_{r\alpha} := s_{r\alpha} + f(\mathbf{m}_r, \mathbf{m}_{r'}) x_{r'\alpha}. \tag{9.1}$$

The number of messages a neuron received is counted in an additional variable $c_{r\alpha}$. When this number exceeds a specified number c_{min} of necessary messages, the neuron is updated by using it's individual support. After that, the individual support and the counter of received messages are resetted to zero. This procedure is formulated in the following algorithm:

1. Specify simulation parameters:
   set number of annealing steps t_{max};
   set number of received messages before neuron update c_{min};

2. Specify grouping behavior:
   specify $f(\mathbf{m}_r, \mathbf{m}_{r'})$, e.g. by applying AHL;
   enter feature vectors $\mathbf{m}_r, r = 1, \ldots, N$;

3. Initialize network and the annealing process:
   choose number of layers L;
   initialize all $x_{r\alpha}$ with small random values around $x_{r\alpha}(t=0) \in [h_r/L - \epsilon, h_r/L + \epsilon]$;
   initialize all $s_{r\alpha} := 0$ and $c_{r\alpha} := 0$;
   initialize $T := 1$, $\eta := (1+\epsilon)^{-t_{max}}$ and $J := c_{min} + \max_r f_{rr} + \epsilon$;

4. Run annealing process:
   For $t = 1$ to t_{max}
   Do $N \cdot L$ times
   Choose (r, α) randomly and perform:

   (a) neuron update:
   If $(c_{r\alpha} > c_{min})$
   $\xi := \frac{J(h_r - (\sum_{\beta \neq \alpha} x_{r\beta}) + s_{r\alpha}}{(J - f_{rr}) * (1+T)}$;
   Update $x_{r\alpha} := \max(0, \xi)$, $s_{r\alpha} := 0$ and $c_{r\alpha} := 0$;

   (b) sending messages:
   For $c = 1$ to c_{min}
   Choose r' randomly and compute $f_{rr'} = f(\mathbf{m}_r, \mathbf{m}_{r'})$;
   For $\beta = 1$ to L
   $s_{r'\beta} := s_{r'\beta} + f_{rr'} x_{r\beta}$ and $c_{r'\beta} := c_{r'\beta} + 1$;

`Decrease` T `by` $T := \eta T$`;`

There are some details in this implementation that have to be emphasized. The first is, that the vertical inhibition is set to $J := c_{min} + \max_r f_{rr} + \epsilon$, ($\epsilon$ is a small positive margin variable) under the assumption, that the specified interaction function responses with a maximal values of $\max_{r,r'} f_{rr'} = f(\mathbf{m}_r, \mathbf{m}_{r'}) = 1$, which can be ensured by normalization with a suitable layer weight. Under this assumption the convergence condition for section 3.2 is fulfilled, if J is bigger than the number of incoming messages plus the self-interaction.
The second detail is, that the annealing process is modeled slightly different than in section 3.5. There the pseudo-temperature is added to the numerator of the neuron update, while in the new implementation the dominator is increased by the multiplication with the factor $(1 + T)$. However, this is no critical difference, since each temperature in one of the implementations can be translated to a unique temperature in the other implementation, e.g. for $f_{rr} = 0$ a value of $T = J$ in the original implementation corresponds to $T = 1$ in the new implementation, while $T = 0$ describes the same nominator in both implementations. The two implementations only differ in the size of the annealing steps. Thereby, it must be considered, that the stochastic character of the individual support $s_{r\alpha}$ causes some permanent noise on the attractor states in the new implementation, such that a kind of blurring effect on the activation thresholds in the pseudo-temperature T occurs. The characteristic phenomena of splitting the layer directions and the activation of the WTA process can still be observed. However, at a certain annealing temperature it can not be clearly decided whether these processes are active or not.
The last detail in the new implementation is, that all messages between the neurons in two columns r and r' are sent in parallel to avoid the multiple computation of the interaction weight $f(\mathbf{m}_r, \mathbf{m}_{r'})$. Thereby, each columns sends c_{min} messages to other columns to make it probable, that each neuron $x_{r\alpha}$ has actually received c_{min} messages, when it is selected for update.
For the new implementation, it can be concluded, that the number of weights that are computed on-line during the simulation is significantly smaller than for the computation of the full interaction matrix of the input pattern, if the number of features in the input is large. In return, more neuron updates have to be performed to reach the output grouping, because of the slower speed of convergence and the stochastic fluctuations in the support, which add a permanent level of noise to the attractor states.
The new simulation algorithm has to deal with some disadvantages, but the following example shows, that the new approach achieves suitable results in problem domains, where the size of input patterns makes the exact simulation impracticable.

9.2 Application on Color Images

The eligibility of the new simulation algorithm is shown on the five images in the first column of Fig. 9.2. These images are a subset of a benchmark data set for

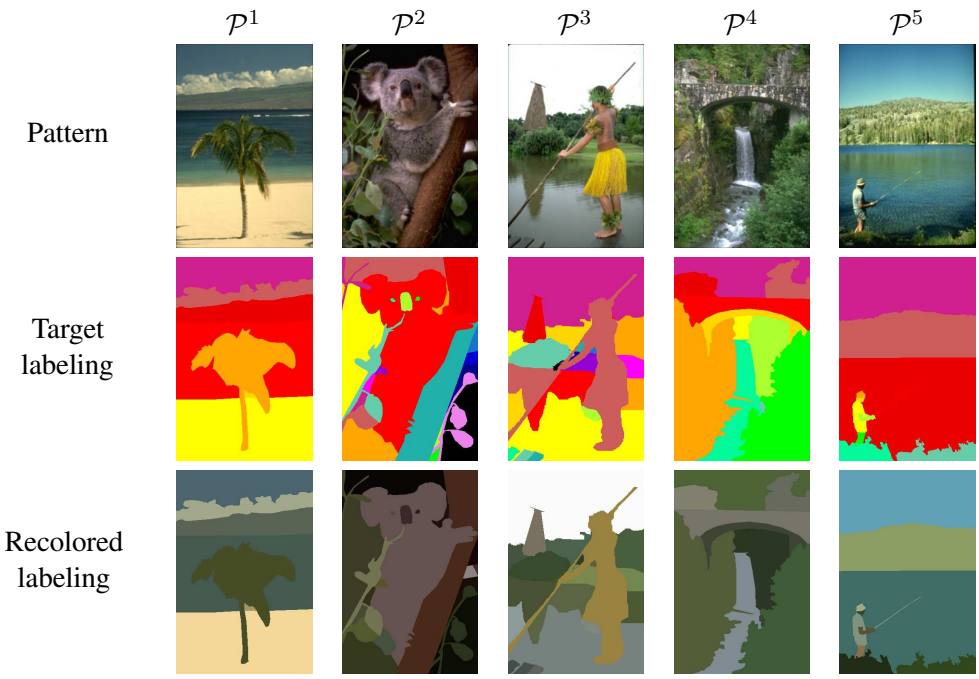


Figure 9.2: Data set for color segmentation [31].

