SearchThisObject: Content Based Image Query Refinement Using Neural Networks

Final Project Report

CPSC 636 – Neural Networks Spring 2010

|  |  |
| --- | --- |
| Folami Alamudun  Department of Computer Science  Texas A&M University  [fola.alamudun@neo.tamu.edu](mailto:fola.alamudun@neo.tamu.edu) | Krishna Ganesula  Department of Computer Science  Texas A&M University  [krisgane@neo.tamu.edu](mailto:krisgane@neo.tamu.edu) |
|  | |

# ABSTRACT

Online Image Searches are based on metadata tags but these tags yield unsatisfactory results majority of the time. A plausible solution is to search the underlying content of the image rather than the metadata, which is primary motivation for Content Based Image Retrieval (CBIR) engines. In this work we propose a CBIR approach that uses information extracted from objects in an image to detect and remove irrelevant results. Image results from a standard metadata based image search engine are preprocessed through segmentation and noise removal techniques. Specialized geometric features for object detection are extracted from each of these images to be used as input to a classifier. The approach has been evaluated for several query objects using both unsupervised technique such as self organizing maps and supervised techniques such as radial basis networks and multi layer perceptrons.

## Author Keywords

Segmentation, Edge Detection, Hough Transform, Multi Layer Perceptrons, Radial Basis Functions, Image Search, Image Retreival.

# INTRODUCTION

Contemporary image search engines such as *Google* and *Yahoo!* depend only on metadata tags associated with the content; this applies to textual and multimedia content such as videos, images and voice content. The associated metadata tags for an image include information such as the title, link address, or the text in the container page. In short, the search makes the assumption that the adjoining text is always correlated with the image and this is not always the case. As a result, there are several cases in which a search query returns several irrelevant images.

Several CBIR engines are being actively researched to address particular issue. However most of these engines use image content like color and texture to characterize image similarity. These variations cannot be ensured to characterize the underlying objects in an image. This work is strongly motivated by the idea that the actual information need of a user making the query is the object itself; hence the presence or absence of the object(s) in an image is the absolute scoring metric for this approach.

# An overview of our approach

The primary goal was to identify and remove poor results from an image search executed by a standard image search query engine. It is critical for this approach that the image query is associated with a real object and not an abstract idea. The images retrieved by the search engine undergo the following sequence of processing steps -

*Preprocessing* – This is to extract the most significant object from an image. It mainly consists of steps like segmentation, background removal, connected component extraction and cropping.

*Feature Extraction* – Shape based features are extracted from this image using Hough transforms and corner detection methods.

*Similarity Analysis* – In an unsupervised learning method this would mean clustering the images to find similar significant objects. For a supervised technique, this would mean include an additional step of relevance feedback to train the classifier.

*Search Refinement* – Images which are not found to be similar are excluded from the initial set of results and provided to the user.

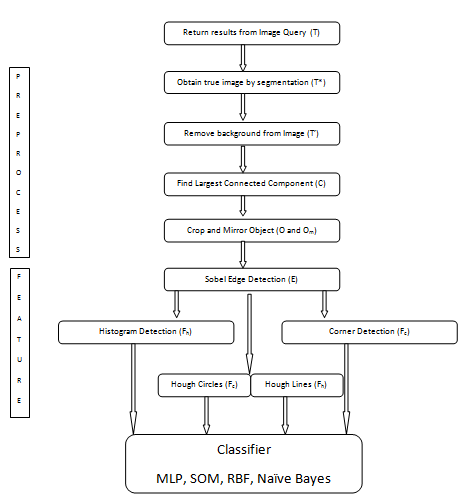


Fig 2.1 Overview of this approach

The steps in Fig 2.1 are explained in detail in the subsequent sections. Section 1 explains various segmentation steps applied. Section 2 gives an overview of the types of features used for classification. Section 4 lists the various classification and clustering methods used. Section 4 provides a summary of the results of this experiment.

# preprocessing

The image preprocessing techniques we have used are heavily customized for detecting the most prominent object in the image. Firstly all image inputs are scaled to a normalized size of 1000x800 pixels and then subjected to certain preprocessing steps. Each image obtained from the preprocessing procedure essentially an individual object that characterizes the image. The preprocessing pipeline includes multiple steps namely segmentation, background removal, connection component extraction and cropping.

## Segmentation

Image segmentation is the widely accepted method to separate the image in to regions of coherent properties in an attempt to detect objects and their components without the knowledge of the object model. Here we implement the statistical region merging algorithm proposed by Nock et.al [ ].

