Hierarchical Feed-Forward Models for Robust Object Recognition

Ingo Bax

Der Technischen Fakultät der Universität Bielefeld vorgelegt zur Erlangung des akademischen Grades *Doktor der Ingenieurwissenschaften*

Dipl.-Inform. Ingo Bax
AG Neuroinformatik
Technische Fakultät
Universität Bielefeld
email: ibax [at] techfak.uni-bielefeld.de

Abdruck der genehmigten Dissertation zur Erlangung des akademischen Grades
Doktor der Ingenieurwissenschaften (Dr.-Ing.)

Der Technischen Fakultät der Universität Bielefeld

- am 31.01.2007 vorgelegt von Ingo Bax
- am 24.05.2007 verteidigt und genehmigt

Gutachter:

- Prof. Dr. Helge Ritter, Universität Bielefeld
- Prof. Dr. Gunther Heidemann, Universität Stuttgart

Prüfungsausschuss:

- Prof. Dr. Ralf Möller, Universität Bielefeld
- Prof. Dr. Helge Ritter, Universität Bielefeld
- Prof. Dr. Gunther Heidemann, Universität Stuttgart
- Dr. Sven Wachsmuth, Universität Bielefeld

Gedruckt auf alterungsbeständigem Papier nach ISO 9706

Acknowledgement

Completing this thesis was only possible due to the help and support of many people. First of all I would like to thank my supervisors Gunther Heidemann and Helge Ritter for their constant support and their substantial and valuable input at different stages of my work. I am also grateful to all my colleagues from the VAMPIRE project for a fruitful cooperation, especially to Holger Bekel, Christian Bauckhage, Sven Wachsmuth, Sebastian Wrede and Marc Hanheide. I would also like to thank all members of the Neuroinformatics Group for providing such an inspiring working environment. Special credits go to Jörg Ontrup for proofreading the manuscript, Oliver Lieske for providing the technical infrastructure that was indispensable for my experimental work, and to Petra Udelhoven who always helped me with various administrative issues. I am also very grateful to Michael Götting who I shared the office with for more than three years and who always made work an enjoyable place to be. Thank you all!

On the personal side, my profound gratitude goes to my parents, my brother Helge, and to Monika for their emotional support and constant assistance!

Notation

Throughout this document, the following notational conventions and typesets are used:

$\vec{x}$	vector
x_i	i-th component of vector $\vec{x}$
$\mathbf{M}$	matrix (also used for the intensity matrix of a gray value image)
$M_{i,j}$	entry in row i, col j of the matrix $\mathbf{M}$
$\mathbb{S}$	set
$dist(\vec{x}, \vec{y})$	Euclidean distance between vectors $\vec{x}$ and $\vec{y}$, i.e. $\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$
n_l	number of layers of the HFM
$n_p(l)$	number of planes on layer l of the HFM
$d_x(l)$	x-dimension of all C-cell planes on layer l of the HFM
$d_y(l)$	y-dimension of all C-cell planes on layer l of the HFM
$d_p(l)$	dimension (in both, x and y direction) of the receptive field profiles on layer l of the HFM
γ_l	"competition strength" parameter of the lateral competitive mechanism on layer l of the HFM
θ_l	"activity threshold" parameter of the lateral competitive mechanism on layer l of the HFM
σ_l	variance of the Gaussian spatial pooling kernel of layer l of the HFM

$P^l_{p\,q}(x,y)$	value at position (x,y) of the receptive field profile between C-cell plane q on layer $l-1$ and S-Cell plane p of layer l
$\hat{S}^l_p(x,y;\mathbf{I})$	preliminary activity of the S-cell at position (x,y) on plane p on layer l of the HFM with image $\mathbf{I}$ as input
$S^l_p(x,y;\mathbf{I})$	final activity of the S-cell at position (x,y) on plane p on layer l of the HFM with image $\mathbf{I}$ as input
$C^l_p(x,y;\mathbf{I})$	activity of the C-cell at position (x,y) on plane p on layer l of the HFM with image $\mathbf{I}$ as input

Contents

CHAPTER 1

Introduction

An overview of the scope and the contributions of this thesis is given. The content of each chapter is summarized shortly.

1.1 Overview

In recent years, computer vision research has made great advances in solving problems of visual recognition in restricted environments like e.g. detecting nonconforming parts on an assembly line or recognizing isolated objects under fixed lighting conditions. However, with computer vision applications about to enter less restricted environments, as in the case of mobile vision systems, autonomous robots, vehicle navigation aids, or surveillance systems, tasks are becoming more difficult. Here, the recognition system has to cope with an input signal of high variability, which is caused by changing illumination, arbitrary view points, cluttered scenes, object deformations, partial occlusions, etc. From an engineering point of view, this situation is difficult: it means that the recognition task cannot be adequately specified anymore, since it is impossible to exhaustively predict all possible configurations of appearances that might occur under combinations of multiple distortions.

A major goal of today's computer vision research is therefore to find approaches that allow for recognition which is *invariant* to distortions. Since the visual system of humans and animals, "optimized" by evolution, seems to have few problems in solving such difficult recognition tasks, it has become popular – and also quite successful – to take into account physiological and psychophysical findings about information processing principles in the brain to build artificial vision systems.

Modern approaches that follow this paradigm often rely on the early findings by Hubel and Wiesel [45], who firstly theorized about the hierarchical organization of the primary visual cortex and identified two types of cells, *simple cells* and *complex cells*, whose layered arrangement plays a key role in visual processing. While simple cells act as feature extractors, selectively responding to oriented bar-like structures that are present within the *receptive field* of the cell, the responses of complex cells are phase invariant and unaffected by small

spatial shifts in position of the stimulus.

Based on these findings, in 1980, Fukushima was one of the first to propose a hierarchical computational model, called *Neocognitron*, which is based on feed-forward processing and mimics the layered organization of the visual cortex. He showed that his model achieves a recognition of handwritten digits which is – to a certain degree – invariant to rotation, spatial shift and deformation of the stimulus. Since then, research has come up with a variety of modifications and enhancements of the original Neocognitron model, which are now opening up the application of the approach for real world computer vision scenarios.

Building upon previous models, in this thesis, we provide a generalized formal framework of a *Hierarchical Feed-Forward Model* (HFM) and apply the approach to challenging object recognition tasks. The main contributions are summarized briefly in the following:

- *Methodology:* For quantifying the success of the model in reducing signal variance, a novel method called *Average Nearest Neighbor Descriptor* (ANND) is proposed that allows measurement of the "apparent complexity" of a given multi-class dataset by analyzing class distributions and computing a single, real-valued descriptor. The method, together with synthetically distorted test datasets which are generated from three natural images databases – covering the domains of hand-written digits, small objects, and human faces – allows for a detailed analysis of the performance of the HFM.

- *Unsupervised optimization of internal model parameters:* We show that a recently proposed unsupervised coding strategy called *Non-negative Matrix Factorization with Sparseness Constraints* [43] can be used to "fine-tune" internal parameters of the HFM in order to account for the properties of a specific image domain. Previous results on unsupervised optimization of internal model parameters have been published in advance in [10].

- *Image patch classification:* We show that the abstract neural feature representation that is provided by the output space of HFM is well suited for building powerful classifiers that are able to achieve a high classification performance on difficult test datasets that contain a significant amount of distortions like noise, affine transformations, and background clutter. For this, we apply standard classification techniques as well as a novel method which learns optimized object representations based on a spectral clustering scheme.

- *Segmentation-free multi-class object detection:* We show that the flexibility of the model definition allows for a straight-forward extension of the model from pure classification to detection. The approach is able to simultaneously perform detection and classification of objects in natural scene images without the necessity of a pre-segmentation. For this, we attach an additional layer to the model, which is trained supervised using a labeled dataset of image patches. We test the approach on a synthetically generated detection task, where objects are randomly embedded into natural scene images. Previous results on segmentation-free multi-class object detection have been published in advance in [11, 8, 9].

1.2 Organization of the Manuscript

This thesis is organized as follows:

- *Chapter 2* gives a short introduction to the problem of visual perception and provides a brief review of models of object recognition which relate to the approach investigated in this thesis.
- *Chapter 3* formally defines the Hierarchical Feed-Forward Model that is used for the investigations in the remaining chapters.
- *Chapter 4* describes a novel, empirical complexity measure for multi-class datasets called *Average Nearest Neighbor Descriptor* (ANND). This allows a convenient analysis of the statistical properties of class distributions in a dataset by computing a single, well-bounded descriptor.
- *Chapter 5* investigates the image normalization capabilities of the HFM. For this, highly distorted image datasets are passed through differently parameterized HFMs and the ANND is used to measure the reduction in complexity that results from projecting images onto the abstract feature space as given by the activity of the output layer of the HFM.
- *Chapter 6* deals with the application of the HFM to the problem of image patch classification using highly distorted natural image datasets. For discrimination tasks, we investigate the use of *Template View Tuned Units* (T-VTUs), which correspond to 1-nearest neighbor classifiers operating on the output space of the HFM. Then we propose *Condensed View Tuned Units* (C-VTUs), which are obtained by applying a spectral clustering scheme on T-VTUs. Finally, for confidence based recognition, i.e. multi-class discrimination with rejection of "unknown" stimuli, *Radial Basis Functions* (RBFs) and *Linear Discriminant Functions* (LDFs) are employed.
- *Chapter 7* applies the HFM to the problem of segmentation-free multi-class object detection. For this, an architecture consisting of a two-layered training branch and a three-layered recognition branch is proposed. The approach is tested on synthetically generated detection images that contain objects that are randomly embedded into images of natural scenes.
- *Chapter 8* summarizes the results of this thesis and sketches potential future research activities.

CHAPTER 2

Background

This chapter gives a short overview of the problem of visual perception and provides a brief review of existing models of object recognition which relate to the Hierarchical Feed-Forward Model that is investigated in this thesis.

2.1 Biological and Computer Vision

Vision is one of the most essential human senses. From a practical point of view, there is not much need to argue for this hypothesis if one considers the number of everyday tasks that would either be impossible or at least very difficult to carry out if we were unable to see. While some of these tasks – for example reading this thesis – solely rely on vision, others are based on a seamless cooperation of vision with other senses. Examples are crossing a street (vision and hearing), playing a ball game (vision, touch, and hearing) or deciding if food is edible (vision and smell). In all cases, our visual system exhibits astonishing degrees of accuracy and flexibility as well as a high level of efficiency. Vision seems to always provide us with a clear window onto reality, as it allows us to constantly *perceive* the state of our environment. Not only for humans but also for the vast majority of animals, light proves to be a rich and vitally important source of knowledge about various aspects of the environment.

For technical systems the exploitation of optical information is equally attractive. In principle, the availability of digital camera equipment and processors with high computational power allows for the development of a new class of "intelligent machines" that are able to actively perceive their environment and, as a consequence, can act and react in a much more flexible way than conventional computer systems do. This is of interest in many application areas. Above all autonomous systems, like for example mobile robots or unmanned vehicles can benefit greatly from visual information. In the area of human machine interaction, optical interfaces can be used to find new ways that allow for a more natural communication with machines. Also for security and biometry, optical information has a key role for the development of new applications.

The subfield of computer science that is devoted to research of this kind is called *computer vision*. It is concerned with automatic processing of digital image data. Typically, digital image data is obtained from image sensors such as cameras, but can also be gained from other sources such as medical imaging devices (e.g., x-ray or computer tomography), any other types of sensor arrays, or it can even be artificially generated. Taking an image or a sequence of images as a starting point, the goal of computer vision is to automatically recover higher level information that is contained in the sensory data. A comprehensive definition of what is meant by "higher level information" is neither possible nor useful in general, but is given by specific application scenarios.

One such scenario is for instance *object recognition*. Here, the goal is to recover information about the properties of physical objects, that are present in the sensor's "field of view" such as identity, position, orientation or others. The computational problem is then defined as an *algorithmic mapping* between the sensory data and the properties that one wants to analyze.

However, as we argue in the following section, this mapping between the sensory data and the desired high level information can by no means be considered trivial and a straightforward solution to the problem is not possible, because visual perception must be understood as an under-specified inverse problem which makes the sensory data inherently ambiguous. In particular, we argue for the following points:

- These ambiguities result from either different objects leading to an identical retinal image, or, different views of the same object leading to different retinal images.
- In order to resolve these ambiguities, *heuristics* are needed which exploit the structure of the environment.
- In biological vision, these heuristics emerge from evolutionary adaptation and learning processes that *actively model the structure of the environment*.

2.1.1 Vision as an Inverse Problem

The task of visual perception is often called an *inverse problem* [80], because the sensory data results from projecting light that is reflected or emitted from the environment onto the sensor plane of a camera, or – in the case of biological vision – onto the retina at the back of the eye. This well-understood process is called *image formation* and is completely determined by the laws of optics. In this sense, the task of deriving properties of objects in the environment from observed light patterns can be formulated as "reversing" the process of image formation by undoing the projective transformations that happened during the formation of the image.

However, from a mathematical point of view, the relation between the sensory data and the environment is not symmetrical, because the light is being reflected from the surfaces of physical three-dimensional objects and projected onto a two-dimensional structure (the array of sensor elements of a camera or the photoreceptor cells of the retina, respectively).

On the one hand, as illustrated in Fig. 2.1 (a), this involves a loss of information and makes the sensory data highly ambiguous, because there are always an infinite number of possible "world configurations" that correspond to a single retinal image.

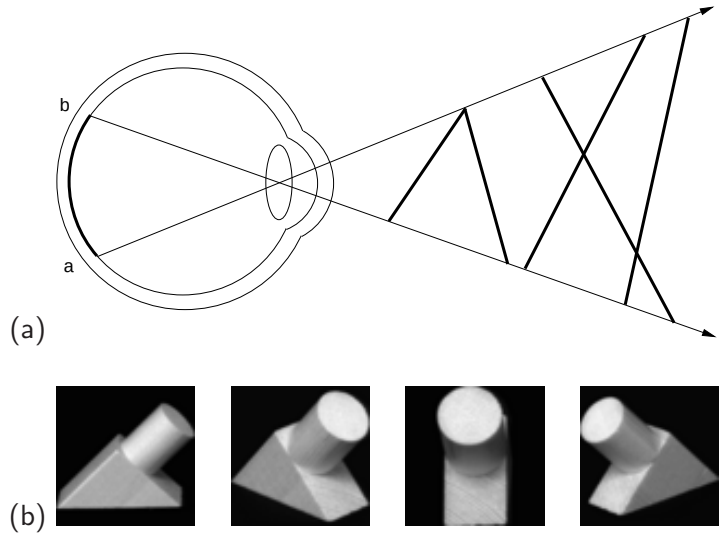


Figure 2.1: *Ambiguities in visual perception. (a) Different line segments lead to an identical retinal image (adapted from [80]). (b) Different views of the same object lead to different retinal images (from [70]).*

On the other hand, as illustrated in Fig. 2.1 (b), one and the same object can lead to drastically different retinal images, depending on the view point or other degrees of freedom that might alter the appearance of the object.

Hence, visual perception must be seen as finding a single suitable solution to an underspecified problem. How this is resolved by biological vision systems is still far from well-understood and a profound understanding of the underlying processes is one of the ultimate goals of cognitive neuroscience.

Whatever the answer might be, from a theoretical point of view the problem can only be solved by applying *heuristic processes* that exploit the structure of the environment in order to rule out unlikely "world configurations" and to end up with a single solution. In other words, for any vision system – artificial or biological – in order to pick the right solution from the infinite set of all theoretically possible ones, it is necessary to rely on a number of *specific assumptions about the structure of the environment*.

Research on artificial object recognition systems within the past decades has produced numerous different approaches to defining such assumptions. However, most often the assumptions are very strict and fixed a priori, leading to systems that can successfully operate in environments that are equally restricted. Many examples of such systems can be found in industrial applications, where e.g. nonconforming parts on assembly lines are detected. Here, the assumptions are highly restrictive in the sense that the objects in question always appear at a specified position, scale, and orientation, and under fixed lighting conditions. If the assumptions are violated, the system fails in generating reliable recognition results.

In contrast, biological vision systems, which, as argued above, also have to rely on restrictive assumptions about the structure of the environment in order to solve the inverse problem, by far outperform any artificial systems known today.

This fact motivates computer vision researchers to analyze the way these assumptions – which appear to be implemented in a much more flexible way and do not seem to be fixed a priori – are encoded in biological vision systems and try to utilize the findings for building artificial vision systems.

This means that a fruitful source for developing new concepts and ideas in the field of computer vision is in fact found in the results of biological, neurophysiological, and psychological research. One prominent example for this paradigm is the field of neural networks research, which has already led to many successful applications in computer vision.

This thesis focuses on a special type of neural network that can be used for visual processing and whose design is inspired by biological vision systems. The basic principle that this type of network employs is that the aforementioned assumptions about the structure of the environment are encoded partly within the overall neural topology of the network architecture and partly within the "neural wiring", i.e. the synaptic connections between neurons of the network. In biological vision systems, the topology, namely the structural organization of the brain, is a result of evolution, whereas the neural wiring is constantly adapted during the life of the organism, as the visual system is exposed to its environment and used to solve recognition tasks that are necessary for survival.

In the following section, we provide a brief review of models of object recognition before we define the generalized *Hierarchical Feed-Forward Model* (HFM) that is investigated in this thesis in Chapt. 3.

2.2 Models of Object Recognition

According to Riesenhuber and Poggio [87], models of object recognition can be roughly divided into two categories: *object-centered models* and *view-centered models*. The latter can be subdivided into another two groups: *feedback-driven models* and *feed-forward models*.

In *object-centered models*, recognition is achieved by creating view-invariant structural descriptions of objects and matching these representations to a database of stored object descriptions. A representative of these kinds of models is found in the works of Biedermann and Hummel [15, 47]. The model, called *Recognition-by-Components* (RBC) is based on decomposing objects into basic geometrical shapes, a similar scheme to that proposed in the classical works of Marr and Nishihara [63]. The main prediction of the model is that recognition of objects is view-point invariant as long as the same structural descriptions can be extracted from each object view [87]. However, from a computational point a view, a major question is how to obtain such descriptions from an input image.

On the other hand, the basic concept of *view-centered models* is that objects are represented by collections of features that are derived from different "image based appearances" of objects, covering for example different view points or different illuminations. Such an image based appearance is referred to as a *view* of an object. According to [87], the recognition performance of view-centered models is then a function of previously seen "training" views of the objects. Since a large number of view-centered models can be found in literature, the authors of [87] make a further distinction, depending on whether the models employ feedback connections or whether they are strictly based on feed-forward processing.

Feedback-driven models are based on an "analysis-by-synthesis" or "hypothesis-and-test" approach. This means that for recognition, the model creates an initial hypothesis in terms

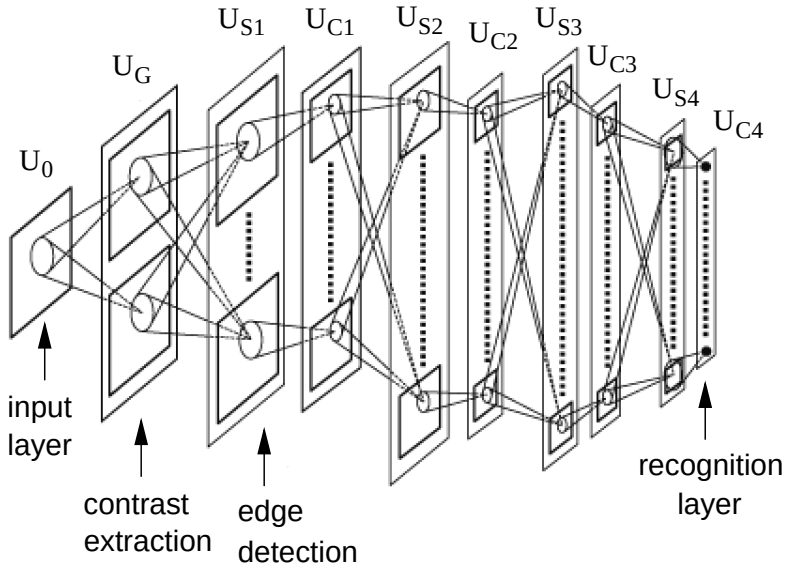


Figure 2.2: *The basic architecture of the Neocognitron model. The model consists of alternating layers of feature-extracting S-planes and spatial-pooling C-planes (adapted from [25]).*

of a neural representation taken from a stored collection of formerly seen object views. Upon comparing this representation to the actual visual input, the initial hypothesis is modified, sometimes in multiple iterations, in order to generate a recognition result. The strategy is employed for instance by the models of Rao and Ballard [83] and Mumford [67]. Slightly different types of feedback connections are present in the *Shifter Circuit* model proposed by Anderson et al. [2, 74]. Here, feedback is used in order to alter the position and the scale of the input stimulus until it finds a match in a database of stored object views.

In contrast, in *feed-forward models*, recognition of input patterns takes place in a strict feed-forward fashion, without employing any feedback connections or recursive loops. For example, in the SEEMORE system, proposed by Mel [65], objects are represented and recognized by histograms over various channels of color, shape, and texture features. The Hierarchical Feed-Forward Model that is investigated in the thesis is also a representative of this class of models. The definition of the model as given in the following chapter is based on previously proposed feed-forward models, two of which are highlighted in more detail in the following sections.

2.2.1 Fukushima's Neocognitron

The *Neocognitron*, firstly proposed by Fukushima in 1980 [25, 28, 26, 27], is one of earliest representatives of a feed-forward view-based recognition model. The basic architecture of the model is sketched in Fig. 2.2.[1]

[1]The model presented here is only a simplified version of the original Neocognitron as proposed in [25], which additionally contains V-planes that are used for gain control of S-cell responses.

Processing of an input stimulus starts at the U_0 plane, which can be interpreted as an array of photoreceptor cells of the retina. The first layer, U_G, performs contrast extraction and the remaining processing architecture is composed of alternating layers of complexity-increasing S-planes and invariance-increasing C-planes.

Each S-plane consists of an array of identical cells which resemble simple cells of the primary visual cortex. They receive as input the activities of cells of the previous layer within a certain *receptive field*. In the lower stages of the hierarchy, S-cells are supposed to extract local features such as edges of certain orientations, whereas at higher levels, the cells respond to more global features, such as larger parts of patterns. Finally, on the output layer, S-cells responds to an entire image of a certain class. The kinds of features that are extracted by S-cells is determined by the variable input connections of the cells (the "weights" or "profiles", as we call them later), which are obtained by learning, as described below.

C-planes of the model consist of arrays of C-cells, which – unlike S-cells – have fixed input connections and resemble the response properties of complex cells as found in the visual cortex. The input connections are designed in such a way that a cell becomes active when at least one S-cell of the previous layer is active within a local sub-window, centered at the position of the C-cell. This way the activity pattern of C-planes is a "blurred copy" of the S-plane activity pattern of the previous layer. This is the essential mechanism which makes the responses of C-cells to a certain degree phase invariant and robust to small spatial shifts of the input stimulus.

For learning in the Neocognitron, which is used to adjust the variable input connections of the S-cells, [2] Fukushima proposed two different approaches, "unsupervised learning" and "learning with a teacher." In case of "unsupervised learning," a winner-takes-all Hebbian learning scheme is applied to train the model layer by layer starting from the left in Fig. 2.2: After presentation of a training pattern, the cell responses of S-cells located within a *hypercolumn*, i.e., within a small area across all planes, are compared and the most responsive S-cell is selected. The input connections of this cell are then adjusted such that the cell responds even more strongly to the stimulus the next time the same pattern is presented. For "learning with a teacher" [28], a human operator manually selects the cells that are supposed to respond to a certain stimulus and again, a Hebbian update rule is applied in order to adjust the input connections.

Despite the fact that the Neocognitron model was successfully applied to the problem of handwritten digit classification, training of the model was reported to be rather difficult [58]. The main reason is that some internal selectivity parameters turned out to be very difficult to adjust. Nevertheless, the Neocognitron model can be seen as a major milestone in the development of biologically motivated hierarchical feed-forward models, as it already employs the two complementary principles of alternatingly creating redundancy by feature extraction and successively reducing variance by spatial pooling. The effectiveness of pooling was also shown in later work by Perret and Oram [81], who used the mechanism for recognition that is invariant to non-affine transformations such as rotation in depth or varying illumination as well.

[2] In the original version of the Neocognitron, V-cells, which have inhibitory connections to S-cells in order to control the gain of the S-cell responses, are also subject to learning.

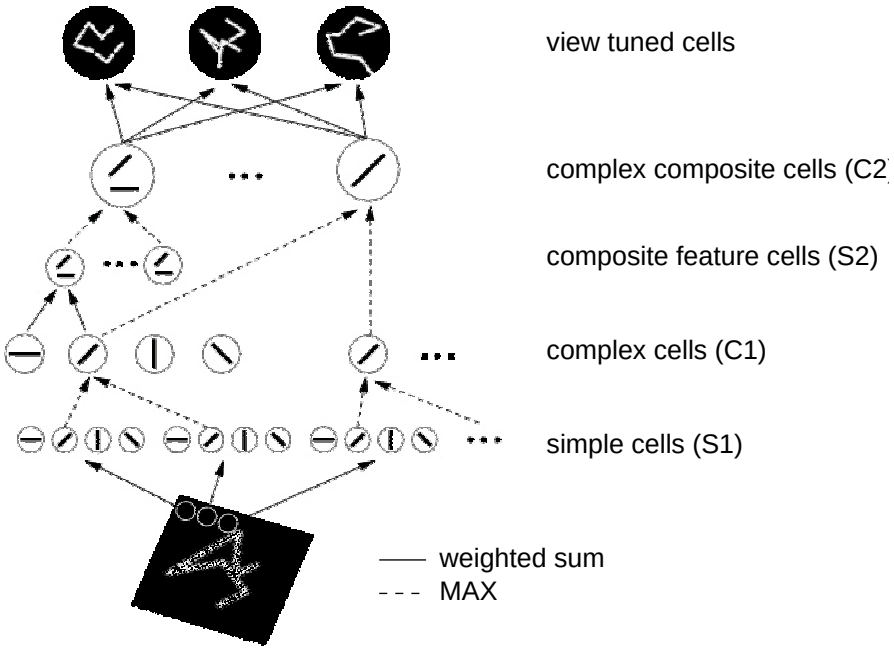


Figure 2.3: *The HMAX model proposed by Riesenhuber and Poggio [85]. Similar to the Neocognitron model, a hierarchy of alternating layers of S-planes and C-planes is used. A major advantage of the model is the use of a non-linear maximum operation for computing C-cell responses.*

2.2.2 The HMAX Model

A modern variation of the Neocognitron model was introduced by Riesenhuber and Poggio in [85, 84, 88]. Tarr [97] named the model *Hierarchical Model and X* (HMAX), where X stands for a highly non-linear maximum operation that is applied for the computation of C-cell responses.

A sketch of the architecture of the model is shown in 2.3. The basic structure, i.e., the alternating arrangement of layers of S-cell planes and C-cell planes, is adopted from the Neocognitron model.

In contrast to the Neocogniton model, the HMAX model employs hard-wired connections on all but the last layer. For S-cells on the first layer, several groups of edge detectors with different receptive field sizes are used (ranging from 7×7 to 29×29 cell positions). C-cells on the first layer pool the responses of groups of S-cells with the same orientations by selecting the maximum response within a certain local region. The role of S-cells on the second model layer is to combine different features from the first layer, yielding "composite features." Here again, the weights are hard-wired. Second layer C-cells again pool over multiple second layer S-cell responses, using the maximum-rule.

Finally, on the last layer of the model, "view-tuned cells" are applied, which receive as input the responses of all second layer C-cells and perform the actual recognition step. For this, Gaussian response functions which operate on the C-cell response space, are used. The

centers of these functions are obtained by recording the final C-cell responses caused by training patterns that are passed through the model. The variance of the Gaussian function can be manually adapted for fine-tuning selectivity.

The HMAX model was successfully applied to the recognition of artificial stimuli like paper-clip objects, where it exhibited a high robustness to distortion of the input pattern. The model was also applied to the recognition of more natural stimuli and also to a categorization task [86]. In summary, the major advance of the HMAX model over the Neocognitron is the novel maximum operation that is employed for computing C-cell responses.

2.2.3 Recent Advances

In more recent literature, a number of advances for biologically motivated feed-forward models can be found.

- *Lateral competition:* In [107], Wersing and Körner proposed a replacement of the maximum operation as used in the HMAX model, which is based on a *winner-takes-most* lateral competition scheme within hyper-columns of S-cells. They showed that this method leads to much more selective cell responses and yields a segmentation of the input in terms of the most dominant features. This high selectivity makes it possible to substantially simplify the initial feature computation performed by S-cells in the first layer. Here, the authors used a single bank of four small scaled odd Gabor filters (spanning approx. 3×3 pixels) and used the absolute values of the responses [3]. For details, the reader is referred to the following chapter, where we adopt the winner-takes-most mechanism for our model definition.

- *Learning of receptive field profiles:* A major drawback of the original HMAX model is the fact that the connections within the lower layers of the model are hard-wired and cannot be automatically adapted to, for instance, a certain image domain. For this, an approach to unsupervised learning of connection weights for the HMAX model can be found in [57], where k-means clustering [61] was used on activity patches of first layer S-cell responses in order to obtain weights for second layer S-cells. In [107] a similar learning scheme based on a variation (see [106]) of *Sparse Coding* [75, 77] was proposed. In both cases, experimental results showed the unsupervised adaptation of connections makes it possible to tune the model to a specific image domain. In Chapt. 5 we also employ such a learning strategy for receptive filed profiles, not only for the second model layer, but also for the first. The learning strategy is based on a recently proposed learning method called *Non-Negative Matrix Factorization with Sparseness Constraints* [43].

2.3 Discussion

In this chapter, a short introduction to the problem of visual perception was given and it was argued that object recognition must rely on a specific set of assumptions about the visual environment in order to solve the inverse problem of visual perception. How such

[3] This approximates quite well the classical "sum-of-squares" complex cell model which relies on combining the responses of pairs of even and odd Gabor filters [24, 44].

assumptions are encoded in biological vision systems is of great interest for computer vision research, because an understanding of the underlying processes would eventually allow us to build more flexible artificial vision systems that can operate in less restricted environments and thus offer a larger range of application possibilities.

Research has yielded a variety of models to explain object recognition: these can be categorized as object-centered or view-centered, where the latter can be subdivided into feed-back driven and feed-forward models. The question which of these models is most plausible cannot be answered easily, but, as argued e.g. in [87] empirical evidence points in the direction of view-based, feed-forward models. Psychophysical data from humans and monkeys (for a review see [98]) suggest view-dependence of object recognition. This is also supported by physiological studies [19], where it was found that cells in the inferotemporal cortex (IT) of monkeys respond to complex objects such as faces. Experimental results by Logothetis et al. [56], who trained monkeys to recognize paperclip objects by presenting example views, also suggest view-dependence.

Evidence for feed-forward processing is given by EEG studies, e.g. [99], where it was shown that humans can solve simple object detection tasks within 150ms, which roughly corresponds to the latency of visual signals being transferred from primary visual cortex to the inferotemporal cortex. According to [87], this result constrains the role of feedback at least in "immediate" object recognition.

CHAPTER 3

A Hierarchical Feed-Forward Model

In this chapter, a generalized formulation of a Hierarchical Feed-Forward Model is presented, which is used for the investigations in the following chapters.

3.1 Formal Definition

The formal definition of the model as described in following sections is based on previously proposed models, but attempts to provide a more general and flexible framework. We first describe the general topological architecture of the model, and then focus on feature-extracting S-cells, lateral competition and finally on spatial pooling C-cells.

3.1.1 Topology

The structure and feed-forward connectivity of the model is illustrated in Fig. 3.1. The hierarchy consists of n_l layers, each holding $n_p(l)$ planes of two types: *S-cell planes* and *C-cell planes*, denoted S_p^l and C_p^l, respectively, where l is the layer index and p the plane index. For notational convenience (see below), the input gray-scale image is formally treated as a C-cell plane and referred to as C_1^0 (setting $n_p(0) = 1$).

The responses of S-cells in a layer always depends on the activity of *all* C-cell planes of the previous layer. In contrast, C-cell responses are computed only from the activity of the corresponding S-cell plane of the same layer.

The elements in each C-cell plane are arranged in a $d_x(l) \times d_y(l)$ grid, whose dimension can be different for each layer (typically, it decreases with an increasing layer index l). The dimension of S-cell planes in a layer l is always equal to that of the C-cell planes of the previous layer $l-1$. For an input image $\mathbf{I}$, we refer to the activity of a single cell located at position (x, y) as $S_p^l(x, y;\ \mathbf{I})$ for S-cells and $C_p^l(x, y;\ \mathbf{I})$ for C-cells respectively.

Processing of an input stimulus takes place in a feed-forward fashion by successively computing the activity of the layers in increasing order. The final output of the network is given by the activity of the C-cell planes of the last layer (shown at the top in Fig. 3.1).

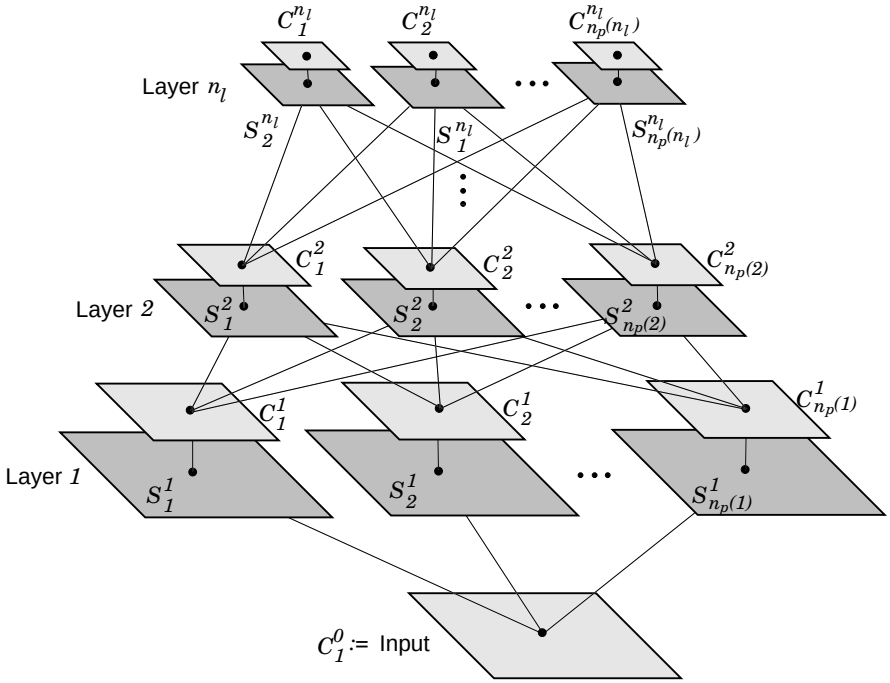


Figure 3.1: *The architecture and feed-forward connectivity of the Hierarchical Feed-Forward Model. It consists of multiple layers, each holding a number of S-cell planes (displayed in dark gray) and corresponding C-cell planes (displayed in light gray). The input stimulus at the bottom is a gray-scale image and the output of the network is given by the activity of the C-cell planes of the final layer at the top.*

Figure 3.2 shows a more detailed view of a single layer l of the hierarchy. The major processing principles that are involved in a layer are *feature extraction* (performed by S-cells), followed by *lateral competition* (among S-cell responses) and *spatial pooling* (performed by C-cells).