segmentation algorithms provided by the image vision group of Berkley California [31]. Each image is accompanied by several human example labelings, where only one of these labelings defines the target labeling for the AHL algorithm. Each image consists of 321×481 pixels described by RGB color, which means a set of 154401 features and $\approx 2.38 \cdot 10^{10}$ lateral interaction weights for the full representation of the interaction matrix.
Obviously, this makes an application of the original simulation algorithm impracticable without subsampling of the input patterns. Therefore, the new implementation is applied for $t_{max} = 200$ annealing steps and the extreme case, that a neuron is updated after it has received at least $c_{min} = 1$ message to compute it's individual support.
Figure 9.3 shows results from the application on an $L = 20$-layered CLM after training AHL on each of the five example patterns. Each row is associated with a different training pattern, while the columns are associated with the test patterns. The grouping qualities of these results are plotted in Fig. 9.6 a). For training, the pixel colors are translated into the hue-saturation-intensity (HSI) color space and each pixel is represented by it's position and the three color dimensions: $\mathbf{m}_r = (\mathbf{p}_r, h_r, s_r, i_r)$. The proximity space of the feature pairs is defined by the local distance

$$d_1(\mathbf{m}_r, \mathbf{m}_{r'}) = \| \mathbf{p}_r - \mathbf{p}_{r'} \| \tag{9.2}$$

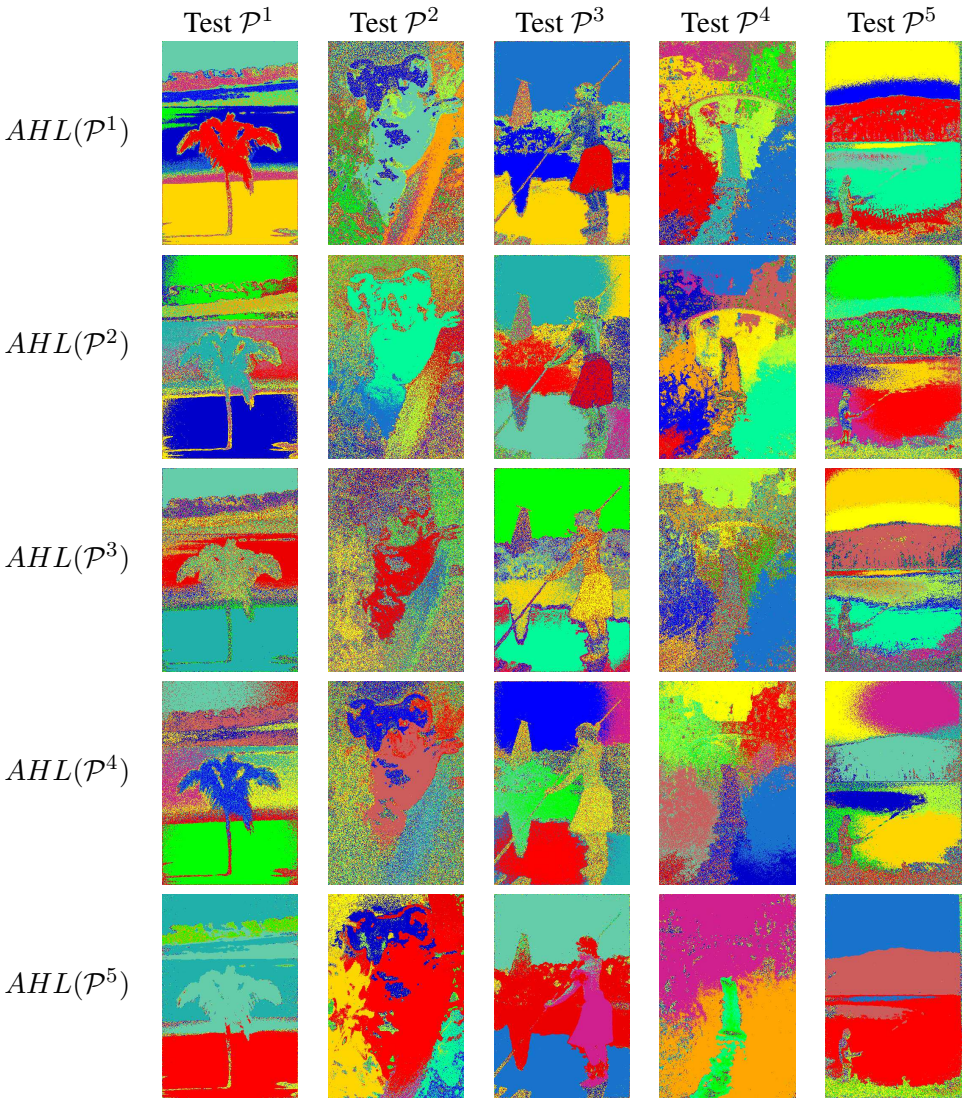


Figure 9.3: Segmentation of HSI color images. The row correspond to the training pattern and its target labeling, while columns correspond to the application of the respective test image.

and the distances in the three color dimensions

$$d_2(\mathbf{m}_r, \mathbf{m}_{r'}) = |h_r - h_{r'}| \tag{9.3}$$

$$d_3(\mathbf{m}_r, \mathbf{m}_{r'}) = |s_r - s_{r'}| \tag{9.4}$$

$$d_4(\mathbf{m}_r, \mathbf{m}_{r'}) = |i_r - i_{r'}| \tag{9.5}$$

In this four dimensional proximity space, AHL is applied with $K = 50$ basis functions and a fixed segmentation control parameter $\Lambda = 1$ for all simulation runs. A remarkable adaptation to the color segmentation of the respective training pattern can be observed in the segmentation results, e.g. the coherent labeling of the palm leaves in pattern one or the koala in pattern two are hard to reproduce with a standard split and merge segmentation algorithms, like the Color Structure Code (CSC). In terms of the generalization, the same grouping behavior between pattern one and three can be achieved. For the other combinations of training and test pattern, strong differences between the target and output labeling are visible, but most of the segmentation errors result from an over- or under segmentation of the test pattern, such that the modification of the control parameter Λ can enhance the grouping result.

9.3 Entropy-based Attention Map

The states of the CLM underlie a permanent fluctuation, because the random sparse support introduces a serious amount of noise onto the CLM dynamics. Therefore, it is not easy to check whether the dynamics has reached a stable output labeling or not and to define a suitable abort criterion for it. A possible strategy might be to simply perform a predefined number of annealing respectively neuron update steps, but as the segmentation outputs in Fig.9.3 show, it is non trivial to specify suitable iteration numbers, because big and coherent groups, like the beach in pattern 1 and the sky in pattern 3 and 5, are switched relative fast into the WTA process, while smaller and diffuse regions, like the border line between beach and sea in pattern 1 or the rock in the background of pattern 3, are switched later into the WTA process or never reach it.

However, in image processing big and coherent regions often form some kind of background and small, but salient regions often show interesting objects. This means, that after the bigger groups are manifested in their layers most of the computation time is wasted on these regions, while the smaller, more interesting regions manifest on a much slower time scale.

As the last chapter has shown, the weighting of the interaction matrix with an attention map can enhance the formation of smaller groups, if they are in the focus of attention.

