An Ad Hoc Image Annotation Engine

Ying Wang
Department of Mathematics, The Ohio State University

Introduction

While significant progress has been made in information retrieval in text documents, tools for indexing multimedia data are still primitive. One of the most difficult but important component of such system is Automatic Image Annotation. Automatic Image Annotation (also known as automatic image tagging) is the process of the computer system automatically assigning keywords to a given digital image, and this is used in image retrieval systems to locate images that are queried. In this work, we propose a new ad hoc algorithm based on machine learning to perform automatic image annotation. Scale Invariant Feature Transform (SIFT) algorithm is employed to extract the key points and feature vectors of each image. We use the k-means clustering technique to construct *visual words*. A set of benchmark training and testing images from Visual Object Classes Challenge 2006 (VOC2006) database has been used to assess the proposed image annotation algorithm. The numerical experiments confirm the effectiveness of the primitive algorithm. Two possible improvements are given, and the numerical experiments suggest the success in improving the performance.

Image Representation Using Visual Words

Key Points and Feature Vectors

Scale Invariant Feature Transform (SIFT) [1] is a method to extract distinctive features from an image which is invariant under scaling, rotation, addition of noise, and illumination change etc. The first step of SIFT is to find the key points of the image. The basic idea of extracting the keypoints is to apply Gaussian filters at different scales of the image. Then the extrema points are taken from the difference of successive Gaussian-blurred images(Difference of Gaussians - DoG), and this is what we call keypoints. Given an image, $I(x,y)$, and a variable-scale Gaussian, $G(x,y,\sigma)$, the scale-space of an image is defined as [1]

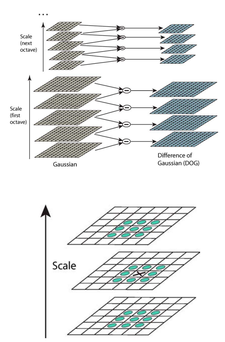
$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y), \qquad (1)$$

where $*$ is the convolution operation in x and y, and

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}, \qquad (2)$$

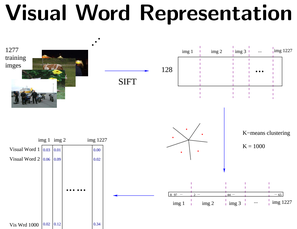
and the Difference of Gaussian function can be calculated by

$$\begin{aligned} G(x,y,\sigma) &= (G(x,y,k\sigma) - G(x,y,\sigma)) * I(x,y) \qquad (3)\\ &= L(x,y,k\sigma) - L(x,y,\sigma), \end{aligned}$$



where k is a constant multiplicative factor which separates two nearby scales. Figure on the right illustrates how this algorithm works. Interest points (called keypoints in the SIFT framework) are identified as local maxima or minima of the DoG images across scales. SIFT assigns a feature vector with 128 elements to reflect the local properties.

Visual Word Representation



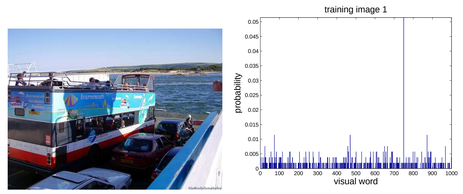
We used the public image data sets from VOC2006 database [2] including predefined training and testing sets. The training set, which is used for training the image annotation machine, includes 1277 images and the testing set includes 2686 images. There are 10 object classes(labels): bicycle, bus, car, cat, cow, dog, horse, motorbike, person, sheep. Every image has of one or more labels.

For every image, the SIFT algorithm reads in the grey-scale intensity matrix and gives a collection of feature vectors. The bag of feature vectors collected from all the training images is then sent to the k-mean algorithm. The k-mean algorithm clusters all the feature vectors into k clusters. In this work, $k = 1000$ is chosen. Now, among the feature vectors belonging to one training image, we can have a count of feature vectors in each cluster. Each cluster can be conceptually understood as a visual word. Therefore, for each training image, there's a visual word distribution associated to this image. This process is shown in Figure on the left. In fact, this visual word distribution is THE representation of the content of this image. In addition to the visual word distribution, there's also a collection of tags assigned to this image.