The objective of this algorithm is to produce a theoretical true image which could have generated the current image. The algorithm uses the individual pixels to separate it into multiple regions with statistical significance. Based on the value of neighboring pixels it is decided whether or not adjacent regions should be merged. If a certain region shows variation in its constituent pixels which is unlikely to be derived from a single Gaussian distribution with ascertained parameters (mean and variance) this region is split into multiple regions. This process is repeated until there are no further regions to split or merge. The set of regions obtained at the end of this region are potential object blobs because they are generated from a singular Gaussian distribution.

## Background Removal

A visual observation of the segmented image revealed that there were a large number of small segments which actually constituted a background but not real objects. Hence it is critical to filter this out as noise and irrelevant subjects of the image in terms of objects.

wever it is not trivial to identify segments of the image which actually belong to the background of the image. An ad hoc method was used for this task, which is based on an assumption that the background is the most prominent color at the four corners of an image. Hence we find the majority color in all four sides of the image frame and assume that it represents the true background color. All the regions which have this colour can now be removed from the segmented image.

## Connected Component Extraction

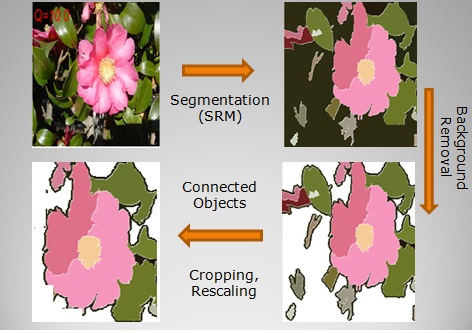
Once the background segments are removed the image is separated into small blobs which are sparsely connected. It could be possible that every one of these blobs specify an individual object. However it is more likely that we are looking for a complex object which is a combination of several such blobs.

To combine the relevant blobs we check the common boundary across adjacent blobs and merge them into a single object if this boundary is lower than the threshold. It might appear that this is a roundabout series of steps – we separate the image into individual blobs during segmentation and merge them back in connected component extraction. But it is significant to note that we have background removal in the middle which considerably reduces the number of connected components, thereby leaving us with very large blobs to analyze. Also at this point we shall pick the most prominent connected blob from the image and consider it to be the representative object for this image.

## Cropping and Mirroring

Having obtained the blob of interest we extract the pixels from the original image and retain its pixel values. Also the central point for the object is identified and the image is translated to center on it. As a result the rest of the image (which is blanked out) moves to the periphery. Now the image is scaled up to normalize the constituent object to the original image size.

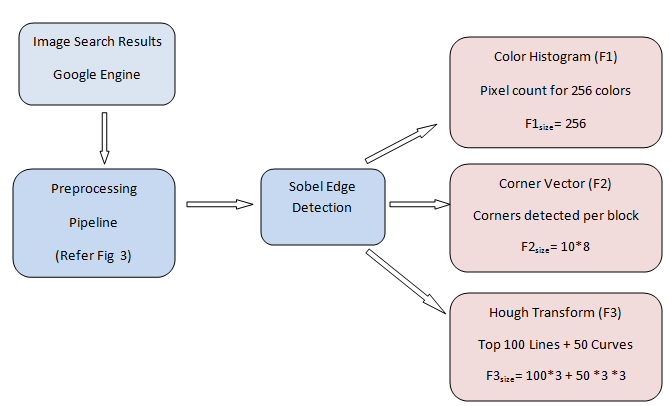
As an additional step we create a new object by swapping the x coordinates to simulate a mirrored object of the exact same size and characteristics. It was observed in several image results for the same object that images were the extremely similar but mirrors of each other (for instance, a car facing with its tail end pointing to the left and a an almost identical structured car with its tail pointing to the right of the image). To ensure that the classifier is invariant to such mirroring effects, we use both objects (original and mirrored).



## Fig 3. Complete Preprocessing Pipeline

# FEATURE EXTRACTION

On completing the preprocessing we have the underlying object isolated and extracted from the image. At this point we need to generate a feature set from this object that can be fed to a classifier. Any feature should satisfy at least two requirements – it should be concise, yet largely lossless representation of the original object and it should provide information characterizing the shape of an object.



## Fig 4.1. Feature Extraction

## Edge Detection

Edge Detection filters are used to detect points in a given image which have discontinuities and sharp changes in the brightness. In our case identifying object edges is critical to assign it a shape model. The edges always provide precise and compact information regarding the structure of this image.