In the following, the computation of S-cell and C-cell responses in a layer is described in detail.

3.1.2 Feature Extracting S-Cells

For featuring extracting S-Cells, in a first step, preliminary responses (denoted $\hat{S}_p^{l-1}$, with $p \in \{1, \ldots, n_p(l)\}$) are computed by means of convolution of preceding C-cell activities (denoted C_q^{l-1}, with $q \in \{1, \ldots, n_p(l-1)\}$) with receptive field profiles. Receptive field profiles are denoted $P_{p\,q}^l$ as depicted by the cone like connections at the bottom of Fig. 3.2 between plane p of layer l and plane q of layer $l-1$. In the following, we consider linear S-Cell models. Formally, the preliminary response of an individual S-cell is given by:

$$\hat{S}_p^l(x, y;\, \mathbf{I}) \;=\; \sum_{q=1}^{n_p(l-1)} \sum_{i=-r}^{r} \sum_{j=-r}^{r} C_q^{l-1}(x+i, y+j;\, \mathbf{I}) P_{p\,q}^l(r+i, r+j), \qquad (3.1)$$

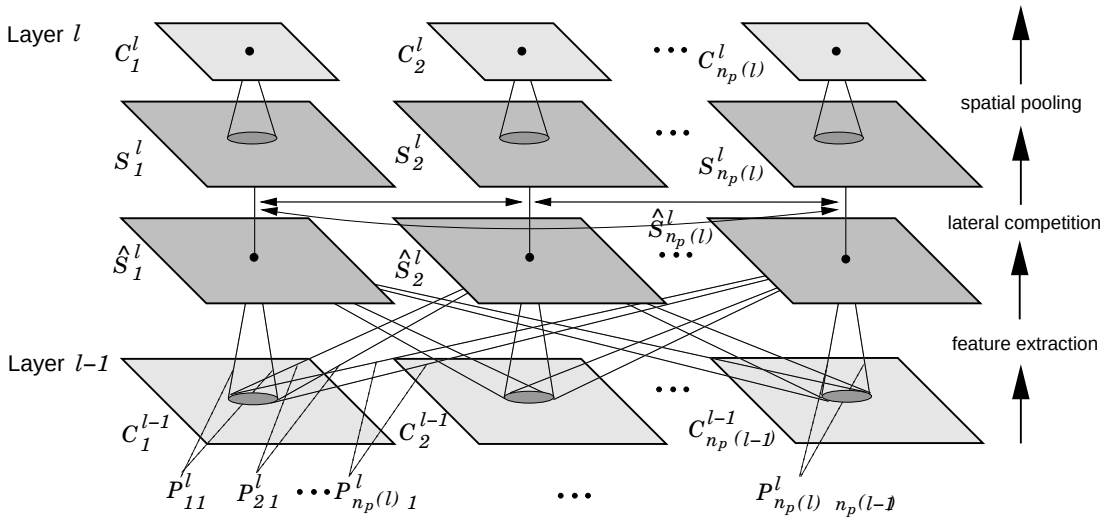


Figure 3.2: *Computation within a layer l: S-cell responses are computed in two steps: First, the activities of the C-cell planes of the previous layer are convolved using different receptive field profiles, one for each plane (denoted $P^l_{p\,q}$, where l is the index of the current layer, p the plane index within the current layer and q the plane index within the previous layer). This step can be interpreted as a feature extraction stage. The cumulative result of the convolutions yields the preliminary responses of S-cells, denoted $\hat{S}^l_p(x, y;\, \mathbf{I})$. In a second step, the preliminary responses are subject to a lateral competitive mechanism, which is performed among all S-cells located at identical positions on the planes of the current layer. The results are the final responses of the S-cells, denoted $S^l_p(x, y;\, \mathbf{I})$. The responses of C-cells (denoted $C^l_p(x, y;\, \mathbf{I})$) is computed using a spatial pooling mechanism that combines the responses of a local neighborhood of S-cells.*

where $r = \frac{d_p(l)-1}{2}$ is the "radius" of the receptive fields, directly derived from the dimension of the receptive field profiles of layer l, $d_p(l)$. $C^l_q(x, y;\, \mathbf{I})$ is set to 0 for out-of-range values of x of y. Note that for the S-cells of the first layer the previous C-cell plane C^0_1 is simply the input stimulus. This model is called linear, because the preliminary responses of S-cells is computed solely by means of convolution and summation.

3.1.3 Lateral Competition

The lateral competitive mechanism (depicted by the edges in the middle of Fig. 3.2 connecting the preliminary S-cell planes with the final S-cell planes) that is used to derive the final responses of S-cells is given by a competition function $LC : \mathbb{R}^{n_p(l)} \times \mathbb{N} \to \mathbb{R}$ that takes as input the activities of a so-called *hyper-column*, i.e. a vector containing all $n_p(l)$ preliminary S-cell activities at position (x, y) of all planes, and a plane index. The output of the competition function then is the final response of the S-cell:

$$S^l_p(x, y;\, \mathbf{I}) = LC((\hat{S}^l_1(x, y;\, \mathbf{I}), \ldots, \hat{S}^l_{n_p(l)}(x, y;\, \mathbf{I}))^T; p). \tag{3.2}$$

Here, we adopt the *winner-takes-most* mechanism proposed in [107]. It is given by:

$$LC(\vec{s}; p) := \begin{cases} 0 & \text{if} \quad M = 0 \quad \text{or} \\ & \frac{s_p}{M} < \gamma_l \quad \text{or} \\ & \frac{s_p - \gamma_l M}{1-\gamma_l} < \theta_l, \\ 1 & \text{else,} \end{cases} \tag{3.3}$$

where $M = \max_p s_p$, γ_l with $0 < \gamma_l < 1$ is the "competition strength", and θ_l is the "activity threshold" common to all planes in layer l.

Even though we adopt the lateral competitive mechanism of [107] (which the authors showed to be superior to others), and are not investigating any other alternatives in this thesis, the above definition of the function LC could be easily replaced by a different approach, such as the maximum operation as used in the HMAX model [85] or the OR-operation of the Neocognitron [25].

3.1.4 Spatial Pooling C-Cells

The activities of cells in a C-cell plane C_p^l are computed directly from the activities of the cells in the corresponding (final) S-cell plane S_p^l. Each C-cell receives as input the responses of S-cells that are located within a *spatial pooling area* (as depicted by the cone-like connections at the top in Fig. 3.2).

The spatial pooling area is determined by a neighborhood function $H_l(x, y; r_l)$ which – given a C-cell position (x, y) in planes C^l – returns a set of S-cell positions within a square of size $r_l \times r_l$ in the S-cell planes S_p^l, i.e.

$$\begin{aligned} H_l(x, y; r_l) = \{(x', y') \mid \quad & f_X x - r_l \leq x' \leq f_X x + r_l, \\ & f_Y y - r_l \leq y' \leq f_Y y + r_l, \\ & x', y' \in \mathbb{N}\}, \end{aligned} \tag{3.4}$$

where $f_X = \frac{d_x(l-1)}{d_x(l)}$ and $f_Y = \frac{d_y(l-1)}{d_y(l)}$ are the sub-sampling ratios between the planes in layer l and $l-1$ in x and y directions respectively. The size of the spatial pooling area is determined by the parameter r_l, which we set to $\frac{6\sigma_l - 1}{2}$, where σ_l is the variance of a Gaussian weighting kernel which can be chosen differently for each layer (see below).

The following formulation for the computation of the response of a C-cell $C_p^l(x, y;\ \mathbf{I})$ was used in the model of [107] and is given by

$$C_p^l(x, y;\ \mathbf{I}) = \tanh\left(\sum_{(x',y') \in H_l(x,y;r_l)} g((x-x')^2 + (y-y')^2; \sigma_l) S_p^l(x', y')\right), \tag{3.5}$$

where $g(x; \sigma_l)$ is the Gaussian weighting kernel computed from the squared distance (in cell positions) of the current cell to the center of the spatial pooling area, i.e.,

$$g(x; \sigma_l) = \frac{1}{\sigma_l \sqrt{2\pi}} e^{-\frac{x}{2\sigma_l^2}}, \tag{3.6}$$

where σ_l denotes the variance for layer l. This definition computes a weighted sum of the responses of S-cells within the spatial pooling area and passes the result through a $\tanh$ function to implement a smooth spatial OR-operation [107].

3.2 Summary

The main advantage of the above definition over previous proposals is that each of the n_l layers are formally treated equally and only differ in that parameters which are summarized in the following:

- $n_p(l)$: This defines the number of planes in layer l (both S-cell planes and C-cell planes).
- $d_x(l)$ and $d_y(l)$: These parameters determine the size of the grid in which C-cells are arranged in layer l. If present, the S-cell planes of the next layer $l+1$ always have the same dimension.
- $d_p(l)$: This parameter defines the size of each receptive field profile $P^l_{p\,q}$, with $p = 1 \ldots n_p(l)$ and $q = 1 \ldots n_p(l-1)$.
- $P^l_{q\,p}$: This is the $d_p(l) \times d_p(l)$ weight matrix of the receptive field profile connecting C-cell plane q of layer $l-1$ and S-cell plane l of layer l.
- σ_l: This parameter defines the variance of the Gaussian pooling kernel as applied for computing C-cell activity.
- γ_l and θ_l: These parameters define the "competition strength" and the "activity threshold" of the winner-takes-most lateral competitive mechanism on layer l, common to all planes.

Before applying the HFM for processing natural images, in the next chapter we first develop a complexity measure for multi-class datasets, which is used in Chapt. 5 for analyzing the success of the HFM in normalizing image patches.

CHAPTER 4

An Empirical Complexity Measure for Multi-Class Datasets

In this chapter a novel method is described that allows the analysis of the statistical structure of class distributions in high-dimensional multi-class datasets. In contrast to former proposals, the measure yields a single, real-valued descriptor between 0 and 1 that can be interpreted as a quantitative judgment of the geometrical complexity of the dataset. The method provides a generic tool for (i) analyzing the "difficulty" of discrimination tasks and (ii) evaluating the quality of feature representations. Therefore the measure can be used as an objective function for optimizing a specific feature extraction technique. The computation of the measure is based on randomly selecting many subsets of data points that are used as reference nodes for an ensemble of simple 1-nearest neighbor classifiers to be applied on the whole dataset. The average accuracy of these classifiers is used to obtain a characteristic graph that reflects the statistical properties of the class distributions. The area above this curve is used to compute the measure which we call Average Nearest Neighbor Descriptor (ANND). Several experiments on toy datasets that show the usefulness of the measure are presented. In Chapt. 5 the ANND is applied on multi-class natural image datasets and it is further used to analyze the quality of the abstract feature representation that is provided by the final C-cell planes of the Hierarchical Feed-forward Model when processing highly distorted natural image datasets.

4.1 Overview

The problem of measuring the complexity of multi-class datasets has been addressed e.g. in the works of Ho et al. [39, 37, 38, 62], where it is pointed out that there are at least three independent factors which affect the complexity (or "difficulty") of discrimination problems:

- *Class ambiguity:* This refers to situations where some classes in the datasets cannot be distinguished using a given feature representation by *any* classification approach. This can either be caused by the intrinsic structure of the dataset or by inadequate

feature measurements.[1] Discrimination problems like this are said to have a non-zero Bayes error, meaning that the class distributions overlap to a certain degree. This sets a natural limit on the lowest achievable error rate.

- *Complexity of decision boundaries:* Even in the case of a zero Bayes error, not all discrimination problems can be considered equally difficult, because implicitly, for defining a classifier a description of the boundaries that separate classes in the feature space is necessary. The more complex these class boundaries are the more difficult the problem must be considered.

- *Sample sparsity and feature space dimensionality:* From a practical point of view a discrimination problem can only be analyzed in terms of a representative dataset that contains example instances of the problem. This means that there always exist unseen examples which are unavailable during analysis. How these examples should be "classified" depends on a set of generalization rules that a corresponding classifier must employ. The smaller the size of the sample (or the higher the dimensionality of the feature space) the sparser the description of the distributions. Therefore, less information is available to constrain the classifier's generalization rules. Small sample size can lead to two different effects: Intrinsically complex problems can appear "simple", whereas intrinsically simple, e.g. linear separable problems might appear to have complex non-linear decision boundaries.

Of these sources of difficulty, sample sparsity can be seen as the most severe one because as long as a complete sample is not available no a priori predictions can be made about the representativeness of the dataset. Therefore, when analyzing a dataset of examples we can never expect to measure the *true complexity* of the underlying problem but only the *apparent complexity* as computed from a fixed, finite dataset [37]. Thus, in the following we concentrate on the first and second of the mentioned factors and assume that we have a dataset at hand which is sufficiently representative.

The problem of characterizing the complexity of a dataset has been approached from two sides: One is to use the performance of classifiers as a measure of complexity, whereas the other attempts to define measures that are independent of the choice of a specific classifier.

A representative of the first class of approaches can be found in the work of Duin et al. [20]. Here, it is argued that a natural characterization of the complexity of a classification problem is provided by the tools that are used to solve it. Following this view, the authors propose to use a set of standard classifiers and measure their performance on a given dataset. From the absolute performance values of the different classifiers the authors obtain what they call a *classifier disagreement matrix*, that — after an embedding into a two dimensional Euclidian space — can be used to compare different classification problems. In practice, the measure can be used to assist the selection of suitable classification techniques when given a classification problem whose properties are unknown. However, the stability and consistency of the approach, especially when changing the set of classifiers or altering their

[1] An example from the field of optical character recognition (OCR) is the distinction between the lower case letter 'l' and the digit '1', which appear similar in some fonts. Using a simple pixel representation for the characters the two classes appear ambiguous. To solve the problem, a more advanced representation is needed that also takes account of the context in which the characters appear.

parameterizations, is not investigated. Therefore, such an analysis is rather qualitative and so far does not appear to be well suited as an absolute, quantitative measure.

Regarding the second type of approach, i.e., measures that are independent of specific classification approaches, a summary of existing methods is given in the work of Ho & Basu [39]. Here, the authors compare different approaches to characterizing the complexity of a dataset which focus on the geometrical complexity of the class boundaries. It is argued that none of the investigated measures alone can sufficiently capture all aspects of complexity that may be present in a dataset. Therefore, the authors proposed a combination of multiple measures (such as Fisher's discriminant ratio [23], Feature Efficiency [38], Mixture Identifiability [95], Non-linearity [40] and several other approaches) to form a multi-dimensional descriptor. A given classification problem can then be seen as a point in a multi-dimensional descriptor space. In experiments it has been shown that for real world datasets the descriptor differs significantly from that of randomly labeled datasets. However, the proposed measure seems to be strongly dependent on the choice of the set of measures and, again, due to the multi-dimensionality of the descriptor, it canot be used directly as an absolute quantitative measure of complexity. In other works (e.g, [62]), the authors use the descriptor space for the definition of a so-called *domain of competence* of a classifier.

In summary, existing methods which characterize the complexity of multi-class datasets are mostly based on multi-dimensional descriptors and are therefore not well suited for the analysis that we intend to carry out in this thesis, since a quantitative measure rather than a qualitative one is required.

The approach introduced in the following is based on a single, real-valued and also well bounded descriptor. The value of the measure is derived from the average accuracy of simple 1-nearest neighbor classifiers that use representations of increasing complexity.

We argue that the measure accounts for the first two of the above mentioned issues, namely "class ambiguity" and "complexity of the decision boundary." With respect to "class ambiguity," the error rate of the 1-nearest neighbor classifier is known to be no worse than twice the Bayes error when the number of nodes is increased to infinity [18, 93].

The measure also accounts for the "complexity of the decision boundary" because the number of reference nodes that the representation consists of corresponds to the number of Voronoi tessellation cells which create the decision boundaries. The more tessellation cells used, the higher the geometrical complexity of the decision boundaries that separate the classes in data space [104, 3].

After describing how the measure — which is called the *Average Nearest Neighbor Descriptor (ANND)* — is computed in the next section, we apply the approach on toy datasets in order to discuss some important properties.

4.2 The Average Nearest Neighbor Descriptor

Let us assume, we are given a dataset $\mathbb{D}$ consisting of examples that are represented by vectors $\vec{x}_i$:

$$\mathbb{D} := \{\vec{x}_i |\ \vec{x}_i \in \mathbb{R}^n, i \in \{1, \ldots, m\}\}, \tag{4.1}$$

where n is the dimension of the data space and m is the total number of examples. Further, labels $l(i) \in \{1, \ldots, c\}$ are assigned to the examples $\vec{x}_i$, with c being the number of classes.

Based on this dataset we now create a set of different 1-nearest neighbor classifiers which use subsets $\mathbb{D}' \subset \mathbb{D}$ of the dataset as nodes for classification. A 1-nearest neighbor classifier is a function $\Phi : \mathbb{R}^n \rightarrow \{1, \ldots, c\}$, which assigns a class identifier to a given data point $\vec{x}$ (based on the current representation $\mathbb{D}'$) by choosing the label of the node that is closest to the input data point with respect to a distance measure $dist : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$

$$\Phi(\vec{x}; \mathbb{D}') := l(\arg\min_i \ dist(\vec{x}_i, \vec{x})), \ \vec{x}_i \in \mathbb{D}', \tag{4.2}$$

where in the following we use the Euclidean distance, $dist(\vec{a}, \vec{b}) :=\| \ \vec{a} - \vec{b} \ \|$. The analysis of the dataset $\mathbb{D}$ based on such classifiers is now done by randomly choosing subsets $\mathbb{D}'$ of different sizes and measuring the classification accuracy of the corresponding 1-nearest neighbor classifiers applied on the whole dataset $\mathbb{D}$. This is expressed by

$$acc(\mathbb{D}') := \frac{1}{m} \sum_{i=1}^{m} \delta_{\Phi(\vec{x}_i, \mathbb{D}'), l(i)}, \tag{4.3}$$

where $\delta_{a,b} = 1$ if $a = b$ and 0 else. The value of $acc(\mathbb{D}')$ is bounded within the interval $[0, 1]$ and can be interpreted as a measure of how well the current choice of nodes represents the dataset. A value of 1 corresponds to maximal accuracy, meaning that all examples of the dataset can be classified correctly using the current representation $\mathbb{D}'$.

The accuracy measure defined in Eq. 4.3 is now used to compute what we call the *Characteristic Accuracy Graph (CAG)* of the given dataset. This is done by varying the size of the representation and averaging the value of the accuracy measure for each size over a number N_{REP} of different randomly selected subsets. The computation of the CAG is summarized in Alg. 1.

Algorithm 1 Computation of the Characteristic Accuracy Graph: CAG($\mathbb{D}$, N_{REP})

```
for size = 1 to m do
    CAG[size] ← 0
    for repetition = 1 to N_REP do
        randomly choose D' ⊂ D with |D'| = size
        CAG[size] ← CAG[size] + acc(D')
    end for
    CAG[size] ← CAG[size] / N_REP
end for
return CAG
```

The array CAG returned by the algorithm has as an argument the size-values ranging from 1 to m. The CAG-values range from 0 to 1. Rescaling the size-values to the interval $[\frac{1}{m}, 1]$, each point of the graph can be interpreted as the average performance of a 1-nearest neighbor classifier that uses a specific fraction of the total data points as nodes for the representation. The curve starts at the size-value of $\frac{1}{m}$, where the representation uses only a single node. Here, assuming an equal number of representatives in each class, we obtain an average accuracy value of $CAG = \frac{1}{c}$. At the other extreme, at the size-value of 1, the representation consists of all data points and trivially we obtain an average accuracy of 1.

The experiments described in the following show that the average accuracy values at the intermediate positions between $\frac{1}{m}$ and 1 form a smooth, monotonically increasing curve,

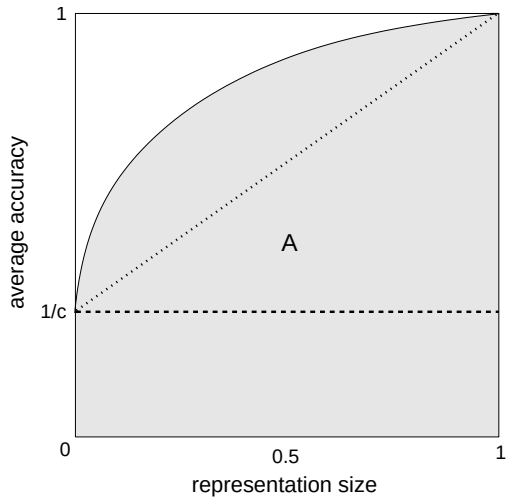


Figure 4.1: *Schematic illustration of a typical Characteristic Accuracy Graph (CAG) and the area enclosed by the curve. The area is used to compute the Average Nearest Neighbor Descriptor (ANND). The CAG curve starts at approx.* $(\frac{1}{m}, \frac{1}{c})$ *and monotonically increases to* $(1, 1)$*. The dotted line represents the theoretical worst case where the dataset contains randomly distributed and randomly labeled (using* c *classes) data points.*

as long as N_{REP} is chosen large enough.[2] In particular, it turns out that in the case of a random dataset, e.g., obtained by using a uniform random distribution and assigning a random labeling to the data points, the curve approximates a straight line connecting the points $(\frac{1}{m}, \frac{1}{c})$ and $(1, 1)$. On the other hand, for a dataset which has a *structured* labeling the curve is located above this straight line.

As illustrated in Fig. 4.1, this property of the CAG motivates us to utilize the value of the area (denoted A in Fig. 4.1) that is enclosed by the curve for the overall measure of complexity of the dataset. The area A can be estimated by:

$$A(\mathbb{D}) = \frac{1}{2m} \sum_{i=1}^{m-1} CAG[i] + CAG[i+1]. \tag{4.4}$$

Based on $A(\mathbb{D})$ we now define the measure — in the following called the *Average Nearest Neighbor Descriptor (ANND)* — as follows:

$$\text{ANND}(\mathbb{D}) := 2\left(1 - \frac{A(\mathbb{D}) - \frac{1}{c}}{1 - \frac{1}{c}}\right). \tag{4.5}$$

The definition is motivated by rescaling the value of the area in a way that leads to invariance to the number of classes c and also bounds the result to the interval $[0, 1]$, where a value of 1 corresponds to the maximal complexity.

[2] For all ANND-experiments presented in the thesis, the parameter N_{REP} is set to 50, which proved to be sufficient for obtaining stable results.

The rescaling is illustrated in Fig. 4.1. The procedure is as follows: First, we subtract $\frac{1}{c}$ from the area $A(\mathbb{D})$ (this is the rectangle enclosed by the dashed line) and rescale the result by dividing it by $1 - \frac{1}{c}$. This makes the measure invariant to the number of classes c. At this point, we obtain a value of approx. 0.5 for randomly labeled data (denoted by the dotted line). For data with structured labeling, we expect the value to be between 0.5 and 1. Therefore, we subtract the current value from 1 and multiply the result by 2 in order to obtain values between 0 and 1 for representing the complexity of the dataset.

4.3 Experiments on Toy Datasets

In this section, we apply the approach to several toy datasets to further illustrate the properties of the proposed measure. The first experiment verifies that we obtain maximum ANND values (close to 1) for datasets that contain randomly distributed classes. This result is invariant to the number of classes, to the dimensionality of the data space and also — as long as sufficiently many points are used – to the number of data points.

In a second experiment we use a 2-dimensional 2-class dataset where the data points are drawn from two Gaussian distributions whose means are varied in distance. In this setup we can analytically compute the theoretical Bayes error of the problem and show that the ANND values obtained for the generated datasets are proportional to the Bayes error. This shows that the measure accounts for the "class ambiguity" issue mentioned at the beginning of the chapter.

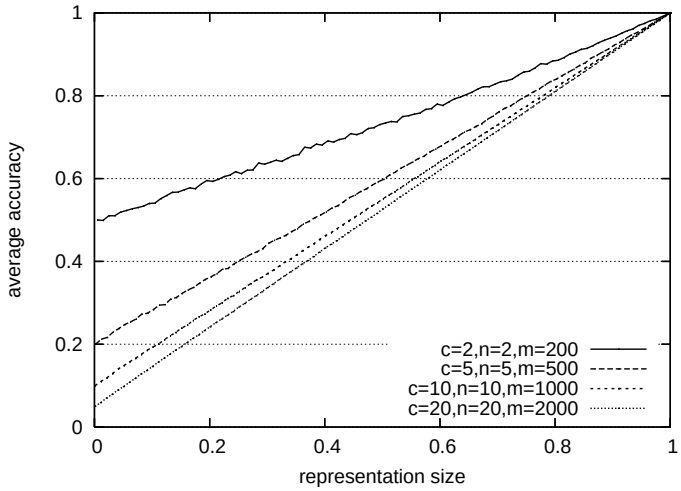
In a third experiment, we use different 2-dimensional datasets that are generated based on gray images that show structures of different complexity levels. First, we use spiral images that have an approximately constant geometrical complexity, but different numbers of classes in order to demonstrate the invariance of the measure with respect to the number of classes. Finally, we use chessboard images in order to generate two-class problems with an increasing geometrical complexity in order to show that the measure accounts for the "complexity of the decision boundary" (the second issue mentioned at the beginning of the chapter) in that it yields higher values the more complex the decision boundary appears.

4.3.1 Experiment 1: The RANDOM-Dataset

The RANDOM-dataset contains c classes and is generated by sampling a number of $\frac{m}{c}$ data points for each class. The dimension of the data points is n and the values are randomly chosen from the interval $[-1, 1]$ using a uniformly distributed random number generator.

For all instances of the RANDOM-dataset we expect the measure to return maximal values (close to 1), since the data does not exhibit any structure which could be exploited by any classification approach.

Figure 4.2 (left) shows four CAGs computed for different settings. In each case, the number of representatives per class is chosen to be 100, i.e. $m = 100 * c$. The number of classes c is set to $2, 5, 10$ and 20 respectively, and the dimension is $n = c$. As expected, the curves approximate straight lines connecting the points $(\frac{1}{m}, \frac{1}{c})$ and $(1, 1)$. The ANND values corresponding to Eq. 4.5 are shown in Fig. 4.2 (right). Again, as expected, all values are close to 1.



c	*n*	*m*	*ANND*
2	2	200	0.999551
5	5	500	0.999378
10	10	1000	0.999367
20	20	2000	0.999378

Figure 4.2: *Left: Characteristic Accuracy Graphs for different instances of the RANDOM-dataset. Classes are distributed following uniform distributions. Right: Corresponding Average Nearest Neighbor Descriptor values.*

This experiment verifies our expectation that random labeling leads to CAGs that approximate straight lines. In the second experiment, we again use datasets that are generated from random distributions but where the labeling exhibits some structure.

4.3.2 Experiment 2: The GAUSS-Dataset

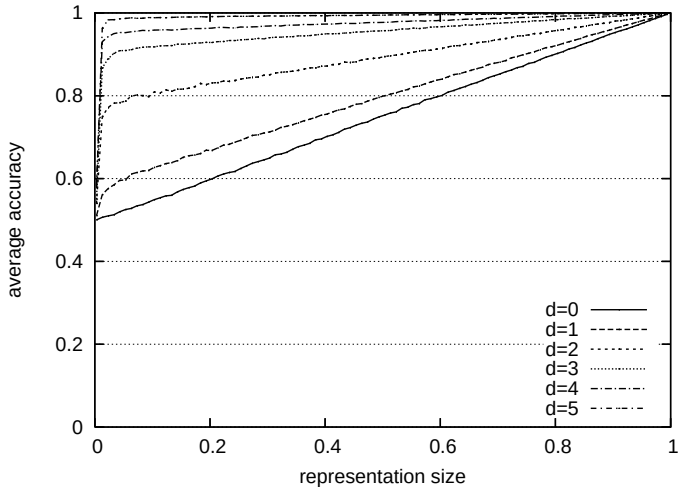
The GAUSS-dataset consist of $c = 2$ classes of data points that are drawn from Gaussian distributions with variance 1 and mean $\vec{\mu}$,

$$G(\vec{x}) = \frac{1}{\sqrt{2\pi}} e^{-\frac{\|\vec{x}-\vec{\mu}\|^2}{2}}, \tag{4.6}$$

where all components of the mean $\vec{\mu}$ are set to 0 for the first class and to $\frac{d}{\sqrt{n}}$ for the second class. The parameter d is then the distance between the means of the two classes. Varying d allows us to introduce structure to the labeling of the dataset in order to test the behavior of the complexity measure.

Figure 4.3 (left) shows examples of CAGs for the GAUSS-dataset. The distance d between the means is chosen to be $0, 1, 2, 3, 4$ and 5, the sample size is $m = 500$, and the dimension is $n = 2$. From the curves it can be seen that in the case of $d = 0$, where the labeling of the dataset does not exhibit any structure (this is equivalent to Experiment 1 with $c = 2$), the CAG forms a straight line between the points $(\frac{1}{500}, \frac{1}{2})$ and $(1, 1)$. As the distance d increases, the curvature of the CAG increases and so does the size of the area enclosed by the CAG.

In Fig. 4.3 (right), the corresponding ANND-values are shown in comparison to the theoretical Bayes error of the problem for the different choices of d, which is given by the overlap of the classes with respect to d:



d	*ANND*	E_{Bayes}
0	0.996	0.9999
1	0.818	0.6171
2	0.475	0.3173
3	0.191	0.1336
4	0.065	0.0455
5	0.030	0.0124

Figure 4.3: *Left: Characteristic Accuracy Graphs for instances of the GAUSS-dataset. Classes are distributed following two Gaussian distributions with variance 1. The distance of the means of the distributions is varied for* $d \in \{0, \ldots, 5\}$. *Right: Corresponding Average Nearest Neighbor Descriptor values and the theoretical Bayes error of the distributions.*

$$E_{Bayes}(d) = \frac{2}{\sqrt{2\pi}} \int_{\frac{d}{2}}^{\infty} e^{-\frac{\|\vec{x}\|^2}{2}} d^n\vec{x}. \tag{4.7}$$

The results show that the decrease of the value of the ANND is proportional to the decrease of the Bayes error. From this we conclude that for the GAUSS-dataset the proposed ANND is able to successfully measure the "structured-ness" of the datasets in the presence of ambiguous classes.

4.3.3 Experiment 3: The COMPLEX-2D-Dataset

Here, we apply the proposed measure on a different class of problems, which have a Bayes error of zero but complex decision boundaries.

For this we generate discrimination problems of dimension $n = 2$ from base images as shown in Fig. 4.4. To obtain a dataset from such an image we randomly sample points from the image plane using a uniformly distributed random number generator. For each point, if the intensity of the pixel at the position is other than white we add the location of the point as an example to the dataset. The labeling of the point is determined by the gray value of the pixel.

First, we generate datasets with $m = 500$ for the spiral images shown in Fig. 4.4 (a-d) and compute the corresponding CAGs, which are shown in Fig. 4.5 (a-d, left). The geometrical complexity of the four problems is approximately equal, just the number of classes c is varied. As expected, the CAG curves each start at an average accuracy value of approx. $\frac{1}{c}$ and rapidly increase to 1 (note that in Fig. 4.5 (a-d, left) the x-axis is restricted to the

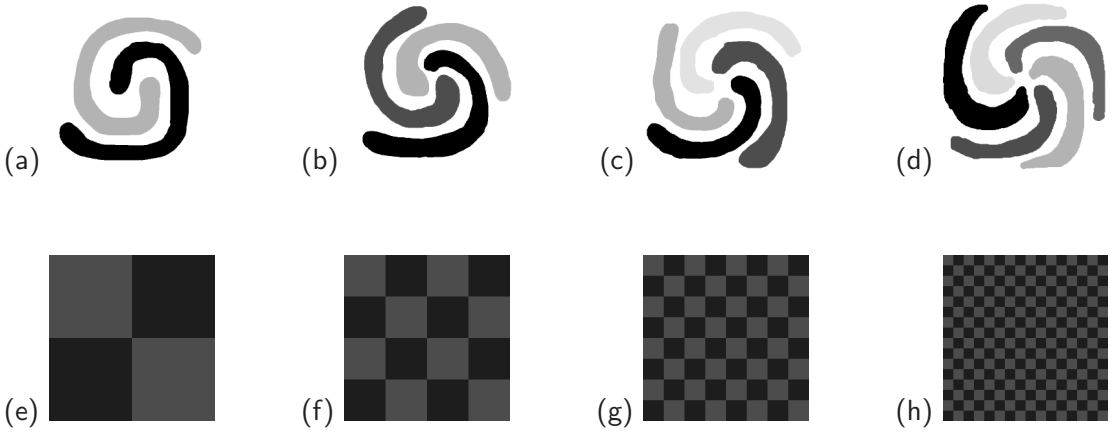


Figure 4.4: *Base images from the COMPLEX-2D dataset, used for generating different two-dimensional test datasets.*

interval $[0, 0.2]$). From the corresponding ANND-values as show in Tab. 4.5 (a-d, right), it can be seen that the measure assigns similar ANND values to four different problems despite the fact that they have different numbers of classes.

