A Framework for Image Classification

Mamoun Awad, Lei Wang, Yuhan Chin and Latifur Khan
The University of Texas at Dallas
lkhan@utdallas.edu

George Chen, and Fehmi Chebil
Nokia Research Center
george.chen@nokia.com

Abstract

Image annotation process requires time and human intervention. In this research we propose a framework to incrementally annotate images in the database based on user feedback. At the beginning users provide some annotations for images manually as a ground truth. Classifier will be trained based on this ground truth. The classifier predicts annotation for new images that are not part of the ground truth. Feedback is collected from the users to increase the size of the training set and then the classifier is retrained. The system strives to capture feedback from users and retrains the classifier on the new training set .Our proposed framework facilitates semi-automatic image annotation

1. Introduction

The development of technology such as digital cameras and mobile telephones equipped with image capturing devices generates huge amounts of non-textual information, such as images. These applications have rapidly penetrated many domains and markets, including commercial and news media photo libraries, scientific and non-photographic image databases, and medical image databases. As a consequence, we face a daunting problem of organizing and accessing these huge amounts of available images. At present, there are few areas in which retrieval and storing mechanisms exist, such as web-image search engines, surveillance applications, and medical imaging. In the medical realm, for example, searching a history of images for a patient or for comparative images during diagnosis can mean life or death. In this environment, annotation and metadata are of critical importance for all images subject to search and retrieval. This search and retrieval should not be based on only visual features of images but also high level semantics of images.

In the early 90's, because of the emergence of large image collections, content-based image retrieval (CBIR) was proposed. CBIR computes relevance based on the similarity of visual content/low-level image features such as color histograms, textures, shapes, and spatial layout [3]. However, the problem is that visual similarity is not semantic similarity. There is a gap between low-level visual features and semantic meanings. The so-called semantic gap is the major problem that needs to be solved for most CBIR approaches [4, 5]. For example, a CBIR system may answer a query request for 'red ball' with an image of a 'red rose'. If we undertake the annotation of images with keywords, a typical way to publish an image data repository is to create a keyword-based query interface addressed to an image database. Images are retrieved if their descriptions/annotations contain (some combination of the) keywords specified by the user.

Image annotation could be automatic, semi-automatic, and manual. Manual annotation is accurate but the process is time consuming and expensive. Automatic annotation techniques are still on research and not reliable. Semi-automatic approaches which involve users' feedback are most practical currently [2].

In this paper, we propose a novel framework that will facilitate semi-automatic image annotation. Our proposed framework will work as follows. At the beginning users provide some annotations for images manually as a ground truth. Classifier will be trained based on this ground truth. A classifier will predict annotation for images that are not part of the ground truth based on current ground truth. Then the classifier will strive to capture feedback from users and will be retrained with more ground truth. It is likely that ground truth will grow incrementally. The system will stop to provide feedback when we notice we have observed enough accuracy on testing set. At that point,

1-4244-0069-4/06/$20.00/©2006 IEEE.

the database is almost fully annotated. Note that the classifier has to be incremental and should operate in real time basis. As a classifier, we will use one of the most successful classifiers, Support vector machine (SVM) along with a K nearest neighbor (KNN).

The paper is organized as follows: Section 2 presents a detailed description of our proposed architecture for annotation. Section 3 presents discussions and comments on future work.

2. Proposed Architecture for Annotation

Our system is composed of five components (See Figure 1). Each component performs a well defined task. Grid segmentation segments the images into n number of grids. In this system we use 16 grids. Next we apply feature extraction module. Then we provide ground truth for the classifier. Finally, the system will support keyword based query or query by example. For our experiments, we use Corel dataset [1].

2.1. Feature Extraction

First, feature extraction is employed for the entire image without doing any segmentation/partition. Next, we extract features for each grid. For grid case, we focus on color and texture features. We ignore shape because our focus is not on object recognition/boundary detection. We extract texture information of each grid using three representative energies in high frequency bands of the wavelet transforms. To obtain these moments, a Daubechies 4-wavelet transform is applied to the L component of the image. After one-level transformation, a grid is decomposed into four frequency bands: the LL (low low), LH (low high), HL, and HH bands. Here, we resample all images to the same resolution before computing wavelet coefficients.

Here, we exploit three sets of features. Feature set 1 is used for the whole image. We use average RGB, LUV color features (dimension=6), texture features (=3), and 10 moments including spatial, central, moment Hu invariant. As a whole, we have a total of 19 features. Feature set 2 and 3 will be used for each grid. For feature set 2, we convert the basic color components (R, G, B) to HSV. With HSV, we calculate the color-histogram bins (7*7) of H and S, respectively. We construct 7X7 bins (=49). Along with 6 color features (RGB, LUV) and texture features (=3), we will end with 58 features/grid for feature set 2. Finally, for feature set 3, we increase histogram bin sizes. Here, we use 10X10 bins. Hence, the total number of features will be 109. In the feature extraction phase, we extract the features for each grid using the INTEL OPENCV software.

