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 underlying metadata content and provide unsatisfactory results for loosely tagged images. A potential solution is to search the underlying content of the image rather than the metadata instead, which is primary motivation for Content Based Image Retrieval (CBIR) engines. In this work we propose a CBIR approach that uses 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 unsupervised technique – Self Organizing Maps and supervised techniques – Radial Basis Networks and MultiLayer Perceptrons.

## Author Keywords

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

# INTRODUCTION

Contemporary Image Search Engines like *Google* depend heavily on metadata associated with an image result. The associated metadata of an image could include the title, link address, or even the text in the container page. In short the search makes an assumption that the adjoining text is always correlated

to the contents of an image which is not necessarily true. Hence, there are several situations where such searches return several irrelevant images as results.

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.

[Add more relevant work here]

# an overview of our approach

The primary goal was to identify and remove poor results from an image search executed a standard image search query engine. It is critical for this approach that the image query is associated with a real object (rather than an abstract idea). The images 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.

These steps are explained in detail in the subsequent sections. Section 1 explains various segmentation steps applied. Section 2 gives an overview of the kind of features used for classification. Section 4 lists the various methods of learning used. Section 4 provides a summary of the results of this experiment.

[Add figure here]

# preprocessing

The image preprocessing techniques we have used are heavily customized for detecting the most prominent object in the image. Firstly all input images 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 [ ].

[Add figure]

[Add formulae??]

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.

[Add figure]

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. [Add figure]

## Cropping and Mirroring

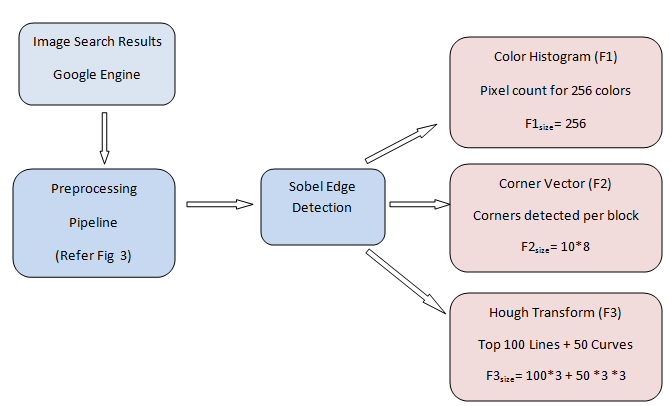
Having obtained the blob of interest we extract the pixels from the original image and retain its pixel values. Also the central point for the object is identified and the image is translated to center on it. As a result the rest of the image (which is blanked out) moves to the periphery. Now the image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

[Add figure]

# 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.



## 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 and marked on the image.

[Add Figure]

[Add Formulae??]

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.

## 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.

[Add Figure]

## Corner Detection

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

[Add Figure]

[Add Formulae]

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

[Add formulae]

## 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.

[Add figure]

## 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.

[Add figure]

# SIMILARITY ANALYSIS

## Unsupervised

## Self Organizing Maps

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

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)**

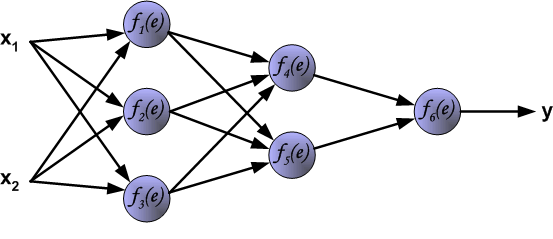
*where   
C(x) = Class of x  
Cj = jth possible* class value  
j = number of possible class values *x = unseen sample feature vector  
a = feature / attribute value   
N = number of features*

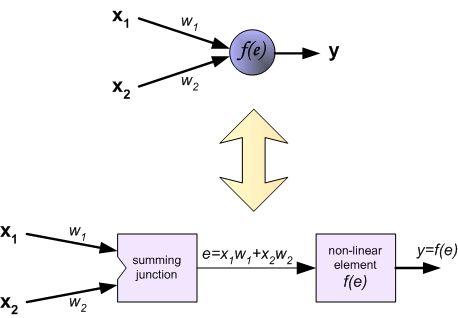
## Multi Layer Perceptron

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.

Figure 5.1 below shows a basic architecture for a multilayer neural network.

 ***figure 5.1***



***figure 5.2***

***where  
x1,x2 = feature vector  
w1,w2 = input weights  
f(e) = is the non-linear element (sigmoid function).***

The sigmoid function at the output of the perceptron places an upper bound on the output of each unit hence the term *“squashing function”*. The equation is given below:

**(*5.2)***

***Where  
x = input  
τ = sigmoid coefficient used for small input values(x < 10-4***).

## Support Vector Machines

Support vector machines is a robust machine learning algorithm used for classification and regression that has been shown to achieve good results in image processing [5]. A support vector machine (SVM) constructs a set of hyperplanes in a [high](http://en.wikipedia.org/wiki/High-dimensional_space) or infinite dimensional space, which can be used for classification, regression or other tasks.

Several approaches have been made to use SVMs in the multimedia data classification domain. Fisher et al [6] used a Gausian mixture model and Moreno et al building on this, derived a kernel distance

based on the Kullback-Leibler (KL) divergence between generative models[7].

We use the Gausian mixture model available in the toolkit with the feature vectors extracted from the Histogram extraction method described in section 4.

## 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

# DISCUSSION

# CONCLUSION

# ReferenceS

1. [1] Richard Nock, Frank Nielsen: Statistical Region Merging. IEEE Trans. Pattern Anal. Mach. Intell. 26(11): 1452-1458 (2004)
2. [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. [3] Siggelkow, S. (2002). Feature histograms for content-based image retrieval. Ph.D. thesis. University of Freiburg, Institute for Computer Science. Freiburg, Germany
4. [4] Konstantinos G. Derpanis. The Harris Corner Detector. York University 2004.
5. [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. [6] Jaakkola, T., Diekhans, M. and Haussler, D., “Using the fisher kernel method to detect remote protein
7. homologies,” in *Proceedings of the Internation Conference on Intelligent Systems for Molecular*
8. *Biology*, Aug. 1999.
9. [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.