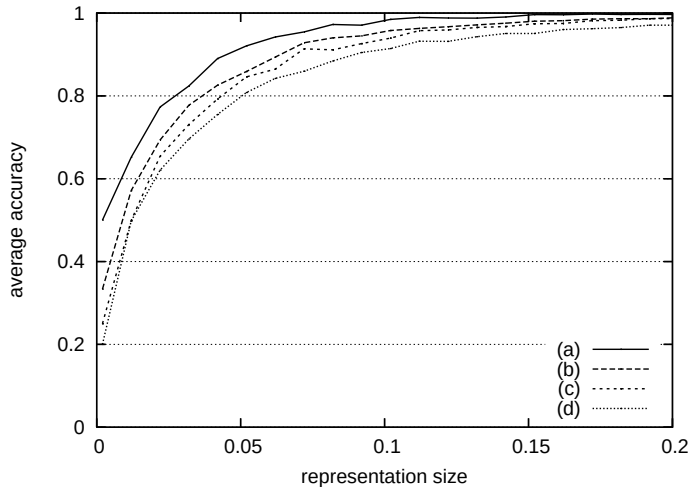
In contrast, the images shown in Fig. 4.4 (e-h) represent four different checkerboard-like distributions. Here, the number of classes is $c = 2$ for all datasets, but the geometrical complexity is varied by increasing the grid dimension of the checkerboard. From the CAGs shown in Fig. 4.5 (e-h), it can be seen that the shape of the CAG flattens for high grid dimensions. In the case of a grid dimension of 16 (Fig. 4.4 (h)), the curve looks similar to the one of the RANDOM-dataset. This is not surprising, since a number of 500 data points is by far not sufficient to describe the structure of 256 grid cells, and the labeling of the data points appears as a random labeling. From the corresponding ANND-values shown in Tab. 4.5 (e-h) it can be seen that the measure assigns higher values to datasets that *appear* to have highly complex decision boundaries.

4.4 Discussion

In this chapter we have proposed an empirical measure that makes it possible to judge the "apparent" complexity of a given multi-class dataset. The measure is derived from the accuracies of simple 1-nearest neighbor classifiers whose node representations are obtained from subsets of increasing size taken from the dataset itself.

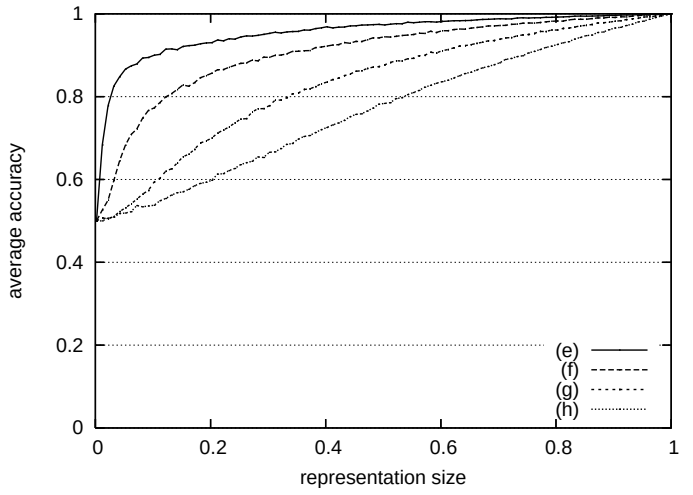
In contrast to former proposals, e.g., by [39] or [20], the measure yields a single, real-valued descriptor that simultaneously accounts for class ambiguity (in the case of a non-zero Bayes error) and for the complexity of the decision boundary — two factors which have been identified by [39] as affecting the complexity of discrimination problems.

Since the measure is free of any additional parameters (the only parameter is the number of repetitions N_{REP}, which must only be chosen large enough in order to ensure stability) it can conveniently be used as an objective function in order to optimize for example the size of a training dataset or the parameters of a feature extraction mechanism.



(a-d)

	ANND
(a)	0.0744
(b)	0.0809
(c)	0.0766
(d)	0.0920



(e-h)

	ANND
(e)	0.1527
(f)	0.4741
(g)	0.9406
(h)	0.9999

Figure 4.5: *Left: Characteristic Accuracy Graphs for the COMPLEX-2D-dataset. The datasets contain $m = 500$ data points that are samples generated from the images in Fig. 4.4. (Right) Corresponding ANND values.*

We argue that the proposed measure provides a highly practical tool for datamining researchers, since it allows analysis of given multi-class data prior to applying a (possibly highly parameterized) classification approach. In the following chapter we apply the ANND measure to analyze natural image datasets and optimizing the parameters of the Hierarchical Feed-forward Model for the task of image normalization.

CHAPTER 5

Image Normalization

In this chapter, the image normalization capabilities of the Hierarchical Feed-forward Model are investigated. The neural activities of the final C-cells of the HFM after processing an input stimulus provide an abstract high dimensional feature representation which is invariant to a substantial range of transformations and distortions that the input stimulus might be subject to. To prove this, multi-class test datasets of highly distorted image patches containing stimuli from three different natural image domains are generated which – in input space – exhibit a poor statistical structure. Passing the images of the datasets through the model and using the Average Nearest Neighbor Descriptor (ANND) as a measure of complexity, it is shown that the HFM is capable of normalizing the images in order to significantly improve the statistical structure of the data. Receptive field profiles of the model are obtained using a recently introduced unsupervised learning method called Non-negative Matrix Factorization with Sparseness Constraints [43]. Profiles learned with this method make it possible to encode knowledge about the current image domain and thus lead to a better performance than fixed receptive field profiles such as those used in former models. This is experimentally confirmed by comparing different choices of receptive field profiles in terms of their success in reducing the apparent complexity of the dataset. The goal of this chapter is to provide a parameterization of the HFM that can be used for the classification and detection experiments in the following chapters. Preliminary results of some of the experiments presented in this chapter have been published in advance in [10].

5.1 Overview

Figure 5.1 show the experimental architecture used in the following and provides an overview of the organization of the chapter. First, in Sect. 5.2 we introduce three publicly available natural image datasets that are frequently used as benchmarks in the computer vision community and that are used for various experiments throughout this thesis. The statistical structure of these datasets is analyzed using the ANND as a measure of complexity. In a series of experiments with synthetic distortions and transformations which are applied on

the datasets it turns out that when adding even a single additional "degree of freedom" to the appearance of the objects a tremendous increase of the complexity can be observed. Moreover, when adding multiple sources of transformations and distortions, the statistical structure is completely lost.

In Sect. 5.3 it is shown that the complexity of the highly distorted test datasets can be reduced again by passing the images through a single- and two-layered HFM and using the activity of the final C-cells of the model as a new representation of the dataset.

In Sect. 5.3.1 a method called Non-negative Matrix Factorization with Sparseness Constraints (NMFSC, [43]) is described. This allows learning of receptive field profiles from natural image data that have similar properties to the receptive fields of biological simple cells as found in visual cortex. As shown in Fig. 5.1, besides learning profiles for the single-layered HFM, higher-order profiles for the second layer of the model can be learned from the outputs of the single layered model.

An example of passing an image through the HFM is discussed in Sect. 5.3.2 and experimental results are given in Sect. 5.3.3, where the performance of the model is evaluated using the ANND and a comparison to other types of receptive field models is made.

5.2 The Statistical Structure of Multi-class Natural Image Datasets

For the experimental investigations in the present and the following chapters, we use as a basis the following three image patch datasets, which are commonly used as benchmarks in computer vision research:

- *The MNIST database of handwritten digits* [53, 52]: A database which is available online[1] and maintained by Yann LeCun (Courant Institute, New York University) and Corinna Cortes (Google Labs, New York). It is composed of a training set of 60,000 examples and a test set of 10,000 examples. The digits are size-normalized and centered in a fixed-size image of 28×28 pixels. From this database, a random subset of 2000 images is selected, rescaled, and embedded into 64×64 frames[2]. Below, this dataset is referred to as MNIST-10. Some example images are shown in Fig. 5.2 (a).

- *The Columbia Object Image Library* [68, 70, 69]: A database of images showing frontal views of small objects recorded using a turntable (rotated in steps of 5 degrees) and controlled lighting conditions. The dataset is available in two versions, one (called COIL-100) contains 7200 color images taken from 100 objects and the other (called COIL-20) contains 1440 gray-scale image images taken from 20 objects. For our experiments we use the COIL-20 library, where each image is rescaled to 64×64 pixels. Some examples are shown in Fig. 5.2 (b).

- *The AT&T Database of Faces* [90, 89]: This database contains 400 gray-scale images of size 92×112 showing frontal views of 40 different people with 10 images each. The database was formerly known as the ORL Face Database. Therefore we refer to the dataset as ORL-40 below. Examples are shown in Fig. 5.2 (c). Since some of the

[1] http://yann.lecun.com/exdb/mnist/

[2] The conversion of the images to 64×64 is done for consistency with the other datasets.

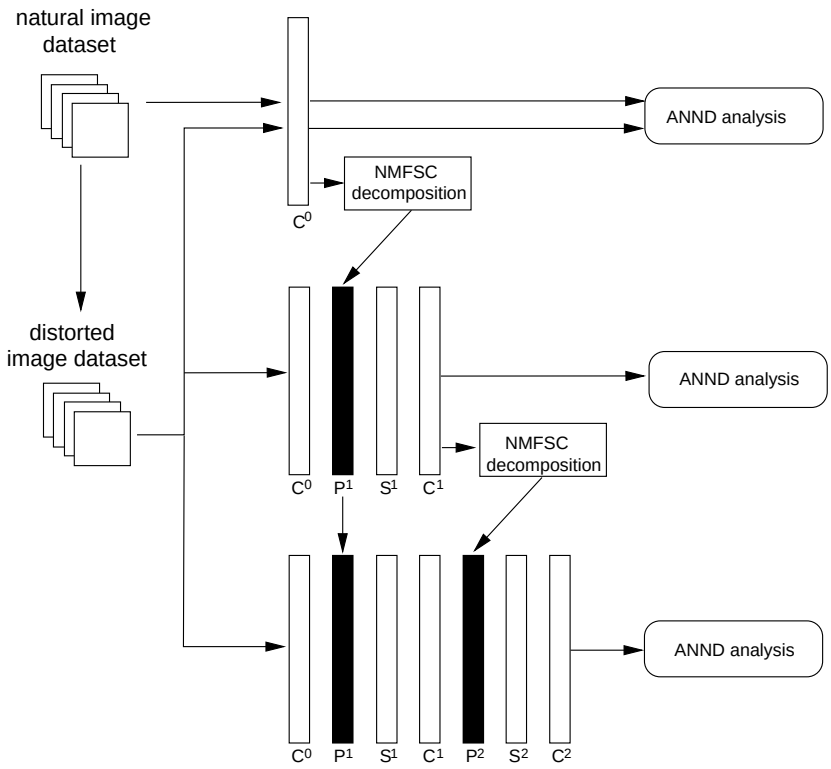


Figure 5.1: *Overview of the experimental HFM architecture used in the present chapter. The Average Nearest Neighbor Descriptor (ANND) is used to analyze the apparent complexity of multi-class natural image datasets under synthetic transformations and distortions (top). The complexity of highly distorted datasets can be significantly reduced by passing the images through a Hierarchical Feed-forward Model (HFM), which consists of a single layer (middle) or of two layers (bottom). Receptive field profiles for the HFM are obtained using an unsupervised learning algorithm called Non-negative Matrix Factorization with Sparseness Constraints (NMFSC, [43]).*

experiments that are conducted in the present and the following chapters require a foreground-background segmentation of the images, the faces are manually segmented and embedded into 64×64 frames. We only use the segmented version of the dataset and refer to it as ORLS-40. Example images are shown in Fig. 5.2 (d).

In order to show that the HFM approach is not restricted to a single image domain, most of the experiments are conducted on all of the three datasets MNIST-10, COIL-20, and ORLS-40. Also, since all datasets are publicly available a good reproducibility of results is given.

5.2.1 Apparent Complexity of Natural Image Datasets

To give some idea of the apparent complexity of natural image datasets, Table 5.1 shows results of an ANND analysis performed on different versions of the three datasets MNIST-10, COIL-20, and ORLS-40.[3] The results for the original, raw datasets of size 64×64 are shown in bold (rows 1, 4, and 7). The apparent complexity of the three datasets differs significantly. The most "difficult" one appears to be ORLS-40 with an ANND value of 0.362. This is not surprising since each class is only represented by 10 different views, which in some cases additionally differ significantly in appearance. The MNIST-10 dataset has a much more dense sampling of 200 examples per class, leading to an ANND value of 0.164. The lowest apparent complexity can be measured for the COIL-20 dataset. Here, a value of 0.113 is obtained, which suggests that the classes are well sampled and in addition the inner-class variance is low.

For comparison, rows 2, 5, and 8 of Tab. 5.1 also show the results for the same datasets, but subsampled to a resolution of 32×32. The values do not differ significantly and are only slightly lower than for the 64×64 resolution. This is probably due to the fact that the Euclidean distance measure which is employed by the ANND algorithm is more stable for the reduced data dimensionality.[4] The ANND value for the original ORL-40 dataset without manual segmentation is shown in row 9 of Tab. 5.1. The apparent complexity is a bit lower for this set, which might be due the background being mostly stable for each class and thus providing a clue to the class identity.

In the following section, we apply different synthetic transformations on the image datasets to see how this affects the apparent complexity. For this, a fixed number of 2000 examples is used for each dataset. In order to create such datasets, randomly selected images from the original datasets are used and Gaussian noise of variance 10 is added to the intensity value of each pixel (the intensity of each pixel is represented by a number between 0 and 255). The datasets are kept balanced, i.e. each class always has the same number of examples. The ANND values for these datasets are shown in rows 3, 6, and 10 (denoted by "GN10"). For the MNIST-10 dataset the apparent complexity increases slightly because the original set already contains 200 examples per class. For the COIL-20 and ORLS-40 datasets, the apparent complexity is significantly reduced since the number of examples per class is larger than in the original set, which means that some examples are almost identical in this base setup. These datasets with a fixed number of 2000 examples are used for the experiments presented in the following section.

[3] For this, we use the Euclidean distance which is applied on the 4096-dimensional image vectors.

[4] This relates to the "curse of dimensionality problem", see [14].

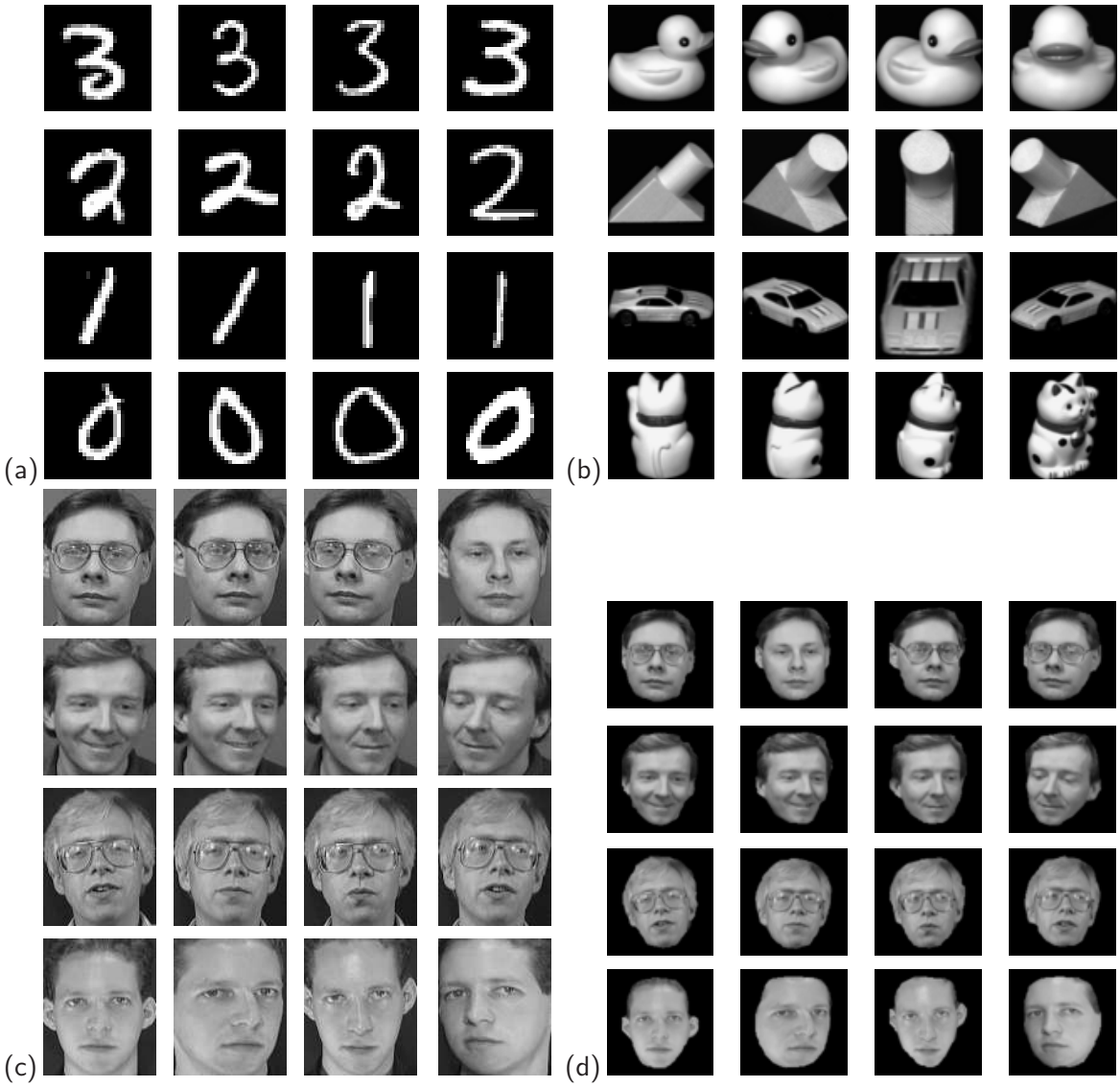


Figure 5.2: *Examples from the multi-class natural image datasets that are used for experiments in this thesis. (a) The MNIST-10 dataset, consisting of 200 examples of each of 10 different handwritten digits. (b) The COIL-20 datasets, consisting of 72 views of each of 20 different small objects. (c) The original ORL-40 datasets consisting of 10 frontal views of each of 40 different people' face. (d) The ORLS-40 datasets, a rescaled and manually segmented version of the ORL-40 dataset. All gray-scale images have a size of* 64×64 *pixels.*

Dataset	*Classes*	*Examples per class*	*Total examples*	*ANND*
MNIST-10, 64×64	**10**	**200**	**2000**	**0.164**
MNIST-10, 32×32	10	200	2000	0.153
MNIST-10, 64×64, GN10	10	200	2000	0.167
COIL-20, 64×64	**20**	**72**	**1440**	**0.113**
COIL-20, 32×32	20	72	1440	0.107
COIL-20, 64×64, GN10	20	100	2000	0.082
ORLS-40, 64×64	**40**	**10**	**400**	**0.362**
ORLS-40, 32×32	40	10	400	0.360
ORL-40, 92×112	40	10	400	0.310
ORLS-40, 64×64, GN10	40	50	2000	0.099

Table 5.1: The apparent complexity of different natural image datasets measured using the Average Nearest Neighbor Descriptor (ANND, see Chapt. 4).

5.2.2 Synthetic Transformations and Distortion

In this section, it is experimentally confirmed that the apparent complexity of natural image datasets is very sensitive to the addition of transformations and distortions to the images. Applying just a single additional source of distortion leads to a tremendous increase of the complexity of the dataset. For the experiment we simulate the following transformations, which commonly appear in practical computer vision setups, for example when observing objects with a camera:

- *Noise:* For simulating the effect that sensor noise has on the complexity of a dataset, we use pixel-wise additive Gaussian noise with an increasing variance. The results in Fig. 5.3 (a) show that the ANND remains quite stable for low variance values, but strongly increases for high values. The ORLS-40 dataset appears to be more sensitive to noise than the other two datasets. This might be due to the fact that classes in the ORLS-40 dataset are distinguished by small details of the face (eyes, nose, etc.) which are corrupted more easily by noise than global shape properties.

- *Rotation, translation, and scaling:* Other transformations of the input stimulus that commonly occur in computer vision setups also strongly affect the apparent complexity of the datasets. Here, we investigate rotation in the image plane, translation in the image plane, and scaling. For rotation, an angle between 0 and a maximum value is chosen for each image and the stimulus is rotated by that angle. The maximum value is increased from 0 to 180 degrees. The resulting ANND values are shown in the plots in Fig. 5.3 (b). For translation in the image plane, an (x, y) pixel offset is randomly chosen where x and y are between $-t$ and t. t is varied between 0 and 10. The results are shown in Fig. 5.3 (c). Finally, for simulating scaling, a random percent value is chosen between $-s$ and s, where s in increased from 0 to 75. The results for this are shown in Fig. 5.3 (d). In all three cases the complexity is strongly affected by the additional source of distortion.

- *Background clutter:* Another source of distortion that must be considered in practical applications occurs when objects appear in front of a cluttered background. This also has a powerful effect on the apparent complexity of the dataset. We simulate this by segmenting the objects from the background (which for our test datasets can be easily done using an intensity threshold) and placing them in front of a 64×64 clutter image patch that is randomly extracted from images of the Art Explosion Photo Gallery [72]. Finally, Gaussian noise of variance 10 is added to each pixel's intensity value. The results for the three datasets are shown in Tab. 5.2, rows 4–6. The apparent complexity strongly increases compared to that of the setup without background clutter (see Tab. 5.1).

- *Multiple sources:* Instead of encountering a single source of additional distortion, in practical computer vision setups multiple sources of transformation often occur simultaneously, which has an even greater effect on the complexity of the datasets. For simulation, we choose the following synthetic transformations, which are applied one after the other: random rotation in the image plane of up to $+/-10°$, random scaling of up to $+/-10\%$, random translation in the image plane of up to $+/-5$ pixels, and additive Gaussian noise with variance 10. The results for the three image datasets are shown in Tab. 5.2, rows 7–9. Additionally applying background clutter leads to a further increase of the complexity, as shown in Tab. 5.2, rows 10–12.[5]

The results of the experiments in this section show that natural image datasets are very sensitive to transformations and distortions of the input stimuli. Adding a single source of distortion such as noise, rotation, scaling, translation, or background clutter leads to a significant increase of the apparent complexity of the dataset in terms of the ANND measure. When multiple sources of distortion are added to an image dataset of limited size, the ANND value quickly approaches value of > 0.5, which means that the statistical distribution of the classes in the Euclidean space is very poor.

For practical computer vision applications which rely solely on statistical methods attempting to model the class distributions on the basis of a training dataset, this has the following implications: *(i)* For each additional source of distortion that the system should be able to tolerate, a tremendous increase of the size of the training dataset is necessary in order to preserve sufficient statistical structure of class distributions that can be exploited during training. *(ii)* When training the system using a limited set of examples, one cannot expect the system to perform well on test datasets which violate the restrictions on the appearance of the objects that have been made during the training phase. When the test images are subject to additional sources of distortion that were not contained in the training set, the internal model of the system is unlikely to account for this additional degree of freedom and recognition is likely to fail.

Therefore, we argue that computer vision systems can greatly benefit from employing heuristic knowledge about natural imagery which is not apparent from a training dataset alone. The Hierarchical Feed-forward Model that is investigated in this thesis makes use of such heuristic knowledge by projecting input stimuli into an abstract high-dimensional feature space. This can be interpreted as a normalization with respect to transformations and distortions that *commonly occur* in natural images. In the remainder of this chapter we

[5] Example images for this are shown in Fig. 6.2.

	Dataset	*Transformations*	*ANND*
1	MNIST-10, 64×64	GN10	0.167
2	COIL-20, 64×64	GN10	0.082
3	ORLS-40, 64×64	GN10	0.099
4	MNIST-10, 64×64	CL, GN10	0.499
5	COIL-20, 64×64	CL, GN10	0.230
6	ORLS-40, 64×64	CL, GN10	0.463
7	MNIST-10, 64×64	RI10, SC10, TR5, GN10	0.211
8	COIL-20, 64×64	RI10, SC10, TR5, GN10	0.331
9	ORLS-40, 64×64	RI10, SC10, TR5, GN10	0.527
10	MNIST-10, 64×64	RI10, SC10, TR5, CL, GN10	0.611
11	COIL-20, 64×64	RI10, SC10, TR5, CL, GN10	0.690
12	ORLS-40, 64×64	RI10, SC10, TR5, CL, GN10	0.747

Table 5.2: ANND values for the three image databases MNIST-10, COIL-20, and ORLS-40 after adding different types of synthetic transformations. CL stands for background clutter, GN10 for additive Gaussian noise of variance 10, RI10 means random rotation in image plane by $+/-10°$, SC10 means random scaling by $+/-10\%$, and TR5 means random translation in the image plane by $+/-5$ pixels.

experimentally investigate the success of the HFM in reducing the apparent complexity of distorted image datasets.

5.3 Image Normalization with the HFM

In this section we investigate the image normalization capabilities of single-layered and two-layered HFMs whose receptive field profiles are obtained from an unsupervised learning method called Non-negative Matrix Factorization with Sparseness Constraints (NMFSC, [43]) which is applied on the input dataset. By computing the ANND on the dataset after projecting the input stimuli into the space of C-cell activities it is shown that *(i)* the HFM is able to successfully recover statistical structure contained in the distorted dataset and that *(ii)* profiles obtained by NMFSC yield a slightly better performance than Gabor filters of randomly initialized receptive field profiles.

In the following it is described how receptive field profiles are learned using NMFSC decomposition, in Sect. 5.3.2 an example of passing an image through the HFM is discussed, and experimental results are given in Sect. 5.3.3.

5.3.1 Learning Receptive Field Profiles

In recent literature there has been an ongoing discussion about whether and how statistical methods can be used to model the formation of receptive field as found in biological simple cells in the primary visual cortex. While the basic properties of simple cells were investigated in the early works of Hubel and Wiesel [45, 46], Barlow [5, 6] was one of the first who

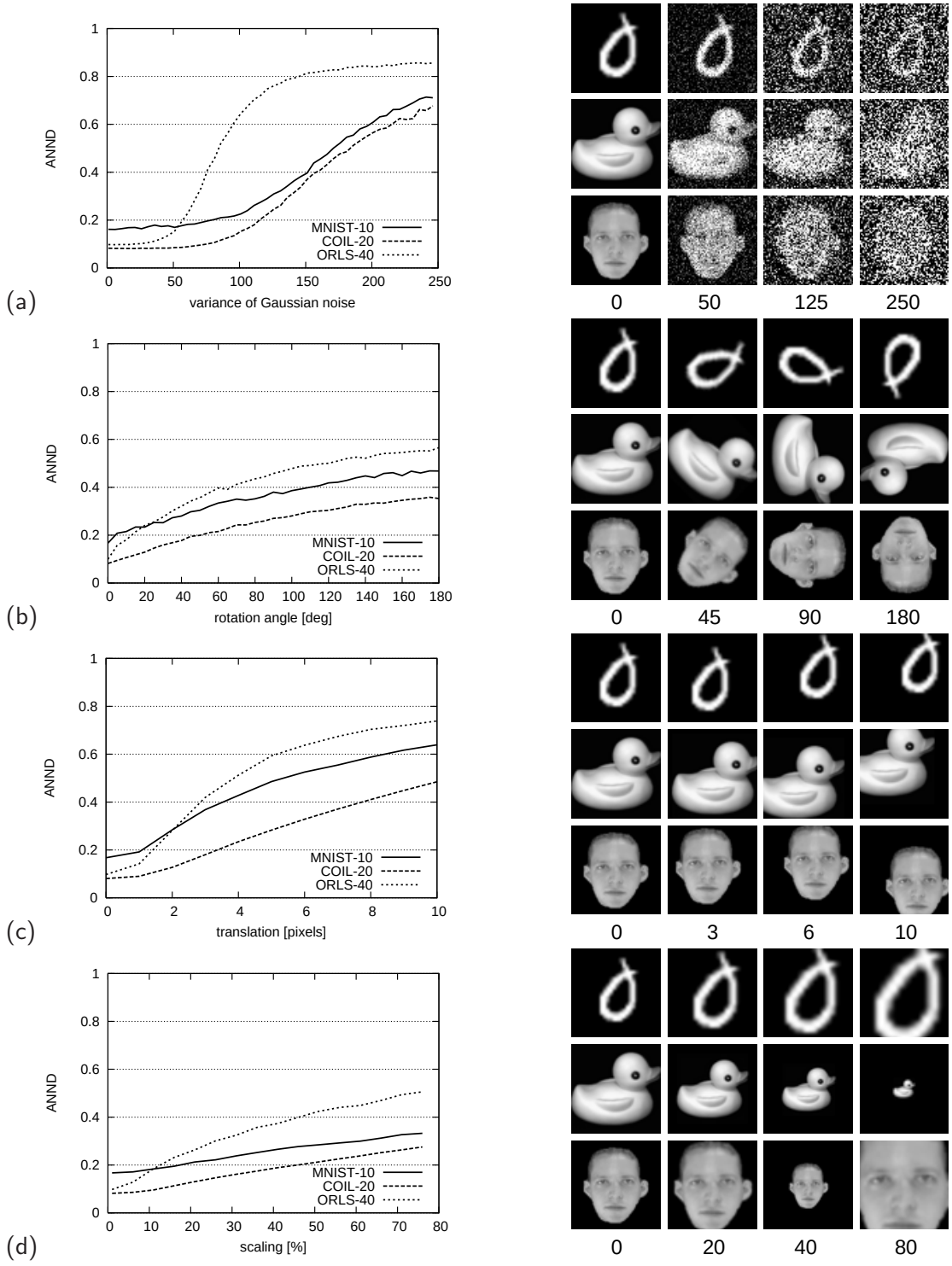


Figure 5.3: *Left: ANND for the three image databases, when (a) adding Gaussian noise of increasing variance to the pixel intensities, (b) randomly rotating the images by +/- an increasing maximum angle, (c) randomly translating the images by +/- an increasing maximum number of pixels, and (d) randomly rescaling the images by +/- an increasing maximum percentage value. Right: Example images (for the extreme values). See text.*

analyzed the behavior of simple cells and suggested that their response properties might emerge from an efficient coding strategy in the sense of information theory.

Since then, a number of proposals have been made for methods that makes it possible to compute receptive field profiles from natural images. Bell and Sejnowski [13] showed that Independent Component Analysis (ICA, [49]) applied on natural image data yields Gabor-like basis components similar to biological receptive fields.

Olshausen and Field [22, 75, 77, 76] suggested using sparse coding to compute an overcomplete basis that accounts for the statistics of natural imagery. In [48, 44, 41] Hyvärinen and Hoyer proposed another model based on sparse coding that learns receptive fields from natural images. In the work of Wersing and Körner [107], a special version of a sparse coding algorithm [106] was successfully applied for learning second-layer receptive field profiles for an HFM similar to the one used in this thesis.

In the following we choose to apply a more recent proposal by Hoyer, called Non-negative Matrix Factorization with Sparseness Constraints (NMFSC, [43]), which relies on an extension of regular Non-negative Matrix Factorization (NMF) [79, 54, 55] and qualitatively subsumes former approaches, because the sparseness of the weight matrix and the latent matrix can explicitly be controlled. This enables the approach to reliably find a decomposition of a natural image ensemble that is made up of constituents that are sparse, localized, oriented, and bandpass [42]. This method is described in the following.

Non-negative Matrix Factorization with Sparseness Constraints

The goal of regular NMF is to decompose a positive data matrix $\mathbf{V}$ into two factors, a weight matrix $\mathbf{W}$ and a latent matrix $\mathbf{H}$ such that $\mathbf{V} \approx \mathbf{WH}$ and $\mathbf{W}$ as well as $\mathbf{H}$ only contain positive values. The data matrix $\mathbf{V}$ has dimension $n \times m$, the weight matrix $\mathbf{W}$ has dimension $n \times r$, and the latent matrix $\mathbf{H}$ has dimension $r \times m$ where n is the dimension of the input space, m is the number of examples, and r is an input parameter that determines the number of basis vectors that the target representation should consist of.

Unlike other linear decomposition methods such as Principal Component Analysis (PCA) or Independent Component Analysis (ICA) [49, 13], the non-negativity constraints employed by NMF lead to a representation of the data that is based on purely additive components. For image datasets this is especially interesting because the method yields a decomposition into sparse local parts that the stimuli consist of [54]. Moreover, applying the method on natural imagery yields a representation that is composed of Gabor-like oriented structures [43], a result which cannot be obtained by for example Hebbian learning or PCA [16].

The quality of the approximation $\mathbf{V} \approx \mathbf{WH}$ can be evaluated by the following error function:[6]

$$E(\mathbf{W}, \mathbf{H}) = ||\mathbf{V} - \mathbf{WH}||^2 = \sum_{i,j} (V_{ij} - (\mathbf{WH})_{ij})^2. \tag{5.1}$$

Minimizing this error function can be done using an iterative algorithm which applies the following multiplicative update rules on the matrices $\mathbf{W}$ and $\mathbf{H}$ in alternating fashion:

$$\mathbf{W} \leftarrow \mathbf{W} \otimes (\mathbf{VH})^T \oslash (\mathbf{WHH}^T) \tag{5.2}$$

[6] Besides using the Euclidian distance as a measure, Lee and Seung [55] also suggest an alternative error function based on the Kullback-Leibler divergence [51]. This option is not investigated here.

and

$$\mathbf{H} \leftarrow \mathbf{H} \otimes (\mathbf{W}^T\mathbf{V}) \oslash (\mathbf{W}^T\mathbf{W}\mathbf{H}) \tag{5.3}$$

where $\otimes$ denotes element-wise multiplication and $\oslash$ element-wise division. As proven in [55], the error of the approximation is non-increasing under these update rules.

An example application in [54] on the CBCL database of face images ([96]) showed that NMF is able to obtain a sparse basis representation which consists of small localized face constituents like e.g., eyes, nose, mouth, etc. However, the author of [43] argues that the sparseness constraints that are employed by regular NMF are only a "side-effect" and that the result cannot always be reproduced for other datasets like for example the ORL face dataset [90, 89], because the sparseness of the basis vectors is not guaranteed. Therefore, he suggested an extension of the original algorithm that allows explicit control of the sparseness of either one, or of both, the weight matrix and the latent matrix.

