Proceedings of the Fourth Electrotechnical and Computer Science Conference ERK'95

25–27 September 1995

Portorož, Slovenia

Volume B

Computer and Information Science
Artificial Intelligence
Robotics
Pattern Recognition
Biomedical Engineering
Numerical Methods
Student Papers

Edited by

Franc Solina and Baldomir Zajc

Slovenia Section IEEE

Ljubljana • Slovenia

Learning to Recognize the Object's Shape

Matevž Kovačič Franc Solina
Computer Vision Laboratory
Faculty of Electrical Engineering and Computer Science
University of Ljubljana, Slovenia
matevz.kovacic@fer.uni-lj.si *franc@fer.uni-lj.si*

Abstract

This paper deals with the problem of recognizing the type of shape of an object. The problem is formalized as a machine learning problem. The descriptions of objects with known type of shapes are presented to the learning algorithm. The algorithm synthesizes knowledge from the presented objects only. The classification procedure obtained in such a way that it can be used to determine the type of shape of unknown objects. The learning algorithm described in the paper is based on kernel density functions as a formalism for representing knowledge about the problem. The results on two domains show that the algorithm is capable of recognising the type of object's shape.

1 Introduction

The motivation for this work emerges from the problem of searching image databases [16]. The amount of data in image databases available on CD-ROMs or Internet has been increasing dramatically over the last few years which makes the search for an object in image databases an important problem. The difficulty stems from the fact that, unless images are indexed in a certain way (usually using a set of keywords), it is extremely difficult to describe, in an operational way, the object we are looking for. A way to alleviate the problem is to give a set of examples representing an object we are looking for and using machine learning techniques to induce rules for determining the objects automatically.

Let us suppose that we have a database of images represented only as a collection of bitmaps and we are interested in images which contain convex objects. If we had a rule which determines convex objects from all other possible objects we could effectively target images by performing segmentation on an image and applying the rule on the objects found by the segmentation procedure.

In this paper we are interested in the problem of automatic construction of rules for classifying objects found by the segmentation procedure. The setting adopted in the paper is as follows. The object produced by the segmentation procedure is described as a vector of parameters $(x_1, \ldots, x_D)$. Each vector belongs to one of the categories (classes). The rule for determining the object's class is a mapping from the set of objects into the set of categories. Humans construct and use rules like the above mentioned rule for determining the convexity of an objects or for determining the sex of a person from the person's photography every day. Unfortunately, the rules used and the underlying formalism used by humans is not known to us. However, a person can easily point out the category of a presented object. This manifestation of human's knowledge about the problem can be used for constructing the set of learning examples in a form of a collection of pairs (object's description, object's class). The automatic construction of rules uses this information for construction of operative rules for determining object's category.

The paper begins with a brief description of the learning from examples setting where the existing learning techniques are divided into two classes: discriminative and generative. It is argued that generative techniques are generally more suitable for the problem than discriminative ones. We focus primarily on real valued vectors as models for objects although the approach is easily extendable to all the models of objects where the distance between two object can be defined. Learning from examples can be formally presented by estimating the density which generated the examples. In Sect 3 the density estimation techniques are presented. A wide range of densities can be presented as a linear combination of Gaussian kernel densities. Sect 3.1 describes the algorithm for approximation of kernel densities from examples. The algorithm is applied, as described in Sect 4, to the problem of determining the convexity of a 2D object and the problem of determining the gross shape of 2D objects. The 2D shape of objects considered is represented by superellipses. For the first problem the analytical solution is easily found as opposed to the second problem of determining the gross shape of an object (for example rounded, squared or diamond shaped). In this second problem the catego-

ry of an object depends on several parameters of the shape model and furthermore, the category of some objects is doubtful for humans. The paper concludes with a discussion of open research issues associated with the recognition of the object's category from images.

2 Learning from Examples

In learning from examples setting we have a set of examples of a concept which is a set of pairs (object, category). A learning algorithm uses the set of examples to induce a set of rules. A rule is a mapping from a set of objects to the set of categories. The induced rules can be used to predict categories of unknown examples. In the paper we use the following variation of the learning problem: the training examples are represented by d-dimensional vectors $\{(x_1, \ldots, x_d)\}$, and where each vector has one of M class labels attached. The task is to design a system that classifies unlabeled vectors. In machine learning this is the *supervised* learning problem addressed by a variety of algorithms such as decision trees [10], rule based systems [2] or neural networks [9].

