in: *Proceedings of the 1st International Workshop on Statistical Techniques in Pattern Recognition*, (STIPR'97), P. Pudil, J. Novovicova, J. Grim (eds.), pp.113--118, Prague, 1997.

Learning Object Representations by Clustering Banana Wavelet Responses

Gabriele Peters, Norbert Krüger
Institut für Neuroinformatik,
Ruhr-Universität Bochum, Germany

Abstract

For object recognition systems it is essential to have an internal representation of the object to be recognized. We introduce a system which learns such a representation from training images of an object class. Every image is preprocessed with banana wavelets. The result is a description of local areas of an image in terms of curved lines. These features are the input to a clustering algorithm which learns to seperate features specific for an object class from features generated accidentally by variations in background or illumination. This leads to a representation of an object class which can be visualized in form of a line drawing. The representation is sparse, for the most part free of redundancies and independent of varying backgrounds and illuminations in the training images. It comprises representative features only, and has already been utilized successfully for object recognition tasks.

1 Introduction

The human visual system posseses the remarkable property that it can recognize objects even if they produce completely different patterns on the retina, due to varying illuminations or partial occlusions, for example. To be invariant against such variations, the recognition process needs an internal representation of the class the object belongs to. We suggest a mechanism which describes how the formation of such an internal representation can take place. We introduce an algorithm which learns local features of an object class from some training images of instances of this class. The result is a 2D-representation for the presented view of the object class in the form of a curved line drawing (see figures 5 and 7), which can be used for object recognition [3]. We are following biological analogies in establishing our model, e.g., there are hints for calculating curvature in the visual cortex [1]; other analogies to biological systems are discussed in [4]. Our system is divided into the preprocessing with banana wavelets (section 2 and 3) and the algorithm which learns the representation by a statistical analysis of the data gained by the preprocessing (section 4). In section 5 we show the functioning of our model with two object classes as examples. Finally, we list some properties of our system (section 6).

2 What is a Banana Wavelet?

Our preprocessing tool is a set of banana wavelets. A banana wavelet $B_{(\alpha,c)}$ is a complex valued function which depends on two parameters, direction (α) and curvature (c)[1] . It is constructed

[1]In [5] and [3], where banana wavelets and their relationship to Gabor wavelets are described in more detail, a banana wavelet is defined with four parameters. There the two parameters frequency and size are also kept

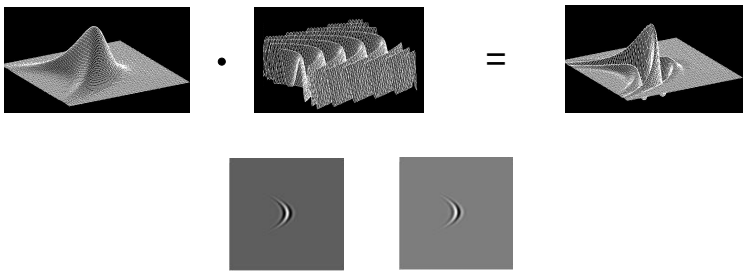

Figure 1: A banana wavelet is the product of a curved Gaussian and a curved wave function. Top: Real part of a banana wavelet. Bottom: Real and imaginary part of a banana wavelet depicted as gray level images with white pixels encoding high values.

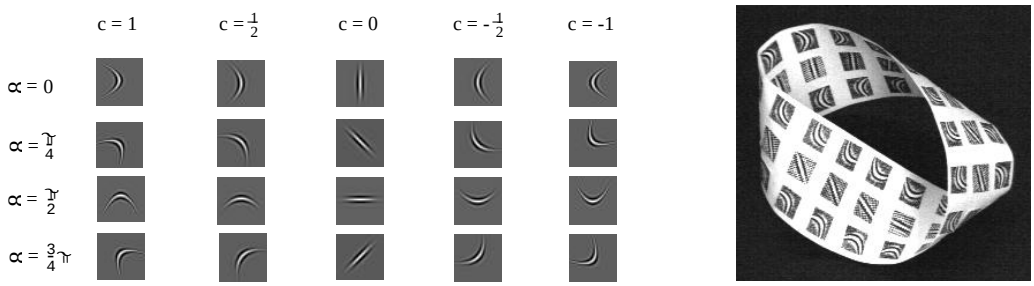

Figure 2: Left: This banana plant consists of four different directions and five different curvatures. Only the real part of each wavelet is shown. Right: The space established by (α, c) forms a Möbius topology, here shown for 16 orientations and 3 curvatures.