We implement the Sobel operator that uses a pair of 3x3 convolution masks, one each to estimate the gradient in rows and columns. The convolution mask is slid over the image, manipulating a square of pixels at a time. The points with the highest estimated gradient are treated as edge pixels.

Although edge detection doesn’t directly provide us an actual feature set, its use is crucial to modify the original bitmap. Computationally it is much easier to analyze or manipulate an edge image rather than the fully populated object.

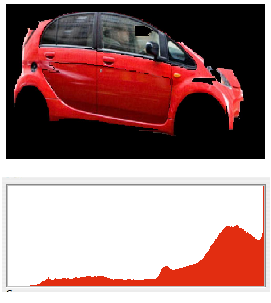


## Fig 4.2. Edge Detection

## Histogram Extraction

It is important to note that an attempt was made to use all the pixels (1000x800) in the edge image as input without extracting specific features. But this was abandoned due to the high dimensionality and the lack of attributes characterizing the actual shape of the object.

In particular to reduce the dimensionality of the input bitmap the histogram method was used. A color histogram was computed of with dimension of 256 representing multiple color values between white and black. The values represent the number of pixels containing this pixel value in the image. The original object image was used to compute the histogram as the edge image caused loss of a majority of colored pixels.



## Fig 4.3 Color Histogram

## Corner Detection

Points of Interest are closely correlated to object shapes, which is critical for modeling objects as specific pattern.



Fig 4.4 Corner Extraction

Here we use the Harris Corner Detector Technique []. This technique utilizes vertical and horizontal edges created by the Sobel Operator. Those edges are then blurred to reduce the effect of external noise. The resulting edges are then combined together to form an energy map that contains peaks and valleys. The peaks indicate the presence of a corner.

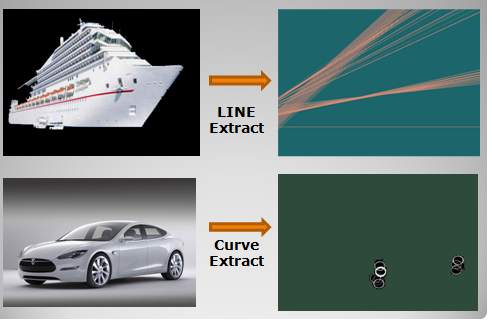
Unfortunately the number of corners varied largely between different kinds of images. To overcome this problem, the top 100 corners were chosen from every image. For each coordinate (x,y) the input dimension thus is 200 based on this feature vector.

## Hough Transform

Hough transforms are a strong motivation to opt for geometrically shaped objects in the image. They can be used to detect lines and curves in an image which provides ample information about the shape of the object. Noise in the image data or the edge detector may lead to missing points or pixels on the desired curves as well as spatial deviations between the ideal line/circle/ellipse and the noisy edge points as they are obtained from the edge detector. For these reasons, it is often non-trivial to group the extracted edge features to an appropriate set of lines, circles or ellipses. The purpose of the Hough transform is to address this problem by making it possible to perform groupings of edge points into object candidates by performing an explicit voting procedure over a set of parameterized image objects

## For Lines

We fit in 100 lines for a single image. As a line is characterized a point and the slope, we use three parameters per a line. Thus the feature vector is of length 300.



## Fig 4.5 Hough Transform

## For Circles

Hough transforms have a limitation in identifying circles. We need to specify the radius of the circle to detect it in an image. Thus we run Hough transforms with multiple radii and get results for r=10,20,30. Even here a circle can be categorized by the center (x,y) and its radius which can be fed to the classifier. The dimension for the top 50 circles of each radius together is 50x3x3 = 150.

# SIMILARITY ANALYSIS

## Unsupervised Learning

Self organizing feature maps (SOMs) also known as Kohonen maps is a type of artificial neural network trained using unsupervised learning. They are useful for [visualizing](http://en.wikipedia.org/wiki/Scientific_visualization) low-dimensional views of high-dimensional data, similar to [multidimensional scaling](http://en.wikipedia.org/wiki/Multidimensional_scaling).

We employ self organizing maps as an alternative approach to supervised learning by clustering the dataset. SOMs use a neighborhood function to preserve the [topological](http://en.wikipedia.org/wiki/Topology) properties of the input space. It this way, we expect that similar images generated by the feature extraction methods will be grouped in the same cluster.