The classic measure of performance of such algorithms is how well they predict on average the unseen class labels–classification error is typically estimated on an independent test set of examples. As far as the classification accuracy is concerned it seems that the point of diminishing returns is being reached since the recent studies show that the best algorithm of each genre achieve very similar classification accuracy on standard domains (see [15], [8]). In learning to predict the class it is sufficient to learn how to *discriminate* between different classes. It is therefore the *differences* between classes that are important rather than the description of classes themselves. The decision trees and neural networks are two very popular models which belong to the category of discriminative algorithms. The differences in d-dimensional vector space are described by decision surface that partition vector space into regions associated with a single class. Consider the 2-D example depicted in Fig 1 taken from [13]. There are two class labels represented by A and B. Different algorithms induce different decision boundaries. In the classification of unseen instances the assumption that each region is homogeneous is used. The vector with an unknown category is classified in the category of its region. In the case of novel input, for instance in the top-right region of Fig 1 it is not obvious to which class it belongs. Discriminative models would assign the class label to it without noticing its own limitations. The alternative approach to direct discrimination is to learn the *generative* models for each of the classes. In generative models case the goodness of fit of the model w.r.t.

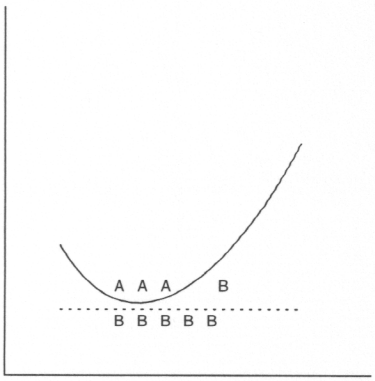

Figure 1: A simple 2-D classification problem

training examples is defined in terms of how vector components vary as a function of the classes as opposed to discriminative models where the opposite is true. An explicit model is defined which specifies how vectors are generated for each class. In this manner, classification boundaries are just an implicit by-product depending on the response of each class model at a particular vector. Moreover, the response to a vector in the top right region in Fig 1 should be low which is a correct generalisation based on learning examples from Fig 1. This situation is frequent in the problem of classifying objects. Generally, the user would provide the examples of the classes of his interest only. The segmentation procedure returns all the possible objects and consequently all possible classes in the image, the number of which is usually much greater than the number of classes provided by the user. The discriminative classifier by definition classifies in one of the provided classes. It was experimentally shown that the discriminative algorithms are not able to cope with the problem since they assign high probability to the objects which belong to the unspecified class (see [13], [5]).

Formally, an optimal classifier is one which, for any input x, returns

$$\arg\max_k \{p(\omega_k|x)\}, \tag{1}$$

i.e. for any description of an object it can calculate which class ω_k out of M possible classes generated the object. Note that an optimal classifier need not compute exact probabilities, but only assign the greatest probability to the correct class given the description of the object. Nonetheless, the probabilities are of importance if a classifier is only a part of an overall decision system. Bayes' rule can be used for the estimation of the posterior probability of the object's class ω_k given the description of the object:

$$p(\omega_k|x) = \frac{f^k(x)\, p(\omega_k)}{\sum_{i=1}^{M} f^i(x)\, p(\omega_i)} \tag{2}$$

where M denotes the number of classes in the training set. The density $f^k(x) = p(x|\omega_k)$ is defined such that $p(X < x) = \int_{-\infty}^{x} f^k(x)\,dx$ where X is the real-valued variable and x represents value of X (see [13]). The density is the generating model of the domain and describes completely how examples from each class are generated. The Bayes' rule and the generating model therefore enable classification even if the novel classes are present in the testing set of examples. There are two problems to be solved: how the densities are approximated from the training examples and after obtaining densities there is still a decision problem in terms of determining at what level one rejects data as being from a particular class. We will focus on the former problem in the rest of the paper.

