# Learning Models for Object Recognition

Pedro F. Felzenszwalb
Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139
pff@ai.mit.edu

## Abstract

*We consider learning models for object recognition from examples. Our method is motivated by systems that use the Hausdorff distance as a shape comparison measure. Typically an object is represented in terms of a model shape. A new shape is classified as being an instance of the object when the Hausdorff distance between the model and the new shape is small. We show that such object concepts can be seen as halfspaces (linear threshold functions) in a transformed input space. This makes it possible to use a number of standard algorithms to learn object models from training examples. When a good model exists, we are guaranteed to find one that provides (with high probability) a recognition rule that is accurate. Our approach provides a measure which generalizes the Hausdorff distance in a number of interesting ways. To demonstrate our method we trained a system to detect people in images using a single shape model. The learning techniques can be extended to represent objects using multiple model shapes. In this way, we might be able to automatically learn a small set of canonical shapes that characterize the appearance of an object.*

## 1. Introduction

In this paper, we consider the problem of object recognition. Our main goal is to explore how classical recognition systems can benefit from learning techniques. In particular, we concentrate on recognizing objects using the Hausdorff distance as a shape comparison measure.

Comparing shapes using the Hausdorff distance has proven to be a reliable and efficient method for object recognition (see [8, 14, 7]). In this context, shapes are sets of points obtained from images using edge detection. Some examples are shown in Figure 1. Typically, an object is represented in terms of an ideal shape, which is referred to as the model. A new shape is classified as an instance of the object if the Hausdorff distance between the model and the new shape is small. We show that such object concepts can be seen as halfspaces (linear threshold functions) in a transformed input space. This makes it possible to use a number of standard algorithms to learn object models. While we concentrate on using the Hausdorff distance for shape comparison, our techniques can be modified to compare shapes using the Chamfer distance (see [1, 3]), or the generalization of the Hausdorff distance introduced by Olson in [12].

Most recognition systems simply use an instance of the object as the model. In that case, it might be necessary to manually edit the model to correct for noise and remove background clutter. We believe that an intelligent system should be able to learn from natural examples. Moreover, in some cases it is not clear what would be a good model for a particular object. By using learning techniques we can estimate the model using training examples alone. The training examples for a particular object are divided into a set of positive examples, and a set of negative examples. Figure 2 shows some of the examples we used to train a system to detect people. Note that the positive examples have large variation in shape, as they come from different people and each person can be in a number of different poses. Each positive example also has background clutter which should not be part of the model.

Our learning method only assumes that there exists some model which can be used for recognition. The model is estimated from the training examples. When a good model exists, we are guaranteed to find one that provides (with high probability) a recognition rule that is accurate. This notion is made precise using the PAC model of learnability, which we describe in Section 5. The learning approach generalizes the Hausdorff distance in a number of interesting ways. These generalizations are described in Sections 4 and 5. We demonstrate our method by training a system to detect people in images using a single shape model. In the last section, we discuss how our techniques can be extended to represent objects using a number of canonical shapes.

**Figure 1: Example of edge detection for shape extraction.**

## 2. Related Work

Shvaytser [16] considered learning visual concepts in binary images. Most of his results are negative, showing that many simple visual concepts are not learnable (in polynomial time) from positive examples alone. This includes concepts similar to the ones we consider. We show that interesting concepts can be learned by using both positive and negative examples.

Gavrila and Philomin [6] use a measure similar to the Hausdorff distance for shape comparison. Instead of learning a shape model from training examples, they simply store a large number of positive examples in memory (on the order of a thousand). A new shape is considered an instance of the target object if it is similar to any of the stored shapes. In contrast, we show that using learning techniques we can build a single shape model which captures the information from all examples. When a single shape model is not good enough, it might be possible to learn a small set of canonical shapes, as discussed in Section 8.

Jacobs, Weinshall and Gdalyahu [9] considered the general problem of characterizing the appearance of an object by storing a number of example shapes (or images). They were interested in methods that can select a small number of examples which characterize an object well. It was pointed out that standard clustering techniques don't perform well when using robust shape comparison measures such as the partial Hausdorff distance. Our techniques provide a principled way to generate canonical shapes that characterize the appearance of an object.

Most recent learning methods in computer vision are not based on classical recognition techniques. For example, Papageorgiou and Poggio [13], Schneiderman and Kanade [15], and Viola and Jones [17] developed systems that yield good recognition accuracy. These methods introduced new representations for visual concepts. In this paper we concentrate on learning concepts that are represented by two-dimensional shapes. In particular, our learning method is motivated by systems that use the Hausdorff distance and similar measures for recognition.