Since the neurons exchange messages to collect their individual support, the attention based weighting of the connections can be implemented by changing the frequencies of sending messages for the columns of the CLM. However, if the focus of attention is changed, the groups that were in the focus of attention before

may leave the WTA process and can be driven backwards to an undecided assignment by the DC-modes of the CLM-dynamics. To prevent this, a heuristics is introduced that removes features that are not in the focus of attention from the dynamics by freezing the respective columns of the CLM. This is done practically by setting the update probabilities of the columns as well as the frequencies of sending messages to the components of the attention map. These modifications of the simulation algorithm brings us to the following reformulation of the implementation (modifications are highlighted):

1. Specify simulation parameters:
   set number of annealing steps t_{max};
   set number of received messages before neuron update c_{min};

2. Specify grouping behavior:
   specify $f(\mathbf{m}_r, \mathbf{m}_{r'})$, e.g. by applying AHL;
   enter feature vectors $\mathbf{m}_r, r = 1, \ldots, N$;

3. Initialize network and the annealing process:
   choose number of layers L;
   initialize all $x_{r\alpha}$ with small random values around $x_{r\alpha}(t=0) \in [h_r/L - \epsilon, h_r/L + \epsilon]$;
   initialize all $s_{r\alpha} := 0$ and $c_{r\alpha} := 0$ and $p_r(0) := 1$;
   initialize $T := 1$, $\eta := (1+\epsilon)^{-t_{max}}$ and $J := c_{min} + \max_r f_{rr} + \epsilon$;

4. Run annealing process:
   For $t = 1$ to t_{max}
   update attention map $\mathbf{p}(t)$;
   Do $N \cdot L$ times
   choose random variable k for [0,1];
   repeat choose column r randomly until $p_r(t) > k$;
   Choose α randomly and perform:

   (a) neuron update:
   If $(c_{r\alpha} > c_{min})$
   $\xi := \frac{J(h_r - (\sum_{\beta \neq \alpha} x_{r\beta}) + s_{r\alpha}}{(J - f_{rr}) * (1+T)}$;
   Update $x_{r\alpha} := \max(0, \xi)$, $s_{r\alpha} := 0$ and $c_{r\alpha} := 0$;

   (b) sending messages:
   For $c = 1$ to c_{min}
   choose random variable k from [0,1];
   repeat choose column r' randomly until $p_{r'}(t) > k$;
   compute $f_{rr'} = f(\mathbf{m}_r, \mathbf{m}_{r'})$;

```
        For β = 1 to L
            s_{r'β} := s_{r'β} + f_{rr'} x_{rβ}  and  c_{r'β} := c_{r'β} + 1;
    Decrease T by T := ηT;
```

Thereby, the following heuristics is applied for the update of the attention map: It starts with a uniform attention of $\forall r: \quad p_r(0) = 1$ and estimates the up-to-now output labeling of the features after each annealing step by:

$$\forall r: \ \hat{\alpha}(r,t) := \text{argmax}_\alpha x_{r\alpha}(t). \tag{9.6}$$

Then it computes the entropy of the output labels in the (here 3×3) neighborhood $\mathcal{N}(\mathbf{m}_r)$ by

$$\forall r: \quad e_r(t) = \sum_{\alpha=1}^{L} P(\alpha = \hat{\alpha}(r',t)|\mathbf{m}_{r'} \in \mathcal{N}_r) log P(\alpha = \hat{\alpha}(r',t)|\mathbf{m}_{r'} \in \mathcal{N}_r), \tag{9.7}$$

where $P(\alpha = \hat{\alpha}(r',t)|\mathbf{m}_{r'} \in N(\mathbf{m}_r))$ is the probability of label α in the neighborhood of feature $\mathbf{m}_r$, and updates the attention map with the normalized entropy map

$$\forall r: \quad p_r(t) = e_r(t) / \max_r e_r(t). \tag{9.8}$$

The effect of this heuristics is, that at the beginning of the dynamics the attention is distributed over all features, because the neuron activities are initialized randomly. When the dominant groups reach the WTA process, they start to form regions of coherent labels, which decreases the labeling entropy in these regions and removes them from the CLM-dynamics. The entropy of labeling stays high only at the borderline of homogeneous regions, such that the activities on the borderline are updated and activities within the groups are frozen.
Figure 9.4 shows, how the output labeling of the CLM changes, if the entropy-based attention map is applied to control the simulation process. The learning and simulation parameters of the CLM are the same as in the first simulation run. Figure 9.5 shows the corresponding attention maps of these output labelings and 9.6 b) shows the quality values of the output labelings to compare them with the results from the first simulation run a).
For some of the results, e.g. when training and testing is performed on the patterns 1 and 3, even smaller groups are assigned coherently to a single layer, where the attention map mainly describes the relevant contours in the input image at the end the simulation process. For these examples the attention heuristics brings a speedup of the grouping process. However, the gain of quality is too small to have a reasonable effect on the quality plot.