In order to apply an explicit sparseness constraint on the weight matrix, a sparseness factor μ_W can be chosen between 0 and 1 and in each optimization step the application of Eq. 5.2 is replaced by the following two steps:

1. Set
$$\mathbf{W} \leftarrow \mathbf{W} - \mu_W(\mathbf{W}\mathbf{H} - \mathbf{V})\mathbf{H}^T. \tag{5.4}$$
2. Project each column of W to be non-negative, have an unchanged L_2 norm, and an L_1 norm of μ_W.

The same can be done for the latent matrix by choosing a sparseness factor μ_H between 0 and 1 and replacing Eq. 5.3 by:

1. Set
$$\mathbf{H} \leftarrow \mathbf{H} - \mu_H\mathbf{W}^T(\mathbf{W}\mathbf{H} - \mathbf{V}). \tag{5.5}$$
2. Project each row of H to be non-negative, have an L_2 norm of μ_H, and an unchanged L_1 norm.

For projecting a vector to have target L_1 and L_2 norms, Hoyer describes a special iterative algorithm which finds, for a given input vector, the closest (in the Euclidian sense) non-negative vector that has the given L_1 and L_2 norm. The reader is referred to [43] for details.

As also demonstrated in [43], applying the NMFSC method on ON-OFF-channel separated, contrast filtered natural image data, the method – under an appropriate parameterization of μ_W and μ_H – can be used to obtain a basis representation that consists of vectors which have similar properties like the receptive fields of biological simple cells found in the mammalian visual cortex [76, 22, 77, 5, 6].

We accordingly adopt this method for our purposes in order to obtain receptive field profiles for the HFM. As shown in Fig. 5.1, besides computing profiles from natural image patches which are used on the first layer of the model, we also use the method to compute higher-order profiles from the outputs of a single-layered model to be used on the second layer of the HFM. The procedure is described in detail in the following.

ON-OFF-Channel Separated Data

As illustrated in Fig. 5.4, for computing receptive field profiles with the NMFSC algorithm, ON-OFF-channel separated data is extracted by sampling a large number of random activity patches from C-cell planes after feeding the HFM with random images from the training dataset. After applying the NMFSC algorithm on this data the receptive field profiles are extracted from the resulting weight matrix.

The first step is to create a data matrix $\mathbf{V}$ using Alg. 2. The algorithm gets as input a training set $\mathbb{D}_{train}$, the total number m of examples to be extracted, and the layer index l for which the profiles shall be computed. For each example a random image from the dataset is chosen as well as a random position on the C^{l-1} cell planes. At this position a patch of size $d_p(l) \times d_p(l)$ is extracted from each plane. Then, for each one of these patches the mean activity is computed and subtracted from each activity value. For storing the values in the matrix column that corresponds to the current example, a positive value is written into a reserved ON-slot. For a negative value the sign is changed and it is written into a reserved OFF-slot. The output of the algorithm is a matrix $\mathbf{V}$ containing the ON-OFF-channel separated data. On this matrix the NMFSC algorithm is then applied, setting the desired target number of basis components to the number of planes of layer l, i.e. $r = n_p(l)$.

The output of the algorithm is a weight matrix $\mathbf{W}$ and a latent matrix $\mathbf{H}$. $\mathbf{H}$ is discarded and from the weight matrix $\mathbf{W}$ receptive field profiles P^l are created using Alg. 3. The values are extracted from the columns of $\mathbf{W}$ in the same order as the data was stored during construction of the data matrix $\mathbf{V}$. Here, the OFF-channel is subtracted from the ON-Channel.

Examples of first-layer receptive field profiles that are obtained by applying the described procedure on the three datasets MNIST-10, COIL-20, and ORLS-40 are shown in Fig. 5.5. For the top three rows, the number of planes of layer l, $n_p(l)$, is set to 16 and the dimension of the profiles, $d_p(l)$, is set to 9. For the bottom three rows, $n_p(l)$ was set to 6 and $d_p(l)$ to 5. The results show that NMFSC decomposition yields sparse, localized, Gabor-like profiles.

Typical examples of second layer receptive field profiles are shown in Fig. 5.6. Here, the outputs of a single-layered HFM (using the profiles shown in the bottom three rows of Fig. 5.5) were used to create the data matrix. The number of planes of the second layer, $n_p(2)$ was varied between 8, 16, and 32. It can be seen that the decomposition yields sparse, localized, and oriented spots that are distributed across the 6 input planes.

To obtain the above results, the number of examples, m was set to 5000 and a sparseness constraint was applied on the weight matrix by setting $\mu_W = 0.5$. The sparseness of the latent matrix was left unconstrained. Additional experiments which are not discussed any further here revealed that slight changes to the parameters μ_W and μ_H do not have a string effect on the properties of the basis components. A previous experimental setup is described in [10], where the above parameterization ($\mu_W = 0.5$ and leaving the sparseness of the latent matrix unconstrained) was found to be optimal for a similar task, where second layer receptive field profiles were learned and the classification performance of the resulting HFM was used as an optimization criterion.

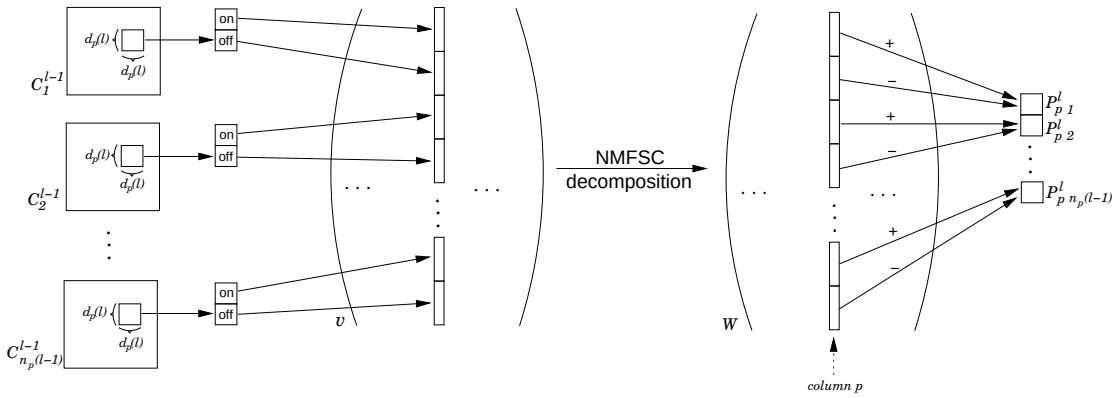


Figure 5.4: *Application of the NMFSC algorithm for creating receptive field profiles for layer l of the HFM using ON-OFF-channel separated data. A column of the data matrix $\mathbf{V}$ is created by extracting an activity patch of size $d_p(l) \times d_p(l)$ from each C-cell planes of layer $l-1$. The data is split into an ON- and an OFF-Channel, as explained in the text.*

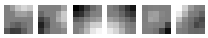
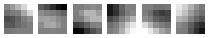
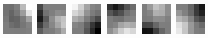
Dataset	$d_p(1)$	$n_p(1)$	
MNIST−10	9	16	
COIL−20	9	16	
ORLS−40	9	16	
MNIST−10	5	6	
COIL−20	5	6	
ORLS−40	5	6	

Figure 5.5: *Examples of receptive field profiles for the first layer of the HFM learned using NMFSC-decomposition, which is applied on ON-OFF-channel separated patches which are extracted from random positions of input images taken from the MNIST-10, COIL-20, and ORLS-40 datasets. The best performance in the current setup is obtained when the number of planes $n_p(1)$ is set to 6 and the dimension of the profiles $d_p(1)$ is set to 5. These profiles are shown in the bottom three rows.*

Algorithm 2 Constructing a data matrix for NMFSC decomposition ($\mathbb{D}_{train}$, l, m)

for $i \leftarrow 1$ to m **do**
  $\mathbf{I} \leftarrow$ random image from $\mathbb{D}_{train}$
  $off_x \leftarrow$ random number between 0 and $d_x(l-1) - d_p(l)$
  $off_y \leftarrow$ random number between 0 and $d_y(l-1) - d_p(l)$
  for $p \leftarrow 1$ to $n_p(l-1)$ **do**
    $sum \leftarrow 0$
    for $x \leftarrow 0$ to $d_p(l) - 1$ **do**
      for $y \leftarrow 0$ to $d_p(l) - 1$ **do**
        $sum \leftarrow sum + C_p^{l-1}(x + off_x, y + off_y;\ \mathbf{I})$
      end for
    end for
    $mean \leftarrow \frac{sum}{d_p(l)^2}$
    for $x \leftarrow 0$ to $d_p(l) - 1$ **do**
      for $y \leftarrow 0$ to $d_p(l) - 1$ **do**
        $j^+ \leftarrow (p-1) * n_p(l-1)^2 + y * n_p(l-1) + x$
        $j^- \leftarrow (n_p(l-1) + p - 1) * n_p(l-1)^2 + y * n_p(l-1) + x$
        if $C_p^{l-1}(x + off_x, y + off_y;\ \mathbf{I}) > mean$ **then**
          $V_{j^+ i} \leftarrow C_p^{l-1}(x + off_x, y + off_y;\ \mathbf{I}) - mean$
          $V_{j^- i} \leftarrow 0$
        else
          $V_{j^+ i} \leftarrow 0$
          $V_{j^- i} \leftarrow -(C_p^{l-1}(x + off_x, y + off_y;\ \mathbf{I}) - mean)$
        end if
      end for
    end for
  end for
end for
return $\mathbf{V}$

Algorithm 3 Obtaining receptive field profiles from a weight matrix ($\mathbf{W}, l$)

for $p \leftarrow 1$ to $n_p(l)$ **do**
  for $q \leftarrow 1$ to $n_p(l-1)$ **do**
    for $x \leftarrow 0$ to $d_p(l) - 1$ **do**
      for $y \leftarrow 0$ to $d_p(l) - 1$ **do**
        $j^+ \leftarrow (q-1) * n_p(l-1)^2 + y * n_p(l-1) + x$
        $j^- \leftarrow (n_p(l-1) + q - 1) * n_p(l-1)^2 + y * n_p(l-1) + x$
        $P^l_{p\,q}(x, y) \leftarrow W_{j^+ p} - W_{j^- p}$
      end for
    end for
  end for
end for
return P^l

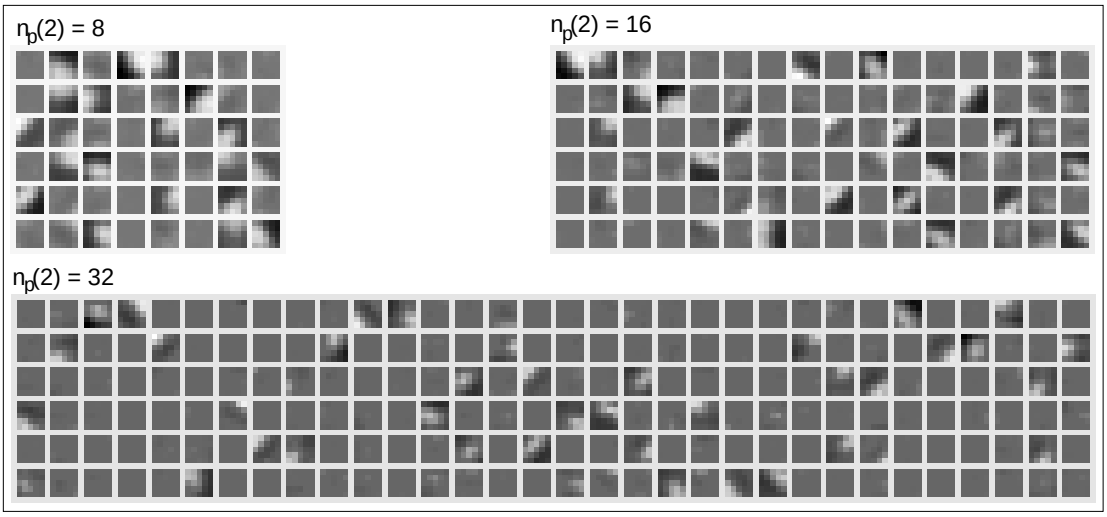


Figure 5.6: *Typical examples of second-layer receptive field profiles obtained by processing an input image dataset by a single layered HFM where the number of planes* $n_p(1)$ *is set to* 6*. Patches are extracted from each* C^1 *cell plane, ON-OFF-channel separation is applied and the resulting vectors are used as input for the NMFSC decomposition algorithm. (For this example, the COIL-20 dataset was used.)*

5.3.2 Processing Example

Figure 5.7 shows an example of passing a distorted test image from the COIL-20 dataset through a two-layered HFM. The profiles of the model are obtained using the NMFSC decomposition procedure as described above.

As can be seen from the activities of the cells on the $\hat{S}^1$ planes, the profiles of the first layer are tuned to different edge orientations. Positive and negative response peaks can be observed at locations where corresponding edges are present.

In the next step, the activities of the $\hat{S}^1$ cells are subject to the *winner-takes-most* lateral competitive mechanism (WTM, [107]). The resulting activities are shown in the S^1 cell planes. The WTM-mechanism yields a "segmentation" of the input with respect to the locally dominant feature.

The last processing step in the first layer of the model is the application of the spatial pooling mechanism, which is implemented by Gaussian convolution and sub-sampling. The result for this is shown in the C^1 cell planes.

As the experimental results in the following section show, we have at this point already achieved a significant normalization of the input image with respect to local transformations and distortions of the image by using the combined activities of the C^1 cell planes as an abstract high-dimensional representation of the input stimulus.

A further normalization is carried out by the second layer of the model. Here, multidimensional receptive field profiles are applied on the output of the first layer. The "meaning" of second layer profiles can be described as follows: The NMFSC decomposition extracts

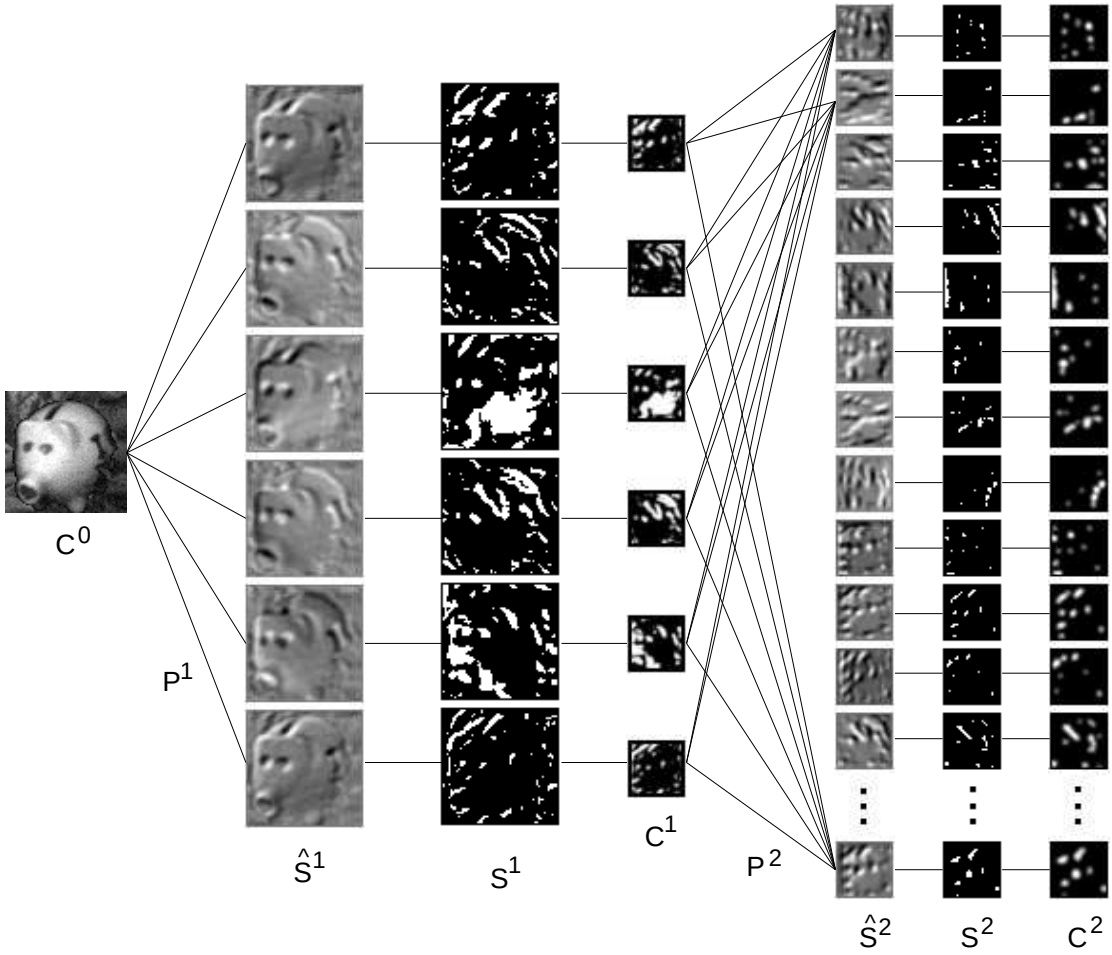


Figure 5.7: *Example of passing a distorted test image from the COIL-20 dataset through a two-layered HFM, whose profiles were obtained using NMFSC decomposition. See text for discussion.*

"typical" local activity patterns that occur on the C^1 cell planes. This leads to a complex ensemble of profiles which for instance respond to co-occurrences of multiple first-layer features (e.g., the meeting points of edges, etc.). The responses of the second layer profiles are shown on the $\hat{S}^2$ cell planes.

Again, the WTM mechanism implements a local competition among the features and yields a segmentation of the input in terms of the most dominant feature. The result of this step is shown in the S^2 cell planes. Finally, the spatial-pooling mechanism is applied on the S^2 cell planes (here, the parameterization was chosen such that no further reduction of the plane size is carried out). The final output of the two-layer HFM is shown on the C^2 cell planes.

We can summarize the underlying strategy which is employed by each layer of the HFM as follows: C-cells achieve a certain degree of "invariance" to spatial translations of local parts of the input stimulus using sub-sampling and smoothing. (This is intuitively clear since for example a translation of a part of the input stimulus by 4 cell positions only appears as a translation of 2 cell positions on a C-cell plane which has been reduced in size by a factor of 2.) However, the spatial pooling mechanism also leads to a loss of – possibly discriminative – information. Therefore, the role of S-cells is to compensate for this loss of information beforehand by performing feature extraction in order to create redundancy in the representation (obviously, the total dimensionality of the S-cells planes is much higher than that of the input C-cell planes of the previous layer).

The experimental results presented in the following section show that the HFM is capable of successfully preserving discriminative information during the normalization.

5.3.3 Experimental Results

For the experimental results as presented in Tab. 5.3, we use the highly distorted test datasets with multiple sources of distortion and transformation generated from the MNIST-10, the COIL-20, and the ORLS-40 datasets by applying random scaling, rotation, translation, background clutter, and noise (for example images see Fig. 6.2). As discussed in Sect. 5.2.2, these datasets have ANND values larger than 0.5 which means that – in input space – they have poor statistical structure (see Tab. 5.2, rows 10–12).

In the following it is shown that the HFM is capable of significantly reducing the complexity of these datasets. For this, we pass the images of the three datasets through different versions of the HFM and use the activities of the final C-cell planes as a new representation of the datasets. Each experiment is repeated 10 times.[7] The average ANND values and the standard deviations are shown in the last two columns of Tab. 5.3.

The column labeled "space" denotes the output space from which the dataset representation is obtained. C^0 (rows 1,9 and 17) refers to the input space (Here, the ANND values are identical to those shown in Tab. 5.2, rows 10–12). C^1 refers to a single-layered HFM and C^2 refers to a two-layered HFM. In the columns "Profiles" different types of receptive field models for the first and the second layer of the HFM are shown. Their performance is investigated in the following.

[7] Each repetition involves generating a new test dataset and recomputing all receptive field profiles.

			Layer 1			Layer 2				
	Dataset	*Space*	*Profiles*	n_p	d_p	*Profiles*	n_p	d_p	*ANND*	*std.dev.*
1	MNIST-10	C^0	-	-	-	-	-	-	0.611	0.0023
2		C^1	RANDOM	6	5	-	-	-	0.416	0.0401
3a		C^1	GABOR	6	5	-	-	-	0.411	0.0072
3b		C^1	GABOR	6	3	-	-	-	0.431	0.0270
3c		C^1	GABOR	4	5	-	-	-	0.413	0.0184
3d		C^1	GABOR	4	3	-	-	-	0.408	0.0170
4		C^1	NMFSC	6	5	-	-	-	0.409	0.0131
5		C^1	NMFSC*	6	5	-	-	-	0.402	0.0121
6		C^2	NMFSC*	6	5	RANDOM	32	5	0.300	0.0669
7		C^2	NMFSC*	6	5	NMFSC	32	5	0.285	0.0197
8		C^2	NMFSC*	6	5	NMFSC*	32	5	**0.275**	0.0134
9	COIL-20	C^0	-	-	-	-	-	-	0.670	0.0013
10		C^1	RANDOM	6	5	-	-	-	0.254	0.0368
11a		C^1	GABOR	6	5	-	-	-	0.229	0.0029
11b		C^1	GABOR	6	3	-	-	-	0.241	0.0100
11c		C^1	GABOR	4	5	-	-	-	0.229	0.0213
11d		C^1	GABOR	4	3	-	-	-	0.228	0.0123
12		C^1	NMFSC	6	5	-	-	-	0.225	0.0140
13		C^1	NMFSC*	6	5	-	-	-	0.218	0.0116
14		C^2	NMFSC*	6	5	RANDOM	32	5	0.195	0.0443
15		C^2	NMFSC*	6	5	NMFSC	32	5	0.180	0.0206
16		C^2	NMFSC*	6	5	NMFSC*	32	5	**0.174**	0.0056
17	ORLS-40	C^0	-	-	-	-	-	-	0.747	0.0085
18		C^1	RANDOM	6	5	-	-	-	0.443	0.0421
19a		C^1	GABOR	6	5	-	-	-	0.377	0.0115
19b		C^1	GABOR	6	3	-	-	-	0.390	0.0038
19c		C^1	GABOR	4	5	-	-	-	0.381	0.0072
19d		C^1	GABOR	4	3	-	-	-	0.379	0.0055
20		C^1	NMFSC	6	5	-	-	-	0.357	0.0433
21		C^1	NMFSC*	6	5	-	-	-	0.351	0.0221
22		C^2	NMFSC*	6	5	RANDOM	32	5	0.339	0.0320
23		C^2	NMFSC*	6	5	NMFSC	32	5	0.331	0.0421
24		C^2	NMFSC*	6	5	NMFSC*	32	5	**0.311**	0.0209

Table 5.3: *Results of the image normalization experiment. Three highly distorted natural image datasets are passed through different variants of the HFM. These differ in the number of layers as well as the employed receptive field profile model. The best performance values (shown in bold) for all datasets are obtained using a two-layered HFM with receptive field profiles that are computed using NMFSC decomposition applied on undistorted example images. See text for discussion.*

Results for a Single-Layered HFM

The results for applying a single-layered HFM on the test datasets using profiles that are obtained by NMFSC decomposition as described in Sect. 5.3.1 are shown in rows 4, 12, and 20 of Tab. 5.3. Compared to the result obtained on the C^0 space a significant reduction of the ANND is achieved for all three datasets. For the "NMFSC" setup the optimal parameterization of the number and the dimension of the profiles was found to be $n_p(1) = 6$ and $d_p(1) = 5$.

An additional improvement of the "NMFSC" setup can be obtained by using the original raw image datasets without synthetic transformations, distortions, and background clutter for training the profiles. This is denoted "NMFSC*" in Tab. 5.3 and the results are shown in rows 5, 13, and 21. In all cases, the performance is slightly better than for the regular "NMFSC" setup. Obviously, providing "clean" training data leads to a better encoding of the properties of the image domain, which might alleviate responses to "unknown" patterns like for example the background clutter.

For comparison we also provide results for two alternative profile models: random profiles and fixed Gabor profiles (denoted "RANDOM" and "GABOR" in Tab. 5.3):

Random profiles, for which the results are shown in rows 2, 10, and 18, are obtained by assigning random values to all elements of P^1 and then normalizing the profiles to have an L_1 norm of 0 and an L_2 norm of 1. Examples of 6 random first layer profiles of dimension 5 are shown in Fig. 5.8 (c). From the resulting ANND values it can be seen that using random profiles on the first layer of the HFM leads to a significant improvement compared to the input space, but the reduction of the complexity and also the standard deviation of the ANND is not as good as for the "NMFSC" and "NMFSC*" setups.

The results for using fixed Gabor profiles (as e.g. done in the work of Wersing & Körner [107]) are shown in rows 3, 11, and 19 (a–d). Here, the number of profiles $n_p(1)$ is set to 4 and 6 and for the dimension $d_p(1)$, 3 and 5 are used.[8] The orientations of the even Gabor filters are set to 0°, 45°, 90° and 135°. The profiles are shown in Fig. 5.8 (a) and (b). In order to reproduce the original setup described in [107], a slight modification of the computation of the S-cell responses (Eq. 3.1 in Chapt. 3) was undertaken, such that the absolute value of the profile responses is taken. The resulting ANND values show that the "GABOR" setup clearly outperforms the "RANDOM" setup in terms of the ANND and the standard deviation of the result. However, the performance remains slightly below that of the "NMFSC*" setup.

Results for a Two-Layered HFM

In the results presented above, the best setup for the single-layered HFM turned out to be the NMFSC* setup using 6 profiles at a dimension of 5. Therefore, this parameterization of the first layer is used for the investigations of the two-layered model.

For this, we compare the three setups "NMFSC" (profiles trained on the distorted test dataset) "NMFSC*"(profiles trained on the raw datasets), and "RANDOM" (as above,

[8] In [107], the authors used $d_p(1)$=3 and $n_p(1)$=4. For better comparison, we also provide results for the same parameterization as used for the other profile models that are investigated here. The results show however, that increasing the number of profiles or the dimension does not significantly influence the performance of the Gabor profiles.

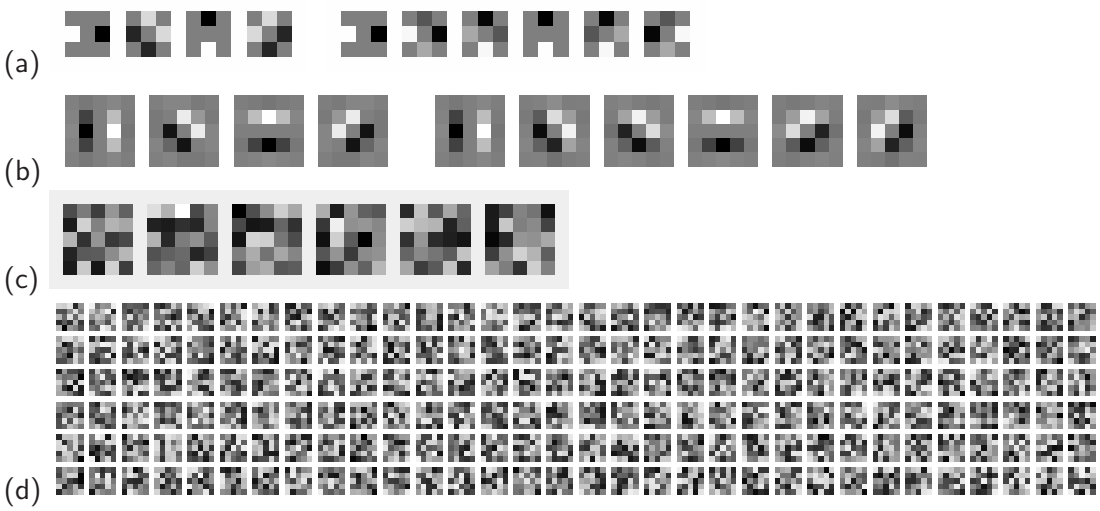


Figure 5.8: *Alternative receptive field profiles. (a) First-layer Gabor filters of dimension $d_p = 3$ (for $n_p = 4$ and $n_p = 6$). (b) First-layer Gabor filters of dimension $d_p = 5$ (for $n_p = 4$ and $n_p = 6$). (c) Set of 6 first-layer random profiles of dimension 5. (d) Set of 32 random profiles of dimension 5 to be used as second layer profiles, where the number of planes of the first layer is 6.*

profiles are chosen randomly and normalized plane-wise to have an L_1 norm of 0 and an L_2 norm of 1, examples are shown in Fig. 5.8 (d)). In all cases we choose to set the dimension of the profiles $d_p(2)$ to 5 and the number of the profiles $d_n(2)$ to 32.[9]

From the results it can be seen that the "NMFSC*" setup (rows 8, 16 and 24) yields the best performance in terms of the ANND as well as the standard deviation. The regular "NMFSC" setup (rows 7, 15, 23) and the "RANDOM" setup (rows 6, 14 and 22) remain slightly below the "NMFSC*" setup but still have a reasonable performance.

Discussion

From the above results we can conclude that the apparent complexity of highly distorted image datasets of fixed size can indeed be significantly reduced by using the HFM approach. The best performance is achieved when using a two-layered HFM with profiles that are learned from the input dataset using the NMFSC decomposition method as described in Sect. 5.3.1. Here, it was found that by using the raw image datasets instead of the highly distorted test dataset for training the profiles it is possible to further improve the quality of the encoding.

For comparison, we also considered randomly chosen receptive field profiles for both the single- and the two-layered model. The results obtained for this setup are also remarkably high, though not quite as high as for the NMFSC setup and also not as stable (in terms of the standard deviation of the ANND).

[9] As shown in the experiments in Chapt. 6, setting the number of planes of the second layer to 32 is not necessarily the optimal choice. For some applications, a further improvement can be achieved when even more planes are used.

For the single layered model we also compared the results to a former approach which relies on a set of oriented Gabor filters used as receptive field profiles. Here, we reproduced the original setup by Wersing & Körner [107] and showed that the ANND remains slightly below that for the NMFSC setup, but still a reasonable and stable performance can be reached.

Looking at the lowest ANND values obtained for the three datasets it can be seen that the approach works significantly better for the COIL-20 dataset than for the MNIST-10 and ORL-40 datasets. This can be explained by the special properties of the datasets: The objects of the COIL-20 datasets differ significantly not only in global shape but also in textural details, whereas the digits of the MNIST-10 can only be distinguished by shape and the faces of the ORLS-40 datasets only differ in textural details. Therefore the success of the HFM in reducing the ANND value is most successful for the COIL-20 dataset.

5.4 Discussion

In the first part of this chapter we have used the previously introduced ANND measure in order to analyze the statistical structure of three multi-class natural image datasets covering the domains of handwritten digits, small objects and faces.

We have simulated a number of transformations and distortions that commonly occur in natural imagery using synthetic manipulations of the images. Here it was found that by adding just a single additional "degree of freedom" to an image dataset the apparent complexity in terms of the ANND measure increases dramatically. When applying multiple sources of transformation and distortion such as scaling, rotation, translation, noise, and background clutter, the statistical structure of a dataset of fixed size becomes very poor.

In the second part of this chapter we showed that the HFM is able to significantly improve the statistical structure of such highly distorted datasets by exploiting heuristic knowledge about natural imagery. This knowledge is encoded partly in the topology of the model and partly in the receptive field profiles that the model employs for feature extraction.

To prove this, an experimental setup was used where the highly distorted test datasets were passed through different single- and a two-layered HFMs. Here it was shown that a recently proposed unsupervised learning scheme called Non-negative Matrix Factorization with Sparseness Constraints can be used to adapt the model to the image domain in order to improve the quality of the normalization.

Throughout this chapter the proposed Average Nearest Neighbor Descriptor (ANND) proved to be a useful tool for measuring the suitability of the HFM for the task of image normalization. It allowed us to conveniently compare different parameterizations of the HFM and different receptive field profile models using multi-class datasets. However, the ANND is a rather abstract quantifier that is well suited for use as a relative comparison measure rather than an absolute performance measure. Therefore, in the following two chapters we apply the HFM for concrete classification tasks using the parameterization that has been found to be optimal for the normalization task described in this chapter.

Parameter Details

The parameter details for the experimental results presented in Tab. 5.3 and shown in Tab. 5.4.

	MNIST-10	*COIL-20*	*ORLS-40*
Single-layered HFM, setups "NMFSC", "NMFSC" and "RANDOM"*			
layer 1 S-cell model	LINEAR	LINEAR	LINEAR
$d_x(1)$	16	32	32
$d_y(1)$	16	32	32
$n_p(1)$	6	6	6
$d_p(1)$	5	5	5
γ_1	0.91	0.91	0.91
θ_1	0.1	0.1	0.1
σ_1	1.2	1.2	1.2
Single-layered HFM, setup "GABOR"			
layer 1 S-cell model	ABS	ABS	ABS
$d_x(1)$	16	32	32
$d_y(1)$	16	32	32
$n_p(1)$	4,6	4,6	4,6
$d_p(1)$	3,5	3,5	3,5
γ_1	0.91	0.91	0.91
θ_1	0.1	0.1	0.1
σ_1	1.2	1.2	1.2
Two-layered HFM, setups "NMFSC", "NMFSC" and "RANDOM"*			
layer 1 S-cell model	LINEAR	LINEAR	LINEAR
$d_x(1)$	16	32	32
$d_y(1)$	16	32	32
$n_p(1)$	6	6	6
$d_p(1)$	5	5	5
γ_1	0.91	0.91	0.91
θ_1	0.1	0.1	0.1
σ_1	1.2	1.2	1.2
layer 2 S-cell model	LINEAR	LINEAR	LINEAR
$d_x(2)$	16	32	32
$d_y(2)$	16	32	32
$n_p(2)$	32	32	32
$d_p(2)$	5	5	5
γ_2	0.91	0.91	0.91
θ_2	0.1	0.1	0.1
σ_2	0.5	0.5	0.5

Table 5.4: Model parameters for the HFM architecture for image normalization as used to obtain the experimental results shown in Tab. 5.3.