## 3. Hausdorff Distance

We will only consider the directed Hausdorff distance, which is defined as follows. Let $A$ and $B$ be two finite point sets. The distance from $A$ to $B$ is given by

$$h(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\|. \quad (1)$$

This measures the maximum distance from a point in $A$ to its nearest point in $B$.

The main strength of the Hausdorff distance is that it does not require that points in $A$ and $B$ exactly match. In particular, moving the points by a small amount only changes the Hausdorff distance by a small amount. This is extremely important for recognizing objects based on their shapes. Locations in an image are quantized, introducing positional errors. Moreover, most objects don't have fixed shapes. The Hausdorff distance can tolerate some shape variation.

One of the problems with the classical Hausdorff distance is that it is not robust with respect to noise or outliers. Sometimes an instance of an object is not fully visible in an image. This can happen because of noise in the imaging process, errors during edge detection,

or some part of the object might be occluded. To deal with this problem, the classical distance is modified by replacing the maximum value in equation (1) with the $K$th ranked value for some parameter $K$. The partial Hausdorff distance from $A$ to $B$ is defined as

$$h_K(A,B) = \underset{a\in A}{K^{th}} \min_{b\in B} \|a-b\|.$$

For example, by taking $K = |A|/2$, the partial distance reflects the median distance from points in $A$ to their closest points in $B$.

# 4. Object Recognition

We can extract shape information from an image using edge detection (see [4]). Some examples are shown in Figure 1. In this domain, the problem of recognition is to decide if a shape $B$ is an instance of a particular object. Suppose we have an ideal shape $A$ which is a model for the target object. We can say that $B$ is an instance of the object if the partial Hausdorff distance from $A$ to $B$ is at most some threshold $d$. One problem with this definition is that cluttered areas in an image will be recognized as instances of the object. This is because if $B$ is very dense, there will be a point in $B$ near every point of $A$. In practice, this problem can be solved by checking not only that $h_K(A,B) \leq d$, but also that $h_L(B,A) \leq d$. The distance from $B$ to $A$ is called the reverse distance.

In our framework, we start with decisions based only on $h_K(A,B)$. We will show that such decisions can be seen as linear threshold functions in a transformed input space (here the input is the shape $B$). The linear threshold functions that are equivalent to decisions of the form $h_K(A,B) \leq d$ have coefficients that are 0 or 1. By allowing for arbitrary coefficients we generalize the partial Hausdorff distance in two interesting ways. First, the measure can give different importance to different parts of a shape. Moreover, negative coefficients allow the measure to reject cluttered image areas, by saying that $B$ should not have points at some locations. This makes it unecessary to use the reverse distance as described above.

Note that $h_K(A,B) \leq d$ holds exactly when at least $K$ points from $A$ are at distance at most $d$ from some point in $B$. It is common to interpret this decision function using the notion of dilation. The dilation of $B$ by $r$ is denoted $B^r$, and it consists of the set of points that are at distance at most $r$ from some point in $B$. Using the concept of dilation, $h_K(A,B) \leq d$ holds when at least $K$ points from $A$ are contained in $B^d$. Now remember that $A$ and $B$ correspond to points from images. We can represent these sets as $n$ dimensional vectors, where $n$ is the number of pixels in the images. Let $\mathbf{A}$ be the vector representation of $A$. $\mathbf{A}$ is a $0,1$ vector, where the $i$-th dimension indicates if the $i$-th pixel in the image is in $A$. Using the vector representation of our sets, the number of points from $A$ contained in $B^d$ is exactly the dot product $\mathbf{A} \cdot \mathbf{B^d}$. So we can represent our decision function based on the Hausdorff distance as a linear threshold function,

$$h_K(A,B) \leq d \quad \Leftrightarrow \quad \mathbf{A} \cdot \mathbf{B^d} \geq K. \tag{2}$$