## Supervised Learning

A supervised training method was adopted to improve the performance of the system by using feedback from the user to determine sample class labels during training. This was achieved by providing the user with a set of initial results (20 images) based on the input query. The images selected by the user are labeled as positive examples while rejected images are labeled as negative examples. These samples are used to train the classifier. The supervised classifiers used for this project are described below.

## Naïve Bayes

Naïve Bayes is a probabilistic classifier that uses Bayesian statistical properties of a training data set to predict the classification of an unseen example. A crucial assumption made by the Naïve Bayes classifier is that the statistical properties of observed attributes are independent of one another. This assumption makes the calculation of statistical properties more tractable but will evidently prove deficient in datasets where this assumption does not hold.

We expect that this assumption holds because all features generated using the feature extraction methods used in this project are independent of each other.

The calculation of Bayesian statistics for the training data set is a key component of the algorithm. For this computation, each unseen example is assigned the target class with the maximum conditional probability (equation [5.1]). This formula gives the computation of the prior probability for a given example

**(5.1)**

*where   
C(x) = Class of x  
Cj = jth possible* class value  
j = number of possible class values *x = unseen sample feature vector  
a = feature / attribute value   
N = number of features*

## Multi Layer Perceptron

A multi layer network of interconnected simple processors (perceptrons), an artificial neural network, is a learning algorithm that can be used to solve a wide range of complex computational problems. The artificial neural network model is widely used across multiple disciplines as a tool for solving a variety of problems including pattern classification, clustering and Categorization, function approximation and a host of other applications.

We use a 5 layer network each consisting of 100 perceptrons as a classifier. The size of the input layer used during experimentation was dependent on the feature vector generated by the feature extraction methods.

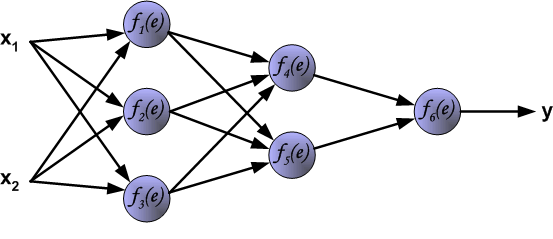


Fig 5.1 Multilayer Perceptron

Figure 5.1 above shows a basic architecture for a multilayer neural network. Our customized network contains 2 hidden layers with 100 nodes each and one output node

## Radial Basis Function Networks

Radial Basis function networks are faster than Multilayer perceptrons and have been used widely in CBIR solutions. The centers are chosen based on the relevance of objects to create a matching object model for the solution.

## Ensemble Averaging

Bootstrap aggregating meta-algorithm is used in machine learning to improve the accuracy and stability of regression and classification algorithms. Bootstrapping is a computationally intensive statistical inference tool used for re-sampling by constructing a number of [re-samples](http://en.wikipedia.org/wiki/Resampling_%28statistics%29) of the original dataset (and of a size less than to the original dataset), each of which is obtained by [random sampling with replacement](http://en.wikipedia.org/wiki/Random_sampling_with_replacement) from the original dataset.

We employ bootstrapping to improve the performance of the unsupervised learning methods described in section 5.2. The results obtained from bootstrapping are presented in the experimental results section of this paper.

# RESULTS

To begin with the features were fed to an unsupervised SOM as unlabelled inputs. The goal was to visualize the results obtained and investigate what cluster the relevant object was clustered into. Due to the presence of a common object across multiple images it was expected that the relevant images would form a majority cluster which could subsequently be labeled as relevant images. However it was found that the majority clusters were not always pertaining to the critical object. It was also found that similar objects were actually grouped into multiple clusters. This was disappointing and suggested that the variance in object structures, sizes and orientation could be the inherent cause.

The solution was to have a relevance feedback mechanism where a user is asked to select 10 relevant results from the top 20 and these are used to classify the rest of the 500 test samples. The assumption is that the user shall ascertain the object invariance by classifying different orientations as the same object. For this experiment out of the top 20 images, images which were contained the object were handpicked and used as positive training examples and the rest were used as negative training examples. Having designed a training and test set it was necessary to have a unique metric to qualify the performance of the approach which is explained in the next section.