2.2. Ground Truth Annotation

In ground truth phase, we try to choose quality images that are related to a specific concept and annotate them manually. The goal here is to choose the grids that are mostly related to specific concept. For example, in Figure 2, we choose ground truth for the concept "Bear". As we can see, not all the 16 grids are related to "Bear"; hence, we only choose grid 1 and 5 for the image in part A, and grids 5, 6, and 10 for image in part B. On the other hand, for the whole image case, we choose the whole image as a ground truth.

2.3. SVM Training

After collecting reasonable number of ground truth for each concept, we start the process of training. We use Support Vector Machines (SVM) for training. The training process is very fast because only the basic ground truth grids are involved here. For the purpose of feature selection and testing, our system provides several options. For example, the user has three different feature sets (see Section 2.1: Feature Extraction). Also the user has the choice to choose between SVM and K nearest neighbor (KNN) in training and prediction. In addition, the user can choose between the whole and 16 grid schemes.

2.4. K Nearest Neighbor (KNN) Training

K Nearest Neighbor is a supervised learning algorithm where the result of new instance query is classified based on majority of *K*-nearest neighbor category. The purpose of this algorithm is to classify a new object based on its attribute values and training samples. *K* Nearest neighbor algorithm uses neighborhood classification as the prediction value of the new query instance. *KNN* works based on the minimum distance from the query instance to the training samples to determine the *K*-nearest neighbors. In image annotation, we compute the *K* nearest images for a given image. We use Euclidean distance as a similarity measure. Depending on the features set of the images, the computation of the distance differs. For Feature set 1 where we extract features for the whole image, we compute the Euclidean distance directly as follows:

$$d(x_i, x_j) = \sqrt{\sum (x_i - x_j)^2} \qquad (1)$$

where x_i and x_j are the feature set of image *i* and image *j*. For feature set 2 where we partition the image into grids, we compute the distance by summing up the Euclidean distance between each two corresponding grids in the images as follows:

$$gd(n,x,y) = \sum_{k=1}^{n} d(x_k, y_k) \qquad (2)$$

where *n* is the number of grids and *d*(*x*,*y*) is the Euclidean distance between feature sets *x* and *y*, and x_k is the grid k of image x. After gathering *K* nearest neighbors, we take simple majority voting of these K-nearest neighbors to be the prediction of the query instance.

2.5. Keyword/Example based Search

Our system facilitates a keyword search and query by example. The user clicks on a concept keyword and the related/closed images to that concept are displayed. Figure 3 presents a keyword search results. The user clicks on a concept keyword and the related/closed images to this concept are displayed. Prediction here is based also on either SVM or KNN in which the user can choose. Notice that the grids on the right side of the images in Figure 3 represents the feedback box in our architecture. In this portion of our interface, the user can express his satisfaction of the results and can add new ground truth to the database. Here positive feedback is taken into consideration. For example, checking a grid means that grid is relevant to the concept that is being queried. After a reasonable number of ground truth is added, new training round is performed. The idea here is to increase the training set incrementally by utilizing users' feedback. User can also retrieve relevant images based on query by example (QBE). In QBE testing phase, we test our system by searching based on Query-By-Example. We have a fixed list of example images. Once we click on one of these images, the closest images are displayed. The similar image will be determined based on the Euclidean distance between QBE image and the test image. As mentioned in Section 2, the distance is computed differently depending on the feature set chosen. For feature set 1 where we extracted features for the whole image, we compute the distance using Eq. 1. For feature set 2 and 3 where we partition the images into grids, we use Eq. 2 to compute the distance. The lower the distance, the more similar will be the image with the query image. For example, Figure 4 presents the results of searching the closest images to a given image. The image on the top is the example, and the two rows of images are the closest images to it.

We consider the color features in feature set 2 and 3 more than the shape and texture. Because of that different concepts with similar colors might be shown in the results. For example, our system might mistakenly show building results for the concept people. That is because the brown color is dominant in many building and people images. To mitigate that, we capture the user feedback at the end of each query results. The user can evaluate the results by choosing those grids/images that are relevant to the example image (or keyword) the most. As a result of that, the training set increases and a new training round is applied. The system performance enhances using the feedback, training set is incrementally increased, and the search results improves.

3. Discussion and Future Work

The result we obtained is promising, and depends on the feature we use. In the case of whole image extraction (for feature set 2 & 3), we notice that the results are concept dependent. For example, images of concepts with similar colors might be predicted incorrectly. For example, the rock may be confused by the building because both the rock and the building have similar colors. On the other hand, concepts that has distinct color features, such as snow and sky, get perfect results, 100% accuracy (see Figure 3, which shows the prediction results for the concept "snow"). On the other hand, for grid based feature extraction case, results will be satisfactory if images have multiple multiple concepts.