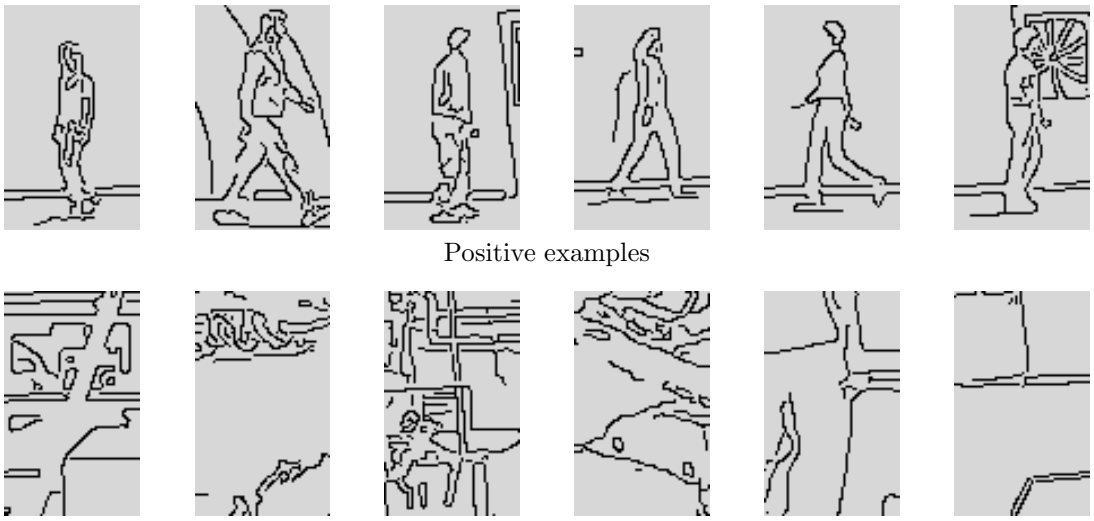
Note that the input to the linear threshold function is $\mathbf{B^d}$, and not $\mathbf{B}$. As mentioned above, decisions based on the partial Hausdorff distance correspond to the linear threshold functions with $0,1$ coefficients. By allowing $\mathbf{A}$ to be an arbitrary vector, we obtain a generalization of the partial Hausdorff distance. In the classical distance, every part of the model shape is equally important. In the linear threshold function, a large value in $\mathbf{A}$ might indicate a very discriminating feature. Also, negative values in $\mathbf{A}$ indicate that instances of the target object shouldn't be dense everywhere. In the next section we will obtain yet another generalization of the partial Hausdorff distance as a consequence of our learning techniques.

The importance of the relation in equation (2) is the connection between a robust measure of shape similarity and simple concepts defined by linear threshold functions. Historically, linear threshold functions or halfspace concepts were considered too simple to yield good classification systems (see [9]). We have shown that we can represent any concept defined by the partial Hausdorff distance as a halfspace concept, by transforming the input space in an appropriate manner. Intuitively, dilating the input shapes yields a more robust representation. For example, let $B$ be obtained by perturbing the points in $A$. The dot product of $\mathbf{A}$ and $\mathbf{B}$ can be arbitrary. On the other hand, the dot product of $\mathbf{A}$ and $\mathbf{B^d}$ reflects the similarity between the two shapes (as long as $d$ is large enough).

# 5. PAC Learning

We say that an algorithm learns concepts from labeled examples if it can find (with high probability) recognition rules that are accurate. The PAC learning model makes this intuition precise. We give a brief description of the model here, see [10] for an in depth introduction.

Let $X$ be the space of all possible examples. We assume that examples are drawn at random from some unknown distribution $\mathcal{D}$. Each example is labeled as positive or negative, according to a concept $c \in \mathcal{C}$, where $\mathcal{C}$ is a class of possible concepts. The learning



Positive examples

Negative examples

**Figure 2: Some of the random examples used to train the people detection system.**

problem is to find a hypothesis $h$ which agrees with $c$ on most examples. The error between $h$ and $c$ is defined as

$$error(h) = \Pr_{x \in \mathcal{D}}[c(x) \neq h(x)].$$

A learning algorithm has access to a set of random examples that are labeled according to $c$ and should output a hypothesis with small error. We say that $\mathcal{C}$ is PAC learnable if for every concept $c \in \mathcal{C}$ and distribution $\mathcal{D}$ over $X$ we can find a hypothesis $h$ that with probability at least $1-\delta$ has error at most $\epsilon$. This captures the idea that $h$ is *probably approximately correct* (PAC). The number of labeled examples necessary to compute $h$ should not be too large, but it is allowed to grow as $\epsilon$ and $\delta$ get small.

The two parameters $\epsilon$ and $\delta$ correspond to two types of failures which can occur. The error parameter $\epsilon$ is necessary because sometimes two concepts $c_1$ and $c_2$ differ only at examples which are very rare according to $\mathcal{D}$. In this case, the random labeled examples might not give any clue as to which of the two is the correct concept. The confidence parameter $\delta$ is necessary because occasionally the learning algorithm might be unlucky and get unrepresentative examples. For example, there is a small chance that all random examples are equal. In that case, the learning algorithm will have very little information about the target concept.

