# MULTICLASS RECOGNITION WITH MULTIPLE FEATURE TREES

Guan-Lin Li, Jia-Shu Wang, Chen-Ru Liao, Chun-Yi Tsai, and Horng-Chang Yang

Department of Computer Science and Information Engineering,
National Taitung University, Taiwan, R.O.C.
{u10011135,u10011147,u10011121}@ms100.nttu.edu.tw;
{cytsai, hcyang}@nttu.edu.tw

***ABSTRACT***

*This paper proposes a multiclass recognition scheme which uses multiple feature trees with an extended scoring method evolved from TF-IDF. Feature trees consisting of different feature descriptors such as SIFT and SURF are built by the hierarchical k-means algorithm. The experimental results show that the proposed scoring method combing with the proposed multiple feature trees yields high accuracy for multiclass recognition and achieves significant improvement compared to methods using a single feature tree with original TF-IDF.*

***KEYWORDS***

*SIFT, SURF, K-means, TF-IDF*

## 1. INTRODUCTION

For machine intelligent applications, one of the critical issues is to achieve high accuracy in multi-object recognition. This paper, motivated by Nister's previous work[3], proposes an algorithm comprising multiple hierarchical k-means feature trees with an improved TF-IDF[4] scheme. The TF-IDF scheme takes account of not only the occurrence frequency of an item / feature but also the inverse of the frequency of documents containing the item/features to the total document. Such an item/feature is apparently good for distinguishing objects from multiple classes. In our study, the proposed method with improved TF-IDF scheme and multiple feature trees significantly improves the accuracy of recognizing multiple objects.

## 2. RELATED RESEARCH

For decades, feature descriptors are widely used in image matching and object recognition applications. One of the classical descriptors is the state-of-the-art SIFT[2] scheme. In the SIFT scheme, local features are detected and extracted by looking for optima in the pyramid of scale spaces generated by the difference of Gaussian (DoG) method. Each SIFT feature consists of orientations computed based on the local image gradient directions around a detected optimum. As the SIFT descriptor is proved efficient and effective for image matching, object recognition, motion tracking, and related fields in machine intelligence, various adaptations are proposed Similar to SIFT, the SURF scheme proposed by Bay[5] generates the pyramid of scale spaces by discrete wavelet transform and approximates the determinant of Hessian blob detector by an integer evaluation to save computing cost. The two feature descriptors are adopted in the paper for object representation. The vocabulary tree proposed by Nister[3] is to categorize all training features into hierarchical clusters by k-means. In the progress of tree construction, the training set is divided into a certain number of clusters in each level. The aim of hierarchical clustering by k-means is not only to enhance the distinctiveness among cluster of features, but also save the searching cost in classification time with TF-IDF scoring for testing data.

## 3. THE PROPOSED METHOD

### 3.1. Build the Feature Tree

The feature tree adopted in the proposed method is constructed by hierarchical k-means. The basic operation of k-means, illustrated by Figure 1, is started by choosing k (in this case k=3) feature points randomly as initial cluster centroids. Each feature point is assigned to the cluster whose centroid is closest to it. Then, each centroid is recomputed. The process iterates until, the difference of consecutive positions for each centroid converges or a termination condition is reached. After finishing k-means clustering in the first layer of the hierarchical tree, it continues to apply k-means clustering with the identical degree in successive layers to construct the hierarchical feature tree, as illustrated in the Figure 2 and Figure 3. Apparently each cluster, represented by its centroid, is a node in the feature tree. A node stop growing if the number of its members is less than a specific value or its level reaches a default upper bound. The detail algorithm for constructing a feature tree is shown in Table 1.

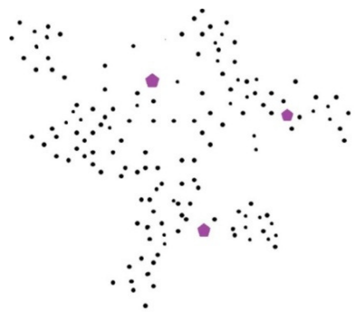

Figure 1. Randomly choose three features as initial cluster centroids.

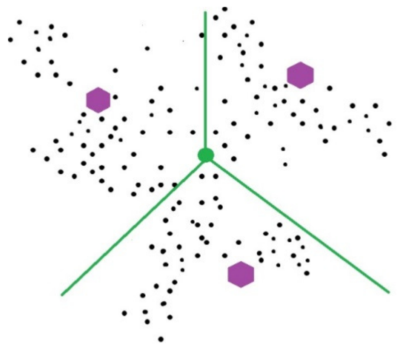

Figure 2. Layer one clustering.