CHAPTER 6

Image Patch Classification

In this chapter, the Hierarchical Feed-forward Model is applied to the problem of image patch classification. The image normalization capabilities of the HFM can be exploited for building powerful classifiers that operate on the output space of the HFM and achieve a high classification performance on difficult test datasets while using training datasets of small size. For this, comparatively simple classification techniques prove to be sufficient for outperforming standard approaches that operate on the raw input image space. First, we investigate the use of so-called Template View Tuned Units (T-VTUs) that correspond to simple 1-nearest neighbor classifiers in the C-cell plane activity space of the HFM. Next, we propose a method that utilizes a spectral clustering technique in order to obtain Condensed View Tuned Units (C-VTUs) allowing for high recognition rates on severely distorted test datasets using only a very low number of units. For confidence based recognition, i.e. classification with the ability to reject "unknown" stimuli, two possibilities are investigated: (i) Linear Discriminant Functions (LDFs), which are trained supervised from T-VTU representations, and (ii) Radial Basis Functions (RBFs), whose centers are obtained from C-VTUs.

6.1 Overview

Figure 6.1 provides an overview of the experimental architecture used in this chapter. First, in Sect. 6.2 we pass training image patches from the MNIST-10, the COIL-20, and the ORLS-40 datasets through the HFM and record the activity of the final C-cell planes to obtain so-called *Template View Tuned Units* (T-VTUs, [107, 85]). For recognition, image patches from a test dataset are passed through an identically parameterized HFM and the resulting activities are matched against the stored T-VTU representation using a nearest-neighbor decision rule [18].

Experimental results on severely distorted test datasets show that T-VTUs in the C-cell plane activity space of a single- and a two-layered HFM significantly outperform the standard 1-nearest neighbor classifier operating on the raw input image space. This verifies the predictions of the ANND analysis that was carried out in the preceding chapter and

shows that the HFM can be used to create more powerful classifiers by applying a simple 1-nearest neighbor technique at different levels of the HFM.

However, one problem of the T-VTU approach is that during recognition for each image patch, its final C-cell plane activity must be matched to *all* T-VTUs that the current representation consists of, which is computationally expensive. As the experimental results in Sect. 6.2 reveal, in order to achieve a reasonable classification accuracy on the chosen datasets, a large number of units has to be used.

Therefore, in Sect. 6.3 a novel method is proposed that uses a spectral clustering technique for reducing the number of units used for representing the classes in the C-cell plane activity space. The strategy of the method is as follows: First, a large number of T-VTUs is generated from a distorted set of training image patches. Then, spectral clustering is used to partition each class representation into disjoint subsets and finally, the means of the partitions are computed and used as new units for classification. Since these units do not correspond to individual views of the objects any longer, they are called *Condensed View Tuned Units* (C-VTUs) below. In an experiment it is shown that C-VTUs outperform T-VTUs for small representation sizes.

In Sect. 6.4, we provide a comparison the results on the image patch classification task to an eigenspace method called "VPL" in order to show that the HFM outperforms standard classification techniques that solely rely on analyzing the statistical properties of the training datasets in input image space.

In the above formulation of the problem of image patch classification it was assumed that each image patch always contains an object belonging to the "known" set of classes and the task was to choose the correct class identifier from this set. However, in practical computer vision setups, it is most often required to also be able to detect the case when the input image contains an "unknown" object or no object at all. In Sect. 6.5 we refer to this extension of the discrimination problem as *confidence based recognition*.

One way to achieve confidence based recognition is to employ a supervised learning strategy for optimizing a set of *Linear Discriminant Functions (LDFs)* using a given T-VTU representation as a training set. Each one of these functions is then responsible for "detecting" one specific class and it is trained to respond strongly to examples of "its" class and to respond weakly to examples of other classes. This way, the responses of LDFs can be interpreted as confidence values and rejection of "unknown" stimuli can then be achieved by introducing a threshold parameter on the responses of LDFs. This approach is discussed in Sect. 6.5.1.

In Sect. 6.5.2, we investigate an alternative option that does not rely on supervised learning but directly uses a given C-VTU representation for constructing a set of *Radial Basis Functions (RBFs)*. Again the responses of these functions can be interpreted as confidence values and a threshold parameter makes it possible to achieve classification and also rejection of unknown stimuli.

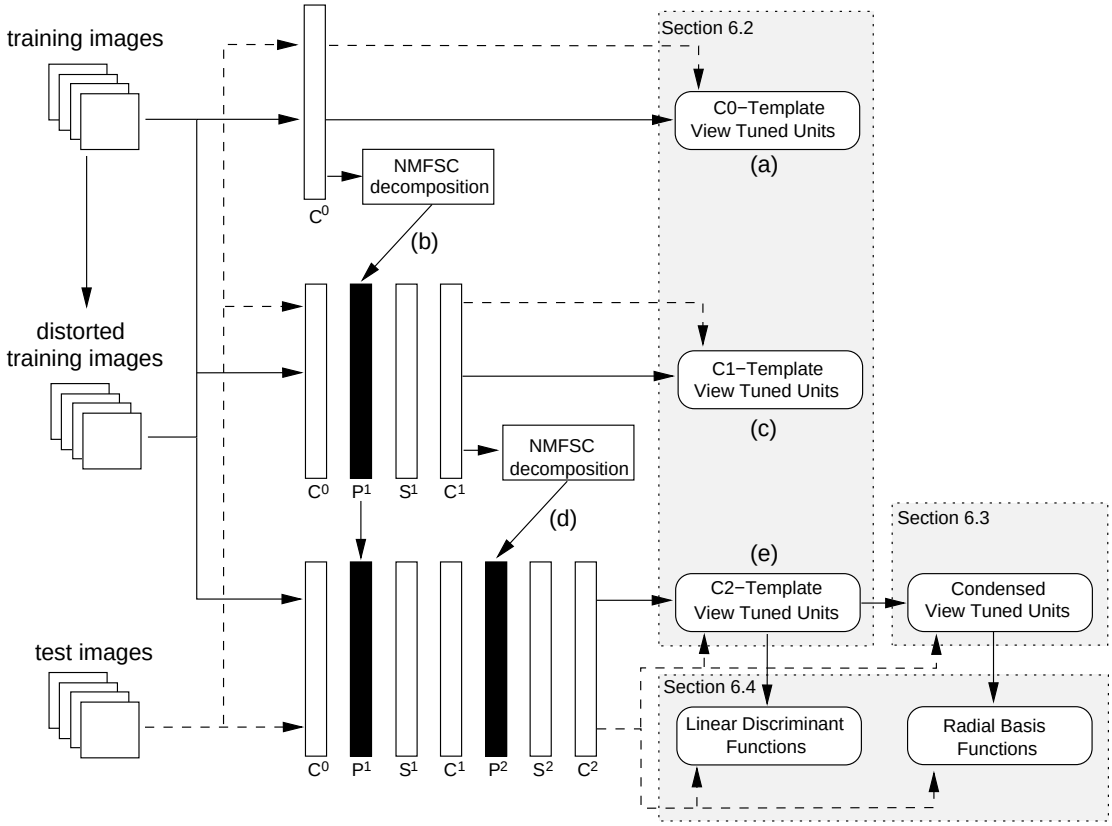


Figure 6.1: *Experimental architecture used for the experiments in this chapter. Solid lines depict training cues and dashed lines test cues. Three variants of Template View Tuned Units (T-VTUs) are obtained by recording the final C-cell plane activity from C^0, C^1, or C^2 of the HFM after processing either raw or distorted training images. Section 6.2 deals with an experimental comparison of the classification abilities of these three representations. In Sect. 6.3 a method is described for creating Condensed View Tuned Units (C-VTUs) from C2-Template View Tuned Units. Finally, Sect. 6.5 describes confidence based recognition using linear discriminant functions and radial basis functions.*

6.2 Template View Tuned Units

Given a collection of m training image patches $\mathbf{T}_i$ of size $d_x \times d_y$,

$$\mathbb{D}_{train} = \{\mathbf{T}_i \mid \mathbf{T}_i \in \mathbb{R}^{d_x} \times \mathbb{R}^{d_y}, i = 1, \ldots, m\}, \tag{6.1}$$

and a function $l_{train}(\mathbf{T}_i) \in \{1, \ldots, c\}$ storing the class identifier of the ith image chosen from a set of c class identifiers, we consider three different T-VTU representations that are obtained by passing the training image patches through the HFM and recording the C-cell plane activities at the input layer C^0 (which is simply the vectorial representation of the input image) at the first layer C^1 and at the final layer C^2:

$$\mathbb{T}_{C^{\{0,1,2\}}} = \{(C^{\{0,1,2\}}(\mathbf{T}_i), l_i) \mid \mathbf{T}_i \in \mathbb{D}_{train}, l_i = l_{train}(\mathbf{T}_i)\}, \tag{6.2}$$

Each one of these three T-VTU representations consists of a set of tuples each containing a C-cell plane activity vector and a class identifier. These T-VTU representations can be used to classify a given test image patch $\mathbf{I}$ by passing it through the HFM and matching the C-cell plane activities at the corresponding layer against all T-VTUs using the nearest neighbor decision rule. For this we define the following three classifiers:

$$\Phi^{TVTU}_{C^{\{0,1,2\}}}(\mathbf{I}) := \arg\min_{l_i} \; dist(C^{\{0,1,2\}}(\mathbf{I}), \vec{t_i}), \; \forall(\vec{t_i}, l_i) \in \mathbb{T}_{C^{\{0,1,2\}}}. \tag{6.3}$$

Given a set of t test images

$$\mathbb{D}_{test} = \{\mathbf{I}_i \mid \mathbf{I}_i \in \mathbb{R}^{d_x} \times \mathbb{R}^{d_y}, i = 1, \ldots, t\}, \tag{6.4}$$

together with a function $l_{test}(\mathbf{I}_i) \in \{1, \ldots, c\}$ storing the true class identifiers of the test images, the classification accuracy is measured by:

$$acc^{TVTU}_{C^{\{0,1,2\}}}(\mathbb{D}_{test}) := \frac{1}{t} \sum_{i=1}^{t} \delta(\Phi^{TVTU}_{C^{\{0,1,2\}}}(\mathbf{I}_i), l_{test}(\mathbf{I}_i)), \tag{6.5}$$

where $\delta(a, b) = 1$ if $a = b$ and 0 else.

6.2.1 Test Datasets

In the following we compare the classification performance of the three classifiers defined in Eq. 6.3 using the following three test datasets, each consisting of 2000 images:

- $\mathbb{D}^{MNIST-10}_{test}$: The original MNIST-10 dataset consists of 200 examples of each of the digit 0 to 9. For this test set, for each of the 10 classes, 200 examples are randomly selected from the odd-numbered images (the even numbered images are used for training). Additionally, each of these 2000 image patches is subject to the following synthetic transformations: Random rotation in image plane by $+/-$ 10^o, random scaling by $+/-$ 10 %, random translation of $+/-$ 5 pixels, randomly selected background clutter (taken from the Art Explosion Photo Gallery [72]), and pixel-wise additive Gaussian noise with variance 10. Some example test images for the MNIST-10 dataset are shown in Fig. 6.2 (a).

- $\mathbb{D}_{test}^{COIL-20}$: This dataset consists of 100 views of each object from the COIL-20 dataset which are again randomly selected from the odd-numbered views. The images are subject to the same synthetic transformations as for the $D_{test}^{MNIST-10}$ dataset. Example images are shown in Fig. 6.2 (b).

- $\mathbb{D}_{test}^{ORLS-40}$: For this dataset, 50 example views are generated for each of the 40 persons contained in the ORLS-40 dataset. Again, the images are randomly selected from the odd-numbered views and the same synthetic transformations are added as for the $\mathbb{D}_{test}^{MNIST-10}$ and the $\mathbb{D}_{test}^{MNIST-10}$ datasets. Fig. 6.2 (c) shows some examples.

6.2.2 Experiment: Classification with Undistorted T-VTUs

For a first T-VTU classification experiment, we use for training undistorted raw images from the three datasets MNIST-10, COIL-20, and ORLS-40 and for testing the severely distorted datasets as described in the preceding section. The raw training images are chosen randomly from the even-numbered images of the three sets. Here, no synthetic transformations are applied. The number of images in each training set (which corresponds to the number of units in the T-VTU representation) is varied, starting at the number of classes (10, 20, and 40 for MNIST-10, COIL-20, and ORLS-40 respectively) and is increased in steps of 40 until a representation size of 2000 units is reached.[1]

As depicted in Fig. 6.1, each training dataset is first used to create a C^0-T-VTU representation by simply normalizing the images (a). Simultaneously, the training images are used to create the receptive field profiles P^1 of the single-layered HFM by NMFSC decomposition (b). Then, the training images are passed through the single-layered HFM in order to create a C^1-T-VTU representation (c) and to obtain second layer receptive field profiles P^2 (d). The final C^2-T-VTU representation for each training set is obtained by passing the images through the two-layered HFM (e).

For testing, the training image patches (bottom left in Fig. 6.1) are passed through the identical architecture and matched against the three T-VTU representations in order to obtain the classification results and measure the classification performance according to Eq. 6.5.

Figure 6.3 shows the results of the experiment. The three curves in each of the three figures (a), (b), and (c) represent the classification accuracies $acc_{C^0}^{TVTU}$, $acc_{C^1}^{TVTU}$, $acc_{C^2}^{TVTU}$ vs. the number of units. The vertical bars represent the standard deviation for 5 repetitions, using differently chosen training and test datasets.

It can be seen that for all three image domains the classifiers operating on the C-cell plane activity space of the HFM ($\Phi_{C^1}^{TVTU}$ and $\Phi_{C^2}^{TVTU}$) clearly outperform the standard 1-nearest neighbor classifier which uses the raw image space ($\Phi_{C^0}^{TVTU}$). Also, it can be observed that the $\Phi_{C^2}^{TVTU}$ classifier slightly outperforms the $\Phi_{C^1}^{TVTU}$ classifier for each of the three image domains.

Table 6.1 shows a summary of the final classification accuracy for the three classifiers and the three datasets for a representation size of 2000 units. For the MNIST-10 and the COIL-20 datasets, the $\Phi_{C^2}^{TVTU}$ classifiers exhibits an improvement in performance of approx. 100% compared to the $\Phi_{C^0}^{TVTU}$ classifier. The fact that the performance increase is rather low for

[1] All datasets are always kept balanced, i.e. all classes are represented by the same number of examples.

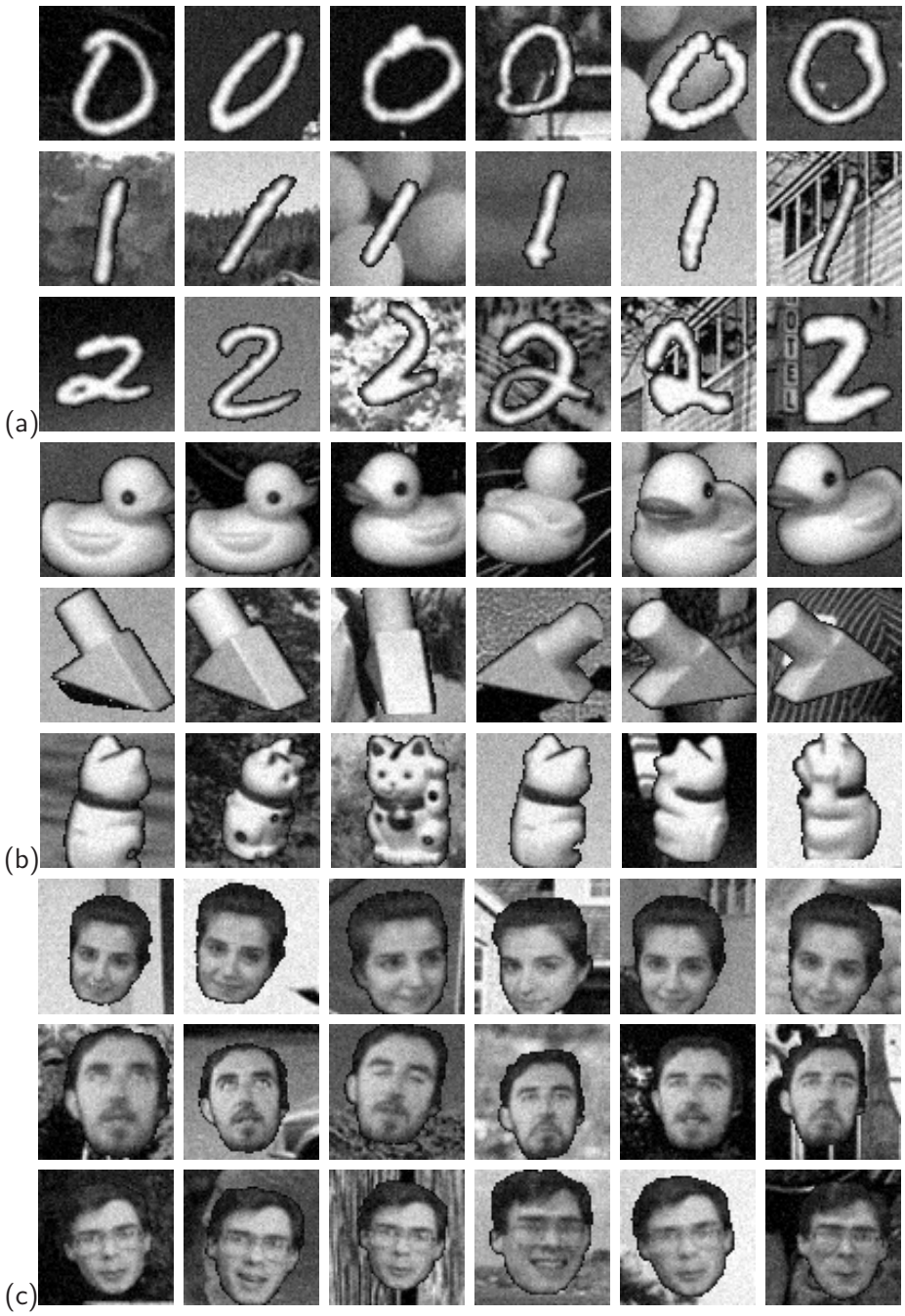


Figure 6.2: *Example images from the three test datasets used for the T-VTU classification experiment. (a) Dataset* $\mathbb{D}_{test}^{MNIST-10}$. *(b) Dataset* $\mathbb{D}_{test}^{COIL-20}$. *(c) Dataset* $\mathbb{D}_{test}^{ORLS-40}$. *The images are randomly selected from the odd-numbered views of the original datasets and subjected to synthetic transformation such as rotation in image plane, scaling, translation in image plane, background clutter, and additive Gaussian noise. See text for details.*

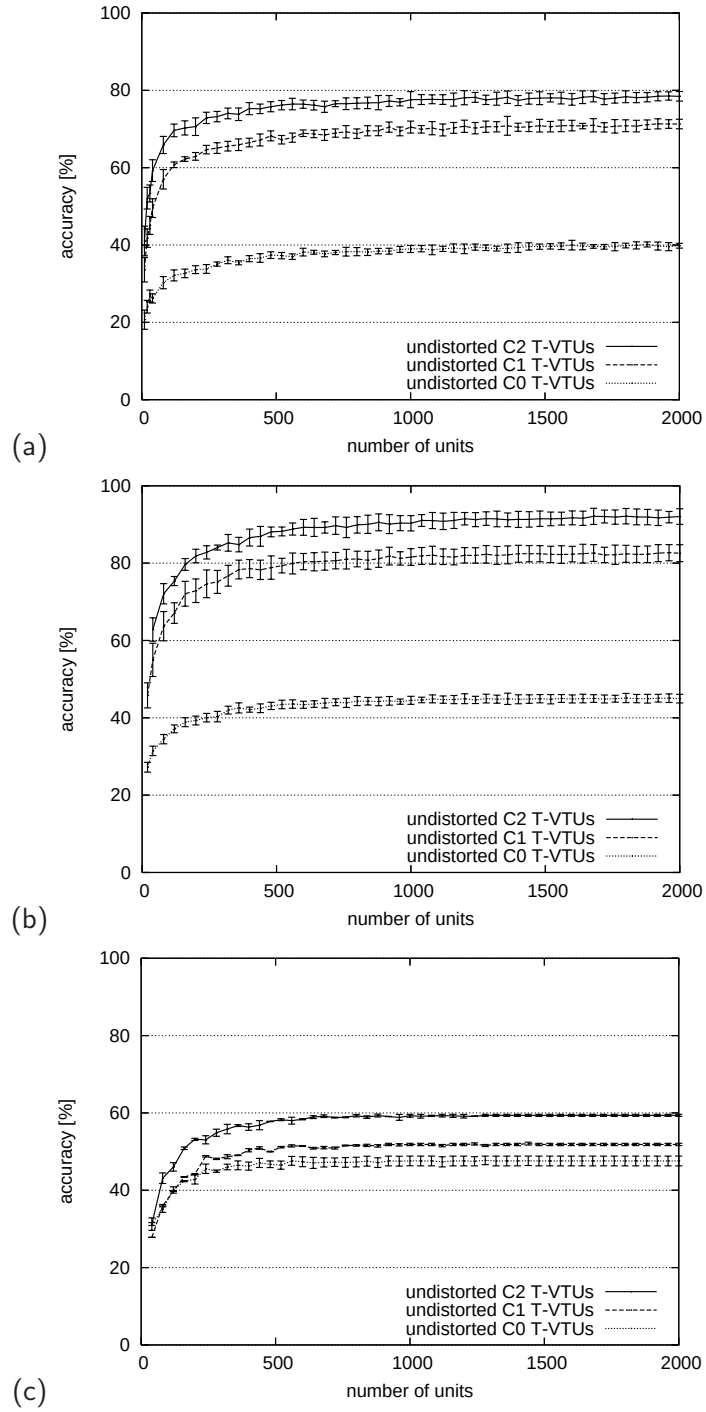


Figure 6.3: *Comparison of the classification performance of three variants of undistorted T-VTUs which operate at three different levels of the HFM: the C^0 (which is simply the raw input image space), the C^1 space, and the C^2 space. (a) Results for the MNIST-10 dataset. (b) Results for the COIL-20 dataset. (c) Results for the ORLS-40 dataset. In each case the training sets consists of an increasing number of undistorted raw images that are randomly selected from the even-numbered images of the original training datasets. The test datasets are composed of 2000 images taken from the odd-numbered views, which are subject to severe synthetic transformations (see Fig. 6.2).*

Dataset	# units	C^0, undistorted	C^1, undistorted	C^2, undistorted
MNIST-10	2000	38.91%	70.48% (+81.1%)	77.63% (+98.8%)
COIL-20	2000	44.52%	81.64% (+83.3%)	90.32% (+102.9%)
ORLS-40	2000	47.63%	51.95% (+9.1%)	59.30% (+24.5%)

Table 6.1: Comparison of the final classification accuracy of undistorted C^0, C^1 and C^2 T-VTUs, using a representation size of 2000 units. For the C^1 and C^2 the percentage of improvement over the C^0-classifier is shown.

the ORLS-40 dataset is not surprising keeping in mind that the training sets only contain 5 different views of each person. Also, for undistorted T-VTUs and for representation sizes above approx. 400 units, no further improvement is possible, because identical images are used.

6.2.3 Experiment: Classification with Distorted T-VTUs

From the results of the experiment described above we have seen that the HFM is capable of successfully compensating distortions the test stimuli were subjected to. For all three image domains, the best classification accuracies were achieved by the $\Phi_{C^2}^{TVTU}$ classifiers operating on the C-cell plane activity space of the second layer of the HFM. In a second experiment we now investigate whether an additional improvement can be achieved by using distorted training data, i.e. applying synthetic transformations not only to the test image patches but also to the training image patches.

The results of this experiment are shown in Fig. 6.4, where the accuracies achieved by the $\Phi_{C^2}^{TVTU}$ classifier using undistorted training data are compared to the accuracies obtained when applying the following synthetic transformations also on the training data: Random rotation in the image plane by $+/-$ 10^o, random scaling by $+/-$ 10%, and random translation of $+/-$ 5 pixels.[2]

From these results it can be seen that by using distorted training data improvements in final accuracy can be achieved and that the stability of the results can be improved (Note that the standard deviations are significantly smaller for the distorted training dataset). As summarized in Tab. 6.2, for the MNIST-10 and COIL-20 dataset the improvement of the final classification accuracy (using a representation size of 2000) is only minor, because high values are already reached by the undistorted T-VTUs. However, for the ORLS-40 dataset the improvement of the final accuracy is much higher. This shows that synthetic transformations offer a good means for compensating for poor class sampling in a dataset (recall that in the original training data we had only 5 different views for each one of the 40 classes).

On the other hand it can also be observed that for small representation sizes, adding synthetic transformations to the training data does not pay off: the accuracy remains slightly below that obtained when using the undistorted raw image.

[2] Adding background clutter and additive Gaussian noise to the training images does not change the average accuracy, but only leads to larger standard deviation of the results under repetition.

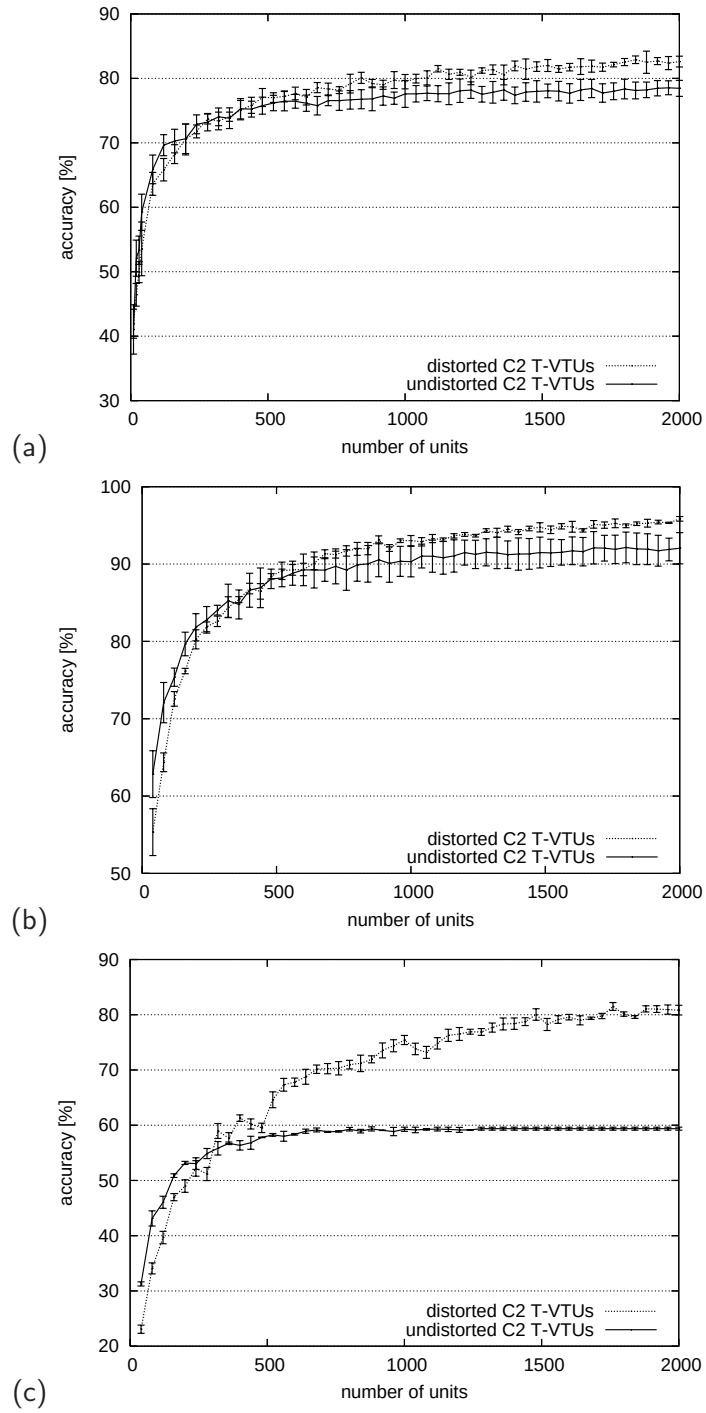


Figure 6.4: *Comparison of the classification performance of the C^2 classifier using undistorted vs. distorted training data. (a) Results for the MNIST-10 dataset. (b) Results for the COIL-20 dataset. (c) Results for the ORLS-40 dataset. See text for details.*

Dataset	# units	C^2, undistorted	C^2, distorted
MNIST-10	2000	77.63%	82.61% (+6.4%)
COIL-20	2000	90.32%	95.85% (+6.1%)
ORLS-40	2000	59.30%	80.84% (+36.3%)

Table 6.2: Comparison of the final classification accuracy of undistorted and distorted C^2 using a representation size of 2000 units.

6.2.4 Summary

In this section, we have investigated the classification performance of Template View Tuned Units (T-VTUs) using severely distorted test datasets from three different image domains. The datasets were generated by applying synthetic transformations to the images taken from the MNIST-10, the COIL-20, and the ORLS-40 datasets. In a first experiment we found that T-VTUs operating on the C-cell plane activity space of the second layer of the HFM exhibit a significantly better performance compared to first-layer T-VTUs and the standard nearest neighbor classifier in the raw input image space.

Secondly, we have found that applying synthetic distortions to training images leads to improvements in performance. For training datasets that already have well represented classes the improvement is only minor, whereas a large improvement can be achieved for datasets such as the ORLS-40 dataset, which have a poor class sampling. However, the improvement in accuracy can only be observed for large numbers of T-VTUs. For small representation sizes distortions of the training data lead to a loss in performance.

We can conclude that T-VTUs can be successfully used to classify image patches but only when large class representations are used. This is clearly a drawback of the T-VTU approach because for classification each test image patch's final C-cell plane activity must be matched to all T-VTUs in order to obtain the result. This is computationally expensive and leads to limitations for practical applications. To overcome this problem in the following section we investigate a method that allows one to reduce the number of units used for classification.

6.3 Condensed View Tuned Units

The idea of Condensed View Tuned Units (C-VTUs) is to create simplified representations of the classes in the C-cell plane activity space that consist only of a low number of units. For this, we start off from a distorted T-VTU representation and apply a spectral clustering technique in order to successively divide each class in the representation into disjoint subsets. The means of these partitions then serve as a new representation, called a C-VTU representation. As shown in experiments below, this method makes it possible to significantly reduce the size of the representation while keeping up a high classification performance.

In the following the spectral clustering technique that is used for computing partitions from a given T-VTU representation is described. Then, in Sect. 6.3.2, we experimentally compare the classification performance of C-VTUs obtained by the spectral clustering technique to that of standard T-VTUs.

6.3.1 Spectral Clustering

The spectral clustering approach as introduced in [94] describes the problem of clustering in the context of graph theory. From a set of m data points – or objects – a fully connected, undirected graph is constructed whose (real-valued) edge weights are defined using a measure of similarity between two objects. The clustering problem is then defined as a problem of finding an optimal *cut* that partitions the graph into two disjoint sets by simply removing all edges that connect the two parts. The classical approach to finding such an optimal cut is to minimize the degree of cumulative similarity between two partitions $\mathbb{A}$ and $\mathbb{B}$, measured by the following local criterion that calculates the total sum of the edge weights,

$$cut(\mathbb{A}, \mathbb{B}) = \sum_{\vec{u} \in \mathbb{A}, \vec{v} \in \mathbb{B}} w(\vec{u}, \vec{v}), \tag{6.6}$$

where $w(\vec{u}, \vec{v})$ is the edge weight between objects $\vec{u}$ and $\vec{v}$. An earlier approach in [108] utilized the minimization of this criterion for data clustering. The authors noticed, however, that the criterion favors cutting out small isolated sets of objects. To avoid this effect, the authors of [94] proposed a new, global criterion called the *normalized cut*,

$$Ncut(\mathbb{A}, \mathbb{B}) = \frac{cut(\mathbb{A}, \mathbb{B})}{assoc(\mathbb{A}, \mathbb{V})} + \frac{cut(\mathbb{A}, \mathbb{B})}{assoc(\mathbb{B}, \mathbb{V})}, \tag{6.7}$$

where $assoc(\mathbb{A}, \mathbb{V}) = \sum_{\vec{u} \in \mathbb{A}, \vec{t} \in \mathbb{V}} w(\vec{u}, \vec{t})$ denotes the total connection of objects in a partition $\mathbb{A}$ to all objects in the graph; $assoc(\mathbb{B}, \mathbb{V})$ is defined correspondingly. The authors argue that minimizing the normalized cut criterion can be viewed as a tradeoff between the association within each partition and the disassociation between the partitions, which leads to a much better quality of partitions.

As shown in [94], the computational appeal of this new criterion is that minimization can be expressed in terms of a generalized eigenvalue problem,

$$(\mathbf{D} - \mathbf{W})\vec{y} = \lambda \mathbf{D}\vec{y}, \tag{6.8}$$

where $\mathbf{W}$ is the $m \times m$ edge weight matrix and $\mathbf{D}$ is a $m \times m$ diagonal matrix with $D_{i,i} = \sum_j W_{i,j}$. The problem can be rewritten as a standard eigenvalue problem,

$$\mathbf{D}^{-\frac{1}{2}}(\mathbf{D} - \mathbf{W})\mathbf{D}^{-\frac{1}{2}}\vec{z} = \lambda \vec{z}, \tag{6.9}$$

where $\vec{z} = \mathbf{D}^{-\frac{1}{2}}\vec{y}$. The solution of the clustering problem can be derived from solving (6.8) for the eigenvector corresponding to the second smallest eigenvalue. By applying a threshold on the entries of the eigenvector, we can then decide which data point belongs to which partition.[3]

Spectral clustering can also be used for finding multiple partitions of a data set by recursively repeating the process described above. In [94] this is referred to as the *Recursive Two-way N-cut*, which we utilize for building the C-VTU representations below. In [94], the authors also mention a different approach to finding multiple partitions, which they call the *Simultaneous K-way Cut*. For this, more than one eigenvector (with increasing eigenvalue,

[3] For finding an optimal threshold, one possibility is to choose different threshold values and compute the normalized cut criterion (6.7) for each choice.

starting from the second smallest) is considered for finding partitions. Assigning the data points to clusters then involves using another clustering method on rows of the eigenvector matrix. One possibility for this is using K-means clustering (originally introduced in [61]) as suggested in [94] and evaluated in more detail, e.g., in [71].