As described in the last section, we are interested in learning concepts of the form

$$c(B) = \begin{cases} \text{positive} & \text{if } \mathbf{A} \cdot \mathbf{B^d} \geq K \\ \text{negative} & \text{otherwise} \end{cases} \qquad (3)$$

For now, assume that $d$ is known. In that case, we can see $\mathbf{B^d}$ as the input to our concepts. This makes the concepts be linear threshold functions. Linear threshold functions split the input space with a hyperplane. Instances on one side of the hyperplane are labeled as negative, and instances on the other side are labeled positive. The problem of learning halfspaces is one of the most studied topics in learning theory. For example, Linear Programming can be used to learn linear threshold functions in the PAC sense (see [2]). Namely, by using a large enough number of training examples, we can obtain a hypothesis that has error at most $\epsilon$ with confidence $1-\delta$. The number of examples that are necessary grows linearly in $n$ (the dimension of the input space) and polynomial in $1/\epsilon$ and $1/\delta$. In practice we use the perceptron algorithm (see [11]). The perceptron algorithm has many nice properties. It is conceptually simple and easy to implement. The algorithm is very fast and uses little memory. Moreover, the perceptron algorithm can tolerate noise, and learns a good hypothesis even when there is no concept that classifies all examples correctly (see [5]).

If we don't know $d$ in advance we must learn richer concepts. In general, we consider a small set of possible dilation parameters $d_1, \ldots, d_m$. For example, we can assume that $d$ is not very large and only consider positive integer values. Picking $d_i$ can be seen as a problem of model selection. A standard technique to solve this problem is to use cross-validation. In this context, our original set of training examples is divided into two sets, corresponding to a training set and a validation set. We learn a hypothesis for each possible $d_i$ using

the training set, and measure its performance using the validation set. The final hypothesis is the one which performs best on the validation set.

Another way to allow for an arbitrary dilation parameter yields a nice generalization of our concepts. Define a new input space, given by the concatenation of the $\mathbf{B^{d_i}}$. The new inputs live in a $nm$ dimensional space, where $n$ is the number of pixels in the original input images, and $m$ is the number of dilation parameters being considered. To be precise, let

$$T(B) = [\mathbf{B^{d_1}} \cdots \mathbf{B^{d_m}}].$$

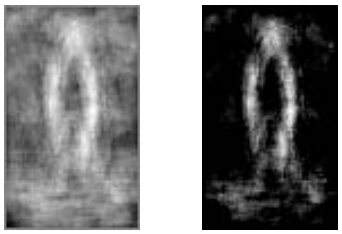
Our new concepts are linear threshold functions of $T(B)$. The coefficients of these functions can be written as $[\mathbf{A_1} \cdots \mathbf{A_m}]$. Note that we can represent any concept of the form in equation (3) if $d = d_i$ for some $i$. Simply let $\mathbf{A_j} = 0$ for all $j \neq i$ and $\mathbf{A_i} = \mathbf{A}$. The threshold remains the same.

The linear threshold functions of $T(B)$ generalize our old concepts in an intuitive way. The new concepts can capture the idea that some features of the model are better localized than others. A positive value in $\mathbf{A_i}$ indicates that there should be a feature in a particular area of $B$. The area is a disc with radius $d_i$. Since there are coefficients corresponding to different dilation parameters, we can represent shapes that have both rigid and deformable parts. The rigid parts are represented by coefficients in $\mathbf{A_i}$, where $d_i$ is small, while the deformable parts are represented by coefficients in $\mathbf{A_j}$, where $d_j$ is large.

# 6. Object Detection

The problem of object detection is to locate all instances of an object in an image. Once we have learned a model to recognize the target object we can use it to solve the object detection problem. For each possible location in an image, we check if the image patch around it is an instance of the target object. Note that there are many locations in an image, a typical image has around $10^4$ locations. Most locations don't contain the target object, so the distribution of examples seen by the recognition system is very skewed. To correctly classify all locations in an image the recognition system must have a very small false positive rate.

In the PAC framework, the error of a hypothesis is measured with respect to an arbitrary distribution $\mathcal{D}$ over the examples. We only assume that the training examples come from the same distribution as the test examples. If we want to train models for object detection we should use many more negative examples than positive ones. This ensures that we will get a small number of false positive detections overall.



