5. Semantic-Based Image Feature

In previous chapters, we have described low-level visual features extracted from images and middle-level features based on visual words in feature spaces. Now, we focus on bridging the semantic gap between low-level features extracted from images and high-level concepts involved in human intention in this chapter. The basic idea of our approach is to extract semantic-based image features by automatically annotating semantic labels in images. A set of labels is predefined to describe the contents of images, and each image is automatically annotated by these labels to extract semantic information for the image.

In this chapter, we propose a novel method of image annotation based on a semi-supervised and hierarchical approach. We construct individual hierarchical classifiers associated with labels, and an image can be computed the confidence value of annotation according to the classifier. We also design a semi-supervised learning for the construction of hierarchical classifiers by using both labeled and unlabeled images in the stage of learning.

5.1. Semantic Gap and Image Annotation

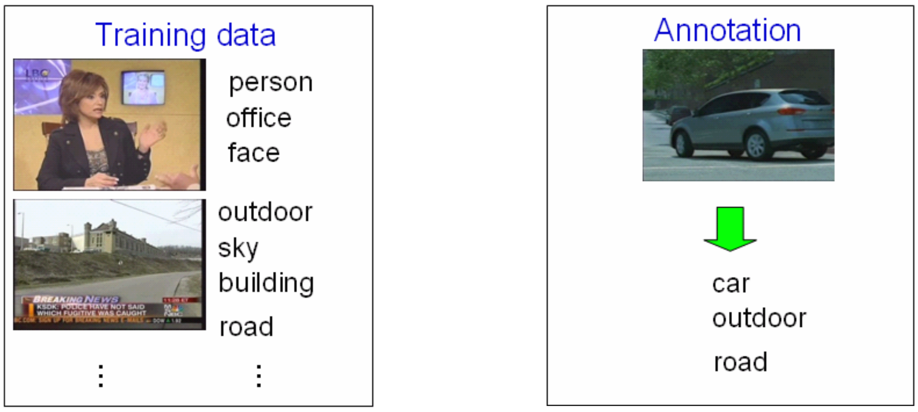
Semantic gap [Smeulders et al. 00] [Lin et al. 07] is one of the most difficult tasks in image understanding and retrieval. In computer vision, only visual features can be directly extracted from images. However, humans recognize or understand an image according to semantic concepts. For example, humans will prefer to specify a query containing forest and sky rather than describe the query with 80% green and

20% blue. How to bridge the semantic gap in image understanding and retrieval is important but difficult.

There are two main potential approaches to the problem of semantic gap. The first is to extract semantic-based information for images, namely automatic image annotation, and the second is to interactively refine the retrieval results according to the user feedbacks, namely relevance feedback. This chapter aims at proposing an approach of image annotation based on a semi-supervised learning to extraction semantic-based image feature. Moreover, the discussion of relevance feedback is presented in Chapter 6.

In automatic image annotation, a set of annotation labels needs to be predefined for describing the semantic contents of images. The process of image annotation can be identified to two stages: learning and annotation, illustrated in Figure 5-1. In the learning stage, a set of training data, which consists of images and their corresponding labels illustrated as Figure 5-1(a), can be used to learn the model between the visual contents of images and their assigned labels. Then, in the annotation stage, the unlabeled image is tested by the learned model in order to assign the annotation labels as shown in Figure 5-1(b).

In general, an image can be associated with multiple labels, e.g., an image can be annotated by “sky”, “forest”, “water”, “tree”. Let the entire image set $D=\{I_1, \ldots, I_N\}$ contain N images, and the label set $L=\{L_1, \ldots, L_H\}$ contains H predefined annotation labels. The goal of image annotation is to assign the vector $\mathbf{v}_\mathbf{s}=\{v_1, \ldots, v_H\}$, for an image I, where v_h, $1 \le h \le H$, means the confidence value of that image I can be annotated by label L_h. Each confidence value v_h of annotation can be either the 0-1 assignment (i.e., $v_h \in \{0,1\}$) or a real value in [0, 1]. Note that the formulation of our problem is described in Section 5.3.1.