For the purpose of building the C-VTU representations, we define a function $split_{spec}(\mathbb{D})$ which performs a single spectral clustering step, i.e. splitting a given set $\mathbb{D}$ into two partitions. For the implementation of the spectral clustering method, the Jacobi algorithm [50, 82] was used for solving the eigenvalue problem. The edge weight matrix was computed by:

$$W_{i,j} = \exp\left(-\frac{dist(x_i, x_j)}{\sigma}\right), \tag{6.10}$$

where σ was chosen such that $exp(-\frac{m}{2\sigma}) = 0.5$, with $m = max(dist(x_i, x_j))$, assigning a similarity value of 0.5 to half of the maximum distance between any two objects.

Based on the function $split_{spec}$, Alg. 4 describes how a C-VTU representation is obtained using the clustering method.

Algorithm 4 Computation of Condensed View Tuned Units: C-VTU($\mathbb{T}$, n)

1: **for** $i \leftarrow 1$ to c **do**
2: $\quad \mathbb{K}_i \leftarrow \{\ (\vec{x}_j, l_j) \mid \vec{x}_j \in \mathbb{T},\ l_j = i\ \}$
3: **end for**
4: **for** $j = c + 1$ to n **do**
5: $\quad i^* \leftarrow \arg\max_i |\mathbb{K}_i|$
6: $\quad (\mathbb{K}_{i^*}, \mathbb{K}_j) \leftarrow split_{spec}(\mathbb{K}_{i^*})$
7: **end for**
8: $\mathbb{C} \leftarrow \{\ (\vec{\mu}_i, l_i) \mid \vec{\mu}_i = \frac{1}{|\mathbb{K}_i|} \sum_{(\vec{x}_j, l_i) \in \mathbb{K}_i} \vec{x}_j,\ i = 1 \ldots n\ \}$
9: **return** $\mathbb{C}$

The algorithm takes as input a T-VTU representation $\mathbb{T}$, i.e., a set consisting of tuples of vectors and corresponding class labels, and a number n denoting the desired target number of units.

In a first step (lines 1–3), the algorithm starts by defining c sets $\mathbb{K}_i$ each of which contains all examples belonging to class i. In the following (lines 4–7), the algorithm successively creates additional sets by splitting the set that contains the most examples (this is expressed in line 5). Splitting of sets is repeated until a target number n of sets have been created. Finally (line 8), the means of the n sets are computed and combined to a set $\mathbb{C}$ which is the resulting C-VTU representation.

Figure 6.5 shows four snapshots of the clustering process for a 2-dimensional spiral-like data distribution containing 2 classes. Data points (boxes depict the first class, crosses the second) correspond to the T-VTUs that are the input to the algorithm. Bold crosses depict the means of the cluster, which are the current C-VTUs. For visualization purposes, the decision borders of the corresponding tessellation cells are included. It can be seen that the representation becomes increasingly detailed, and after 32 clustering steps all data points are located inside a tessellation cell of the correct class.

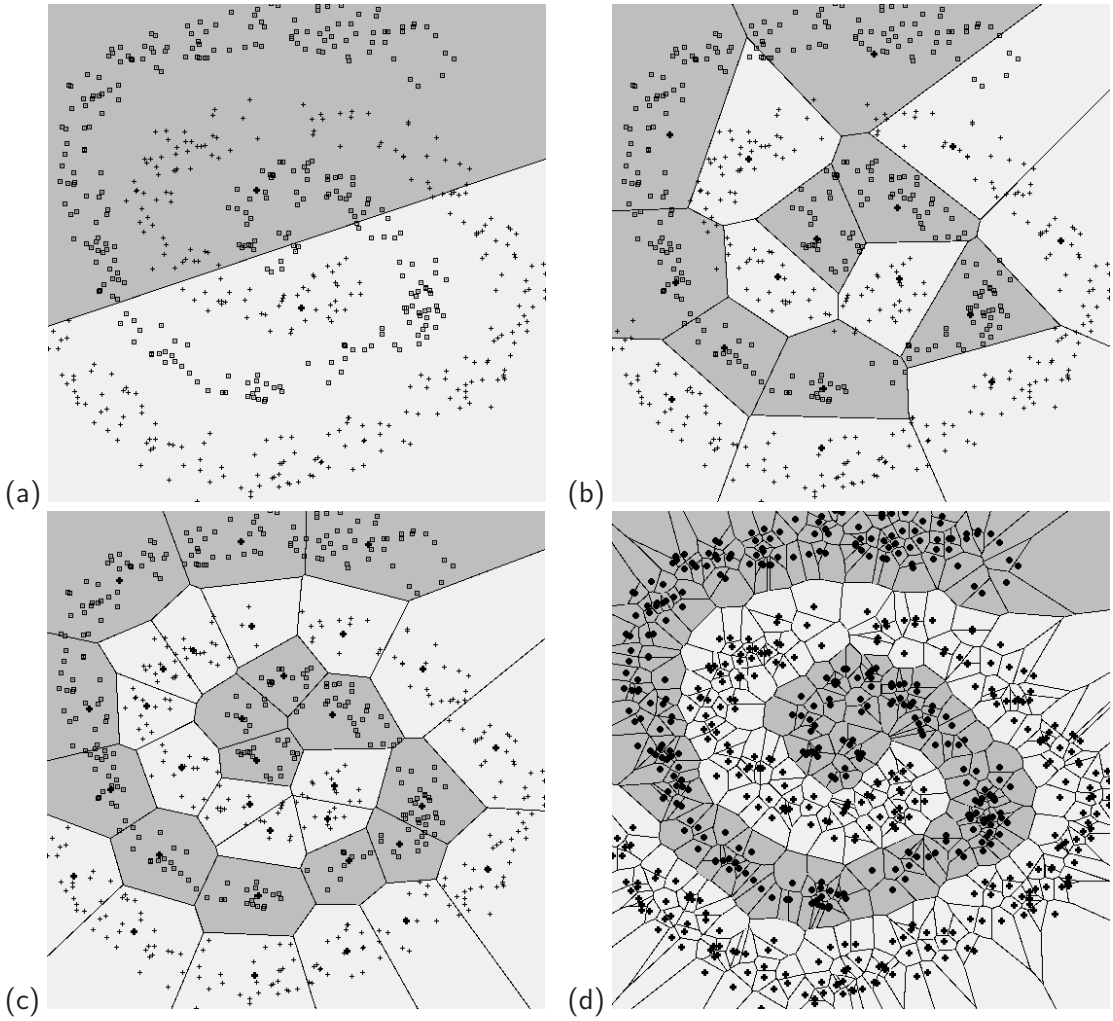


Figure 6.5: *Toy example for creating a C-VTU representation for a 2-dimensional dataset containing 2 classes. Boxes denote examples of the first class and crosses for the second. Bold crosses denote the means of the clusters and lines visualize the decision borders. (a) Initial representation with 2 C-VTUs, (b) after 16 steps, (c) after 32 steps and (d) final representation after 500 steps.*

Dataset	Representation size	distorted T-VTUs	C-VTUs	Increase
MNIST-10	400	75.07 %	80.52 %	7,25 %
COIL-20	400	86.83 %	93.45 %	7,61 %
ORLS-40	400	61.29 %	70.06 %	14.31 %

Table 6.3: Comparison of classification accuracy of second layer distorted T-VTUs and C-VTUs for a representation size of 400 units. (The C-VTUs are obtained by spectral clustering applied on the full set of 2000 T-VTUs.)

6.3.2 Experiment: Classification with C-VTUs

In this section, the classification performance of C-VTUs is experimentally compared to that of T-VTUs. Here, we consider the classifiers operating on the second layer of HFM only.

Analogous to Eq. 6.3 we define the C-VTU classifier operating on the second-layer C-cell plane activity space as follows:

$$\Phi_{C^2}^{CVTU}(\mathbf{I}) := \arg\min_{l_i} \; dist(C^2(\mathbf{I}), \vec{t}_i), \; \forall(\vec{t}_i, l_i) \in \mathbb{C}_{C^2}, \tag{6.11}$$

and analogous to Eq. 6.5, we can measure the classification accuracy on a test dataset $\mathbb{D}_{test}$ by

$$acc_{C^2}^{CVTU}(\mathbb{D}_{test}) := \frac{1}{t}\sum_{i=1}^{t} \delta(\Phi_{C^2}^{CVTU}(\mathbf{I}_i), l_{test}(\mathbf{I}_i)). \tag{6.12}$$

The C-VTU representations are obtained from distorted second layer T-VTU representation with 2000 units. The target number of C-VTUs is varied, starting at the number of classes (10, 20 and 40 for MNIST-10, COIL-20, and ORLS-40 respectively) and is increased in steps of 40 until the full representation size of 2000 units is reached.

Figure 6.6 shows the results of the experiment. For each of the datasets MNIST-10, COIL-20, and ORLS-40, the two plots shows the classification accuracy of T-VTUs (Eq. (6.5)) and C-VTUs (Eq. (6.12)). Again, the standard deviation of the accuracy under 5 repetitions is visualized by horizontal bars. From the result, it can be seen that for small representation sizes C-VTUs significantly outperform the distorted T-VTUs. In Tab. 6.3, the accuracies for a representation size of 400 units are compared.

6.3.3 Summary

In this section we have investigated a method that allows a significant reduction in the number of units needed for representing classes in the C-cell plane activity space of a two-layered HFM. The method takes as input a large T-VTU representation, which is obtained by adding synthetic transformations to a training dataset of images and successively partitions the representation class-wise using spectral clustering [94]. Although for training an initially large set of T-VTUs has to be generated, from which the C-VTU representation is computed, for recognition the computational benefit lies in the reduced number of units that have to be matched for every input image. With this method, for the three test datasets, a reasonable classification performance can be reached using approx. 400 units.

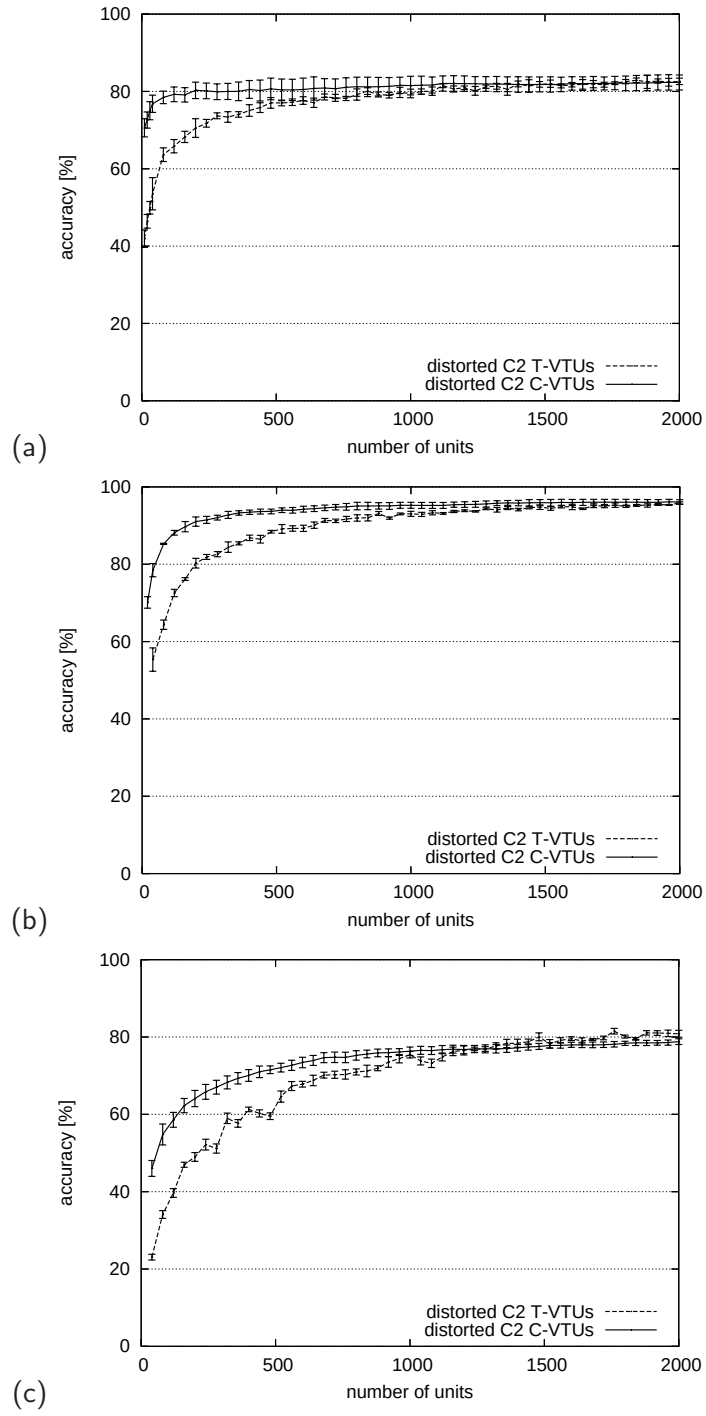


Figure 6.6: *Comparison of the classification performance of second layer T-VTUs and C-VTUs for the three datasets MNIST-10 (a), COIL-20 (b), and ORLS-40 (c). See text for details.*

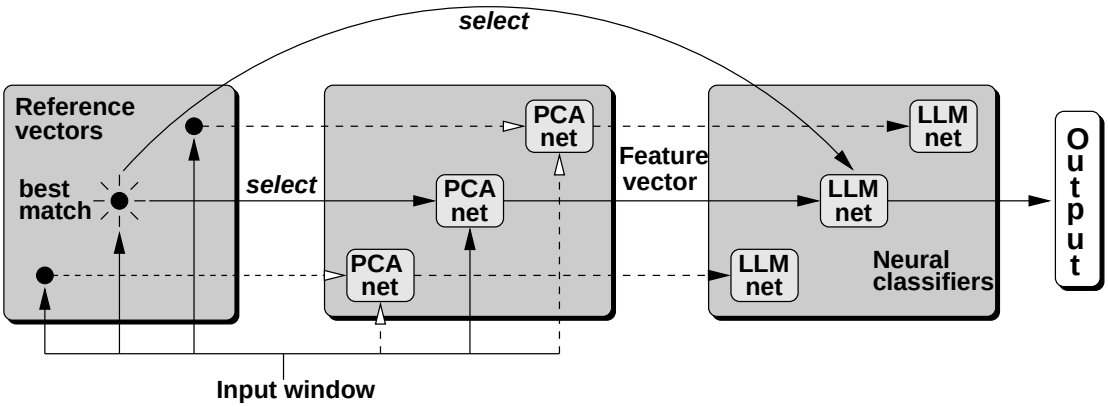


Figure 6.7: *Sketch of the three-stage VPL classification architecture which relies on Vector Quantization, Local Principal Component Analysis and Local Linear Maps. See text and [29] for details.*

6.4 Comparison to an Eigenspace Classification Approach

In order to provide a comparison of the results presented above, to a conceptually different classification approach, in this section we apply an eigenspace based classification method on the same datasets.

For this, we choose an approach called "VPL", which is a highly optimized eigenspace classification architecture that was proposed by Heidemann in [29] and that was successfully applied for different image patch classification tasks in a number of computer vision setups [35, 33, 7, 12, 31]. In contrast to former eigenspace classification approaches that rely on global eigenspaces (e.g. [101, 68]), the VPL architecture is based on local eigenspaces that are used for feature extraction and Local Linear Map (LLM, [66]) neural networks for classification.

As depicted in Fig. 6.7, the architecture involves three processing stages. In the first stage, called "V", a special winner-takes-all based vector quantization technique (called "AEV", [34]) is applied on the training images in order to partition the datasets into a number of clusters. In the next stage, called "P", Principal Component Analysis (PCA) is applied on the images in each cluster. For the computation of the Principal Components, Sanger's algorithm [91] is applied, an approach that relies on Hebbian learning to successively compute the leading eigenvectors of the auto-correlation matrix of the data. In the final stage, called "L", for each cluster the images are projected onto the corresponding local eigenspace and the resulting representation is used to train LLM classifiers [66]. The VPL architecture has three main parameters: the number of clusters V in the first stage, the number of Local Principal Component P in the second stage, and the number of LLM-nodes L in the third stage. For further details on the VPL architecture, the reader is referred to [29].

For the experiment we consider two setups: In the first, different VPL classifiers for each of the three datasets MNIST-10, COIL-20, and ORLS-40 are trained on datasets with 500, 1000, 1500, and 2000 undistorted images (the same as used in Sect. 6.2.2). The VPL

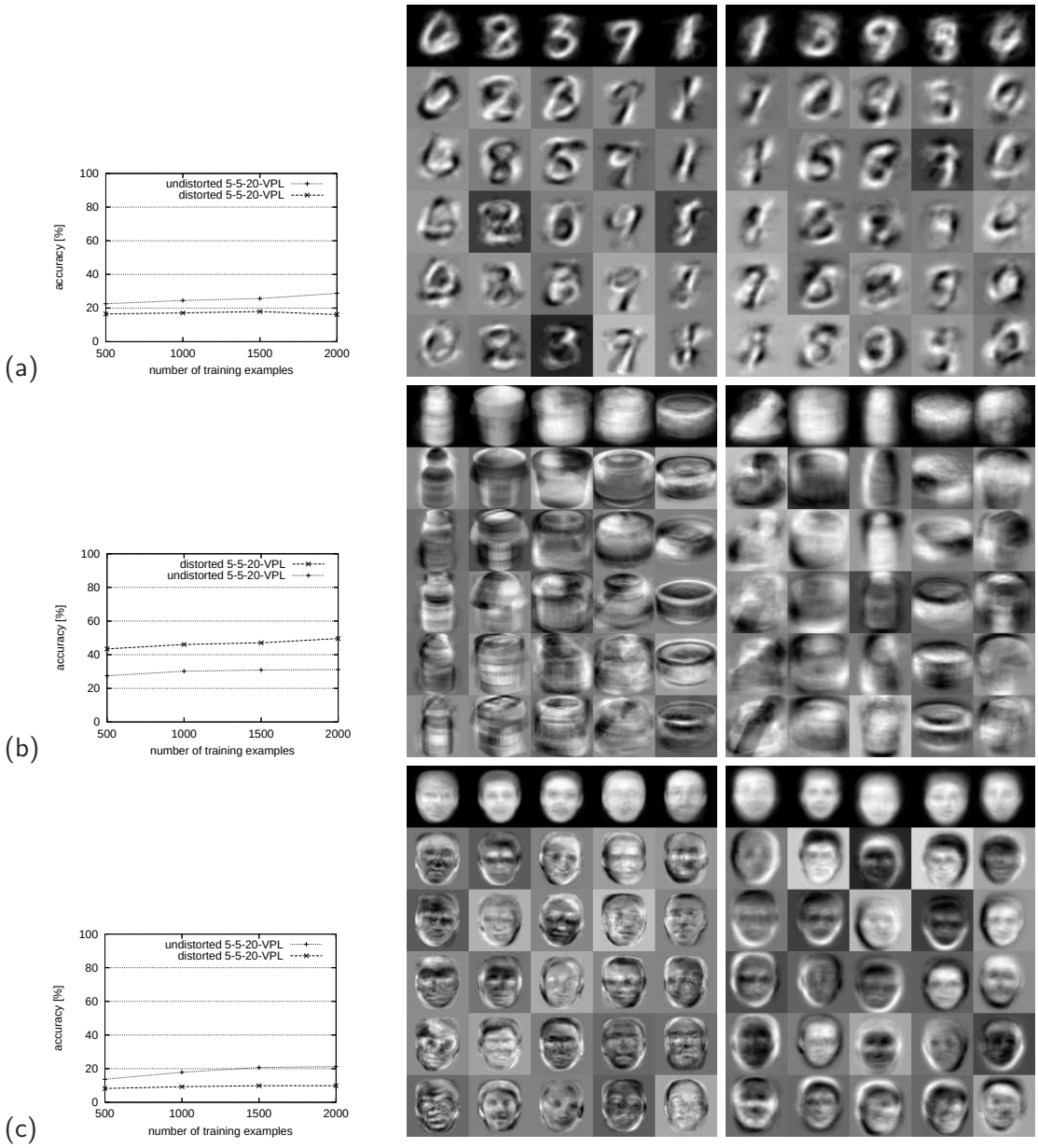


Figure 6.8: *Results for applying the VPL classifier on the MNIST-10 (a), the COIL-20 (b), and the ORLS-40 (c) datasets. For the curves labeled "undistorted", raw images are used for training the classifiers and for the curves labeled "distorted" highly distorted training images are used. Visualizations of the internal states of the trained classifiers using undistorted training images are shown in the middle column and in the right column for distorted training images. In each visualization, the top row shows the Vector Quantization prototypes, and the corresponding Local Principal Components are arranged column-wise below. See text for discussion.*

parameters are chosen to be $V = 5$, $P = 5$ and $L = 20$.[4] The internal states of the classifiers obtained for this setup are shown in the middle column of Fig. 6.8. Plots of the classification performance of the classifiers applied on the highly distorted test datasets with background clutter (Fig. 6.2) are shown in the left column of Fig. 6.8 (curves labeled "undistorted").

The second setup is identical to the first, except that distorted images are used for training the classifiers (see Sect. 6.2.3). The resulting performance is plotted in the curves labeled "distorted", and the visualizations of the internal states are shown in the right column of Fig. 6.8.

The results of the experiment show that the VPL classifiers exhibits a relatively weak performance on the highly distorted test datasets. In all cases, the performance clearly remains below that obtained by the HFM approach as presented above. From the visualization of the internal states of the trained classifiers it can be seen that the eigenvectors for the undistorted training data appear to be too specific to account for the severe distortions contained in the test images. On the other hand, this problem cannot be compensated for by using the distorted training images. Here, the eigenvectors appear blurred and seem to fail to extract discriminative features that can be utilized for successful classification.

For the COIL-20 dataset an increase in performance from 31% to 49% (for using 2000 training images) can be observed when using distorted training images instead of raw images. For the other datasets we even encounter a severe drop in performance. (From 28% to 16% for the MNIST-10 dataset, and from 21% to 10% for the ORLS-40 dataset.)

In summary, for neither one of the two setups can a reasonable performance be obtained using the VPL classifier. We conclude that eigenspace approaches alone are not sufficient for solving discrimination problems where the test images are subject to such severe distortions, transformations and background clutter as used for the experiments with the HFM above. The HFM achieves a significantly better performance since it makes use of heuristic knowledge about the general structure of natural imagery. This knowledge cannot be extracted from a training dataset alone using statistical methods such as Principal Component Analysis.

6.5 Confidence based recognition

This section discusses two methods that allow for confidence based recognition, i.e. classification with rejection of "unknown" stimuli: Linear Discriminant Functions and Radial Basis Functions. Linear Discriminant Functions are obtained from supervised learning using T-VTU representations and Radial Basis Functions are directly derived from C-VTU representations. We experimentally compare the rejection performance of the two options on the distorted test datasets as already used in the preceding experiments, but with 2000 additional "rejection" images that do not contain objects, but only random clutter.

[4] This parameterization of the VPL was found to offer a good trade-off between training time, size of the classifier, and performance.

6.5.1 Linear Discriminant Functions

For classification with linear discriminant functions, we define a set of functions $LDF_i : \mathbb{R}^{d_x(2)*d_y(2)} \rightarrow \mathbb{R}$, which take as input a second layer C-cell plane activity vector $\vec{x}$ and return a real-valued response that can be interpreted as a *confidence score*:

$$LDF_i(\vec{x}) := \vec{x}^T \vec{w}^i + b_i, \; with \; i = 1, \ldots, c \tag{6.13}$$

where c is the number of classes. $\vec{w}^i$ is a called the *weight vector* and b_i the *bias* for class i. These free parameters are optimized by supervised learning using a given T-VTU representation, such that a high response is given to an example whose class identifier is equal to i and a low response if the example belongs to a different class. The weight vectors $\vec{w}^i$ and biases b_i can be found by minimizing the following error function with respect to a given second layer T-VTU representation:

$$E(\vec{w}^1, \ldots, \vec{w}^c, b_1, \ldots, b_c; \mathbb{T}_{C^2}) := \frac{1}{2} \sum_{(\vec{x},l) \in \mathbb{T}_{C^2}} \sum_{i=1}^{c} \left(\Delta(l,i) - LDF_i(\vec{x})\right)^2, \tag{6.14}$$

where $\Delta(i,j) = 0.9$, if $i = j$ and 0.1 else.

After optimizing the error function,[5] we can classify a given input image based on the maximum response of the linear discriminant functions:

$$\Phi^{LDF}(\mathbf{I}) := \arg\max_i \; LDF_i(C^2(\mathbf{I})), \; with \; i = 1, \ldots, c. \tag{6.15}$$

Further, we can reject an image as an "unknown stimulus," if the maximum response of the linear discriminant functions remains below a threshold θ_{LDF}:

$$LDF_{\Phi^{LDF}(\mathbf{I})}(\mathbf{I}) < \theta_{LDF}. \tag{6.16}$$

In Sect. 6.5.3 we experimentally investigate classification with rejection using linear discriminant functions.

6.5.2 Radial Basis Functions

In the preceding section it was described how supervised learning can be used to obtain a set of linear discriminant functions from a given T-VTU representation that allows classification with rejection using a threshold on the responses of these functions. In this section, we investigate an alternative possibility: radial basis functions that are obtained directly from C-VTUs and that do not require supervised learning.

Analogous to the preceding section, we define a set of radial basis functions $RBF_i : \mathbb{R}^{d_x(2)*d_y(2)} \rightarrow \mathbb{R}$, which again take as input a second layer C-cell plane activity vector $\vec{x}$ and return a real-valued response. However, here the number of functions is not equal to the number of classes as for the LDFs, but equal to the size of the C-VTU representation:

[5] For optimization, we use the Numerical Recipes [82] implementation of the Fletcher-Reeves-Polak-Ribiere algorithm, which is based on the conjugate gradient method [36].

$$RBF_i(\vec{x}) := exp\left(-\frac{dist(\vec{x},\vec{x_i})}{2\sigma}\right),\ with\ (\vec{x}_i, l_i) \in \mathbb{C}_{C^2},\ i = 1,\ldots,|\mathbb{C}_{C^2}| \tag{6.17}$$

where σ is the radius of the RBF which is set fixed to $\sigma = d_x(2) * d_y(2)$. In this setup, each RBF gives high responses to stimuli whose second layer C-cell plane activity vectors are "close" to the corresponding C-VTU that the function has been created from. Based on this set of functions, we can define the following classifier, which returns the class label l_{i^*} of the RBF function with the maximum response:

$$\Phi^{RBF}(\mathbf{I}) := l_{i^*},\ with\ i^* = \arg\max_i\ RBF_i(C^2(\mathbf{I})),\ i = 1,\ldots,|\mathbb{C}_{C^2}|. \tag{6.18}$$

Again, we can reject an image as an "unknown stimulus" if the maximum response of the "winning" RBF is below a threshold θ_{RBF}:

$$RBF_{\Phi^{RBF}(\mathbf{I})}(\mathbf{I}) < \theta_{RBF}. \tag{6.19}$$

In the following section, we experimentally compare the performance of the above defined LDFs and RBFs.

6.5.3 Experiment: Classification with rejection using LDFs and RBFs

For the rejection experiment we again use the three distorted test datasets $\mathbb{D}_{test}^{MNIST-10}$, $\mathbb{D}_{test}^{COIL-20}$, and $\mathbb{D}_{test}^{ORLS-40}$ as described in Sect. 6.2.1 but each extended by an additional number of 2000 image patches only containing clutter that is randomly selected from the Art Explosion Photo Gallery [72]. For all clutter image patches, we set the true class identifier $l_{test}(\mathbf{I})$ to -1. Some example clutter image patches are shown in Fig. 6.9.

The T-VTU representations used for training linear discriminant functions are obtained from the distorted training datasets as described in Sect. 6.2.3. The C-VTU representations used for creating radial basis functions are obtained from T-VTU representations of size 2000.

ROC analysis

For evaluating the classification and rejection performance of the two classifiers Φ^{LDF} and Φ^{RBF}, which have as free parameters the rejection thresholds θ_{LDA} and θ_{RBF} respectively, we use special types of ROC curves, called *Precision vs. Recall* curves which are extended to the multi-class case in a straight-forward way.

For creating such a *Precision vs. Recall* curve, the classification and rejection experiment is repeated for different threshold values and for each choice of θ_{LDA} or θ_{RBF} respectively, the following quantities are aggregated:

- *True positive (TP):* In this case, *(i)* the test image patch contains an object, *(ii)* the object is classified correctly and *(iii)* the output of the "winning" classification unit is above the chosen threshold value:

$$l_{test}(\mathbf{I}) \neq -1\ \wedge\ l_{test}(\mathbf{I}) = \Phi^{LDF}(\mathbf{I})\ \wedge\ LDF_{\Phi^{LDF}(\mathbf{I})}(\mathbf{I}) \geq \theta_{LDF} \tag{6.20}$$

Figure 6.9: *Some example clutter image patches to be rejected as "unknown stimuli".*

for the linear discriminant functions and

$$l_{test}(\mathbf{I}) \neq -1 \ \wedge \ l_{test}(\mathbf{I}) = \Phi^{RBF}(\mathbf{I}) \ \wedge \ RBF_{\Phi^{RBF}(\mathbf{I})}(\mathbf{I}) \geq \theta_{RBF} \tag{6.21}$$

for the radial basis function setup.

- *False positive (FP):* Either an object is detected, but classified incorrectly, or, the image patch does not contain an object, but the "winning" classification unit's response is above threshold:

$$(l_{test}(\mathbf{I}) \neq -1 \ \wedge \ l_{test}(\mathbf{I}) \neq \Phi^{LDF}(\mathbf{I})) \ \vee \ (l_{test}(\mathbf{I}) = -1 \ \wedge \ LDF_{\Phi^{LDF}(\mathbf{I})}(\mathbf{I}) \geq \theta_{LDF}) \tag{6.22}$$

and

$$(l_{test}(\mathbf{I}) \neq -1 \ \wedge \ l_{test}(\mathbf{I}) \neq \Phi^{RBF}(\mathbf{I})) \ \vee \ (l_{test}(\mathbf{I}) = -1 \ \wedge \ RBF_{\Phi^{RBF}(\mathbf{I})}(\mathbf{I}) \geq \theta_{RBF}) \tag{6.23}$$

respectively.

- *False negative (FP):* The image patch contains an object, but all classification units' responses are below threshold:

$$l_{test}(\mathbf{I}) \neq -1 \ \wedge \ LDF_{\Phi^{LDF}(\mathbf{I})}(\mathbf{I}) < \theta_{LDF} \tag{6.24}$$

and

$$l_{test}(\mathbf{I}) \neq -1 \ \wedge \ RBF_{\Phi^{RBF}(\mathbf{I})}(\mathbf{I}) < \theta_{RBF}, \tag{6.25}$$

respectively.

Based on these three values, *Precision* (also called *Positive Predictive Value*) and *Recall* (also called *Sensitivity* or *True-Positive Rate*) are defined as follows:

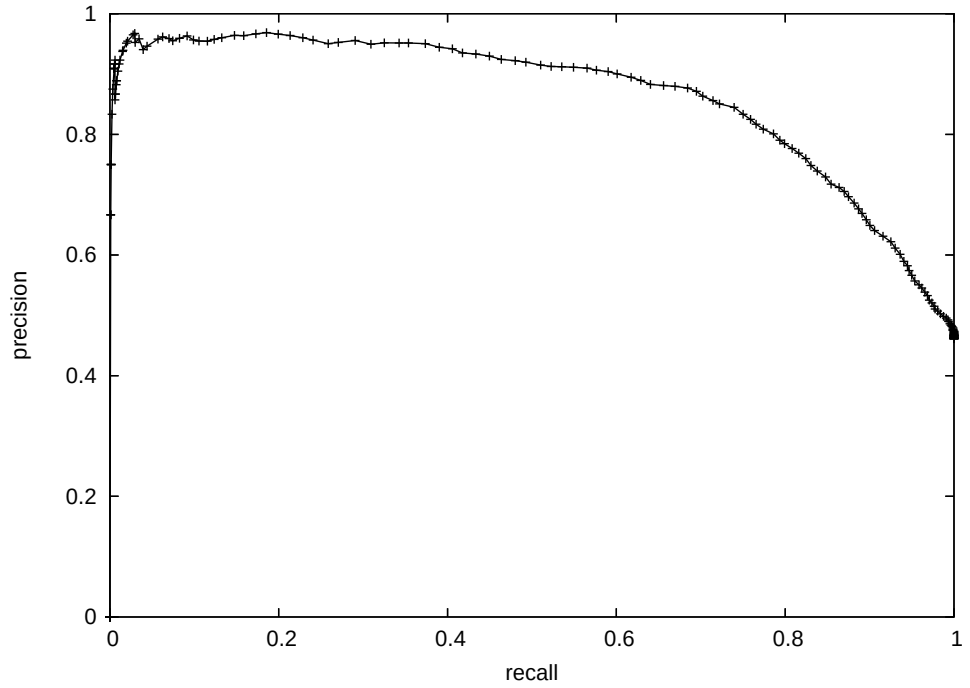


Figure 6.10: *An example Precision vs. Recall curve. Here, 200 radial basis function were used on the COIL-20 dataset.*

$$Precision := \frac{TP}{TP + FP} \tag{6.26}$$

$$Recall := \frac{TP}{FP + FN} \tag{6.27}$$

An example *Precision vs. Recall* curve for the radial basis function setup using a representation size of 200 and the COIL-20 dataset is shown in Fig. 6.10.

A convenient way of obtaining a single performance value based on an ROC curve is to estimate the *area under the curve* (AUC). As discussed in e.g. [21] this is a well established technique that allows a relative comparison of different classifiers. The AUC value for the example curve depicted in Fig. 6.10 is approx. 0.872.[6]

Results

Table 6.4 (a) shows AUC values for different settings obtained in the RBF setup for the three datasets MNIST-10, COIL-20, and ORLS-40. Here, the size of the representation, i.e. the number of radial basis functions is varied, using the values 100, 200, 400, 1000, and 2000. Also, the number of planes $n_p(2)$ on the second layer of the HFM is varied, using

[6] For algorithmic details on how to compute AUC values from given ROC curves, the reader is referred to [21].

the values 16, 32, and 64. Each value is averaged over 5 repetitions of the experiment with differently generated datasets.