(a) (b)

**Figure 3: Illustration of a model. Bright pixels correspond to large coefficients, and dark pixels correspond to small coefficients. (a) shows all coefficients of the linear threshold function and (b) shows only the positive coefficients.**

| | Examples | Mistakes |
|---|---|---|
| Positive set | 105 | 3 |
| Negative set | 69870 | 0 |

**Table 1: Performance of a typical classifier on the test data. Mistakes in the positive set correspond to false negatives, and mistakes in the negative set correspond to false positives.**

# 7. Experimental Results

We applied our learning techniques to build a system that can detect people walking in our lab. Positive examples were obtained by capturing several people walking in different directions. We have images from seven people walking in two different directions. The total number of positive examples is 529. The negative examples are all possible image patches from different images which do not contain people. These were images from different enviroments. The total number of negative examples is 349.344. Some of the examples are shown in Figure 2.

We learned concepts of the form in equation (3). The value for the dilation parameter $d$ was automatically selected using cross-validation, as described in Section 5. We considered five different possible dilation parameters, corresponding to a dilation of 1,2,3,4 or 5 pixels. To evaluate our techniques we performed multiple training trials. In each trial 80% of the examples were used as training data (selected at random), and the remaining 20% as test data. One of the learned models is shown in Figure 3. The model corresponds to the coefficients of the linear threshold function. A dilation parameter of $d = 3$ was selected (automatically) in each trial. There was very little variaton in performance over the different trials. Table 1 displays the performance of a typical hypothesis on the test data.