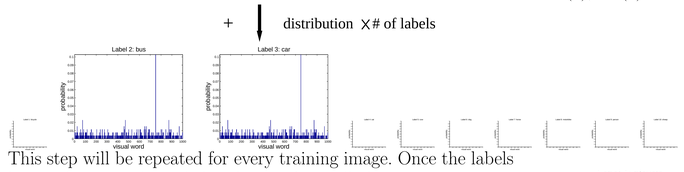
An Ad Hoc Image Annotation Algorithm

Building Visual Word Distribution for Each label

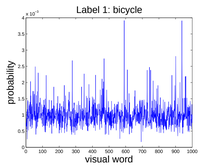
Once the visual word distributions for all the training images are built, it's ready for the labels to learn their visual word distributions. We use an example to illustrate this process.



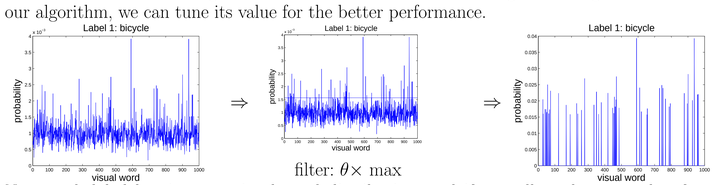
Labels: bus(2), car(3)



This step will be repeated for every training image. Once the labels learnt their visual word distribution from all the training images, each of the label will have a visual word distribution on its own, for example, the visual word distribution of label 1 is shown in Figure on the right. Heuristically, this Figure shows the probability that each visual word is associated to label 1, and it can also be conceptually understood as the probability that label 1 should be assigned when each visual word appears in an image, i.e. P(l_1 should be assigned when w_i appears in an image).



The fluctuation at the bottom of this figure could be noise. To get rid of the fluctuation, we can impose a filtering parameter, call it θ, and everything below $\theta\times$ max will be filtered out from the distribution as shown in Figure below. Note that the parameter θ give one degree of freedom of our algorithm, we can tune its value for the better performance.

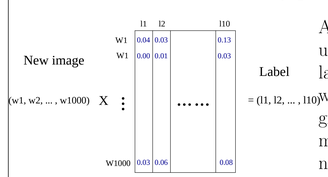


Now, each label has its own visual word distribution, and if we collect them together for all the labels, we get a matrix . Each column of this matrix represents the visual word distribution for each label. Heuristically, the (i,j)-entry means P(l_j should be assigned when w_i appears in an image).

Automatically Annotating the Testing Images

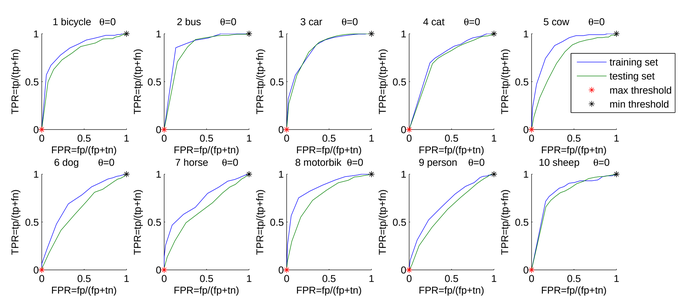
When a testing image is given to the annotation engine, the engine first compute its visual word distribution as what's done to each training image. Then this row vector containing the visual word distribution of this testing image is multiplied by the visual word distribution matrix obtained in the previous step. Figure below shows the details. After the multiplication, we get a new row vector. The j-the entry gives the confidence that label j should be assigned to this testing image, and the reason is the following:

$$\mathrm{P}(l_j \text{ is assigned}) = \sum_{i=1}^{1000} \mathrm{P}(w_i) \cdot \mathrm{P}(l_j \text{ is assigned when } w_i \text{ appears}).$$



After the confidence is calculated for each label, we will use a decision making threshold to help us to give the final labeling decision: $l_j \geq \max_{i=1..10}(l_i)\times$ threshold $\Rightarrow$ l_j will be assigned. Note that the decision making threshold gives the second degree of freedom. We will vary this parameter to evaluate the effectiveness of our algorithm at the next step.