Under these variations, the highest AUC values achieved by RBFs are 0.760 for the MNIST-10 dataset, 0.912 for the COIL-20 dataset, and 0.969 for the ORLS-40 dataset. In this setup the most difficult dataset appears to be the MNIST-10. This can be explained by the fact that for most digits only a small portion of the image plane is occupied by the stimulus itself whereas the remaining area is filled with background clutter for the test images as shown in Fig. 6.2. This makes it more difficult to distinguish positive cases from negative ones (pure clutter images) for this dataset.

From the results it can also be seen that increasing the representation size, i.e. the number of functions used, does not lead to a significant improvement in performance. Even over-fitting effects can be observed for some cases when the representation becomes too detailed.

Increasing the number of planes $n_p(2)$ of the second layer of the HFM (which corresponds to an increase of the feature space dimensionality) also does not lead to improvements. This can be explained by the fact that the Euclidian distance measure as used for computing the responses of RBFs becomes unstable for the high dimensional feature space.

The results for the LDF setup are shown in Tab. 6.4 (b). Here, the size of the training T-VTU representation is varied, using the values 100, 200, 400, 1000, and 2000 and again the number of second layer plains is varied, using the values 16, 32, and 64.

The highest AUC values achieved by LDFs are 0.755 for the MNIST-10 dataset, 0.934 for the COIL-20 dataset, and 0.867 for the ORLS-40 dataset. Here, in contrast to the RBF setup the values increase significantly with both the size of the training T-VTU representation and the dimensionality of the features space.

Since LDFs rely on the dot product between the weight vector and the C-cell plane activity vector the performance benefits from increasing the feature space dimensionality, because the probability increases that the classes are linear separable in the C-cell plane activity space.

6.5.4 Summary

In this section we have investigated two methods for utilizing the HFM for classification with rejection of "unknown" stimuli.

The first option was to create radial basis functions (RBFs) from C-VTU representations by directly using C-VTUs as centers. Recognition relies on thresholding the responses of the units which are computed from passing the Euclidian distance between the C-cell plane activity of the test stimulus and the center of the RBFs through an activity function.

Experimental tests on sets of 4000 test images (2000 positive and 2000 negative cases) showed that RBFs achieve high recognition performance values in terms of the area under ROC curves. However, large numbers of RBF units have to be used for representing each class and during recognition the responses of all units have to be computed, which is computationally expensive. The advantage of RBFs is that no supervised learning step is required.

The second alternative investigated were linear discriminant functions (LDFs) which are trained supervised on target responses using T-VTU representations. The performance values achieved by LDFs are almost comparable to those of RBFs as long as the training

Dataset	Number of radial basis functions	$n_p(2) = 16$	$n_p(2) = 32$	$n_p(2) = 64$
MNIST-10	100	0.749	0.743	0.742
	200	0.756	0.747	0.742
	400	0.755	0.753	0.751
	1000	0.759	**0.760**	0.752
	2000	0.753	0.747	0.747
COIL-20	100	0.849	0.861	0.851
	200	0.872	0.878	0.877
	400	0.886	0.896	0.887
	1000	0.889	0.909	0.905
	2000	0.896	**0.912**	0.907
ORLS-40	100	0.873	0.879	0.881
	200	0.902	0.909	0.912
	400	0.930	0.935	0.938
	1000	0.954	0.960	0.962
	2000	0.963	0.967	**0.969**

(a)

Dataset	Number of training T-VTUs	$n_p(2) = 16$	$n_p(2) = 32$	$n_p(2) = 64$
MNIST-10	100	0.502	0.492	0.583
	200	0.506	0.523	0.655
	400	0.515	0.623	0.703
	1000	0.544	0.651	0.729
	2000	0.587	0.689	**0.755**
COIL-20	100	0.732	0.786	0.821
	200	0.743	0.808	0.850
	400	0.754	0.824	0.878
	1000	0.782	0.847	0.915
	2000	0.846	0.901	**0.934**
ORLS-40	100	0.639	0.708	0.747
	200	0.593	0.735	0.826
	400	0.492	0.678	0.821
	1000	0.592	0.677	0.832
	2000	0.660	0.766	**0.867**

(b)

Table 6.4: *Comparison of AUC values obtained by radial basis functions (a) and linear discriminant functions (b) for classification with rejection using the three image domains MNIST-10, COIL-20, and ORLS-40. The highest AUC values achieved for each dataset are highlighted in bold font. See text for discussion.*

T-VTU representation is large enough and the dimensionality of the feature space given by the C-cell plane activity space is high enough such that linear separability of the classes is possible. Although the supervised optimization step is computationally expensive, the LDF approach is well suited for practical applications, because each class is represented by one single unit.

6.6 Discussion

In this chapter we have used the image normalization capabilities of the HFM in order to solve difficult image patch classification problems using comparatively small training datasets and highly distorted test datasets.

First, we have investigated the use of simple 1-nearest neighbor classifiers operating on the raw image space and on the output space of single- and two-layered HFMs. Since the nodes of these classifiers correspond to the C-cell plane activities caused by individual object views they are called *Template View Tuned Units* (T-VTUs, [107, 85]). The experimental results for this setup showed that the performance of the 1-nearest neighbor classifier can be greatly enhanced by projecting the input images onto the output space of the HFM.

In a next step, we have used a spectral clustering technique in order to reduce the computational effort that is required by the T-VTU approach during recognition. Here, a significant reduction of the number of units was possible while preserving most of the the classification performance obtained using the T-VTU approach with a large number of units.

A comparison of the results on the same datasets to a different classification approach called "VPL" [29] suggest that eigenspace based methods are insufficient for tolerating such a high number of distortions of the test images, when trained on small sized training datasets and using the input image space. Here, the HFM together with View Tuned Unites by far outperformed the VPL approach.

In the last part of the chapter we have investigated the use of Linear Discriminant Functions (LDFs) as well as Radial Basis Functions (LDFs) for the problem of confidence based recognition, i.e. classification with the ability to reject "unknown" stimuli using a threshold parameter. Experiments on test datasets that contained 50% negative examples showed that with both approaches, a high recognition performance can be achieved. The advantage of the RBF approach is that no additional training phase is required since the RBF representation is directly obtained from a given C-VTU representation. However, the recognition process is still computationally expensive, despite the reduction achieved by the spectral clustering method.

In contrast, the LDF approach requires an additional training phase, but yields as a result only a single unit for each class. However, the dimensionality of the output space of the two-layered HFM (the number of planes $n_p(2)$) has to be increased in order to achieve a performance that is comparable to that of the RBF approach.

Another advantage of the LDF approach is that the computation of unit responses rely on the dot-product. Therefore, it is possible to use LDF weight vectors as receptive field profiles in a three-layered HFM with increased input dimensionality in order to perform segmentation-free multi-class object detection in highly cluttered scenes. This option is investigated in the following chapter.

In summary, we have shown that the HFM offers a good possibility for high performance

classification of image patches. The approach allows the use of relatively small training datasets and is robust to significant distortions and transformations of the test images. This makes the method interesting for real world computer vision applications.

Parameter Details

If not stated otherwise, the parameterization of the HFM that was used for the experiments in this chapter is identical to the one used in Chapt. 5. Refer to Tab. 5.4.

CHAPTER 7

Detection

In this chapter, the Hierarchical Feed-forward Model is applied to a segmentation-free multi-class detection problem, i.e. finding and classifying objects in larger images. For this, a three layered HFM architecture with an increased input dimensionality is proposed. Receptive field profiles of the first two layers of the architecture are inherited from a two-layered training model which processes distorted training image patches in order to obtain receptive field profiles using NMFSC decomposition (see Chapt. 5). On the third layer of the detection architecture, linear discriminant functions (LDFs) are used as receptive field profiles that are obtained by supervised learning in the output space of the two-layered training model (see Chapt. 6). Finally, recognition results are obtained using a maxima detection scheme on the output C-cell planes of the three layered architecture which exhibit activity peaks at locations that correspond to the positions of specific objects that are present in the input image. Experimental tests are carried out on synthetically generated test images that are obtained by randomly embedding objects from different image domains into highly cluttered natural scene images. Earlier results and parts of the material presented in this chapter have been published in advance in [11, 8, 9].

7.1 Overview

Multi-Class object detection is a problem that involves solving two subtasks simultaneously: *(i)* Finding the location of an object in the image and *(ii)* determining its class.

Many common approaches to multi-class object detection employ a strategy to split up the process into two independent subtasks: a localization stage and a subsequent identification stage. Usually, the localization stage utilizes a coarse heuristic that makes it possible to generally distinguish "objects" from "non-objects" from the perspective of the current application scenario.

For example, in previous work [31, 32, 12] a system for online object learning and recognition was proposed, in which it was assumed that objects are highly textured and appear in front of a homogeneous background. Under this premise the locations, i.e. the centers of the

objects in terms of image coordinates could be found using a segmentation scheme that is based on context-free, purely data-driven saliency measures (see [33] for details). The subsequent identification stage was then implemented by applying an image patch classification method on "candidate" regions computed during the localization stage.

Even though such an approach proves to be quite efficient in terms of computation time and online applicability it is nevertheless highly restricted to the application scenario. The sequential split-up into the two stages implies that foreground-background segmentation must always be done in a purely data-driven and unsupervised manner, i.e. without taking into account the special properties and the features that are needed in order to discriminate between the objects. This limitation becomes apparent when one tries to apply such a sequential model to multi-class object detection in highly cluttered scenes. Natural imagery imposes significantly higher difficulty than, e.g., a textured background, because it exhibits the same natural power spectrum [100] as the objects themselves and the data-driven distinction between objects and non-objects becomes just as difficult as the classification task itself. The architecture that is proposed in this chapter overcomes this limitation by solving both tasks not sequentially, but simultaneously.

A conceptually different approach to object detection in cluttered scenes was proposed by Viola and Jones [103, 102]. Here, simple local features are used in combination with a boosting algorithm [64, 92] to achieve efficient localization of objects in real world scenes. However, training of the weak classifiers is very time-consuming and the approach is only feasible for detection of a single object class (such as faces). Application to multi-class problems, e.g., detecting and identifying different objects, is not investigated. Boosting mechanisms have also been applied to the problem of generic object recognition [78], i.e., deciding if an object belonging to a specific category is present in a given image or not. Again, an extension of the approach to simultaneous processing of multiple categories seems difficult. Another example of single category detection was presented by Agarwal and Roth [1], who use part-based sparse representations for finding cars in images.

In contrast to the above works, the approach presented in this chapter allows simultaneous detection and identification of multiple object classes without the necessity of a pre-segmentation. The strategy employed bears some similarity to the method proposed in the works of Lowe [59, 60]. Lowe uses special types of features (called SIFT features) for object detection and identification. A large number of features that are invariant to a substantial range of transformations is extracted from an image and matched against a database of examples using an efficient lookup mechanism. Instead of storing specific feature ensembles in a database, the approach described below uses linear discriminant functions whose weight vectors encode signatures of objects in the C-cell plane activity space of the HFM.

7.2 An Architecture for Segmentation-Free Multi-Class Object Detection

In the preceding chapter, a learning scheme was described for linear discriminant functions (LDFs) that allow a high classification accuracy in the output space of a two-layered HFM under severe distortions of the test images. In the following, it is demonstrated how these linear discriminant functions can be used as receptive field profiles in a three layer HFM

with an increased input dimensionality in order to perform segmentation-free detection of multiple object classes in larger images that contain severe background clutter.

Figure 7.1 shows a sketch of the proposed architecture. The upper part of the architecture depicts the training branch, which processes a set of distorted training image patches of size 64×64 and successively creates first- and second-layer receptive field profiles as well as a set of linear discriminant functions (LDFs).[1] The bottom part of Fig. 7.1 depicts the recognition branch, which employs a three layered HFM with an increased input dimensionality of 256×256. Receptive field profiles of the first and second layers are directly inherited from the training branch. On the third layer, LDFs are used as receptive field profiles. Here, each profile takes the role of a "grandmother cell", serving as detector for a specific object class.

To use LDFs as receptive field profiles P^3 on the third layer of the recognition model we simply plug in the components of the optimized weight vectors $\vec{w}^i$ from Eq. 6.13. [2]

$$P^3_{pq}(x,y) = w^p_j, \quad \begin{array}{l} with\ j = (q\ n_p(3) + p)\ d_p(3)^2 + y\ d_p(3) + x, \\ where\ p = 1 \ldots c,\ q = 1 \ldots n_p(2),\ x,y = 1 \ldots d_p(3). \end{array} \tag{7.1}$$

Since the LDF functions have bias components (the b_i in Eq. 6.13) we slightly modify the linear simple cell model and replace Eq. 3.1 in Sect. 3.1.2 for the third model layer with:

$$\hat{S}^3_p(x,y;\mathbf{I}) \ = \ b_p + \sum_{q=1}^{n_P(2)} \sum_{i=-r_X}^{r_X} \sum_{j=-r_Y}^{r_Y} C^2_q(x+i,y+j) P^3_{p\,q}(r_X+i, r_Y+j). \tag{7.2}$$

Additionally, we skip the WTM competition on the third layer by simply setting $S^3_p(x,y;\mathbf{I}) = \hat{S}^3_p(x,y;\mathbf{I})$. This is needed because the binarization caused by the WTM mechanism would discard the confidence information required. In the current setup, a slightly different competitive mechanism is used on the third layer of the model, which is implemented in the algorithm for activity peak detection described in Sect. 7.2.3. The algorithm is used to obtain the final recognition results, i.e. the positions and the class labels of detected objects.

In the following section a method for the synthetic generation of detection test images is described; these are used in the experiments below for evaluating the performance of the proposed architecture. In Sect. 7.2.2, an example of passing a detection image through the recognition branch of the trained architecture is used to illustrate how the model achieves simultaneous detection and identification of objects contained in an input image. The activity peak detection scheme is described in Sect. 7.2.3. Finally, experimental results using detection test images from three different image domains are presented in Sect. 7.3.

7.2.1 Synthetic Detection Test Images

The synthetic generation of a detection dataset is carried out by embedding randomly chosen examples from a given set of test image patches into images of natural scenes taken from the Art Explosion Photo Gallery [72]. For the experiments in this chapter the three natural image datasets MNIST-10 [52], COIL-20 [70], and ORLS-40 [90, 89] are used.

[1] The training branch is identical to the architecture discussed in the preceding chapter.

[2] Note that since the dimension of the receptive field profiles $d_f(3)$ needs to be odd, the chosen dimension of the final C-cell planes of the training branch ($d_x(2)$ and $d_y(2)$) must be odd.

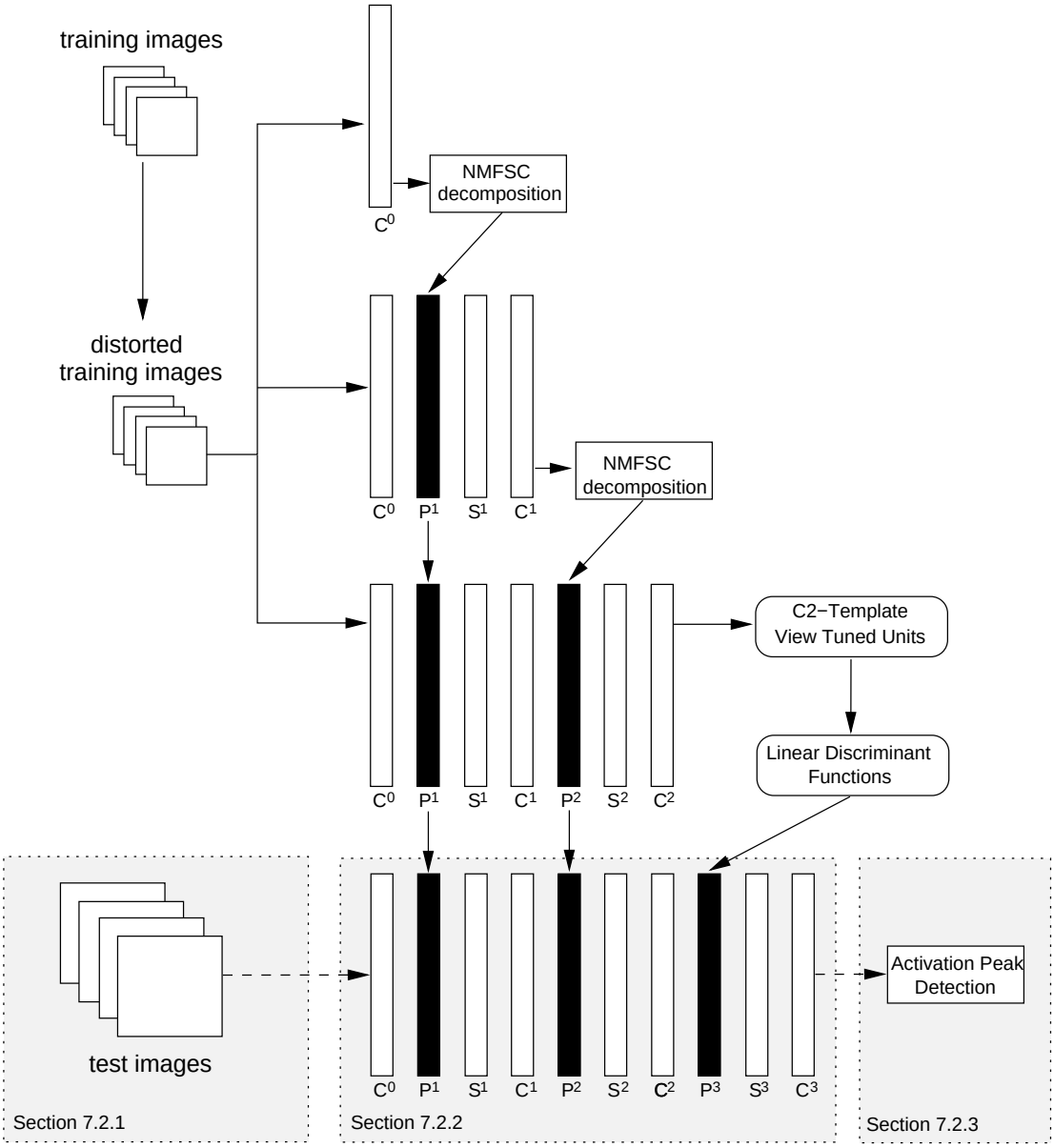


Figure 7.1: *Sketch of the architecture for segmentation-free multi-class detection. The training branch (top) consists of a two-layered HFM with an input dimensionality of* 64×64*. The recognition branch (bottom) uses a three layered model with an increased input dimensionality of* 256×256*. Receptive field profiles employed in the recognition branch are inherited from the training branch. See text for further details. The generation of synthetic detection test images is described in Sect. 7.2.1, an example for processing a detection image is discussed in Sect. 7.2.2, and Sect. 7.2.3 deals with the activity peak detection scheme that is used for obtaining the final detection results.*

The generation procedure requires a foreground-background segmentation of the original object images from the database, which – in our case – can be easily done using an intensity threshold. In order to provide "ground truth" for the evaluation of the recognition results, for each test image a list of the embedded objects is stored, with each entry containing the class identifier and the center position (in image coordinates) of the embedded object.

For the current setup an image size of 256×256 was chosen, where the original size of the image patch dataset is 64×64. For each generated image, first a random scene image is selected from a pool of 1000 scene images from the Art Explosion Photo Gallery. Then, five random examples from the patch dataset are chosen and embedded at random positions in the scene image. The generation algorithm is designed such that a spatial overlap of the objects is prevented. Some example detection test images are shown in Fig. 7.2.

7.2.2 Detection Example

Figure 7.3 shows an example of passing a detection test image generated from the COIL-20 dataset through the trained three layered HFM with an input dimensionality of 256×256. The number of planes of the first layer is set to 6 and the spatial resolution is decreased by 25% relative to the input image. The activities of the first layer C-cell planes C^1 exhibit small isolated patches of activity corresponding to the locally dominant edge orientation that the respective receptive field profile is tuned to. On the second layer, the number of planes is set to 64 and the spatial resolution is again decreased by 25% relative to the previous layer. The activity pattern on the C^2 C-cell planes is qualitatively similar to that of the first layer but more sparse in general. This results from the fact that on the first layer the ABS simple cell model is applied, whereas on the second layer the linear model is applied. Also, the receptive field profiles on the second layer have more specific responses than on the first layer since more planes are involved in the lateral competition mechanism (see Chapt. 5).

Finally, on the third layer of the model the LDFs are applied as receptive field profiles. Since there is exactly one LDF for each class, the plane index directly corresponds to the class identifier. Strong local activity peaks can be observed on the C^3 C-cell planes at locations that correspond to the positions of the respective objects in the input image.[3]

7.2.3 Activity Peak Detection

In order to obtain the final recognition results for an input test image a detection of local activity peaks on the final C-cell planes C^3 of the three layered HFM is performed. Alg. 5 is used for computing a set of detection results given a test image $\mathbf{I}$ and a response threshold θ.

The algorithm iterates over all planes of the third layer and over all cells in each C-cell plane (lines 2–29). Each cell's activity is compared to that of the 8 neighboring cells of the same complex cell plane (lines 5–12). If none of these cells has a higher activity than the current cell, the algorithm performs an additional check whether no higher maximum can be found on any other C-cell plane within a region of $+/-$ 1 cell positions (lines 14–22). If the current cell passes this check and if its activity is above the selected threshold θ (line 24), an object is detected and a triple is added to the result set, containing the object's

[3] On the third layer, the linear simple cell model is applied. The model is parameterized such that no further reduction of the spatial resolution is employed on the final layer.

Figure 7.2: *Examples of synthetically generated detection test images for the segmentation-free multi-class detection experiment. For generation, random odd numbered images from the MNIST-10 (a-c), COIL-20 (d-f) and ORLS-40 (g-i) datasets were used and embedded at random positions into real world scene images taken from the Art Explosion Photo Gallery. In the examples shown here, no synthetic transformations were applied on the test image patch datasets.*

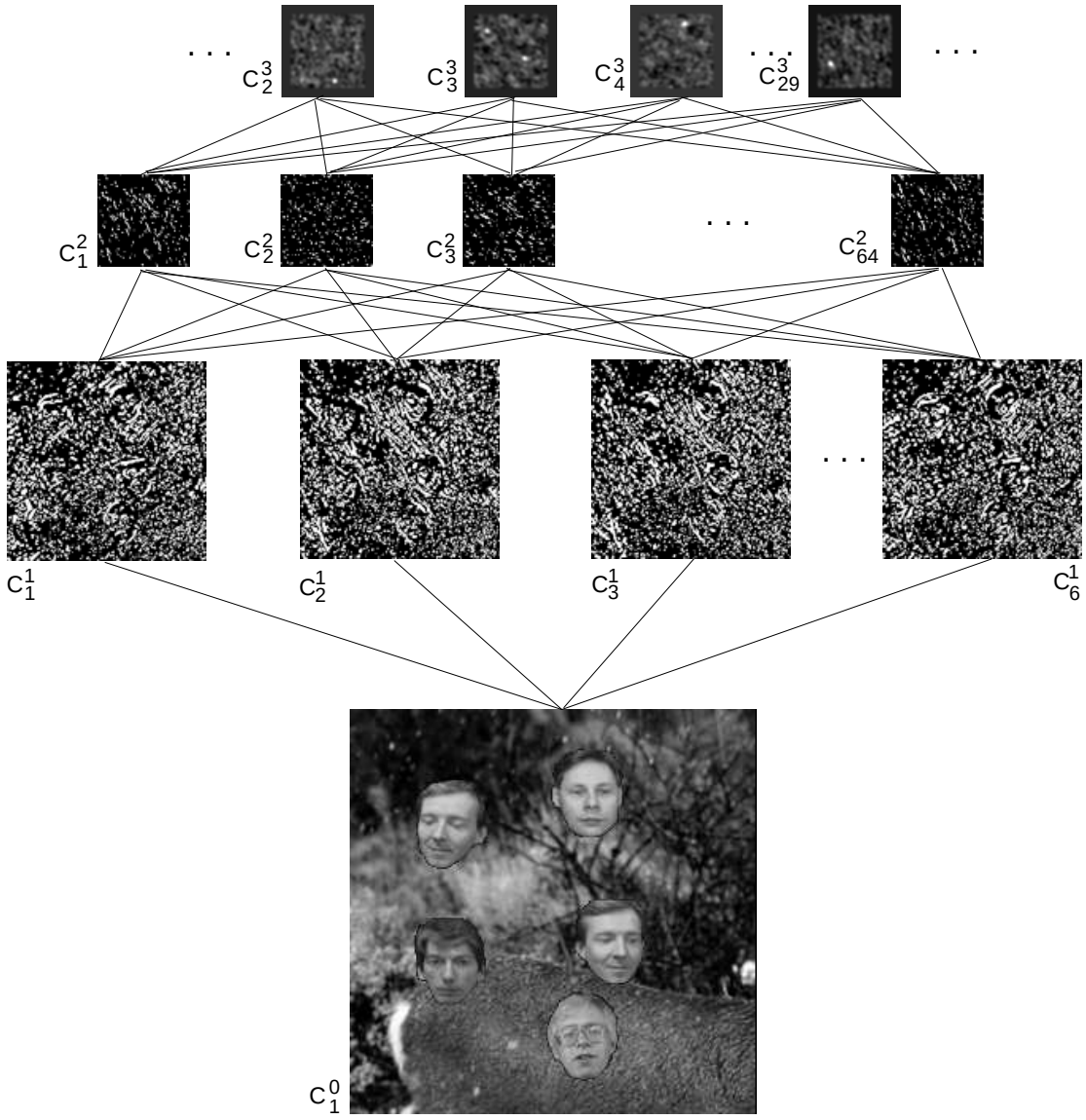


Figure 7.3: *Example of passing a detection image generated from the ORLS-40 patch dataset through the recognition branch of the trained architecture. For simplicity, only selected C-cell planes of each layer are shown. At the output layer at the top, only those C-cell planes that correspond to the classes that occur in the input image (2, 3, 4, and 29) are shown. It can be seen that the C^3 planes clearly exhibit activity peaks at locations where the corresponding person's face is present. See text for further explanation.*

Algorithm 5 Activity Peak Detection: APD($\mathbf{I}$, θ)

$\mathbb{M} \leftarrow \emptyset$
for $p \leftarrow 1$ to $n_p(3)$ **do**
  for $x \leftarrow 1$ to $d_x(3)$ **do**
    for $y \leftarrow 1$ to $d_y(3)$ **do**
      $max \leftarrow true$
      for $xx \leftarrow -1$ to 1 **do**
        for $yy \leftarrow -1$ to 1 **do**
          if $xx \neq 0 \wedge yy \neq 0 \wedge C_p^3(x, y; \mathbf{I}) < C_p^3(x + xx, x + yy; \mathbf{I})$ **then**
            $max \leftarrow false$
          end if
        end for
      end for
      if $max = true$ **then**
        for $q \leftarrow 1$ to $n_p(3), q \neq p$ **do**
          for $xx \leftarrow -1$ to 1 **do**
            for $yy \leftarrow -1$ to 1 **do**
              if $xx \neq 0 \wedge yy \neq 0 \wedge C_p^3(x, y; \mathbf{I}) < C_q^3(x + xx, x + yy; \mathbf{I})$ **then**
                $max \leftarrow false$
              end if
            end for
          end for
        end for
      end if
      if $max = true \wedge C_p^3(x, y; \mathbf{I}) > \theta$ **then**
        $\mathbb{M} \leftarrow \mathbb{M} \cap \left(\frac{x * d_x(0)}{d_x(3)},\ \frac{y * d_y(0)}{d_y(3)},\ p \right)$
      end if
    end for
  end for
end for
return $\mathbb{M}$

x,y-position (in terms of input image coordinates) and the class identifier (which is the index of the current plane). Finally, the algorithm returns the detection results as a set of triples.

7.3 Results

For a systematic evaluation of the performance of the proposed architecture for segmentation-free multi-class detection, the following experiment was carried out: For each of the three image domains MNIST-10, COIL-20, and ORLS-40, 500 detection test images were generated as described in Sect. 7.2.1 using randomly selected images from the odd numbered views of the raw image databases (examples are shown in Fig. 7.2).

The recognition architecture was trained using distorted training images (the same as used for the experiments in the previous chapter, see Sect. 7.2.1 for details). The detection

test images were then passed through the three layered HFM and Alg.5 was used to obtain the recognition results for a varying threshold θ.

Similar to the confidence based classification experiment in the preceding chapter, for each chosen threshold value the following quantities were aggregated by matching the recognition results obtained from each detection test image against the "ground truth" which was stored during generation of the detection test images:

- *True positive (TP):* A detected object exists in the test image at the specified location and the class identifier is correct.
- *False positive (FP):* Either a detected object is not present in the test image at the specified location or an object is present but the class identifier is incorrect.
- *False negative (FN):* An object is present in the test image, but the architecture has not detected an object at that position.

In order to judge whether a detected object position matches the ground truth (see above), we tolerate an inaccuracy of $+/-$ 4 cell positions (this corresponds to a shift of 7% of the object size). [4]

Analogous to the confidence based classification experiment we can again create a Precision vs. Recall curve based on these three values. The results for the three datasets are shown as solid curves in Fig. 7.4. The area under the curve values (AUC) are 0.347 for the MNIST-10 dataset, 0.682 for the COIL-20 dataset, and 0.886 for the ORLS-40 dataset. For comparison, the dashed curves in Fig. 7.4 show the results for the same experiment, but using a black background instead of a natural scene image. Here the AUC values are 0.739 for MNIST-10, 0.959 for COIL-20, and 0.947 for ORLS-40.

From the results it can be seen that the best overall performance is obtained for the ORLS-40 dataset. Even though this dataset consists of 40 classes and for training only 5 different views of each person are available, the architecture achieves an astonishingly high detection performance. One explanation for this is that the outside shape of the faces remains very stable across all classes, which makes the detection task comparatively easy. This is supported by the fact that the performance difference between the settings "clutter" and "black" is only minor. As a consequence the model has more "representational capacity" left over for encoding specific face features that make it possible to distinguish between different persons.

The results for the COIL-20 dataset are not as good as for the ORLS-40 dataset, but still a high performance is reached. Here, a much larger difference between the "clutter" and the "black" setting can be observed. This is probably due to the fact that a larger variety of different shapes occurs in the dataset which the model needs to encode. This causes a higher risk of false positives resulting from objects which may be present in the background image and which might look similar to objects in the training set.

For the MNIST-10 dataset the performance is much lower than for the other two. For the "black" setting a reasonable performance can be reached, whereas the performance for the "clutter" setting is much lower. This can be explained by the special properties of the

[4] The reconstruction of the original position of an object in the input image is only coarse, because the spatial resolution of the final C-cell planes of the HFM is only 25% of the input image (see line 25 in Alg. 5).

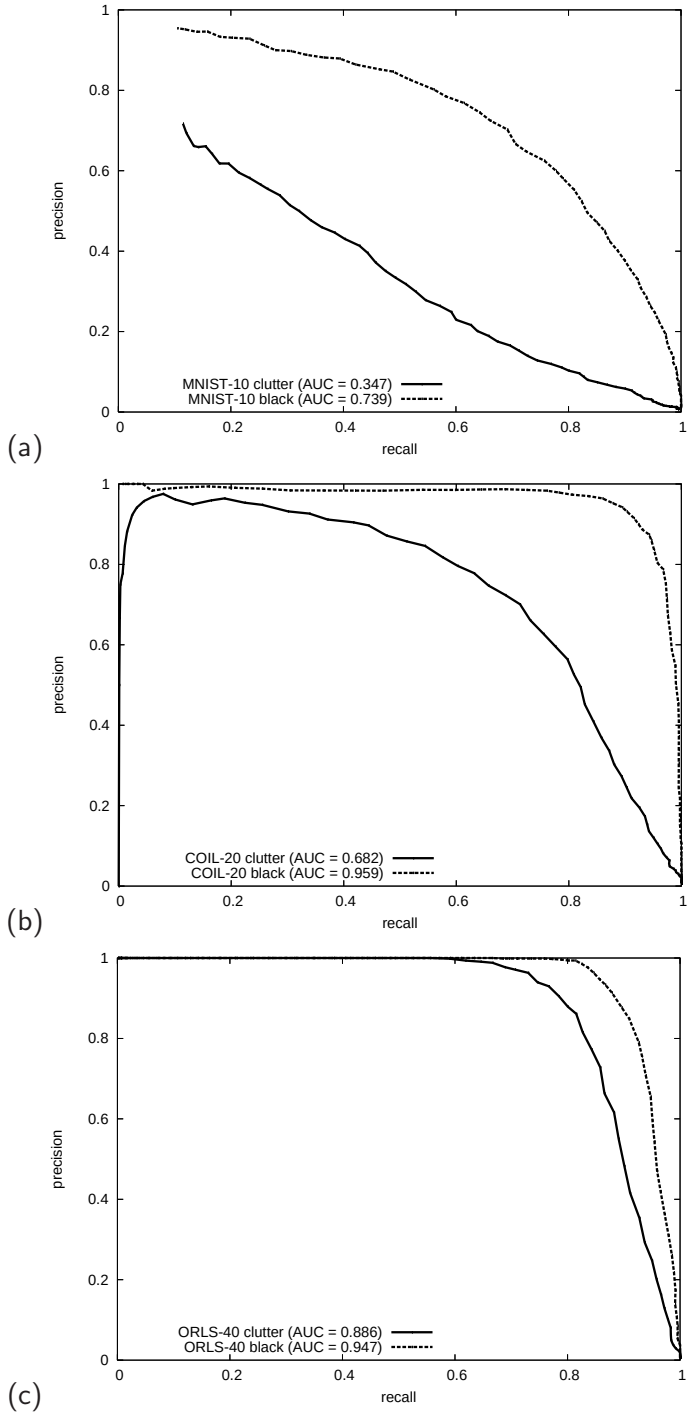


Figure 7.4: *Precision vs. Recall plots for the segmentation free multi-class detection experiment for the three test datasets MNIST-10 (a), COIL-20 (b), and ORLS-40 (c). Detection test images (examples are shown in Fig. 7.2) were created using undistorted images from the odd numbered examples and were embedded into natural scene images (solid curves labeled "clutter"). For comparison, each plot also includes the detection results for a black background without clutter (dashed curves labeled "black"). The architecture was trained on distorted images patches taken from the even numbered views.*

dataset. Even though there are only 10 classes, handwritten digits exhibit a very challenging variety of different shapes, not only across different classes, but also within a single class. Since handwritten digits can only be distinguished by shape properties as opposed to textural properties, the MNIST-10 detection dataset must be considered a very difficult benchmark. The special properties of the MNIST-10 dataset are accounted for in the parameterization of the model by using a large reduction of the plane resolution on the first layer. This leads to an effective increase of the receptive field size of the profiles on the first layer allowing them to capture global shape properties rather than textural details.