## Quality Metric - Goodness

Applying the usual method of error classification to this problem would be largely ineffective. Our goal is to discard irrelevant images from the final set. To ensure that the filtering process is efficient we introduce a novel metric of Goodness of our approach. It is defined as the ratio of the proportion of correctly classified negative examples to the incorrectly classified positive examples. This tells us how many filtered out images are actually irrelevant and worth elimination. As the number of positive samples and the negative samples are the same,

Goodness (µ) =

*where*

*TN is the true negatives in the confusion matrix*

*TP is the number of true positives*

*FP is the number of false positives*

*FN is the number of true negatives.*

A quick look at the above expression tells us that greater the value of µ, more efficient the method. A value of 1 or less is essentially low quality because it could mean that we are losing a large number of positive samples while eliminating the negative set of results. The following subsections give an overview of the performance in various cases.

## Varying the classifier

The performance varied drastically across various classifiers. Naïve Bayes was used as a statistical reference point to provide us an idea about the efficiency of a classifier. The major neural networks used were Multi Layer Perceptrons(MLP) and Radial Basis Function Networks (RBF) which are widely chosen for CBIR tasks[ ]. Fig. provides an overview of the results based on the average Goodness, maximum and minimum Goodness measures for each of the classifiers.

Fig 6.1 Goodness values for various classifiers

Clearly Naïve Bayes lags far behind the neural network approaches in this case. It is interesting to observe that MLPs outperform the RBFs in certain best case scenarios, they are not always reliable. The choice of classifier thus is dependent upon the kind of task and whether the user is looking for maximum performance or reliability. Also another factor to be considered while implementing these techniques in practice is efficiency in terms of time.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Naïve Bayes | MLP | RBF |
| Train Delay (s) | 3 | 100.8 | 12.2 |
| Test Delay (s) | 2 | 180.2 | 8.4 |

Table 6.1 Time Duration for Classifiers

As shown in Fig. MLPs take up much longer than both RBFs and Naïve Bayes approaches. This has been a consistent limitation in applying MLPs in practice, and considering the practical relevance of this work, this is an extremely significant factor.

## Varying the feature vector

As explained in the previous section the goal of the feature extraction method was to model the object’s shape through the feature.

Fig 6.2 Performance for different features

As expected the basic color histogram did not give us good results. This is probably because the histogram suggests similarity between the image characteristics but fails to distinguish when the images are related in terms of the color distribution but the underlying objects are unrelated. For example a grayscale image of a car is actually related to a colored image of the same, but this relation cannot be captured using a histogram. Further the objects in question have the same shape but can have an entirely different color composition for different instances of a single kind of object.

The corner detection method gave much better results. However it did not match up the results obtained by Hough transforms, because of the loss of information. Corner points were grouped into bins of 10x8 regions to reduce the dimensionality which resulted in loss of location information. A future approach could be to control the number of corners by effectively customizing the Harris Corner detector approach to obtain a limited number of corners and feed them to the neural networks as we do in the case of Hough transforms.

Hough transforms gave the best results in this scenario. This was partly because the objects chosen (ships, bicycles, and cars) had geometrical significance in terms of lines and curves. The top hundred lines obtained by Hough transforms gave much better results than the corners because the locality information was preserved better. The results from Hough Circles were not as consistent although it outperformed the lines in a few cases. This is because the radius of a circle needs to be specified in addition to detecting the circles and if this fails to match with the radius of the actual circle in the image, this information is never captured.

A combination of feature vectors was also fed to various classifiers to investigate the performance in such cases. Combining the Hough Transforms and Hough lines gave much better overall results than using either of these methods. Adding Corners to this mix also improved the results in some scenarios, but also showed loss of quality for a few others. Adding the histogram to the combination didn’t show any substantial increase in the quality of the results.

## Varying the dataset

As a part of this evaluation we used three kinds of objects – cars, bicycles and ships. The objects can be expected to have a fixed geometrical model that can be identified by the classifier.

It was found that the Naïve Bayes method gave similar results for each of the datasets. Interestingly varying the classifier didn’t having a drastic effect in the Goodness observed. However in certain cases the classifiers behaved inconsistently for different datasets. This was overcome by using the ensemble averaging method gave a more reliable and normalized set of results.

The performance of the dataset varied considerably with the kind of feature vector. For instance, the curves worked really well for bicycles and cars; however it performed badly with a ship which is expected due to the lack of circles in the geometric model of a ship. The lines gave poor results for bicycles but worked well for cars and ships. Extracted corners gave consistent results for different datasets. An additional dataset of birds was used to verify the behavior on an object that didn’t possess a certain geometric model and didn’t have a fixed shape. The corners method performed well here and surprisingly the histogram method also gave good results in this scenario. Using a combination of feature vectors like lines and curves proved to be consistently good and acceptable for all geometric objects.