(a). Learning stage using training data. (b). Annotation stage to unlabeled data.

Figure 5-1. Two stages of image annotation.

Image annotation can be regarded as concept detection, which discovers or detects what concepts are included in an image by annotating textual information. The task of image annotation that associates text to the semantic contents of images has been used as an intermediate step to image retrieval [Srikanth et al. 05]. That forms the semantic-based image features in retrieval and is potential to be the solution of the problem of semantic gap.

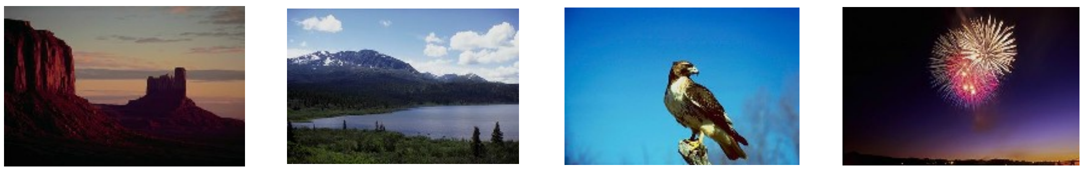


Figure 5-2. The different image contents with the label "sky".

Unfortunately, it is a difficult task to build a model that can describe the contents of images with semantic labels, even with a single label. Regarding the semantic meanings involved in images with a single label, the contents are not often homogeneous. For example, Figure 5-2 shows the four images that all contain the same label “sky”, but their semantic contents, we do ignore their foregrounds here, are very different – sunset, cloud or cloudless, blue sky, and night. Moreover, it must be more complex if many labels are mixed. Hence, our opinion is to consider only one label in annotation and try to model the different contents of images for the single label. Our approach is to build individual hierarchical classifiers each of them associated with a semantic label. Using an individual classifier with a label can reduce the complexity of the learning problem, and the hierarchical approach can divide the learning into several sub-problems that could match the different contents of images.

Image annotation is often considered a supervised learning problem in most of the state-of-the-art approaches [Carneiro and Vasconcelos 05]. A main drawback of the supervised learning approach for image annotation is that a large amount of training images is necessary to avoid overfitting. However, it is often difficult to manually annotate a large set of images. This reason motivates us to design a semi-supervised learning approach by integrating labeled and unlabeled images to reduce the amount of the training images in the learning stage of image annotation.

There are two main advantages of our proposed method for image annotation. The first is the scalability in the training due to individual classifiers associated with labels. When a new annotation label is added to describe the semantic contents of images, we only need to train the new classifier associated with the new label. Besides, additional labeled data can be added to re-train an existing classifier to improve the accuracy of the annotation classifier. The second is to reduce the amount of the labeled images in the learning due to the proposed semi-supervised approach. That is very important because the labeled data are sometimes difficult to be collected. This advantage can make the annotation system more flexible.

5.2. Semi-Supervised Learning

Semi-supervised learning could be simply defined that the classifier is learned by both labeled and unlabeled data [Cohen et al. 04]. Here, we briefly categorize semi-supervised learning approaches as two types. The first is that the learning model is based on supervised learning and extra unlabeled data are added to help the learning. The second is that the learning approach is based on unsupervised clustering and extra labeled data are added to improve it. In the former, how does the unlabeled data help the learning classifier is the critical issue. Also, in the latter, how does the labeled data help the structure analysis of the unlabeled data is the most important. Our proposed approach for image annotation belongs to the latter – the unsupervised clustering with extra labeled data – by applying a small amount of labeled images to improve the clustering task. Now, we introduce the two types of semi-supervised learning and their related works as the follows. Note that there have been several good articles of surveys for semi-supervised learning [Zhu 05] [Haffari 06] [Basu 05].

5.2.1. Using supervised learning

We first discuss the related works of the first type that the learning model is based on supervised learning and extra unlabeled data are added. There have many different approaches for supervised learning, and these approaches can be revised to be semi-supervised learning by integrating unlabeled data for training. Cohen et al. provided a new analysis that shows under what conditions unlabeled data can be used in learning to improve classification performance [Cohen et al. 04]. They also showed that using unlabeled data can be detrimental to classification performance if some conditions are violated.