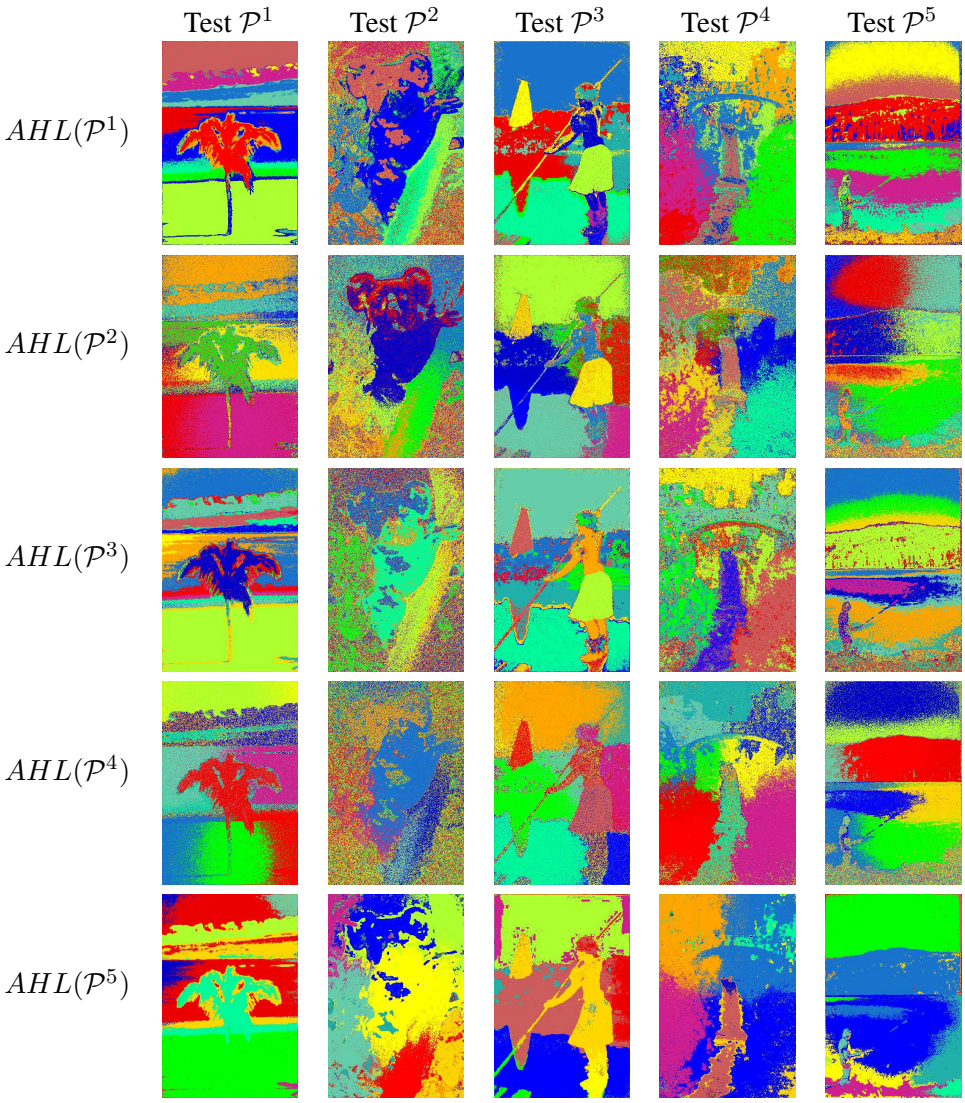


Figure 9.4: Segmentation result under control of the CLM-dynamics with the entropy map (9.7).

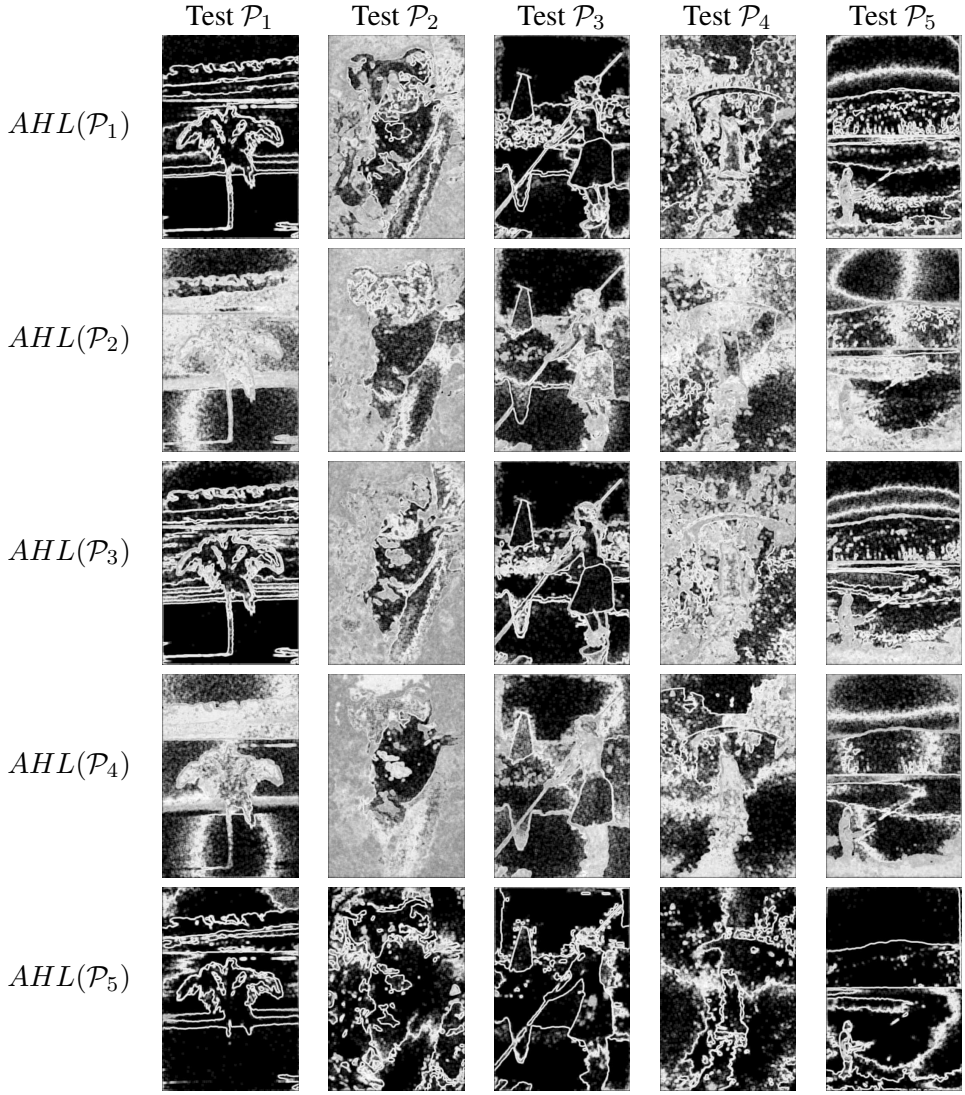


Figure 9.5: Entropy maps of the output labelings in Fig. 9.4.

More striking is the relative strong loss of quality for other patterns, especially, when training is performed with pattern 2 and 4, where some of the bigger regions are split into several regions with a diffuse separation line. This effect can be explained by the fact, that long ranged inhibitory interactions and short ranged excitatory connections exist between the features within these groups. In the first simulation, the excitatory interactions in the interior of the groups are stronger than the long range inhibitory connections between the opposite border regions, such that the whole groups are assigned to the same layer.
In the case of the entropy-based attention control, most of the inner group excitatory connections are removed and the long ranged inhibitory connections dominate, such that the groups are split to different layers of the CLM. Since there are still short ranged positive interactions between features on both sides of the separation lines, there is a random exchange of features between these layers, which causes the diffuse junction of the labels.

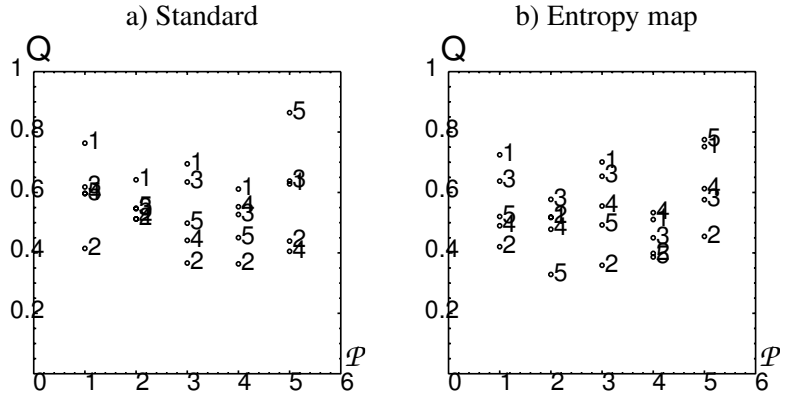


Figure 9.6: Quality of the grouping results in Fig. 9.3 (a), where the dynamics is simulated without the control of the entropy map (9.7), and Fig. 9.4 (b), where the dynamics is influenced by the entropy map.