For the results that were presented in this experiment, one has to keep in mind that *simultaneous classification and detection* without any further means of localization — such as either low level segmentation or high level cues — is a much more difficult task compared to mere *classification*. While simple classification has to discriminate only a finite and a priori known number of objects, simultaneous detection must deal with an infinite number of a priori unknown distractors. In the present test scenario, we expose the system to the maximally difficult type of distractors: arbitrary natural imagery, which imposes significantly larger difficulty than, e.g., a textured background. In addition, natural background leads to a much higher risk of false positives, because objects similar to the ones in the training set may be present in background images.

7.4 Discussion

In this chapter we have proposed an architecture that allows for segmentation-free multi-class detection of objects in natural scene images. The architecture is based on a two-layered training HFM which processes distorted image patches from a training set in order to create first- and second-layer receptive field profiles by unsupervised NMFSC decomposition. Using the outputs of the training model, linear discriminant functions (LDFs) are computed using supervised learning. Recognition is achieved by a three layered HFM with an increased input resolution which re-uses the first- and second-layer receptive field profiles obtained in the training phase and utilizes LDFs as detector units on a third model layer.

The feasibility of the approach was demonstrated in an experiment where synthetic detection test images were created using objects from the three image domains that were randomly embedded into natural scene images. The performance of the architecture was evaluated by measuring the area under the Precision vs. Recall curve. The results showed that the approach is especially well suited for the segmentation-free detection and identification of faces. Here a very low number of training views for each class is sufficient in order to obtain a high recognition performance. The approach also works well for the domain of natural objects where also a reasonable recognition performance can be reached. The domain of handwritten digits turned out to be much more challenging than the other two image domains. Here the performance clearly remains below that achieved for the other two domains.

Parameter Details

Table 7.1 shows the main model parameters used for the experiments presented in this chapter. The parameterization is almost identical for all three image domains. Except for

the MNIST-10 dataset, the larger reduction of the plane resolution is done on the first layers (see discussion above).

	MNIST-10	*COIL-20*	*ORLS-40*
Training model			
layer 1 S-cell model	LINEAR	LINEAR	LINEAR
$d_x(1)$	15	32	32
$d_y(1)$	15	32	32
$n_p(1)$	6	6	6
$d_p(1)$	5	5	5
γ_1	0.91	0.91	0.91
θ_1	0.1	0.1	0.1
σ_1	1.2	1.2	1.2
layer 2 simple cell model	LINEAR	LINEAR	LINEAR
$d_x(2)$	15	15	15
$d_y(2)$	15	15	15
$n_p(2)$	64	64	64
$d_p(2)$	5	5	5
γ_2	0.91	0.91	0.91
θ_2	0.1	0.1	0.1
σ_2	0.5	0.5	0.5
Test model			
layer 1 S-cell model	LINEAR	LINEAR	LINEAR
$d_x(1)$	60	128	128
$d_y(1)$	60	128	128
$n_p(1)$	6	6	6
$d_p(1)$	5	5	5
γ_1	0.91	0.91	0.91
θ_1	0.1	0.1	0.1
σ_1	1.2	1.2	1.2
layer 2 simple cell model	LINEAR	LINEAR	LINEAR
$d_x(2)$	60	60	60
$d_y(2)$	60	60	60
$n_p(2)$	64	64	64
$d_p(2)$	5	5	5
γ_2	0.91	0.91	0.91
θ_2	0.1	0.1	0.1
σ_2	0.5	0.5	0.5
layer 3 simple cell model	LINEAR, with bias	LINEAR, with bias	LINEAR, with bias
$d_x(3)$	60	60	60
$d_y(3)$	60	60	60
$n_p(3)$	10	20	40
$d_p(3)$	15	15	15
γ_2	-	-	-
θ_2	-	-	-
σ_3	0.5	0.5	0.5

Table 7.1: Model parameters for the HFM architecture for segmentation-free multi-class object detection.

CHAPTER 8

Conclusion

In this chapter, a brief summary and a discussion of the results of this thesis are given and potential future research activities are discussed.

8.1 Summary

In this thesis we have investigated the application of a *Hierarchical Feed-Forward Model* (HFM) to different computer vision tasks such as image normalization, image patch classification, and segmentation-free, multi-class object detection. In contrast to other methods for pattern recognition which purely rely on a statistical analysis of given training datasets in order to build models of object classes, the HFM approach employs a complementary strategy: Input stimuli are projected into an abstract, high-dimensional feature representation. This feature representation proves useful, because it incorporates *a-priori knowledge* about the statistical structure of natural imagery in order to "simplify" the object manifolds in the representation space. This can be interpreted as a normalization with respect to distortions of the input that *typically* occur, e.g., when observing objects with a camera, where the resulting images are subject to sensor noise, affine transformations, or local deformations.

One part of this a-priori knowledge is encoded in the model's topology which – in close analogy to the neurophysiological organization of biological vision systems – consists of a hierarchy of alternating layers of feature-extracting S-cells and spatial-pooling C-cells. Another portion of a-priori knowledge is encoded in receptive field profiles of S-cells, i.e. the synaptic connections between different layers of the model. In analogy to the adaptive behavior of biological simple cells, unsupervised learning methods can be applied in order to obtain "optimized" receptive field profiles which are adapted to the current image domain.

In the following, we briefly summarize the major contributions and results of this thesis.

Model Definition

The generalized definition of the model in Chapt. 3 is inspired by previously proposed hierarchical feed-forward models like the Neocognitron model by Fukushima [25], the HMAX model by Riesenhuber and Poggio [84, 85, 88] and, more recently, the model by Wersing and Körner [107]. From the latter work, we adopted the mechanism for lateral competition among S-cell responses which, in [107], was shown to be superior to the maximum operation as employed by the original HMAX model.

Compared to previous models, the definition provided in Chapt. 3 is more flexible in the sense that (*i*) an arbitrary number of layers can be used, (*ii*) each layer is formally treated as equal, (*iii*) the reduction in spatial resolution from one layer to the next can be controlled more explicitly by directly defining the number of cells in each plane, and (*iv*), the exact choice of receptive field profiles is not part of the basic model definition, but kept flexible. This allows for a better analysis of different choices of receptive field profile models (see Chapt. 5).

Evaluation Methodology

In Chapt. 4, we have proposed a novel complexity measure for multi-class datasets, called the *Average Nearest Neighbor Descriptor* (ANND), which computes for a given multi-class dataset a single, well-bounded and real-valued complexity score. The computation of the score is based on the average accuracy of collections of simple 1-nearest neighbor classifiers that are generated from the data itself. As illustrated using several toy examples, the measure is highly stable and accounts for different factors which have been identified in previous literature as affecting the complexity of discrimination problems, such as "class ambiguity," "complexity of decision boundaries," "sample sparsity," and "feature space dimensionality" [39].

We argue that the ANND measure provides a highly practical tool for data-mining researchers, because the method is free of crucial parameters and allows one to receive a judgment of the "difficulty" of the discrimination task at hand prior to applying a specific classification approach to the data. Further, the measure can be used for analyzing the adequacy of feature measurements, optimizing free parameters of feature extraction methods, optimizing the size of a training dataset, and making predictions about the generalization performance that can be expected when training a classifier on a specific dataset.

Image Normalization

In Chapt. 5, we have used the ANND measure in order to quantify the usefulness of the abstract feature representation that is provided by the output space of HFM – or, in other words, to measure the success of the HFM in "simplifying" the above mentioned object manifolds. For this, we created an experimental setup in which three different multi-class natural image datasets – covering the domains of handwritten digits, small objects and human faces – were subjected to substantial synthetic transformations. First, the ANND measure was used for analyzing the effects that these transformations have on the complexity of the datasets. Here, it turned out that even after adding only a slight mount of distortion, datasets of limited size quickly lose their statistical structure, meaning that the object manifolds in image space become too complex to be sufficiently described by small numbers

of reference examples. Next, when passing these highly distorted datasets through the HFM and using the activities of the final C-cell planes as new representations, it was shown that much of the statistical structure of such datasets – analyzed in terms of the ANND – can be recovered again.

Of course, the success of the HFM in normalizing multi-class natural image datasets depends on the parameterization of the model, and especially on the number of layers and planes used as well as on the types of receptive field profiles. Here, the ANND proved useful for optimizing model parameters and comparing different types of receptive field profile models. For a single layered model, we experimented with Gabor profiles (as used extensively in previous work, e.g., [107, 48, 44, 17]), normalized random profiles (as a baseline), and finally, profiles, which were obtained using an unsupervised learning method called *Non-Negative Matrix Factorization with Sparseness Constraints* (NMFSC, [43]). Here, it transpired that the latter exhibit a slightly better performance than the other approaches, as unsupervised learning obviously allows additional "tuning" of profiles to the image domain, yielding slightly better response properties than "general" Gabor filters.

Extending the experimental setup by an additional layer, and using normalized random profiles and NMFSC-profiles on the second layer, it was shown that an additional reduction of the apparent complexity of natural image datasets can be achieved. For the second layer, we again found that the NMFSC profiles lead to lower and more stable ANND values than random profiles. In summary, in Chapt. 5, it was shown that the HFM is capable of significantly reducing the apparent complexity of natural image datasets.

Image Patch Classification

From the experiments that were carried out in Chapt. 5, we obtained an "optimal" parameterization of the HFM, both in terms of selecting topological parameters as well as choosing an appropriate receptive field profile model. Using this parameterization, in Chapt. 6 we applied the HFM approach to the problem of image patch classification. For this, we first constructed several discrimination tasks and investigated standard classification approaches operating on the output C-cell activity space of the HFM. Here, so-called *Template View Tuned Units* (T-VTUs, [107, 84, 85, 88]), which correspond to simple 1-nearest-neighbor classifiers, proved to exhibit a high classification accuracy on challenging test datasets. However, the number of chosen units, i.e., the number of reference nodes stored in the representations of the classifiers, had to be quite large in order to achieve a reasonable performance. Due to the high dimensionality of the feature representation, the T-VTU approach has some computational disadvantages. Therefore, we proposed new types of units, called *Condensed View Tuned Units* (C-VTU), which were obtained by applying a spectral clustering scheme [94] to large T-VTU representations. With this method, it was possible to significantly reduce the number of units while preserving most of the classification performance.

In a next step, we extended the experimental setup in order to investigated *confidence based recognition*, i.e. classification with rejection of "unknown" input patterns. For this task, we added clutter images to the test datasets and applied *Radial Basis Functions* (RBFs) as well as *Linear Discriminant Functions* (LDFs). While RBFs are constructed directly from C-VTU representations, LDFs require an additional supervised learning step based on a T-VTU representation. The experimental results showed that for both approaches, a high

recognition performance could be achieved.

In summary, the HFM approach was found to be well suited for image patch classification tasks where a limited number of training examples is available and the test images are subject to severe distortions like affine transformations, noise, and background clutter.

Segmentation-Free Multi-Class Object Detection

In Chapt. 7 we applied the HFM to the problem of segmentation-free multi-class object detection, i.e., locating and identifying objects in natural scene images. For this, we constructed an experimental setup in which synthetic detection images were generated from multi-class natural image datasets by segmenting the objects and embedding them at random positions into images of natural scenes (For examples, see Fig. 7.2).

For recognition of objects in such detection images, a special processing architecture was proposed, consisting of a training and a recognition branch. In the training branch, a two-layered HFM which operates on image patches is used for creating a set of LDF units (as described in Chapt. 6). The recognition branch employs a three-layered HFM with an increased input dimensionality. Receptive field profiles for the first and the second layer of the model are inherited from the training branch. Since the computation of the responses of an LDF unit relies on applying the dot-product between the final C-cell activities and the weight-vector of the unit, it is possible to directly use each weight vector as a third-layer receptive field profile. Each unit then serves as a detector cell for a specific object class. On the output C-cell planes of the three-layered recognition model, activity peaks can be observed at locations corresponding to the positions of a specific object in the input image. Finally, the detection results are obtained by applying an algorithm for activity peak detection on the final C-cell planes.

To evaluate the performance of the proposed method, we generated a large number of detection images using the MNIST-10, the COIL-20, and the ORLS-40 datasets and created Precision-vs-Recall plots by matching the recognition results against ground truth obtained during the generation process. Using the area under these curves as a measure of performance, interestingly, the best results were obtained on the ORLS-40 dataset – despite the fact that this dataset consists of 40 classes and only provides 5 training views of each person. The fact that the accuracy for the ORLS-40 test images was higher than for the other two datasets was explained by the special properties of the dataset. Faces have a rather stable global shape which makes it comparatively "easy" to detect them in scene images, resulting in a low risk of false positives. For the objects of the COIL-20 dataset, the global shape is not as invariant across classes which – in our experimental setup – led to a lower detection performance. The MNIST-10 dataset was found to be the most difficult dataset for the detection experiments. Even though the dataset only consists of 10 classes and a large number of training images is available, handwritten digits exhibit a large inner-class variance and additionally lack any textural features by which the classes can be distinguished.

Application in Practical Computer Vision Setups

The experimental results that have been presented in this thesis suggest that the HFM approach provides a framework well-suited to many practical computer vision setups. This

is especially true for object recognition applications, where certain restrictions on the appearance of the objects – as often required by former approaches – cannot be enforced any longer. For instance, in the field of autonomous systems which operate in natural environments, recognition systems have to deal with an input signal that is subject to high variability. This variability is caused by multiple degrees of freedom that the objects in question are subject to, such as varying orientation, arbitrary viewing distances and angles, uncontrolled lighting conditions, etc. Additionally, the input images are subject to other sources of distraction such as sensor noise or light reflections. Here, it is impossible to exhaustively cover the variability using training datasets of limited size. Therefore, processing methods are needed which employ a-priori knowledge about natural imagery, in combination with the ability to learn object representations from small-sized training datasets.

Further, the feed-forward nature of the approach allows a highly efficient computation of the recognition results. The computation of the cell responses in each layer of the HFM offers excellent possibilities of optimization, because the main operation is convolution and the computation can be parallelized plane-wise. This make the approach especially useful for high-throughput real-time applications.

8.2 Perspectives

In this final section, we briefly outline some potential future research activities:

Color Vision

Despite the fact that color provides a rich source of additional discriminative information in many application scenarios, in this thesis we have restricted our investigations to the processing of gray-scale images only. However, in theory, the current formal definition of the HFM already allows for the processing of color images. Color processing can be implemented in a straight-forward way by using, instead of a single C-cell plane C^0 on the input layer of the HFM, multiple planes where each of them corresponds to one component of a chosen color space. This would imply that already on the first layer of the model, multi-dimensional receptive field profiles are used.

Open questions in this regard are, which color space should be used (RGB, HSV, or other representations), and whether the NMFSC decomposition algorithm as described in Chapt. 5 is capable of producing suitable multi-dimensional receptive field profiles for the first model layer.

Motion Recognition

Analogous to the straight-forward extension of the model for color processing, the HFM could potentially also be used for motion processing. For this, each frame of an input motion sequence could be used as an input plane for the HFM. Based on this input, multi-dimensional receptive field profiles could be used (and possibly also learned using the NMFSC decomposition algorithm), which respond to spatio-temporal features that occur in the input sequence (e.g., see [73]).

A problem with this approach is however that the quantization of the motion sequence into a fixed number of discrete frames introduces a limitation that cannot be easily overcome

without substantially extending the current formal definition of the HFM topology. How such an extension could be achieved is an interesting question for future research activities.

Recognition of Partly Occluded Patterns

The experimental results that were presented in this thesis proved that the HFM is capable of tolerating high degrees of transformations and distortions of the input stimuli. Thus, the approach is also able to deal with minor partial occlusions of input patterns. However, for larger partial occlusions, as might also be encountered in practical applications, the HFM is likely to fail to produce reliable recognition results. The reason for this is that recognition on the final layer of the HFM is still *holistic*, meaning that the complete set of feature must be present in order for the correct unit to give a high response.

To overcome this limitation, the final recognition layer of the HFM could be modified in such a way that instead of a single classification unit for each object, several units could be trained on designated *local parts* of the stimulus. Using an additional cell for combining the responses of these local classifiers into a single response by incorporating the offsets of the parts would then correspond to implementing a generalized Hough transform [4] in the wiring of the HFM (similar as in [59, 60]).

For this parts-based extension of the HFM, a key question is how to arrange the parts on which the local classifiers are trained. (An exploratory study using this option suggested that partitioning the final cell planes in a simple grid-like fashion, e.g., 2×2 or 3×3, was not sufficient to produce results comparable to the holistic approach.)

Application in Hybrid Systems

In Chapt. 7, it was demonstrated how the HFM approach can be applied to the problem of segmentation-free multi-class object detection. For this, we would like to point out that in many practical computer vision setups, additional information cues are available from other system components which can be used to restrict the "attention" of the recognition system to a certain *region of interest* in the sensor's field of view. These additional information cues might either be derived from user interaction (e.g., in previous work, a system for the recognition of pointing gestures was developed [31]), or from data driven saliency cues, such as described in [33, 30].

A popular approach for encoding such information cues is the use of a so-called *attention map* [105, 33]. As demonstrated for instance in [105] for the HMAX model, such an attention map can easily be incorporated into the processing stream of the HFM, simply by multiplying the responses of the final layer by the corresponding activities of the attention map. This coupling of the HFM architecture with other processing modules in a hybrid system is another interesting perspective for future work.

Bibliography

[1] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *Proceedings of the 7th European Conference on Computer Vision*, pages 113–130, London, UK, 2002. Springer-Verlag.

[2] C. H. Anderson and D. C. Van Essen. Shifter circuits: A computational strategy for dynamic aspects of visual processing. *Proc. Natl. Acad. Sci. USA*, 84:6297–6301, 1987.

[3] Franz Aurenhammer. Voronoi Diagrams - A Survey of a Fundamental Geometric Data Structure. *ACM Computing Surveys*, 23(3):345–405, 1991.

[4] D. H. Ballard and C. M. Brown. *Computer Vision*. Prentice-Hall, Englewood Cliffs, NJ, USA, 1982.

[5] H. B. Barlow. Possible principles underlying the transformation of sensory messages. *Sensory Communication*, pages 217–234, 1961.

[6] H. B. Barlow. Unsupervised learning. *Neural Comp.*, 1:295–311, 1989.

[7] I. Bax, H. Bekel, and G. Heidemann. Recognition of gestural object reference with auditory feedback. In *Proc. Int'l Conf. Neural Networks*, pages 425–432, Istanbul, Turkey, 2003.

[8] I. Bax, G. Heidemann, and H. Ritter. A hierarchical feed-forward network for object detection tasks. In H. Szu, editor, *Proc. SPIE Conf. on Independent Component Analyses, Wavelets, Unsupervised Smart Sensors, Neural Networks*, volume 5818, pages 144–152, Orlando, Florida, 2005.

[9] I. Bax, G. Heidemann, and H. Ritter. Face Detection and Identification Using a Hierarchical Feed-forward Recognition Architecture. In *Proc. IEEE Int'l Joint Conf. Neural Networks*, pages 1675–1680, Montral, Qubec, Canada, 2005.

[10] I. Bax, G. Heidemann, and H. Ritter. Using non-negative sparse profiles in a hierarchical feature extraction network. In *Proc. 9th IAPR Conf. Machine Vision Applications*, pages 464–467, Tsukuba Science City, Japan, 5 2005.

[11] I. Bax, G. Heidemann, and H. Ritter. Hierarchical feed-forward network for object detection tasks. *Optical Engineering*, 45(6), 2006.

[12] H. Bekel, I. Bax, G. Heidemann, and H. Ritter. Adaptive Computer Vision: Online Learning for Object Recognition. In C. E. Rasmussen et al., editor, *Proc. DAGM 2004*, pages 447–454, Tübingen, Germany, 2004. Springer.

[13] A. J. Bell and T. J. Sejnowski. The independent components of natural images are edge filters. *Vision Research*, 37(27):3327–3338, 1997.

[14] R. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, Princeton, NJ, USA, 1961.

[15] I. Biederman. Recognition-by-components: A theory of human image understanding. *Psych. Rev.*, 94:115–147, 1987.

[16] B. Blais, L. N. Cooper, and H. Shouval. Formation of Direction Selectivity in Natural Scene Environments. *Neural Comp.*, 12(5):1057–1066, 2000.

[17] M. Carandini and D. J. Heeger. Summation and division by neurons in primate visual cortex. *Science*, 264:1333–1336, 1994.

[18] T. M. Cover and P. E. Hart. Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, IT-13(1):21–27, 1967.

[19] R. Desimone. Face-selective cells in the temporal cortex of monkeys. *Cogn. Neurosci.*, 3:1–8, 1991.

[20] R. P. W. Duin, E. Pekalska, and D. M. J. Tax. The characterization of classification problems by classifier disagreements. In J. Kittler, M. Petrou, and M. Nixon, editors, *Proc. Int'l Conf. Pattern Recognition*, volume 2, pages 140–143, Cambridge, United Kingdom, 2004.

[21] T. Fawcett. Roc graphs: Notes and practical considerations for data mining researchers. Technical Report Technical Report HPL-2003-4, HP Laboratories, Palo Alto, CA, USA., 2004.

[22] D. J. Field. What is the goal of sensory coding? *Neural Comp.*, 6(4):559–601, 1994.

[23] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annual Eugenics*, 7, Part II:179–188, 1936.

[24] W. T. Freeman and E. H. Adelson. The Design and Use of Steerable Filters. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13:891–906, 1991.

[25] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol. Cybern.*, 36:193–202, 1980.

[26] K. Fukushima. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Networks*, 1(2):119–130, 1988.

[27] K. Fukushima. Neocognitron for handwritten digit recognition. *Neurocomputing*, 51:161–180, 2003.

[28] K. Fukushima, S. Miyake, and I. Takayuki. Neocognitron: A neural network model for a mechanism of visual pattern recognition. *Trans. Systems, Man, and Cybernetics*, 13:826–834, 1983.

[29] G. Heidemann. *Ein flexibel einsetzbares Objekterkennungssystem auf der Basis neuronaler Netze*. PhD thesis, Univ. Bielefeld, Faculty of Technology, 1998. Infix, DISKI 190.

[30] G. Heidemann. Focus-of-Attention from Local Color Symmetries. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(7):817–830, 2004.

[31] G. Heidemann, I. Bax, H. Bekel, C. Bauckhage, S. Wachsmuth, G. Fink, A. Pinz, H. Ritter, and G. Sagerer. Multimodal interaction in an augmented reality scenario. In *Proc. Int'l Conf. Multimodal Interfaces*, pages 53–60, New York, New York, October 14-15 2004. ACM Press.

[32] G. Heidemann, H. Bekel, I. Bax, and H. Ritter. Interactive online learning. *Pattern Recognition and Image Analysis*, 15(1):55–58, 2005.

[33] G. Heidemann, R. Rae, H. Bekel, I. Bax, and H. Ritter. Integrating context-free and context-dependent attentional mechanisms for gestural object reference. *Machine Vision and Applications*, 16(1):64–73, 2004.

[34] G. Heidemann and H. Ritter. Efficient Vector Quantization Using the WTA-rule with Activity Equalization. *Neural Processing Letters*, 13(1):17–30, 2001.

[35] G. Heidemann and H. Ritter. Learning to recognise objects and situations to control a robot end-effector. *KI (Artificial Intelligence), special issue on Vision, Learning, Robotics*, 2:24–29, 2003.

[36] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Research Nat. Bur. Standards*, 49:409–436, 1952.

[37] T. K. Ho. Geometrical complexity of classification problems. In E. R. Caianiello, editor, *Proc. 7th Course on Ensemble Methods for Learning Machines*, International School on Neural Nets, 2003.

[38] T. K. Ho and H. S. Baird. Pattern classification with compact distribution maps. *Computer Vision and Image Understanding*, 70(1):101–110, 1998.

[39] T. K. Ho and M. Masu. Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):289–300, 2002.

[40] A. Hoekstra and R. P. W. Duin. On the nonlinearity of pattern classifiers. In *Proc. 13th Int'l Conf. Pattern Recognition*, pages 271–275, 1996.

[41] P. O. Hoyer. *Probabilistic Models of Early Vision*. PhD thesis, Computer Science Department, Helsinki University of Technology, 2002.

[42] P. O. Hoyer. Modeling receptive fields with non-negative sparse coding. *Neurocomputing*, 52-54:547–552, 2004.

[43] P. O. Hoyer. Non-negative Matrix Factorization with Sparseness Constraints. *Machine Learning Research*, 5(37):1457–1469, 2004.

[44] P. O. Hoyer and A. Hyvärinen. A multi-layer sparse coding network learns contour coding from natural images. *Vision Research*, 42(12):1593–1605, 2002.

[45] D. H. Hubel and T. N. Wiesel. Receptive fields of single neurons in the cat's striate cortex. *Journal of Physiology*, 148:574–591, 1959.

[46] D. H. Hubel and T. N. Wiesel. Receptive fields, binocular interaction and functional architecture of the cat's visual cortex. *Journal of Physiology*, 160:106–154, 1962.

[47] J. Hummel and I. Biederman. Dynamic binding in a neural network for shape recognition. *Psych. Rev.*, 99:480–517, 1992.

[48] A. Hyvärinen and P. O. Hoyer. A two-layer sparse coding model learns simple and complex cell receptive fields and topography from natural images. *Vision Research*, 41:2413–2423, 2001.

[49] A. Hyvrinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley, New York, NY, USA, 2001.

[50] C.G.J. Jacobi. Über ein leichtes Verfahren, die in der Theorie der Säcularstörungen vorkommenden Gleichungen numerisch aufzulösen. *Reine und Angewandte Mathematik*, 30:51–94, 1846.

[51] S. Kullback and R.A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[52] Y. LeCun. MNIST handwritten digit database. Avaible online at http://yann.lecun.com/exdb/mnist/.

[53] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[54] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[55] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems*, pages 556–562, 2000.

[56] N. Logothetis, J. Pauls, H. Bülthoff, and T. Poggio. View-dependent object recognition by monkeys. *Curr. Biol.*, 4:401–414, 1994.

[57] J. Louie. A Biological Model of Object Recognition with Feature Learning. Master's thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Cambridge, MA, USA, 2003.

[58] D. R. Lovell, T. Downs, and A. C. Tsoi. An evaluation of the neocognitron. *IEEE Trans. Neural Networks*, 8(5):1090–1105, 1997.

[59] D. G. Lowe. Object recognition from local scale-invariant features. In *Proc. of the International Conference on Computer Vision ICCV*, pages 1150–1157, Corfu, Greece, 1999.

[60] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[61] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symp. Math. Stat. Probab.*, volume 1, pages 281–297, 1965.

[62] E. B. Mansilla and T. K. Ho. On classifier domains of competence. In J. Kittler, M. Petrou, and M. Nixon, editors, *Proc. Int'l Conf. Pattern Recognition*, volume 2, pages 136–139, Cambridge, United Kingdom, 2004.

[63] D. Marr and H. K. Nishihara. Representation and recognition of the spatial organization of three-dimensional shapes. *Proc. R. Soc. Lond. B: Biol. Sci.*, 200:269–294, 1978.

[64] R. Meir and G. Rätsch. An introduction to boosting and leveraging. *Advanced lectures on machine learning*, pages 118–183, 2003.

[65] B. Mel. SEEMORE: Combining color, shape, and texture histogramming in a neurally inspired approach to visual object recognition. *Neural Comp.*, 9:777–804, 1997.

[66] A. Meyering and H. Ritter. Learning 3D-Shape-Perception with Local Linear Maps. In *Proc. Int'l Joint Conf. on Neural Networks*, volume IV, pages 432–436, Baltimore, MA, USA, 1992.

[67] D. Mumford. On the computational architecture of the neocortex. II. The role of corticocortical loops. *Biol. Cyb.*, 66:241–251, 1992.

[68] H. Murase and S. K. Nayar. Visual Learning and Recognition of 3-D Objects from Appearance. *Int'l J. of Computer Vision*, 14:5–24, 1995.

[69] S. A. Nene, S. K. Nayar, and H. Murase. Columbia Object Image Library (COIL-100). Technical Report CUCS-006-96, Dept. Computer Science, Columbia Univ. New York, N.Y. 10027, 1996.

[70] S. A. Nene, S. K. Nayar, and H. Murase. Columbia Object Image Library (COIL-20). Technical Report CUCS-005-96, Dept. Computer Science, Columbia Univ. New York, N.Y. 10027, 1996.

[71] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[72] Nova Development Corporation, 23801 Calabasas Road, Suite 2005 Calabasas, California 91302-1547, USA. *Art Explosion Photo Gallery*.

[73] B. A. Olshausen. Learning Sparse, Overcomplete Representations of Time-Varying Natural Images. In *Proc. Int'l. Conf. Image Processing*, pages 41–44, Barcelona, Spain, 2003.

[74] B. A. Olshausen, C. H. Anderson, and D. C. Van Essen. A neurobiological model of visual attention and invariant pattern recognition based on dynamic routing of information. *Neuroscience*, 13(11):4700–4719, 1993.

[75] B. A. Olshausen and D. J. Field. Sparse coding of natural images produces localized, oriented, bandpass receptive fields. Technical Report CCN-110-95, Department of Psychology, Cornell University, Ithaca, New York, 1995.

[76] B. A. Olshausen and D. J. Field. Emergence of simple cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

[77] B. A. Olshausen and D. J. Field. Natural image statistics and efficient coding. *Network*, 7:333–339, 1996.

[78] A. Opelt, M. Fussenegger, A. Pinz, and P. Auer. Weak Hypotheses and Boosting for Generic Object Detection and Recognition. In T. Pajdla and J. Matas, editors, *Proc. ECCV 2004*, pages 71–84. Springer, 2004.

[79] P. Paatero and U. Tapper. Positive matrix factorization: a non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5:111–126, 1994.

[80] S. E. Palmer. *Vision Science – Photons to Phenomenology*. MA: Bradford Books/MIT Press, Cambridge, MA, USA, 1999.

[81] D. Perrett and M. Oram. Neurophysiology of shape processing. *Img. Vis. Comput.*, 11:317–333, 1993.

[82] W. H. Press, B. P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing, second edition*. Camebridge University Press, Cambridge, MA, USA, 1992.

[83] R. Rao and D. H. Ballard. Dynamic model of visual recognition predicts neural response properties in the visual cortex. *Neural Comp.*, 9:721–763, 1997.

[84] M. Riesenhuber and T. Poggio. Are cortical models really bound by the "binding problem"? *Neuron*, 24:87–93, 1999.

[85] M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Nat. Neurosci.*, 2:1019–1025, 1999.

[86] M. Riesenhuber and T. Poggio. CBF: A New Framework for Object Categorization in Cortex. In S-W. Lee, H. H. Buelthoff, and T. Poggio, editors, *Proc. Biologically Motivated Computer Vision. First IEEE International Workshop*, pages 1–9, Seoul, Korea, 2000.

[87] M. Riesenhuber and T. Poggio. Computational models of object recognition in cortex: A review. Technical Report A.I. Memo No. 1695, C.B.C.L. Paper No. 190, MIT AI Laboratory, Center for Biological and Computational Learning, 2000.

[88] M. Riesenhuber and T. Poggio. Models of object recognition. *Nat. Neurosci.*, 3:1199–1204, 2000.

[89] F. Samaria. *Face Recognition Using Hidden Markov Models*. PhD thesis, Trinity College, University of Cambridge, 1994.

[90] F. Samaria and A. Harter. Parameterisation of a stochastic model for human face identification. In *Proceedings of 2nd IEEE Workshop on Applications of Computer Vision*, Sarasota, FL, USA, 1994.

[91] T. D. Sanger. Optimal Unsupervised Learning in a Single-Layer Linear Feedforward Neural Network. *Neural Networks*, 2:459–473, 1989.

[92] Robert E. Schapire. A brief introduction to boosting. In *Proc. Sixteenth Int'l Joint Conf. on Artificial Intelligence*, pages 1401–1406, 1999.

[93] G. Shakhnarovish and P. Indyk T. Darrell. *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice*. MIT Press, Cambridge, MA, USA, 2005.

[94] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[95] S. P. Smith and A. K. Jain. A test to determine the multivariate normality of a data set. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):757–761, 1988.

[96] K.-K. Sung. *Learning and Example Selection for Object and Pattern Recognition*. PhD thesis, MIT, Artificial Intelligence Laboratory and Center for Biological and Computational Learning, Cambridge, MA, 1996.

[97] M. Tarr. News on views: pandemonium revisited. *Nat. Neurosci.*, 2:932–935, 1999.

[98] M. Tarr and H. Bülthoff. Image-based object recognition in man, monkey and machine. *Cognition*, 67:1–20, 1998.

[99] S. Thorpe, D. Fize, and C. Marlot. Speed of processing in the human visual system. *Nature*, 381:520–522, 1996.

[100] D. J. Tolhurst, Y. Tadmor, and T. Chao. Amplitude spectra of natural images. *Ophthalmic & Physiological Optics*, 12(2):229–232, 1992.

[101] M. Turk and A. Pentland. Eigenfaces for Recognition. *J. Cognitive Neuroscience*, 3:71–86, 1991.

[102] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. Conf. Computer Vision and Pattern Recognition CVPR*, 2001.

[103] P. Viola and M. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[104] G. Voronoi. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. *Reine und Angewandte Mathematik*, 133:97–178, 1907.

[105] D. Walther, L. Itti, M. Riesenhuber, T. Poggio, and C. Koch. Attentional selection for object recognition - a gentle way. In *Proc. 2nd Workshop on Biologically Motivated Computer Vision (BMCV'02), Tuebingen, Germany*, 2002.

[106] H. Wersing, J. Eggert, and E. Körner. Sparse coding with invariance constraints. In *Proc. Int'l Conf. Neural Networks*, pages 385–392, Istanbul, Turkey, 2003.

[107] H. Wersing and E. Körner. Learning optimized features for hierarchical models of invariant object recognition. *Neural Comp.*, 15(7):1559–1588, 2003.

[108] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1101–1113, 1993.