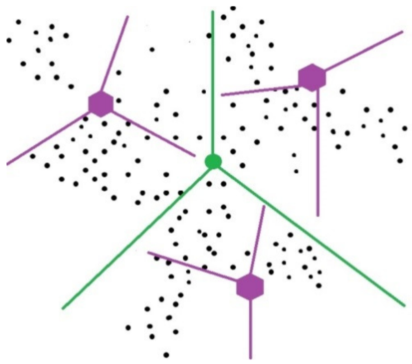

Figure 3. Layer two clustering.

Table 1. Build the Feature Tree

```
Input :
1.  P: input feature data
2.  H: number of tree layer
3.  T: initial feature tree
Output :
1.  hktree: the output feature tree
Temporal:
1.  Hktree : type definition of the feature tree, owning clustering function to save
    center or leaf node data
2.  C : the set of cluster centers
3.  Kmeans(P) : k-means processing for input data and returning centers
4.  A(C,P) : assigning data to the center closest to it, and returning data clusters
--------------------------------------------------------------------------------
Hktree HKTree(P,H,T){
    Hktree hktree
    C = Kmeans(P)
    hktree.group = A(C,P)
    IF P.size > limit or T.layer < limit {
        T→next = hktree
        HKTree(hktree.group,H+1,T→next)
    }
    Else
        Return hktree
}
```

## 3.2. Searching

To search an image in an image base, the image is represented by a features vector which is compared with the features of the feature tree built in the last section as the inverted-index file of the image base. Each feature is categorized to a cluster by following the path from the root to the leaf node, of which each node has the closest distance to the input feature.

## 3.3. Scoring

### 3.3.1. Weighted TF-IDF

The first attempt that we adopt for scoring is the weighted TF-IDF approach based on the original TF-IDF scheme. The scoring policy is defined as follows. Let $f_i$ denote the total number of features that reaches the leaf node $i$ from an input image. Let $M$ denote the total number of image classes, $d_{ij}$ denote the number of all features of a specific image class $j$ clustered in the leaf node $i$, $m_i$ denote the number of all features clustered in the leaf node $i$, $N$ denote the total number of leaf nodes, and $n_j$ denote the total number of leaf nodes containing the specific image class $j$. Then $d_{ij}/m_i$ is the weighting value, and the natural log of $N/n_j$ is the IDF value. Let $Score[j]$ denote the score of the image class $j$. The searching algorithm along with the scoring policy defined in eq. (1) is shown in Table 2. After each feature extracted from an input image reaches the leaf node that it belongs to, the total score for each candidate image class can be evaluated respectively.

Table 2 : Algorithm for searching and scoring

```
FeatureTree Tree
Queue Layer=Tree.root
Queue Feature = TheSetofTotalPoints
DO{
    Node node = Layer.dequeue()
    Point points = Feature.dequeue()
    IF node != leaf_node {
        Assign points to k childs of node
        FOR i=1 to k{
            Layer.enqueue(childs[i])
            Feature.enqueue(points[i])
        }
    }
    ELSE{
        FOR i=1 to N
            FOR j=1 to M
                $\boldsymbol{Score[j] += f_i \times \frac{d_{ij}}{m_i} \times ln\left(\frac{N}{n_j}\right)}$    (1)
    }
}WHILE node is not null
```

### 3.3.2. VT(vocabulary tree)

The vocabulary tree(VT)[3] uses the total number of features that reaches the leaf node $i$ from an input image as the TF value, denoted by $f_i$. Let $D$ denote the total number of all features in the dataset. Then, the second scoring policy adopted for experiment is defined as in eq. (2).

$$\boldsymbol{Score[j] += f_i \times d_{ij} \times ln\left(\frac{D}{m_i}\right)} \quad (2)$$

### 3.3.3. Proposed Scoring Method

The proposed method for scoring adopts identical TF definition in method (a), but improves the IDF. Let $D_j$ denote the total number of features of a specific image class $j$. Then the improved IDF is defined as the natural log of $D/D_j$ to imply that classes with rare features are important. That is, the scoring policy is defined as in eq. (3). The design concern for the proposed scoring method is to generally consider the impact of rare features from the aspect of the whole dataset.

$$\boldsymbol{Score[j] += f_i \times \frac{d_{ij}}{m_i} \times ln\left(\frac{D}{D_j}\right)} \qquad (3)$$

## 3.4. Proposed Multiple Tree Searching

With the three scoring policies, we design two additional multiple-feature-trees schemes, SIFT⊕ SURF and SIFT→SURF, to improve the image retrieval accuracy. The scheme denoted by SIFT ⊕SURF, is to do score normalization for the SIFT and SURF trees, respectively, and sum up the normalized scores as the final score. The scheme denoted by SIFT→SURF, is designed for cascade searching. It represents that the search process on SURF tree activates only if the search result on SIFT tree is unmatched. The reason behinds this scheme is to improve the resistance to distortion of feature representation; i.e., the complementary of distinct types of feature facilitates the enhancement of distinctiveness among different classes and promotes similarity among identical ones.