3 Density Estimation Techniques

If we knew the density function that generated the data we would be able, by using Bayes' rule, to predict the class optimally in the sense of Eq 1. Since we can not assume any particular form of $f^k(x) = p(x|\omega_k)$ we rely on probability estimation techniques (see [12] and [4]). We focus on kernel-based methods which are based on the following: the local estimation of probability density around point a is obtained by placing a "bump" around a. The "bump" K is usually a Gaussian centered at a with fixed bandwidth h, which equals at point x

$$K\left(\frac{x-a}{h}\right)$$

The estimate of probability density which generated a set of points $\{a_i; i = 1, N\}$ is the normalised sum of kernels positioned at $\{a_i\}$

$$\hat{f}(x) = \frac{1}{Nh}\sum_{i=1}^{N} K\left(\frac{x-a_i}{h}\right) \qquad (3)$$

which makes $\hat{f}$ itself a density. For the multidimensional case, the product kernel is commonly used:

$$\hat{f}(x) = \frac{1}{Nh_1\cdots h_D}\sum_{i=1}^{N}\prod_{d=1}^{D} K\left(\frac{x^d - a_i^d}{h_d}\right)$$

where x^d denotes the component in dimension d of vector, and the h_d represents different fixed bandwidths in each dimension.

3.1 Centroid Kernel Classifiers

The kernel density estimation technique is not without disadvantages. For computation of the density estimate at point x all data points must be stored and a distance measure between x and all stored points must be calculated for each class prediction. An obvious simplification of the full kernel model of Eq 3 is to replace clusters of data points by appropriate representatives. This is called the *centroid kernel model*. The centroid kernel model is a *mixture density model* [7] where probability density is estimated by a linear combination of C mixture components, i.e.,

$$f(x) = \sum_{j=1}^{C} \alpha_j f_j(x) \qquad (4)$$

where the α_j are the mixing proportions. Note that the full kernel estimate is a special case of Eq 4. The parameters of the mixture density is estimated by posing the problem as a maximum-likelihood problem, i.e., we seek the parameters which maximize the probability that the model with those parameters generated the training data. The method employed for maximum likelihood estimation of parameters is well-known *Expectation minimisation*-EM method [11] for iteratively calculating model parameters given some initial estimates.

3.2 EM Procedure

Let us remind the reader that for the estimation of probability density only examples from one class are taken into consideration, i.e., each class density estimation is treated as a separate problem, resulting in M density approximations for M classes which are used in Bayes' formula for calculating the probability of each class given example.

The parameters to be calculated in the centroid kernel model are the mixing proportions α_j and the parameters defining each centroid θ_k; let

$$\Phi = \{\alpha_1, \ldots, \alpha_C, \theta_1, \ldots, \theta_C\}$$

denote the set of parameters of centroid kernel model with $C > 0$ components. Let further $\sum_{k=1}^{C} \alpha_k = 1$. Suppose that given set of examples $\{x_i\}$ can be described by a mixture model:

$$p(x_i|\Phi) = \sum_{j=1}^{C} \alpha_j\, p(x_i|\theta_j)$$

As already mentioned, the set of parameters of the mixture density is obtained by maximising the probability that the model with given parameters generated the data, i.e., if we assume the independence of the N training examples of a fixed class we are maximising

$$L(\Phi) = \prod_{i=1}^{N} p(x_i|\Phi) \qquad (5)$$

The direct maximisation of Eq 5 results in a set of non-linear simultaneous equations which have no direct analytical solutions. Hence the iterative techniques are typically employed which result in a local

optimum w.r.t. Eq 5. The EM algorithm [3] is an iterative procedure which produces a suboptimal solution by taking an initial guess at the parameter values of a mixture model and try to estimate which example belongs to which model component; in the next step the values of the parameters are estimated taking into account the results obtained in the first step. These two steps are repeated until the value of $L(\Phi)$ increases.