9.4 Summary

The removal of inner group features enhances the effect of over segmentation. Consequently, the heuristics of attention based control of the CLM dynamics only enhances the grouping performance, if a suitable adjustment between the update rate of the attention map, the segmentation control parameter Λ and the annealing speed of the CLM-dynamics can be found. However, the results have shown, that the strategy of random sparse support makes the CLM, in principle, applicable on high dimensional patterns without a subsampling of the feature vectors.

Chapter 10

Conclusion and Outlook

In this work, perceptual segmentation and grouping processes, like they can be described by the Gestalt Laws, were implemented with the Competitive Layer Model. Several grouping principles were encoded in the excitatory and inhibitive lateral interaction weights of the network, while the vertical competition weights implemented a WTA process between the occurring groups.

Key features of the CLM are an automatic detection of the number of present groups, an annealing process to prevent suboptimal segmentations, which is supported by theoretical knowledge about the convergence and assigning properties of the CLM-dynamics, and a general architecture that can be expanded to a wide range of grouping problems.

The goal of this work was to advance the learning approach of estimating the lateral interaction weights from hand-labeled training patterns.

The practical simulations have shown, how the output labeling arises from a successive activation of splitting and WTA processes during annealing. Thereby, this behavior was very robust against random noise in the lateral interactions weights, because of the high redundant and dynamical character of the CLM.

From this observation the AHL algorithm was derived, which roughly approximates the distributions of excitatory and inhibitive feature-pairs within a pairwise feature proximity space to describe some kind of binary classification of the lateral interaction weights. In this term, the AHL approach competes with more accurate, but also more complex methods, like the SVM.

The advantages of AHL are a simple and fast implementation and the direct control of the complexity and the segmentation level of the learned interaction function. AHL was successfully applied to the problems of point clustering, contour grouping and texture, color and fluorescence cell image segmentation. Generalization properties were guaranteed by external knowledge from the predefined proximity spaces.

Although AHL represents a significant simplification of the learning process, it still leaves some open problems. One is the automatic estimation of the segmentation control parameter Λ. Another is the still extensive effort for the user in designing

suitable feature and proximity spaces.
An approach to this problem might be the combination of the AHL algorithm with the concept of evolutionary learning. A set of general feature and proximity properties might be encoded as chromosomes that are combined to feature and proximity spaces. The fitness of these spaces could be evaluated very efficiently by computing the classification rate of an AHL-based interaction function according to the target labeling.
Since the AHL algorithm shows robustness against errors in the target labeling, also some kind of bootstrapping strategies might be interesting, where the target labeling as well as the features and proximity functions is evolved from random initializations.
If several interaction functions are applied in parallel within the CLM, the WTA behavior enables a classification of different types of object shapes by a kind of perceptron approach. This approach works quite good for clear and simple shapes, like artifical letter contours. For more complex shapes, like the COIL20 data, the situation gets more complicated, where the mere bottom-up grouping process based on the pairwise feature relations has more problems to describe global object properties. Here the estimation of class specific control parameters Λ_c, $c = 1, \ldots, C$ combined with the adaption of the layer weights might be useful to differentiate the competing interaction functions from each other.
Another approach may be to introduce higher order feature relations or top-down backward loops to the CLM architecture. An exemplary scenario of this approach on the detection of circular objects may be to start with the original CLM-dynamics on an AHL-learned interaction function. After the groups have manifested in different layers, some kind of super-features could be extracted from the groups, e.g. the average distance of the features to the center of the groups, and the interaction weights of the elementary features could be modified according to these super-features, e.g., such that all features whose distance to the center deviate drastically are reduced to remove them from the respective group, similar to the application of the attention maps in this work. Such an architecture would require a temporal coordination of the annealing process and the weight modifications from the adaptation of the segmentation control parameter, the application of attention maps and the influence of global top-down information.
One of the main drawbacks of the practical simulation of the CLM-dynamics was the resource demand for the high number of interaction weights. Therefore, further works should try to speed up the simulation algorithm, e.g. by an efficient hardware implementation or a massive parallelization of the actual simulation algorithm. A far goal would be to apply the CLM to image sequences in real time, such that the CLM can trace objects or groups over time by recycling the information from previous grouping results. Together with additional mechanisms for storing information, like position, orientation, type and velocity about perceived objects such an architecture might be developed to a complex model for perception.

Appendix A

Example for the Construction of the Interaction Matrix

Six arbitrary features $\mathbf{m}_r, r = 1, \ldots, 6$ shall be separated into three groups containing three, two and one features. This problem can be described by assigning each feature to a label $\hat{\alpha}(r) \in 1, 2, 3$, where it is assumed, that the features are ordered according to their label which results in the labeling vector:

$$\hat{\alpha}^{ord} = \begin{bmatrix} 1 & 1 & 1 & 2 & 2 & 3 \end{bmatrix}^T. \tag{A.1}$$

Using this labeling in (3.36), a three layered target state of the CLM can be generated containing the layer vectors:

$$\begin{aligned} \mathbf{y}_1^{ord} &= \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}^T \\ \mathbf{y}_2^{ord} &= \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}^T \\ \mathbf{y}_3^{ord} &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T. \end{aligned} \tag{A.2}$$

Now, the block-diagonal correlation matrix for the difference vectors between these layer vectors can be computed according to (5.3) as

$$\hat{F} = \begin{bmatrix} 4 & 4 & 4 & -2 & -2 & -2 \\ 4 & 4 & 4 & -2 & -2 & -2 \\ 4 & 4 & 4 & -2 & -2 & -2 \\ -2 & -2 & -2 & 4 & 4 & -2 \\ -2 & -2 & -2 & 4 & 4 & -2 \\ -2 & -2 & -2 & -2 & -2 & 4 \end{bmatrix}. \tag{A.3}$$

The negative entries of $\hat{F}$ are scaled against the positive ones by the factor $\Lambda = \frac{1}{3-1}$. For any other ordering of the six features, the rows and the columns of (A.3) must be permuted in the same way as the features to make the interaction matrix consistent with the consistency conditions in equation (3.37).

Appendix B

Heuristic Constraints on the Interaction Coefficients c_j

To estimate an lower bound for the control parameter Λ, it is provided, that the sum of all left hand sides in (5.1) is smaller than zero:

$$
\begin{aligned}
\sum_{r} \sum_{\beta \neq \hat{\alpha}(r)} \sum_{r'} f_{rr'} y_{r'\beta} < 0 \qquad & (3.36) \\
\iff \\
\sum_{(\mathbf{m}_r, \mathbf{m}_{r'}) | \hat{\alpha}(r) \neq \hat{\alpha}(r')} f_{rr'} < 0 \qquad & (5.11) \\
\iff \\
\sum_{(\mathbf{m}_r, \mathbf{m}_{r'}) | \hat{\alpha}(r) \neq \hat{\alpha}(r')} \sum_{j} c_j g^j_{rr'} < 0 \qquad & \\
\iff \\
\sum_{j} c_j \sum_{(\mathbf{m}_r, \mathbf{m}_{r'}) | \hat{\alpha}(r) \neq \hat{\alpha}(r')} g^j_{rr'} < 0 \qquad & (5.17) \\
\iff \\
\sum_{j} c_j \sum_{(\mathbf{m}_r, \mathbf{m}_{r'}) \in V_j | \hat{\alpha}(r) \neq \hat{\alpha}(r')} 1 < 0 \qquad & (5.19) \\
\iff \\
\sum_{j} c_j c_j^- < 0 \qquad & \\
\iff \\
\mathbf{c}^T \mathbf{c}^- < 0 \qquad &
\end{aligned}
$$