from a rotated and curved complex wave function and a Gaussian rotated and curved in the same way (see figure 1). We define a banana wavelet as follows:
Definition A *banana wavelet* $B_{(\alpha,c)}$ is a mapping $B_{(\alpha,c)} : \mathbb{R}^2 \to \mathbb{C}$

$$B_{(\alpha,c)}(x,y) := \gamma \cdot \exp\left(-\frac{k^2}{2}\left(\left(\frac{x_c}{\sigma_x}\right)^2 + \left(\frac{y_c}{\sigma_y}\right)^2\right)\right) \cdot \exp\left(i \cdot k \cdot x_c\right) \tag{1}$$

$$\text{with } \begin{pmatrix} x_c \\ y_c \end{pmatrix} := \mathcal{C}_c\left(\mathcal{M}_\alpha \begin{pmatrix} x \\ y \end{pmatrix}\right), \quad \mathcal{M}_\alpha := \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix}, \quad \mathcal{C}_c \begin{pmatrix} x \\ y \end{pmatrix} := \begin{pmatrix} x - c \cdot y^2 \\ y \end{pmatrix},$$

and $\alpha, c, \gamma, k \in \mathbb{R}$.

This means that $\mathcal{M}_\alpha$ carries out a rotation with the angle α, and $\mathcal{C}_c$ performs a curving with the extent c. k determines the frequency of the wave. The set of Gabor wavelets is a subset of the set of banana wavelets for Gabor wavelets are characterized by the parameters frequency and orientation only, i.e., they are uncurved. For our preprocessing we use a *set* of banana wavelets, a *banana plant* (see figure 2). A banana plant can be generated by rotating and curving the mother wavelet $B_{(0,0)}$.

variable whereas here we work with one frequency and one size only.

3 Preprocessing with Banana Wavelets

In every training image we first manually define the salient points of an object, e.g., the left edge of a can or the eyes of a face, so that a landmark in one training image has a corresponding landmark in every other training image. The surroundings of these landmarks are examined for curved lines or edges. This is done by a *banana transform*, i.e., a convolution of the training image with a banana plant. The *response* of a banana wavelet $B_{(\alpha,c)}$ at a position (x_0, y_0) in a gray level image $I(x, y)$ is defined by $\mathcal{R}_{(\alpha,c)}(x_0, y_0) := \left|\left(I * B_{(\alpha,c)}\right)(x_0, y_0)\right|$. By performing a banana transform we receive responses $\mathcal{R}$ at every position in a landmark's surroundings for every banana wavelet of the plant. These responses are arranged in a 4D metric space - the *banana space* $\mathcal{B}$ - which is established by (x, y, α, c) with (x, y) describing pixels in the landmark's surroundings and (α, c) specifying wavelets of the plant. The metric of $\mathcal{B}$, which is needed for clustering, is denoted by d. It is defined by an Euclidean-like distance in $\mathcal{B}$ where the Möbius topology is taken into account. An exact definition can be found in [5].

The banana transform has an important property which we use in our learning algorithm. Suppose there is a line or an edge with a certain orientation and curvature at some pixel position in a landmark's surroundings. Then the response $\mathcal{R}$ at this position is maximal for that banana wavelet of the plant whose orientation and curvature corresponds best to the local structure in the training image. The convolution is a measure for the correspondence between a part of the image and the wavelet. So in the way we use a banana wavelet one can interpret it as a detector of lines and egdes in images which are oriented and curved in the same way as the wavelet.

4 Learning Important Features of an Object Class

While reading the single steps of the algorithm, please, refer to figure 3 where each step is illustrated.

1. Banana Transform: Let m be the number of training images, and suppose there are n landmarks set in each image at salient points of the object. The surroundings of the i-th landmark of the k-th training image is denoted by I_i^k. Now the first step of our algorithm is to perform a banana transform with I_i^k for $i = 1, \ldots, n$, $k = 1, \ldots, m$, i.e., to calculate the values of the response function $\mathcal{R}$ for all images and all landmark's surroundings. (This is the preprocessing described in section 3.)

2. Significant Features Per Instance: From the assumption that a strong response $\mathcal{R}_{(\alpha,c)}(x, y)$ means that there is a structure in the image at position (x, y) with orientation α and curvature c we calculate the set $\mathcal{S}_i^{I^k}$ of *significant features of the i-th landmark's surroundings of the k-th training image* for $i = 1, \ldots, n$, $k = 1, \ldots, m$. An element $(x, y, \alpha, c) \in \mathcal{B}$ is defined to be such a significant feature, i.e., $(x, y, \alpha, c) \in \mathcal{S}_i^{I^k}$, if it causes a response above a preset threshold $t_1 \in \mathbb{R}^+$, i.e., $\mathcal{R}_{(\alpha,c)}(x, y) > t_1$, and if the response represents a local maximum in $\mathcal{B}$. A significant feature may be specific for the object class, and in this case should be "kept in mind", or it just may derive from the background or a reflection caused by the illumination. In this case it should be"forgotten" (see figure 4) .