Support vector machine (SVM) employs the support vectors to find a maximum margin linear boundary to design the classifiers, but the selection of support vector greatly affect the performances of the classifiers. Hence, some researchers applied additional unlabeled to help the choice of the support vectors. Xu and Schuurmans proposed a training method based on semi-definite programming to coordinate labeled and unlabeled data in SVM [Xu and Schuurmans 05]. Zhang and Oles presented their probability analysis of applying both labeled and unlabeled data for SVM [Zhang and Oles 00].

Self-training is a commonly used technique for semi-supervised learning [Zhu 05]. A classifier is first trained with a small set of labeled data, and then unlabeled data are tested by the classifier. The most confident unlabeled data points, together with their predicted labels, are added to enlarge the training set and used to re-train the classifier, and the process repeats. Self-training has been applied to several natural language processing tasks, e.g., Yarowsky applied the self-train method for word sense disambiguation [Yarowsky 95].

Co-training approach needs a basic assumption that features can be split into two subsets, each other conditional independent given the class, and each one is sufficient to train a good classifier [Blum and Mitchell 98] [Zhu 05]. In the beginning, the two separate classifiers are trained with the labeled data on the two sub-feature sets, respectively. Each classifier then classifies the unlabeled data, and ‘teaches’ the other classifier with the most confident unlabeled examples, with the predicted labels. Each classifier is retrained with the additional training examples given by the other classifier, and the process repeats.

5.2.2. Using unsupervised clustering

Next, we discuss the related works of the second type that the learning approach is based on unsupervised clustering and extra labeled data are added, often called semi-supervised clustering. The unsupervised clustering aims to analysis the data structure in the feature space, and, in this type of semi-supervised clustering, the extra labeled data form constraints on the clustering process in order to improve the accuracy of the clustering.

Bilenko et al. employed labeled data to improve the accuracy of *K*-means clustering [Bilenko et al. 04]. Two roles of labeled data used in their work are (i) to be the constraints in data clustering and (ii) to be used for learning the metric in the clustering. In the former, must-link and cannot-link are appended among unlabeled data according to the extra labeled data. In the latter, moreover, the metric to compute the distance of two data is tuned by the labeled data. Qian et al. designed a hierarchical clustering and the labeled data are used to be the constraint in constructing different levels of the classifier [Qian et al. 06]. Jin et al. designed a *K*-means clustering for image annotation by integrating labeled images to be the constraint of clustering [Jin et al. 04].

5.3. Image Annotation

This section presents our approach to the problem of image annotation. We first formulate our problem for image annotation and introduce our approach in overview, and then describe the approach in details in two stages: learning and annotation.

5.3.1. Problem formulation

Following the description and the notation in Section 5.1, we formulate the annotation problem in this section. Let the entire dataset, denoted as D, contain N images. Suppose that H annotation labels $\{L_1, \ldots, L_H\}$ are predefined to describe the semantic contents of the images. Because the number N is usually a huge number, it is hard to annotate all images in D manually. Thus we only annotate a part of images in D with labels $\{L_1, \ldots, L_H\}$, each labeled image may be associated with multiple labels. Let all labeled images associated with L_h as D_h, and D_L is the union of D_h, h=1, …, H. Let unlabeled image set, $D_U = D - D_L$, is the set of images that is not associated with any labels, and $|D_L| << |D_U|$. Note that $D = D_U \cup \mathrm{D}_1 \cup \mathrm{D}_2 \cup \ldots \cup \mathrm{D}_H$ and the intersection of any two D_i and D_j, $i \neq j$, may be non-empty because one image may associated with multiple labels. Moreover, we denote $D_h^{'}$ as the set of labeled images that is not associated with L_i, i.e., $D_h^{'} = D_L - D_h$. In general, we know $|D_h^{'}| \cong (H-1) \cdot |D_h|$ because images with other labels may belong to $D_h^{'}$, and that must yield the imbalance problem for $|D_h| << |D_h^{'}|$ in the learning. Hence, we randomly chose images from $D_h^{'}$ such that $|D_h| = |D_h^{'}|$ to avoid the imbalance problem.