For an upper bound for Λ, it is provided, that the sum of all right hand sides in (5.1) is larger than zero:

$$\begin{aligned}
&\sum_r \sum_{\beta \neq \hat{\alpha}(r)} \sum_{r'} f_{rr'} y_{r'\hat{\alpha}(r)} > 0 && \overset{(3.36)}{\iff} \\
&(L-1) \sum_{(\mathbf{m}_r, \mathbf{m}_{r'}) | \hat{\alpha}(r) = \hat{\alpha}(r')} f_{rr'} > 0 && \overset{(5.11)}{\iff} \\
&(L-1) \sum_{(\mathbf{m}_r, \mathbf{m}_{r'}) | \hat{\alpha}(r) = \hat{\alpha}(r')} \sum_j c_j g^j_{rr'} > 0 && \iff \\
&(L-1) \sum_j c_j \sum_{(\mathbf{m}_r, \mathbf{m}_{r'}) | \hat{\alpha}(r) = \hat{\alpha}(r')} g^j_{rr'} > 0 && \overset{(5.17)}{\iff} \\
&(L-1) \sum_j c_j \sum_{(\mathbf{m}_r, \mathbf{m}_{r'}) \in V_j | \hat{\alpha}(r) = \hat{\alpha}(r')} 1 > 0 && \overset{(5.19)}{\iff} \\
&(L-1) \sum_j c_j c_j^+ > 0 && \iff \\
&(L-1)\, \mathbf{c}^T \mathbf{c}^+ > 0 && \overset{(L-1>0)}{\iff} \\
&\mathbf{c}^T \mathbf{c}^+ > 0
\end{aligned}$$

Appendix C

AHL Learning Algorithm

The input of the AHL algorithm is given by a set of training patterns $\mathcal{P}^i$, where each training pattern consists of a set of N^i features $\mathbf{m}_r^i$, with a target labeling $\hat{\alpha}^i(r) = \hat{\alpha}^i(\mathbf{m}_r^i)$. The distances in the proximity space are computed by a set of P similarity functions $d_p(\mathbf{m}_r, \mathbf{m}_{r'})$, $p = 1, \ldots, P$.
First, the individual mean μ_p and variance σ_p of each distance function is computed on the training set:

$$\mu_p = \frac{1}{\sum_i N_i^2} \sum_i \sum_{r,r'} d_p(\mathbf{m}_r^i, \mathbf{m}_{r'}^i),$$

$$\sigma_p = \sqrt{\frac{1}{\sum_i N_i^2} \sum_i \sum_{r,r'} (d_p(\mathbf{m}_r^i, \mathbf{m}_{r'}^i) - \mu_p)^2}.$$

The distances are then normalized by their variance using

$$a_p = \begin{cases} \frac{1}{\sigma_p} & : \quad \sigma_p > 0 \\ 1 & : \quad \sigma_p = 0 \end{cases},$$

and $D_p^{irr'} = a_p d_p(\mathbf{m}_r^i, \mathbf{m}_{r'}^i)$ is defined as the normalized distance for all p, i, r, r'. The set of distance vectors $\mathbf{D}^{irr'} = (D_1^{irr'}, \ldots, D_P^{irr'})^T$ is quantized using a set of K prototypes $\tilde{\mathbf{d}}_j \in \mathbb{R}^P, j = 1, \ldots, K$. The Voronoi cell set V_j of prototype $\tilde{\mathbf{d}}_j$ is defined as

$$V_j = \{(i, r, r') | \parallel \mathbf{D}^{irr'} - \tilde{\mathbf{d}}_j \parallel < \parallel \mathbf{D}^{irr'} - \tilde{\mathbf{d}}_k \parallel \text{ for all } k \neq j\},$$

which carries the training pairs assigned to this prototype. This can be further subdivided into a set V_j^+, where features are assigned to the same layer, and V_j^-, where they are in different layers [1]:

$$V_j^+ = \{(i, r, r') \in V_j | \hat{\alpha}^i(r) = \hat{\alpha}^i(r')\},$$
$$V_j^- = \{(i, r, r') \in V_j | \hat{\alpha}^i(r) \neq \hat{\alpha}^i(r')\}.$$

[1] Feature-pairs, where both features are assigned to the background (like in Fig. 6.15) or one feature is not assigned to any label, should be treated as assigned to different labels, to prevent spurious support in the the background.

The algorithm can be performed using simple vector quantization, or the more advanced AEV algorithm for resetting unused prototype vectors. For AEV, an activation score A_j is computed for each prototype, based of the population of the Voronoi cell set V_j. Based on an activation threshold Θ_{AEV} and a given update probability P_{AEV} prototypes are then reinitialized.
The target of the learning are the coefficients c_j^+ and c_j^- of the basis interaction prototypes, derived from the vector quantization in the proximity space.
The AHL learning algorithm is implemented by:

1. Initialize $(\tilde{\mathbf{d}}_j)_p(t=0) = \mathcal{N}(a_p\mu_p, 1)$ [2];

2. For iterations $t = 0, \ldots, t_{max}$:

   (a) For all $j \in 1, \ldots, K$:
   set $\tilde{\mathbf{d}}_j(t+1) = \frac{1}{A_j(t)} \sum_{i,r,r' \in V_j(t)} \mathbf{D}^{irr'}(t)$,
   where $A_j(t) = \sum_{i,r,r' \in V_j(t)} 1$;

   (b) For all $j \in 1, \ldots, K$:
   compute $V_j(t+1)$, based on
   the new prototypes $\tilde{\mathbf{d}}_j(t+1)$;

   (c) AEV step with $P_{AEV} = 1 - \frac{i}{t_{max}}$:
   For all $j \in 1, \ldots, K$ with $\frac{A_j}{\sum_k A_k} < \Theta_{AEV}$:
   choose $s \in [0,1]$ randomly;
   if $s < P_{AEV}(t)$:
   set $(\tilde{\mathbf{d}}_j)_p \in \mathcal{N}(a_p\mu_p, 1)$;
   otherwise:
   choose another prototype $\tilde{\mathbf{d}}_n$;
   place $\tilde{\mathbf{d}}_j$ near $\tilde{\mathbf{d}}_n$:
   $(\tilde{\mathbf{d}}_j)_p \in \mathcal{N}((\tilde{\mathbf{d}}_n)_p, 0.01)$;

3. For all $j \in 1, \ldots, K$:
   compute:
   $c_j^+ = \sum_{i,r,r' \in V_j^+} 1$ and $c_j^- = \sum_{i,r,r' \in V_j^-} 1$;

4. For all $j \in 1, \ldots, K$:
   normalize:
   $c_j^+ := c_j^+ / \sum_{i=1}^K c_i^+$ and $c_j^- := c_j^- / \sum_{i=1}^K c_i^-$;

[2] $\mathcal{N}(\mu, \sigma)$ is a normal distribution with mean μ and variance σ.