3. Union Over All Training Images: We define the union $\mathcal{S}^i := \bigcup_{k=1}^m \mathcal{S}_i^{I^k}$ and calculate it for all landmarks $i = 1, \ldots, n$.

4. Clustering: With each set $\mathcal{S}^i$, $i = 1, \ldots, n$, we perform a simple clustering to seperate the *important* significant features of the landmark's surroundings (i.e., invariant features deriving from structures typical for the object class) from *unimportant* significant features (i.e., noisy features deriving from the background, reflections or other variations). We start from the as-

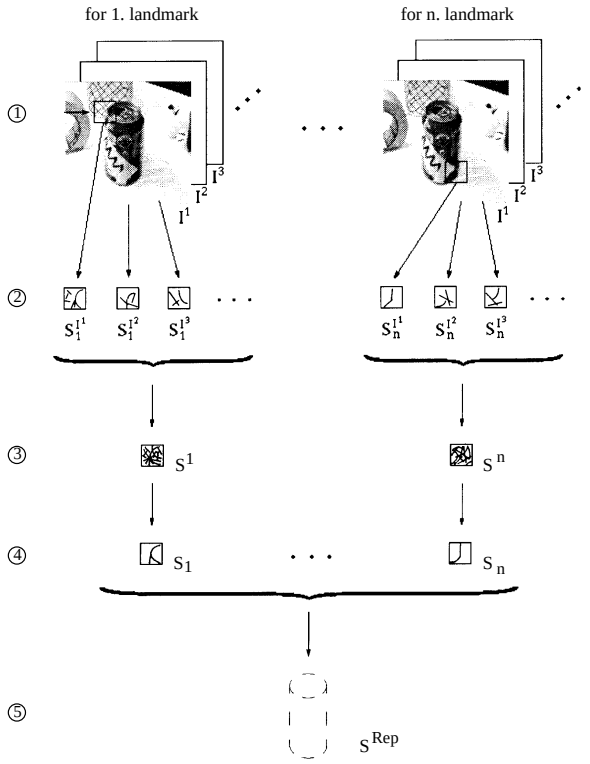

Figure 3: The flowchart of the algorithm. The line drawings are illustrations, not results.

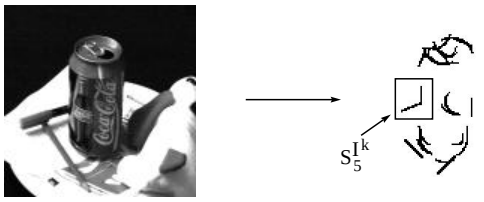

Figure 4: Left: The training image I^k. Right: The generated significant features for this image. The set $\mathcal{S}_5^{I^k}$ for the fifth landmark's surroundings contains a significant feature which is specific for the object class (the left edge of the can) as well as a significant feature which is cleaned out by the learning algorithm later on (the pen in the background).

sumption that the important features are similar in a lot of training images, i.e., they have small distances d in $\mathcal{B}$. The unimportant features are assumed to occur irregularly and be randomly distributed in $\mathcal{B}$ with larger distances d. Thus, by grouping the elements of $\mathcal{S}^i$ into clusters we expect a few large clusters the centroid of which representing an important significant feature, and many more rather small clusters representing all the features caused by chance. Clustering the set $\mathcal{S}^i$ works as follows (the counter l is initialized to 1):

1. Choose an arbitrary element $s \in \mathcal{S}^i$ and form a new cluster $\mathcal{C}_l$ out of it.
2. Add to $\mathcal{C}_l$ successively all remaining elements $\tilde{s} \in \mathcal{S}^i$ with $\exists \; s \in \mathcal{C}_l \; : d(\tilde{s}, s) < t_2$ where $t_2 \in \mathbb{R}^+$ is a preset threshold depending on the size of $\mathcal{B}$.
3. Increment $l := l + 1$ and repeat the steps 1 and 2 with the elements still remaining in $\mathcal{S}^i$ until each element of $\mathcal{S}^i$ is assigned to a cluster.

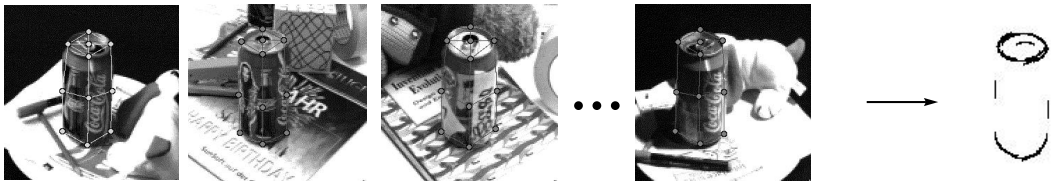

Figure 5: Left: 4 of 60 different training images. 11 landmarks are chosen as shown here. Right: The learned representation for the object class "can". For each learned feature of $\mathcal{S}^{Rep}$ a line with the corresponding orientation and curvature is drawn. Using a SPARC station 20 it takes about 80 s to learn the representation.