For a label L_h, we simply call dataset D_h as positive image set, and $D_h^{'}$ as negative image set. Therefore, the annotation problem can be formulated as the follows. Given an unlabeled image d_{new}, we want to decide which labels the image can be associated with. In general, we can compute

p(L_h is associated with d_{new} | d_{new}), where $1 \leq h \leq H$. (5.1)

Note that the unlabeled image d_{new} could be or not be in the unlabeled data D_U.

5.3.2. Learning stage

In this section, we propose a semi-supervised approach for learning hierarchical classifiers that are used for annotating images. Our approach consists of two basic ideas: (i) build a hierarchical classifier denoted as C_h for each label L_h, and (ii) design a semi-supervised approach by integrating labeled and unlabeled data to learn the hierarchical classifiers. For the classifier C_h associated with the label L_h, all unlabeled images having similar behavior with positive-label images are classified as positive, and those with negative-label images are classified as negative. Figure 5-3 depicts the concept of our work that integrates all labeled and unlabeled data to train the hierarchical classifiers each corresponding to a label individually.

Table 5-1 shows the algorithm of constructing a hierarchical classification C_h for label L_h. In this algorithm, the root node N_{ij}^{h} of the tree C_h initially contains all images in D_h and D_U, and some images in $D_h^{'}$, where $|D_h| = |D_h^{'}|$ for the balance. Then we recursively decide which node needs to be split .If we decide to split a node, we go on to decide how many branches are appropriate to split for the node. Here, K-means clustering is applied to divide a node into several child nodes. We try a range (2 to a constant b) of branch number and calculate the corresponding score to select the best branch number for splitting nodes. The goal of trying a range of branch number is to choose a proper number of branches for splitting the node according to all images of the node. Hence, our proposed semi-supervised approach learns the classifier in the two ways: (i) evaluate the stopping criteria for node splitting according to the positive and negative images set in the node and (ii) split a node by use of K-means clustering in construction of the hierarchical clustering.

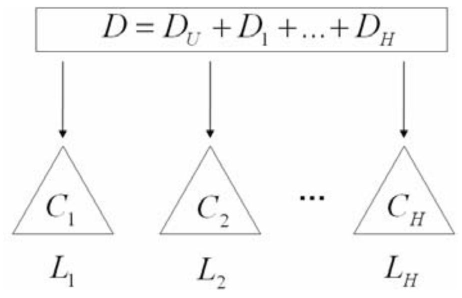


Figure 5-3. Learning hierarchical classifiers C_h for label L_h using both labeled and unlabeled data.

Table 5-1. The algorithm of constructing the hierarchical classification C_h for label L_h.

```
Input:  unlabeled images $D_U$
        positive-labeled images $D_h$
        negative-labeled images $D_h^{'}$
Output: a tree classifier $C_h$ for label $L_h$
Initialization: the root node $D_{11}^h$ contain $D_U \cup D_h \cup D_h^{'}$
// $N_{ij}^h$: node j at level i for label $L_h$.
// construct the tree by splitting each node $N_{ij}^h$.

1. for each leaf node $N_{ij}^h$ not fitting the **stop condition** {
2.     for z = 2 to b {    //b is the max number of branches.
3.         apply **node splitting method** to divide $N_{ij}^h$ into z classes.
4.         compute ***score***$(N_{ij}^h, z)$
           //evaluate how many branches are appropriate for node $N_{ij}^h$.
       }
5.     $z_{ij}^h = \arg\max_z score(N_{ij}^h, z)$
6.     apply ***K*-means clustering** to divide $N_{ij}^h$ into $z_{ij}^h$ classes
       // $N_{ij}^h$ is divided into, without loss of the generality, $N_{i+1,j}^h, \ldots, N_{i+1,z_{ij}^h}^h$.
   }
```

In the algorithm of the learning stage, we need to design three tasks: (i) the method for splitting nodes, (iii) the score function which evaluates how many branches for the node to split are appropriate, and (iii) the stop condition (step 1 in Table 5-1) which checks whether a node needs to be split. Denote the *j*-th node at level *i* of the tree for L_h, with loss of generality, as N_{ij}^{h}. Note that for a node N_{ij}^{h} we use the following notations: d_{ij} is the number of positive images in the node, $d_{ij}^{'}$ is the number of negative images in the node, and u_{ij} is the number of unlabeled images in the node.