In future, we would like to extend our work along the following directions. First, we would like to extract more low level features using MPEG-7 and test the annotation accuracy. Next, we would like to modify SVM/KKN classifier so that it can be used in incremental basis along with real time usage.

Acknowledgement

This research was supported in part by gift from NOKIA Research Center, Irving, Texas, USA.

4. References

[1] http://kdd.ics.uci.edu/ databases/ CorelFeatures/ Corel Features.data.html

[2] A. Natsev, M. Naphade, J. Teslc, "Learning the Semantics of Multimedia Queries and Concepts from a Small Number of Examples," in Proc. of The *13th Annual ACM International Conference on Multimedia (MM 2005)*, Singapore, November 2005, pp. 598-607.

[3] J. R. Smith and S. Chang, "VisualSEEK: A Fully Automated Content-Based Image Query System.", *ACM Multimedia 1996*, pp. 87-98.

[4] L. Zhu, A. Rao, and A. Zhang, "Theory of Keyblock-based Image Retrieval," *ACM Transactions on Information Systems*, vol. 20, No. 2, April 2002, pp. 224-257.

[5] Y. Chin, L. Khan, L. Wang, and M. Awad, "Image Annotations By Combining Multiple Evidence & WordNet" in Proc. of *The 13th Annual ACM International Conference on Multimedia (MM 2005)*, Singapore, November 2005, pp. 706-715.