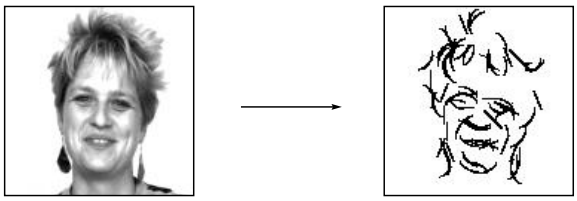

Figure 6: Left: One of the training images. Right: Significant features of this individual face, generated by the algorithm.

The partitioning gained by this clustering is unambiguous (for a proof see [5]). Now we define that an *important cluster* is a cluster with the number of its elements exceeding a preset threshold which depends on the number of elements in $\mathcal{S}^i$. The learned representation $\mathcal{S}_i$, $i = 1, \ldots, n$, for the i-th landmark's surroundings of the object class is defined by the centroids of the important clusters.

5. Learned Representation: The learned representation $\mathcal{S}^{Rep}$ for the object class is just the union of the representation for all landmark's surroundings $\mathcal{S}^{Rep} := \bigcup_{i=1}^{n} \mathcal{S}_i$.

5 Simulations and Results

We would like to demonstrate the usefulness of our algorithm with two object classes, cans and frontal faces. We use 128×128 gray level images and a banana plant with 8 orientations and 7 curvatures. The landmark's surroundings have a size of 17×17 pixels each. For learning a representation of a can we use a set of 60 training images with varying backgrounds, illuminations, and can design and slightly varying elevation angles (see figure 5). To learn the representation of a frontal face we use 30 training images and choose 31 landmarks per face. For an example of significant features of an individual face see figure 6, for the resulting face representation see figure 7.

6 Conclusion and Outlook

The results show that the representation learned is quite independent of varying backgrounds and reflections or shadows caused by different illuminations. Even slight occlusions and rotations of the objects have only a small influence on the result. The special predefinition of features we are looking for (curved lines) reduces the dimensionality of the learning problem. Our representation meets the conditions for a *sparse coding* which has many advantages such as

Figure 7: Left: 4 of 30 different training images. 31 landmarks are chosen for every face as shown in the leftmost image. Right: This is the learned representation for the object class "face". There are some redundancies, e.g., in the area of the eyes and nose. The fact that there is only the left edge of the nose represented whereas the right is not, may derive from the fact that the illumination does not vary randomly in the training images. Note that irregularities such as glasses or a beard are not represented in the result.

redundancy reduction (discussed in [2]). A sparsely coded object is its representation by a small number of binary features (here the presence or absence of a curved line) chosen from a large set of features (all curved lines of all localities, orientations, and curvatures). In comparison to other object recognition systems, e.g., [6], the representation described here has several desirable properties: it comprises only *representative* local features of the object class, it is nearly free of redundancy, it needs less data storage, and it is learned from noisy data by a process which seperates important from unimportant local features. A disadvantage of the system described here lies in the manual positioning of the landmarks. Ways to tackle this problem are suggested in [3].

A representation similar to the one described here is already used successfully in object recognition tasks, e.g., the matching of an object in an unknown scene. The small amount of data the representation consists of enables a fast matching which still can be speeded up by approximating banana wavelet responses by Gabor wavelet responses. All this is described in detail in [3].

References

[1] A. Dobbins, S. Zucker, M. S. Cynader, Endstopped neurons in the visual cortex as a substrate for calculating curvature, *Nature*, vol. 329, pp. 438-441, 1987.

[2] D. Field, What is the Goal of Sensory Coding?, *Neural Computation*, vol. 6, no. 4, pp. 561-601, 1994.

[3] N. Krüger, G. Peters, C. v. d. Malsburg, Object Recognition with a Sparse and Autonomously Learned Representation Based on Banana Wavelets, *Technical Report IR-INI 96-11. Institut für Neuroinformatik*, Ruhr-Universität Bochum, 1996.

[4] N. Krüger, M. Pötzsch, G. Peters, Utilizing Principles of Cortical Processing for Artificial Object Recognition, submitted for *Proceedings of "Information, Theory, and the Brain II"*.

[5] G. Peters, Lernen lokaler Objektmerkmale mit Bananenwavelets, *Technical Report IR-INI 96-09, Institut für Neuroinformatik, (Diploma Thesis)*, Ruhr-Universität Bochum, 1996.

[6] L. Wiskott, J.-M. Fellous, N. Krüger, C. v. d. Malsburg, Face Recognition and Gender Determination, *Proceedings of the International Workshop on Automatic Face- and Gesture recognition*, Zürich, 1995.