# DISCUSSION

The segmentation and object extraction were surprisingly effective and the pipeline of steps worked well together to obtain individual objects. However the query results for more complex objects could contain much more noise and multiple objects which could be hard to separate. At this point we assume that any major connected object returned from an image is significant which has given us considerably good results.

A relevance feedback step is not desired in practice as it needs user intervention. A more practical approach should be to cluster the objects and throw away all the images that do not classify into a substantial cluster. However the variations in objects resulted in poor performance of the SOM which led to the relevance feedback.

Implementing relevance feedback could be done in an online or offline manner. The offline manner chosen above (selecting 10 relevant images) was to avoid multiple passes to the user which would need further user intervention which has a high cost in practical scenarios. The other drawback here is that the training set is extremely small (the user cannot be asked to select relevant results out of a large number of results say 500), thus statistically based classifiers like Naïve Bayes fail to provide a satisfactory solution.

Among the rest of the classifiers it was found that RBFs were the most reliable, giving fairly good performance in terms of average number of correctly filtered out images. MLPs give really good performance for specific problems but are hard to generalize to any kind of objects. Further they are not advisable in practical applications due to the underlying time delay associated with training as well as testing the neural network.

Also the results obtained from Hough transforms are encouraging. In theory any geometrically shaped object can be expressed effectively in terms of lines and curves. The failure of histograms to represent the objects effectively and the improvement by using corners and Hough transforms is a clear indication that these features could play a major role in defining an object model and enabling the classifier to learn this model.

However the inconsistent performance of various classifier and features across multiple kinds of objects is a cause of bother. An attempt was made to replace the original feature vector with a combination of feature vectors or to use an ensemble averaging method for classification. The results from these experiments have been encouraging and tell us this could be the right step to proceed in future.

# CONCLUSION and FUTURE WORK

Image search result refinement can be modeled using the underlying objects, in particular with objects that follow a fixed geometrical model. Shape based features like Hough Transforms and Interest Points can be efficient methods to build such a model. Object invariance is important and can be addressed using relevance feedback from the user. Inconsistency across multiple classifiers and feature detection methods can be addressed using averaging techniques. However image segmentation techniques might need improvement to work with images with a large number of objects or with considerable noise in the background. A possible means to address variation in orientation and size is to introduce object invariance in the input vector as suggested by []. Eventually it is desirable to devise an unsupervised learning approach to automatically refine results without any user intervention.

# ReferenceS

[1] Richard Nock, Frank Nielsen: Statistical Region Merging. IEEE Trans. Pattern Anal. Mach. Intell. 26(11): 1452-1458 (2004)

[2] Sobel, I., Feldman,G., "A 3x3 Isotropic Gradient Operator for Image Processing", presented at a talk at the Stanford Artificial Project in 1968

[3] Siggelkow, S. (2002). Feature histograms for content-based image retrieval. Ph.D. thesis. University of Freiburg, Institute for Computer Science. Freiburg, Germany

[4] Konstantinos G. Derpanis. The Harris Corner Detector. York University 2004.

[5] Chapelle O., Haffner P., Vapnik V. (1997). "SVMs for Histogram-Based Image Classification". *Advances in Neural Information Processing Systems 9*, NIPS 1996, 155-161, MIT Press.

[6] Jaakkola, T., Diekhans, M. and Haussler, D., “Using the fisher kernel method to detect remote protein

homologies,” in *Proceedings of the Internation Conference on Intelligent Systems for Molecular*

*Biology*, Aug. 1999.

[7] P. J. Moreno, P. P. Ho, and N. Vasconcelos (2004) A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications. Neural Information Processing Systems 2003.

[8] [Francesc Serratosa](http://www.informatik.uni-trier.de/%7Eley/db/indices/a-tree/s/Serratosa:Francesc.html), Nicolás Amézquita Gómez, [René Alquézar](http://www.informatik.uni-trier.de/%7Eley/db/indices/a-tree/a/Alqu=eacute=zar:Ren=eacute=.html): Combining Neural Networks and Clustering Techniques for Object Recognition in Indoor Video Sequences. [CIARP 2006](http://www.informatik.uni-trier.de/%7Eley/db/conf/ciarp/ciarp2006.html#SerratosaGA06): 399-405