Node splitting method

An unsupervised clustering is used to divide node N_{ij}^{h} into several classes. Here we adopted *K*-means clustering. To employ it in our work, an image should first be converted into be a vector. In this work, we adopt the smoothed visual-word-based image feature that have described in Chapter 4 to build the region-based representation for an image. Note that either features or unsupervised clustering method are independent of the proposed algorithm. Other methods of unsupervised clustering can also be used in this work, e.g., probabilistic Latent Semantic Analysis (pLSA) [Hofmann 99] [Sivic et al. 05] [Fergus et al. 05] or Latent Dirichlet Allocation (LDA) [Blei et al. 03] [Fei-Fei and Perona 05].

Score function

In order to decide how many branches to split a node, we define a score function to calculate a score for each of branch numbers, and compare the scores to choose the most appropriate number to split the node. Denotes the score of splitting node N_{ij}^{h}

with the branch number z as *score*(N_{ij}^{h} , z). Here, we hope the child nodes can either contain much more positive images than negative images that means this node can present a cluster of images associated with this label, or contain much more negative images than positive images that means this node can present a cluster of images not associated with this label. So we adopt entropy to measure the score for the splitting number. For a node N_{ij}^{h} splitting into z child nodes, we denote the z child nodes as $N_{i+1,j}^{h}, ..., N_{i+1,z}^{h}$. Consider a child node $N_{i+1,p}^{h}$ of node N_{ij}^{h}, we define:

$$E(N_{i+1,p}^{h}) = \text{entropy in } N_{i+1,p}^{h} = (-1)\times(\tau_{i+1,p}\log\tau_{i+1,p} + (1-\tau_{i+1,p})\log(1-\tau_{i+1,p})),$$
$$\text{where } \tau_{i+1,p} = \frac{d_{i+1,p}}{d_{i+1,p}+d'_{i+1,p}}, \tag{5.2}$$

and

$$score(N_{ij}^{h}, \text{z}) = \min_{1\le p\le z} E(N_{i+1,p}^{h}) . \tag{5.3}$$

In Equation (5.3), we use the minimal function because we expect that there exists at least one node with the best criteria in the next level of tree. Other nodes with worse entropy can be divided again. Thus, the best branch number for splitting node N_{ij}^{h} is

$$z^{*} = \arg\min_{2\le z\le b} Score(N_{ij}^{h}, \text{z}) . \tag{5.4}$$

Stop condition

A node containing consistent or unified information means that this node is high confident to classify data. Hence, a node should not be split if it only contains either positive or negative data. We define the stop condition of splitting a node as:

$$Stop(N_{i}^{h}) = \begin{cases} true, & \text{if } \frac{d_i}{d_i+d'_i} > H_S \text{ or } \frac{d'_i}{d_i+d'_i} > H_S \text{ or } (d_i+d'_i) < H_d \text{ ,} \\ false, & \text{otherwise} \end{cases} \tag{5.5}$$

where H_S and H_d are threshold values. If leaf nodes that do not fit the stop condition means that it contains a mixture of positive or negative images, then they need to be split in the algorithm (steps 2 to 6) of Table 5-1.

While dividing a node N_{ij}^h into z^* child nodes using the *K*-means clustering, the semantic label L_h can be grouped into z^* subclasses according to the positive and negative images in the node. A leaf node represents the positive or negative results associated with only single label (due to the stop condition) in a classifier.