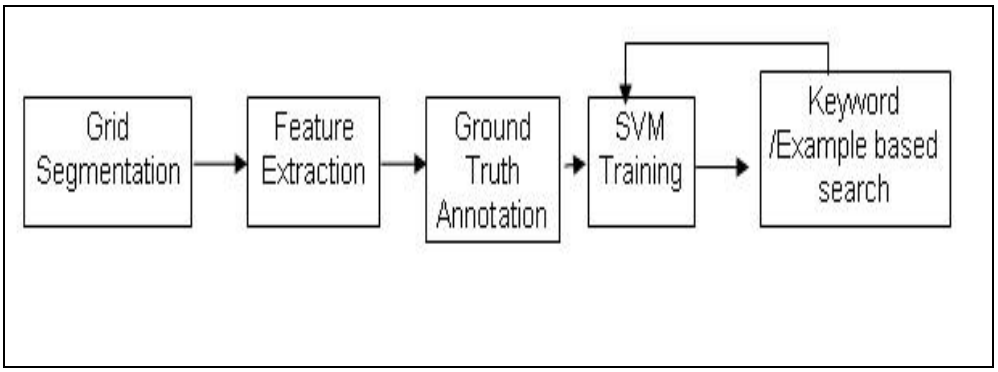

Figure 1 Architecture of Our Image Annotation Mechanism

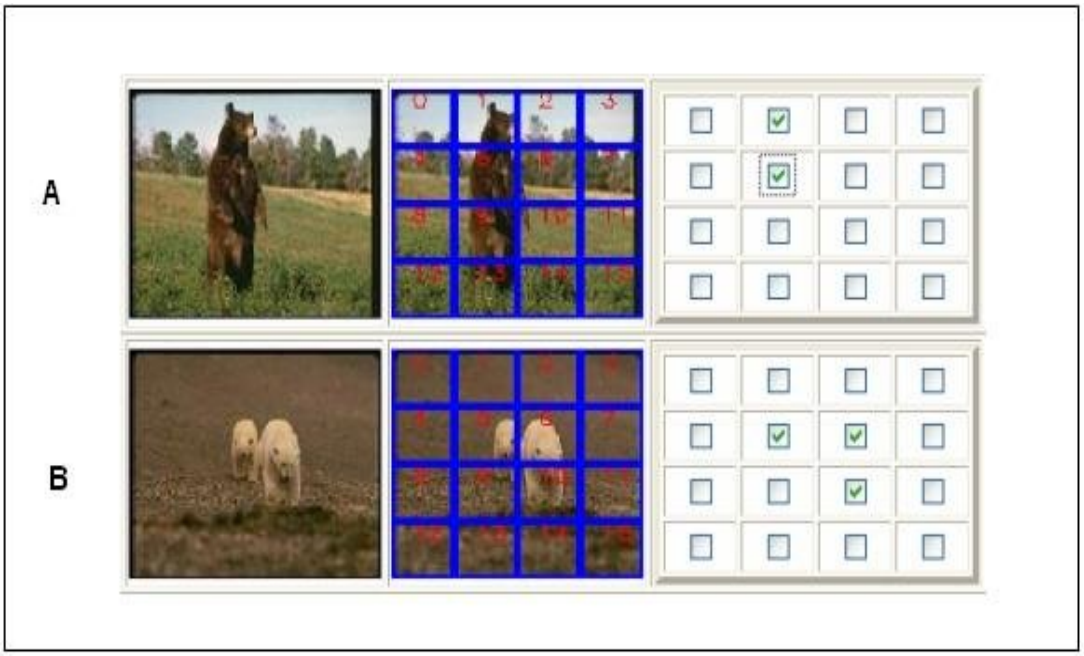

Figure 2 Ground Truth Collection

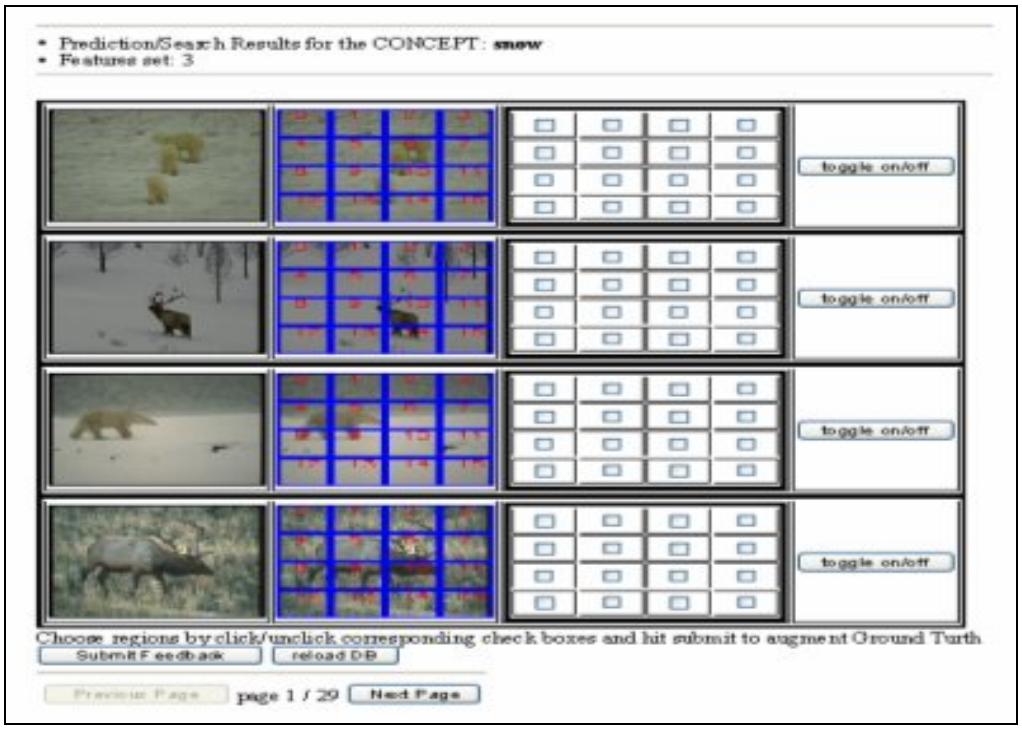

Figure 3 Keyword query based search

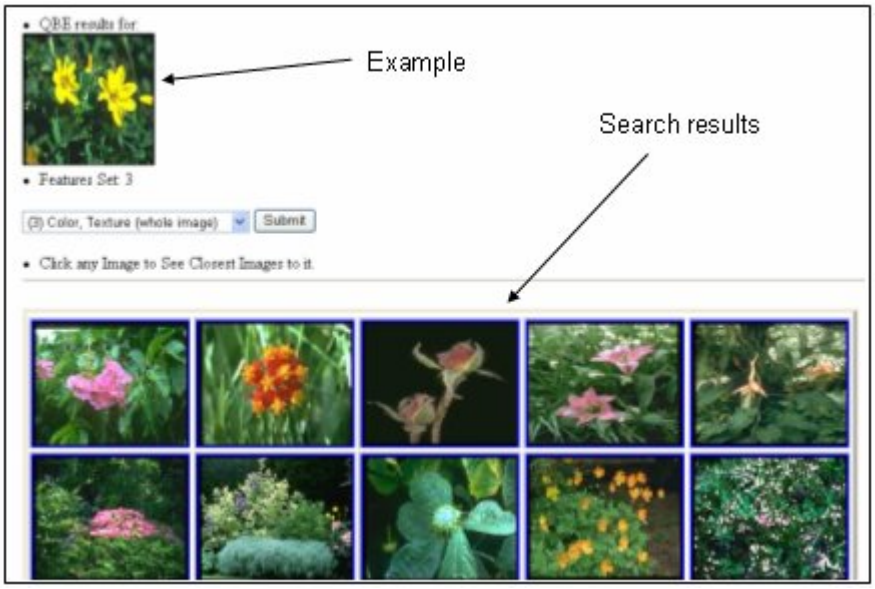

Figure 4 Query by Example Search Results