EM algorithm is a general procedure. We briefly describe the variation for the case when the function in the mixture of Eq 4 are Gaussians, following [11]. Each component of density mixture is defined as

$$p_k(x|\theta_k) = \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma_k|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)\right)$$

where μ_k is the centroid position and Σ_k is the $D \times D$ covariance matrix for the component. In the initialization of the EM procedure the initial value of Φ must be found. Centroid positions are set to C randomly choosen training examples positions. Clusters are formed by assigning each training example to its closest centroid. Covariance matrix for each component is estimated from the points which belong to its cluster. The algorithm proceeds by alternating E and M steps. The E step consists of estimating the component densities for the data. First we calculate the probability that the component k generated the example x:

$$\hat{p}(k|x) = \frac{\alpha_k\, \hat{p}_k(x|\hat{\mu}_k, \hat{\Sigma}_k)}{\sum_{j=1}^{C} \alpha_j\, \hat{p}_j(x|\hat{\mu}_j, \hat{\Sigma}_j)}.$$

Parameters of the centroid are then updated given this revised membership for each data point. The mean of a centroid is updated as

$$\mu_k = \frac{\sum_{i=1}^{N} \hat{p}(k|x_i)\, x_i}{\sum_{i=1}^{N} \hat{p}(k|x_i)}.$$

The position of k-th centroid is updated in the direction towards x weighted depending on how likely example x belongs to k-th centroid. Similarly, the weights are updated as

$$\alpha_k = \frac{\sum_{i=1}^{N} \hat{p}(k|x_i)}{\sum_{j=1}^{C}\sum_{i=1}^{N} \hat{p}(j|x_i)}.$$

The weights can be interpreted as a mass assigned to the component based on summing the memberships of examples to that component. The covariant matrix for a mixture density component is updated as

$$\Sigma_k = \frac{\sum_{j=1}^{N} (x_j-\mu_k)(x_j-\mu_k)^T\, \hat{p}(k|x_j)}{\sum_{j=1}^{N} \hat{p}(k|x_j)}.$$

The EM procedure stops when a local maximum of Eq 5 is reached. The procedure is computationally simple and straightforward to implement. There may be a problem, since in the case when the parameter set Φ is unconstrained $L(\Phi)$ may not be bounded. In our case this happens when $\sigma \to 0$. Hence, Φ must be constrained.

4 Application to Classifying Object's Shape

Suppose we are given a description of an object produced by a segmentation procedure. In our experiments we use the description of 2D objects in a form of superelipsoids. Superelipsoids [1] are suitable for describing a variety of forms by varying parameters a, b and ϵ that define a superelipsoid:

$$\left(\frac{x}{a}\right)^{\epsilon} + \left(\frac{y}{b}\right)^{\epsilon} = 1.$$

Methods for recovering superellipsoids or their 3D equivalents-superquadrics from images are well known ([14], [6]).

The first problem presented to the algorithm is to discriminate between *concave* and *convex* objects. The analytical solution to the problem is known: An object is convex iff $\epsilon < 2$. Despite its simplicity this problem is suitable for the testing purpose. The training data consists of vectors of the form (a, b, ϵ) and the corresponding class labels denoting the convexity/concavity of an object. We randomly generated 100 exemplars with $a, b \in [1, 5]$ and $\epsilon \in [0, 4]$. The generated set is split into training and testing sets. The algorithm uses the training set, which consists of 70% of instances for learning and the remaining 30% for testing. Split was performed 10 times and the results on testing sets were averaged. The algorithm correctly recognizes 97.61 % testing instances, i.e. on average makes only one error per run. The algorithm makes erroneous decisions in cases where ϵ is close to 2.

The next experiment is more interesting since there is no known analytical solution to it. Namely, we are interested in determining the type of object's shape: *rounded*, *square-shaped* or *diamond-shaped* (see Fig 2).

The shape of a randomly generated object (with a, b and ϵ distributed as in the previous experiment) was classified into one of the predetermined classes by the authors. There were cases where the classification was not clear since, by the authors' subjective opinions, certain objects could be classified into more than one class. Again, the data was split 10 times where 70% of instances were used for learning and the remaining 30% for testing. The classification of objects, performed by the algorithm was correct in 82.21% of cases.