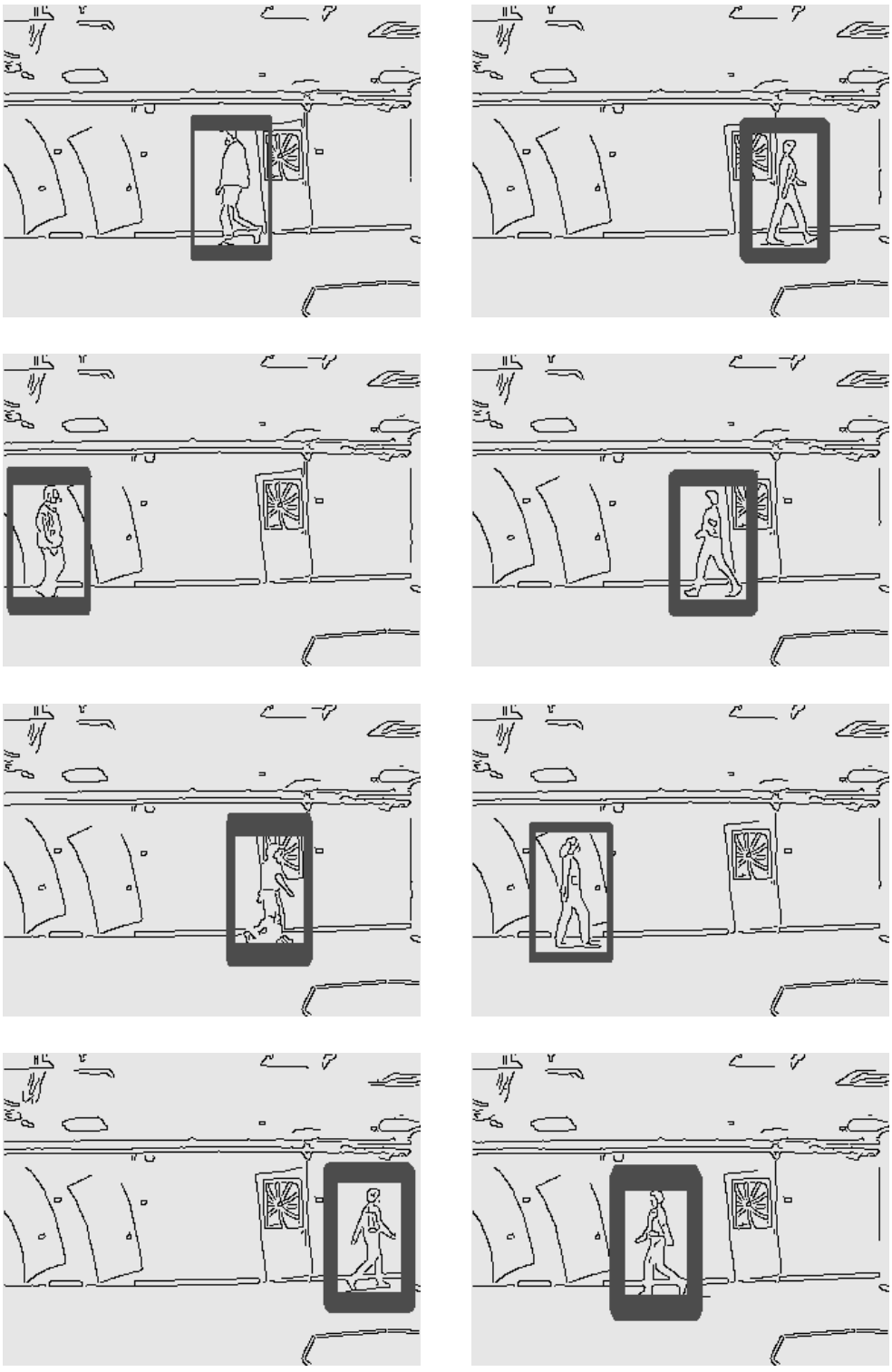


**Figure 4: Detection results.**

The system always had less than 5% false negatives and no false positives.

As described in Section 6, we can use the learned models to detect objects in new images. Some detection results are shown in Figure 4. There is a box around each detected object. Note that objects are detected multiple times across a small set of translations. This makes sense since the training images were not aligned perfectly. We could easily use some heuristic to eliminate the multiple detections. For example, we could select the center position for each cluster of detected locations.

# 8. Multiple Views

In this section we describe a direction for future work. The techniques we have presented so far assume that instances of an object are well described by a two-dimensional shape. This is only true for a limited set of objects. For example, the shape of an arbitrary three-dimensional object can vary due to viewing position. One way to allow for shape variation is to represent objects using a number of canonical views.

We can generalize our framework to represent objects using more than one model shape. Now we say that a new shape is an instance of a target object if it is similar to any of the canonical views of that object. Remember that in our framework, the set of shapes that are similar to a particular model form a halfspace in a transformed input space. Using this idea, the set of shapes which are similar to at least one of the canonical views of an object form a union of halfspaces in the transformed input space. Each halfspace corresponds to one of the canonical views. So the problem of learning a set of canonical views to represent an object can be seen as learning a union of halfspaces.

# References


[1] H. Barrow, J. Tenenbaum, R. Bolles, and H. Wolf. Parametric correspondence and chamfer matching: Two new techniques for image matching. In *IJCAI*, pages 659–663, 1977.

[2] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *J. ACM*, 36(4):929–965, 1989.

[3] G. Borgefors. Hierarchical chamfer matching: A parametric edge matching algorithm. *PAMI*, 10(6):849–865, November 1988.

[4] J. Canny. A computational approach to edge detection. *PAMI*, 8(6):679–698, November 1986.

[5] Y. Freund and R. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296, 1999.

[6] D. Gavrila and V. Philomin. Real-time object detection for smart vehicles. In *ICCV*, pages 87–93, 1999.

[7] D. Huttenlocher. Monte carlo comparison of distance transform based matching measures. In *DARPA*, pages 1179–1184, 1997.

[8] D. Huttenlocher, G. Klanderman, and W. Rucklidge. Comparing images using the hausdorff distance. *PAMI*, 15(9):850–863, September 1993.

[9] D. Jacobs, D. Weinshall, and Y. Gdalyahu. Classification with nonmetric distances: Image retrieval and class representation. *PAMI*, 22(6):583–600, June 2000.

[10] M. Kearns and U. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, 1994.

[11] M. Minsky and S. Papert. *Perceptrons*. MIT Press, 1969.

[12] C. Olson. A probabilistic formulation for hausdorff matching. In *CVPR*, pages 150–156, 1998.

[13] C. Papageorgiou and T. Poggio. A trainable system for object detection. *IJCV*, 38(1):15–33, June 2000.

[14] W. Rucklidge. *Efficient Visual Recognition Using the Hausdorff Distance*. Springer-Verlag, 1996. LNCS 1173.

[15] H. Schneiderman and T. Kanade. A statistical method for 3d object detection applied to faces and cars. In *CVPR*, pages I:746–751, 2000.

[16] H. Shvaytser. Learnable and nonlearnable visual concepts. *PAMI*, 12(5):459–466, May 1990.

[17] P. Viola and M. Jones. Robust real-time object detection. Technical Report 2001/01, Compaq CRL, February 2001.