Assessment of the Algorithm using ROC curves



ROC curve will be used to evaluate the effectiveness of the algorithm. We fix the filtering parameter $\theta = 0$, and vary the decision making threshold to draw the ROC curve, and the result is given in Figure on the left.

Improvements to the Ad Hoc Algorithm

Improvement Through Decision Making

When a testing image is given to our image annotation engine, a confidence that each label should be assigned to this image is calculated, namely l_j. Previously, the way we make the final labeling decision is:

$$\text{if } l_j \geq \text{ threshold } \times \max_{i=1}^{10} l_i, \text{ then } l_j \text{ is assigned to this image.}$$

To improve the performance, we can use the following decision:

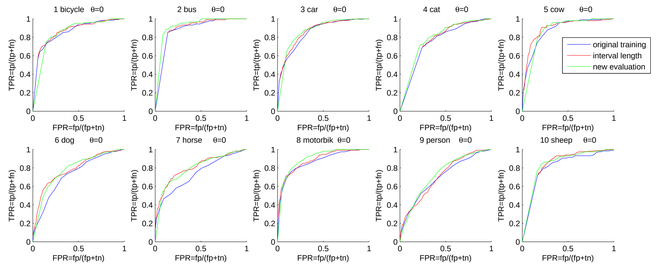
$$\text{if} l_j \geq \min_{i=1}^{10} l_i + \text{ threshold } \times (\max_{i=1}^{10} l_i - \min_{i=1}^{10} l_i), \text{ then } l_j \text{ is assigned.}$$

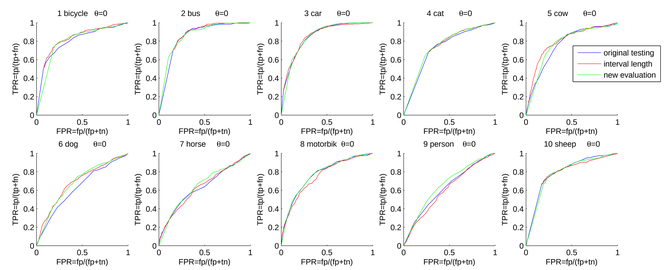
Improvement Through Multiple-Label Scheme

Another way to make the final labeling decision is to compare the visual word distribution of the testing image with that of a single label, or combined labels. This means that we assume an image consists of 1 or 2 labels, then we build the ideal visual word distribution for such images, and then compare the given testing image with the ideal visual word distribution of images with one label or two labels. The decision making step is:

$$\begin{aligned} \text{if dist(testing image,}l_j) &< \min(\text{dist}) + \text{threshold} \times (\max(\text{dist}) - \min(\text{dist}))\\ &\quad \text{then } l_j \text{ is assigned to this image}\\ \text{if dist(testing image,}\{l_j, l_k\}) &< \min(\text{dist}) + \text{threshold} \times (\max(\text{dist}) - \min(\text{dist}))\\ &\quad \text{then } l_j, l_k \text{ are assigned to this image} \end{aligned}$$

The ROC curve for the training images and testing images with this improvement along with improvement 1 are given in Figures below.





Conclusion

We have built an ad hoc automatic image annotation engine and also proposed two possible ways to improve the primitive design. The performance has been tested on a set of benchmark training and testing images. The numerical experiments confirm the effectiveness of the primitive algorithm and the improvements.

Literatur

[1] Lowe, David G., *Distinctive Image Features from Scale-Invariant Keypoints*, International Journal of Computer Vision, 60(2) 91–110, 2004.

[2] Ëveringham, M. and Zisserman, A. and Williams, C. K. I. and Van Gool, L., *The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results*, `http://www.pascal-network.org/challenges/VOC/voc2006/results.pdf`.