The AEV step is omitted by removing step 2(c) from the algorithm. For all the simulations in this work, $\Theta_{AEV} = \frac{1}{2K}$ is chosen and eleven iterations of step 1 to 5 ($t_{max} = 10$) are performed. All summations over (i, r, r') that run over all feature pairs from the training patterns can be approximated by a randomly selected subset of N^{pairs} feature pairs. The number N^{pairs} is called the "number of learning steps per learning phase" (step 2(a) and 3) in the application examples.
The learned interaction is applied to a new test pattern $\mathcal{P}^T$ with N^T features $\mathbf{m}_r, r = 1, \ldots N^T$ by computing the lateral interactions:

$$f_{rr'} = f(\mathbf{m}_r, \mathbf{m}_{r'}) = c^+_{I(\mathbf{m}_r, \mathbf{m}_{r'})} - \Lambda c^-_{I(\mathbf{m}_r, \mathbf{m}_{r'})},$$

where $I(\mathbf{m}_r, \mathbf{m}_{r'})$ is the index of the corresponding proximity prototype

$$I(\mathbf{m}_r, \mathbf{m}_{r'}) = argmin_j \left(\sum_p (a_p d_p(\mathbf{m}_r, \mathbf{m}_{r'}) - (\tilde{\mathbf{d}}_j)_p)^2 \right).$$

Bibliography

[1] P. Bordatz (1966): Texture: A Photographic Album for Artists and Designers. Dover, New York.

[2] C.M. Bishop (1995): Neural Networks for Pattern Recognition. Oxford Press, New York.

[3] J. Canny (1986): A Computational Approach to Edge Detection. IEEE Trans. Pattern Analysis and Machine Learning 8(6): pp. 679-698.

[4] C.Y. Chang, P. C. Chung (2001): Medical Image Segmentation Using a Contextual-Constraint-Based Hopfield Neural Cube. Image and Vision Computing 19: pp. 669-678.

[5] O. Chapelle, V. Vapnik, O. Bousquet, S. Mukherjee (2002): Choosing Multiple Parameters for Support Vector Machines. Machine Learning 46: pp. 131-159.

[6] P.C. Chung, C.T. Tsai, E.L. Chen, Y.N. Sun (1994): Polygonal Approximation Using a Competitive Hopfield Neural Network. Pattern Recognition Vol. 27(11): pp. 1505-1512.

[7] P.S. Churchland, T.J. Sejnowski (1992): The Computational Brain. Cambridge, Mass., MIT Press.

[8] J.G. Daugman (1988): Complete Discrete 2D Gabor Transforms by Neural Networks for Image Analysis and Compression. IEEE Trans. on Acoustics, Speech and Signal Processing, 36(7): pp. 1169-1179.

[9] R. Douglas, C. Koch, M. Mahowald, K. Martin, H. Suarez (1995): Recurrent Excitation in Neocortical Circuits. Science 269: pp. 981-985.

[10] J.H. Elder, S.W. Zucker (1996): Computing Contour Closure. European Conference on Computer Vision: pp. 399-412.

[11] J. Feng (1997): Lyapunov Functions for Neural Nets with Nondifferentiable Input-Output Characteristics. Neural Computation 9(1): pp. 43-49.

[12] C. Gomez, C. Bunks, J.P. Chancelier, F. Delebecque (1999): Engineering and Scientific Computating with SciLab. ISBN: 0817640096, Birkhauser.

[13] R.C. Gonzalez, P. Wintz (1987): Digital Image Processing, Addison-Wesley Pubilsching Company, Reading, MA.

[14] R.L.T. Hahnloser (1998): On the Piecewise Analysis of Linear Threshold Neurons. Neural Networks 11: pp. 691-697.

[15] S. Haykin (1999): Neural Networks - A Comprehensive Foundation. Prentice-Hall Inc.

[16] G. Heidemann, H. Ritter (2001): Efficient Vector Quantization Using the WTA-rule with Activity Equalization. Neural Processing Letters Vol. 13, No. 1: pp. 17-30.

[17] D.D. Hoffman (2000): Visual Intelligence: How We Create What We See. Norton & Company, New York.

[18] T. Hofmann, J.M. Buhmann (1997): Pairwise Data Clustering by Deterministic Annealing. IEEE Transactions on Pattern Analysis and Machine Intelligence Vol. 19 No. 1.

[19] J.J. Hopfield (1982): Neural Networks and Physical Systems with Emergent Collective Computational Abilities. Proc. Natl. Acad. Sci. USA 79: pp. 2554-2558.

[20] R. A. Hummel, S. W. Zucker (1983): On the Foundations of Relaxation Labeling Processes. IEEE Trans. Pattern Analysis and Machine Intelligence 5(3): pp. 267-287.

[21] J. Ivins, J. Porrill (1993): Everything You Always Wanted to Know about Snakes (but Where Afraid to Ask). AIVRU Technical Memo #86, July 1993, Artifical Intelligence Vision Research Unit University of Sheffield, England, S10 2TP.

[22] I.T. Jollife (1986): Principal Component Analysis, Springer, New York.

[23] D. Jones, J. Malik (1992): A Computational Framework for Determining Stereo Correspondence from a Set of Linear Spatial Filters. European Conf. on Computer Vision, May 1992: pp. 395-410, and Image and Vision Computing 10(10), Dec. 1992: pp. 699-708.

[24] G. Kaniza (1979): Organization in Vision. Praeger, New York.

[25] T. Kohonen (2001): Self-Organizing Maps. Springer, Berlin.

[26] J. Kittler, J. Illingworth (1985): Relaxation Labeling Algorithms - A Review. Image Vision Comput. 3: pp. 206-216.

[27] Y. Linde, A. Buzo, R.M. Gray (1980): An Algorithm for Vector Quantizer Design, IEEE Trans. Communications: pp. 84-95.

[28] T. Lindenberg (1998): Edge Detection and Ridge Detection with Automatic Scale Selection. International Journal on Computer Vison, Vol. 30 No. 2: pp. 77-116.

[29] J. Malik, S. Belongie, T. Leung, J. Shi (2001): Contour and Texture Analysis for Image Segmentation. International Journal of Computer Vision 43(1): pp. 7-27.

[30] J. Malik, P. Perona (1990): Preattentive Texture Discrimination with Early Vision Mechanisms. Journal of the Optical Society of America A, 7(5).

[31] D. Martin, C. Fowlkes, D. Tal, and J. Malik (2001): A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. International Conference on Computer Vision Vol. 2: pp 416-423.

[32] P. Meinike, T. Twellmann, H. Ritter (2002): Maximum Contrast Classifiers. Proc. of the Int. Conf. on Artificial Neural Networks, Jose R. Dorronsoro (Eds.), Springer, LNCS 2415: pp. 745-750.