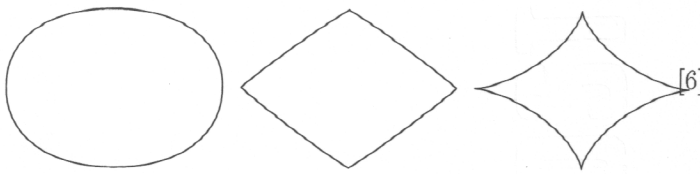

Figure 2: Types of object's shape: rounded, square-shaped and diamond-shaped

5 Conclusion and Further work

With the increase of the size of image databases its seems impossible to index the databases by hand. Automatic procedure for searching the databases is therefore needed. The problem can be tackled by machine learning techniques. Namely, models of an object we are looking for can be automatically induced from instances of object presented to a machine learning algorithm. This is only a preliminary research and leaves several open questions for further work. We assumed fairly simple a structure of the model produced by the recovery and segmentation procedure. It is not clear if this approach can cope with more elaborate structure description. It may be necessary to include a combination of a high-level description with kernel estimation techniques to cope with the problem. Another issue connected to the segmentation procedure is how our approach can cope with objects of different sizes which belong to the same class. We also assumed that the number of centroids in the model is fixed. It would be interesting to investigate the use of Bayesian techniques to aid in the comparison of models of different sizes.

References

[1] A. H. Barr. Superquadrics and angle-preserving transformations. *IEEE Computer Graphics and Applications*, 1(1):11–23, 1981.

[2] P. Clark and T. Niblett. The CN2 induction algorithm. *Machine Learning*, (3):261–285, 1987.

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *J. Royal Stat. Soc. Ser. B. (methodological)*, 39:1–38, 1977.

[4] A. J. Izenmann. Recent developments in nonparametric density estimation. *J. Amer. Stat. Assoc.*, 86:205–224, 1991.

[5] M. A. Kramer and J. A. Leonar. Diagnosis using backpropagation neural networks-analysis and criticism. *Computers in Chemical Engineering*, 14(12):1323–1338, 1990.

[6] A. Leonardis, F. Solina, and A. Macerl. A direct recovery of superquadrics models in range images using recover-and-select paradigm. In Jan-Olof Eklundh, editor, *Proceedings of Third European Conference on Computer Vision, Vol I*, pages 309–318, Stockholm, Sweden, May 1994. Springer.

[7] G. J. McLachlan and K. E. Basford. *Mixture Models: Inference and Applications to Clustering.* Marcel Dekker, New York, 1988.

[8] K. Ng and R. P. Lippman. A comparative study of the practical characteristics of neural network classifiers and conventional classifiers. In R.P. Lippman, J. Moody, and D.S. Touretzky, editors, *Advances in Neural Information Processing 3*, pages 970–976, Los Gatos, CA, 1991. Morgan Kaufmann.

[9] Y. H. Pao. *Adaptive pattern recognition and neural networks.* Addison Wesley, 1989.

[10] J. R. Quinlan. Induction of decision trees. *Machine Learning*, (1):81–106, 1986.

[11] R. A. Redner and H. F. Walker. Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26(2):195–239, 1948.

[12] B. Silvermann. *Density Estimation for Statistics and Data Analysis.* Chapman Hall, New York, 1986.

[13] P. Smyth and J. Mellstrom. Detecting novel classes with application to fault diagnosis. In D. Sleeman and P. Edwards, editors, *Machine Learning-Proceedings of the Ninth International Workshop (ML92)*, pages 416–425, San Mateo, CA, 1992. Morgan Kaufmann.

[14] F. Solina and R. Bajcsy. Recovery of parametric models from range images: The case for superquadrics with global deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(2):131–147, 1990.

[15] S. M. Weiss and I. Kapouleas. An empirical comparison of pattern recornition, neural nets, and machine learning classification methods. In *Proceedings of 11th IJCAI*, pages 781–787, Los Gatos, CA, 1989. Morgan Kaufmann.

[16] A. Yoshitaka, S. Kishida, M. Hirakawa, and T. Ichikawa. Knowledge assisted content-based retrieval for multimedia databases. *IEEE MultiMedia*, 1(4):12–21, Winter 1994.