5.3.3. Annotation stage

Given an unlabeled image d_{new} and the *H* classifiers C_1 to C_H that are learned by the procedure in Section 5.3.2, we denote $p(C_h = true \mid d_{new})$, where *h* is from 1 to *H*, as the confidence that label L_h can be used for annotating the image. Each node in hierarchical classifier C_h is a sub-classifier using *K*-means clustering. We then use the notation $p(C_h = true, N_{ij}^h \mid d_{new})$ as the confidence at the *j*-th node of *i*-th level in classifier C_h using *K*-means clustering. The confidence value $p(C_h = true, N_{ij}^h \mid d_{new})$ is computed by the propagation of classification results in nodes of each level. Then the computation of the confidence value can be recursively formulated as:

$$p(C_h = true \mid d_{new}) = p(C_h = true, N_{ij}^h \mid d_{new}), \tag{5.6}$$

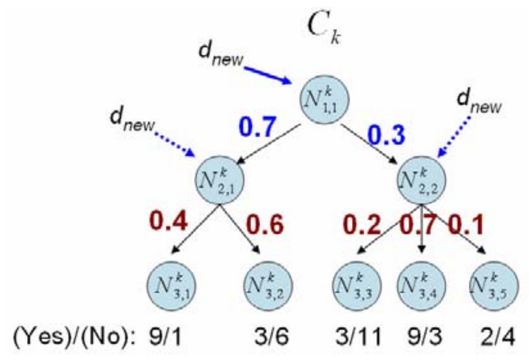
$$p(C_h = true, N_{ij}^h \mid d_{new}) = \begin{cases} \sum_{l=1}^{z_{ij}^h} p(C_h = true, N_{i+1,l}^h \mid d_{new}) \cdot p(N_{i+1,l}^h \mid d_{new}), & \text{if } N_{i+1,j}^h \text{ is not a leaf node}, \\ \dfrac{\#\text{ of images with label } L_h \text{ in node } N_{ij}^h}{|D_h|}, & \text{if } N_{i+1,j}^h \text{ is a leaf node} \end{cases} \tag{5.7}$$

and

$$p(N_{i+1,l}^h \mid d_{new}) = \frac{\left(\text{distance}(centroid(N_{i+1,l}^h), d_{new})\right)^{-1}}{\sum_{j=1}^{z_{ij}^h} \left(\text{distance}(centroid(N_{i+1,j}^h), d_{new})\right)^{-1}}, \tag{5.8}$$

where $\text{distance}(centroid(N_{i+1,j}^h), d_{new})$ is the Euclidean distance from the centroid of the node $N_{i+1,j}^h$ (by *K*-means clustering) to the image feature of d_{new}.

In order to annotate labels for a test image, we can choose first *Y* (equals to 6 in the experiment in Section 7.3.1) labels with the higher confidence values that are computed by Equations (5.6) and (5.7). Moreover, the proposed semantic-based image feature does not annotate images exactly but collect the confidence values of labels associated with the image, which are described in the next section.



$$p(L_h = Yes \mid d_{new}) = 0.7\times 0.4\times\frac{9}{26} + 0.7\times 0.6\times\frac{3}{26}$$
$$+ 0.3\times 0.2\times\frac{3}{26} + 0.3\times 0.7\times\frac{9}{26} + 0.3\times 0.1\times\frac{2}{26} = 0.227$$

Figure 5-4. Illustration of image annotation to compute the confidence that label L_h can be assigned to the new image d_{new}, where d_h=26.

Figure 5-4 takes an illustration for the annotation process. Suppose there be 26 positive-labeled images for learning the classifier C_h, i.e., d_h=26. The new image is tested to compute $p(C_h = true, N_{ij}^h \mid d_{new})$ at each internal node of the classifier for accumulating the confidence $p(C_h = true \mid d_{new})$ in Equation (5.6) and (5.7).

5.4. Image Representation

Given an image *I*, the semantic-based image feature is defined as *H* dimensions that means the *H* labels are used to describe the contents of image *I*. Hence, we collect all the confidence values of annotation for image *I* to be the semantic-based image feature, denoted as $\mathbf{v}_s = \{v_1, ..., v_H\}$, where

$$v_h = p(C_h = true \mid I), \ 1 \le h \le H , \tag{5.9}$$

is computed by Equations (5.6) and (5.7). Note that the collection of confidence values can record all of the information of semantic annotation in order to avoid the effect of the inaccurate classifications.