[33] T. W. Nattkemper, H. Wersing, W. Schubert, H. Ritter (2000): Fluorescence Micrograph Segmentation by Gestalt-Based Feature Binding. Proc. of the Int. Joint Conf. on Neur. Netw. (IJCNN), 1, Como, Italy: pp. 248-254.

[34] T. W. Nattkemper (2001): A Neural Network-Based System for High-Throughput Fluorescence Micrograph Evaluation. Faculty of Technology, University of Bielefeld.

[35] T.W. Nattkemper, H. Wersing, H. Ritter, W. Schubert (2002): A Neural Network Architecture for Automatic Segmentation of Fluorescence Micrographs, Neurocomputing, 48(4): pp. 357-367.

[36] T.W. Nattkemper, A. Saalbach, T. Twellmann (2003): Evaluation of Multiparameter Micrograph Analysis with Synthetical Benchmark Images. Proc. of EMBC2003 (25th Annual Int. Conf. of the IEEE Engineering in Med. and Biol. Soc. Cancun, Mexico.

[37] T.W. Nattkemper, T. Twellman, W. Schubert, H. Ritter (2003): Human vs. Machine: Evaluation of Fluorescence Micrographs. Computers in Biology and Medicine, 33(1): pp. 31-43.

[38] S.A. Nene, S.K. Nayar, H. Murase (1996): Columbia Object Image Library (COIL20). Technical Report CUCS-005-96 Februray 1996.

[39] B.A. Olshausen, D.J. Field (1997): Sparse Coding with an Overcomplete Basis Set: A Strategy Employed by V1? Vision Research 37: pp. 3311-3325.

[40] J. Ontrup, H. Ritter (1998): Perceptual Grouping in a Neural Model: Reproducing Human Texture Perception. Technical Report SFB360-TR-98/6, Neuroinformatics Groups, University Bielefeld.

[41] J. Ontrup, H. Wersing, H. Ritter (2004): A Computational Feature Binding Model of Human Texture Perception. Cognitive Processing, 5(1): pp. 32-44.

[42] R. Opara, F. Wörgötter (1998): A fast and Robust Cluster Update Algorithm for Image Segmentation in Spin-lattice Models without Annealing - Visual Latencies Revisited. Neural Computation 10(6): pp. 1547-1566.

[43] J. Park, H. Cho, D. Park (1999): On the Design of BSB Associative Memories Using Semidefinite Programming. Neural Computation 11: pp. 1985-1994.

[44] S. Peleg, A. Rosenfeld (1978): Determining Compatibility Coefficients for Curve Enhancement Relaxation Processes. IEEE Trans. Syst. Man Cybern. 8: pp. 548-555.

[45] M. Pelillo, M. Refices (1994): Learning Compatibility Coefficients for Relaxation Labeling Processes. IEEE Trans. Pattern Analysis and Machine Intelligence 16(9): pp. 933-945.

[46] M. Pomplun, H. Ritter, B. Velichkovsky (1996): Disambiguationg Complex Visual Information: Towards Communication of Personal Views of a Scene. Perception Vol. 25: pp. 931-948.

[47] V. Rehrmann, L. Priese (1998): Fast and Robust Segmentation of Natural Color Scenes. 3rd Asian Conference on Computer Vision (ACCV'98), Springer, LNCS.1351: pp. 598-606.

[48] F. Rosenblatt (1962): Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. Spartan Books, Washington DC.

[49] A. Rosenfeld, R.A. Hummel, S.W. Zucker (1976): Scene Labeling by Relaxation Operations. IEEE Trans. Syst. Man Cybern. 6(6): pp. 420-433.

[50] H. Ritter (1990): A Spatial Approach to Feature Linking. Proc. International Neural Network Conference Paris Vol.2: pp. 898-901.

[51] A.J. Smola, B. Schölkopf (1998): A Tutorial on Support Vector Regression. NeuroCOLT2 Technical Report Series NC2-TR-1998-030.

[52] D. Terman, D. Wang (1995): Global Competition and Local Cooperation in a Network of Coupled Oscilators. Physica D, 81: pp. 148-176.

[53] T. Twellmann, T.W. Nattkemper, W. Schubert, H. Ritter (2001): Cell Detection in Micrographs of Tissue Sections Using Support Vector Machines. Proc. of. ICANN 2001 Workshop on Kernel and Subspace Methods for Computer Vision.

[54] D. Vernon (1991): Machine Vision: Automated Visual Inspection and Robot Vision. Prentice Hall, New York.

[55] C. von der Malsburg (1981): The Correlation Theory of Brain Function. Internal Report 81-2 Department of Neurobiology, Max Plank Institude of Biophysical Chemistry, Göttingen, and revised in E. Domany, J.L. von Hemmen, K. Schulten (Eds.) (1994): Models of Neural Networks II. Springer, New York: pp. 95-118.

[56] C. von der Malsburg (1999): The What and Why of Binding: The Modeler's Perspective. Neuron, 24: pp. 29-40.

[57] D. Wang, J. Buhmann, C. von der Malsburg (1990): Pattern Segmentation in Associative Memory. Neural Computation (2): pp. 94-106.

[58] D. Wang, D. Terman (1997): Image Segmentation Based on Oscillatory Correlation. Neural Computation 9(4): pp. 805-836.

[59] S. Weng, J.J. Steil (2002): Data Driven Generation of Interactions for Feature Binding. Proc. of International Conference on Artifical Neural Networks (ICANN) 2002: pp. 432-437.

[60] S. Weng, J.J. Steil(2003): Learning Compatibility Functions for Feature Binding and Sensory Segmentation. In Proc. of International Conference on Artifical Neural Networks (ICANN) 2003: pp. 60-67.

[61] H. Wersing (2000): Spatial Feature Binding and Learning in Competitive Neural Layer Architectures. Faculty of Technology, University of Bielefeld, Cuvillier Press.

[62] H. Wersing (2001): Learning Lateral Interactions for Feature Binding and Sensory Segmentation. Advances in Neural Information Processing Systems (NIPS).

[63] H. Wersing, W.J. Beyn, H. Ritter (2001): Dynamical Stability Conditions for Recurrent Neural Networks with Unsaturating Piecewise Linear Transfer Functions. Neural Computation 13(8): pp. 1811-1825.

[64] H. Wersing, E. Körner (2003): Learning Optimized Features for Hierarchical Models of Invariant Recognition. Neural Computation 15(7): pp. 1559-1588.

[65] H. Wersing, J.J. Steil, H. Ritter (2001): A Competitive Layer Model for Feature Binding and Sensory Segmentation. Neural Computation 13(2): pp. 357-387.

[66] M. Wertheimer (1923): Laws of Organization in Perceptual Forms. First published as Untersuchungen zur Lehre von der Gestalt II, in Psychologische Forschung, 4: pp. 301-350, Translation published in W. Ellis (1938): A source book of Gestalt psychology. London: pp. 71-88.

[67] A. Wismüller, O. Lange, D.R. Dersch, G.L. Leinsinger, K. Hahn, B. Pütz, D. Auer (2002): Cluster Analysis of Biomedical Image Time-Series. International Journal of Computer Vision, 46(2): pp. 103-128.