## 4. EXPERIMENTS AND RESULTS

In this paper, the INRIA Holiday dataset[1] is used for experiment. We pick 612 images from 119 image classes. Each class contains at least 4 images. One image in each class is used as test data, and the other 493 images are used for training data. Two well-known feature extraction schemes, SIFT and SURF, are applied to the training data to build a SIFT feature tree and a SURF feature tree by hierarchical k-means, respectively. For each input image, the six highest scores images are outputted. If the test image indeed belongs to any one of the six image classes, we call it "matched". Otherwise, it is unmatched. The accuracy for this experiment is defined as the mean matching rate for all the test data. The experiment shows, as in Table 3, the performance of SIFT ⊕SURF using the proposed scoring method yields higher accuracy compared to the other two scoring methods. Although it is slightly lower than those of applying the first two scoring methods using a single SIFT feature tree, it is resulted from averaging with the low scores from SURF. This issue might be alleviated by adjust the dimension of SURF features. The performance of SIFT→SURF as shown in Table 3, Figure. 4, and Figure. 5 is superior to all other combings of feature trees and scoring methods.

Table 3. Mean accuracy of different weighted scoring method and feature trees schemes

| | **Weighted TF-IDF** | **VT** | **Proposed Scoring Method** |
|---|---|---|---|
| SIFT | 0.605042 | 0.613445 | 0.739496 |
| SURF | 0.336134 | 0.327731 | 0.436975 |
| Proposed **SIFT⊕SURF** | 0.588235 | 0.563025 | 0.739496 |
| Proposed **SIFT→SURF** | 0.638655 | 0.672269 | 0.781513 |

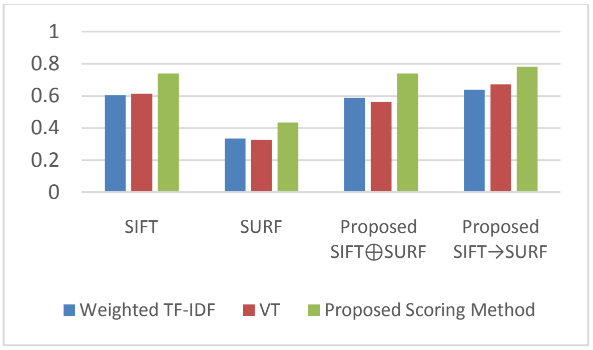

Figure 4. Mean accuracy of feature trees when selecting different scoring methods.

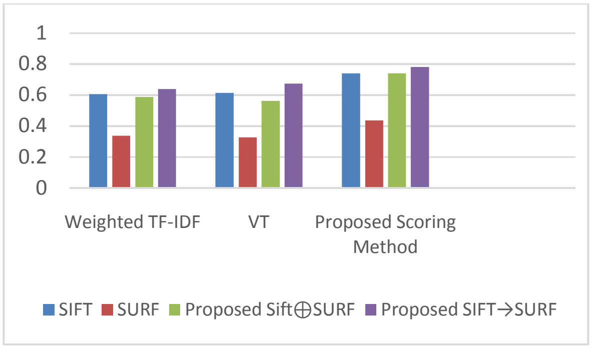

Figure 5. Mean accuracy of scoring methods when applying to various feature trees.

## 5. CONCLUSIONS

Inspecting Figure. 4 and Figure. 5, we can see that applying the three scoring methods on SIFT feature tree yields acceptable results. The outcomes show that although the performance of the single SURF feature tree might be insufficient, the proposed combination of SIFT and SURF trees, SIFT$\rightarrow$SURF, with proposing scoring method outperforms the other methods in the experiment. In addition, we observed that the scoring method has the crucial impact on accuracy, and the selection of feature tree scheme also affects the improvement of accuracy. Besides, due to a certain portion of similarity of SIFT and SURF algorithms, it can be expected that there still exists rooms for improvement if more heterogeneous features descriptors, such as HOG[6], DAISY[7], and covariance[8], are applied.

## REFERENCES

[1] Hervé Jégou, Matthijs Douze, Cordelia Schmid. "Hamming embedding and weak geometric consistency for large scale image search", European Conference on Computer Vision, 2008.

[2] Lowe, D. G., "Distinctive Image Features from Scale-Invariant Keypoints", International Journal of Computer Vision, 60, 2, pp. 91-110, 2004.

[3] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), volume 2, pages 2161-2168, June 2006.

[4] Jones KS, A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation,1972.

[5] Bay, H. and Tuytelaars, T. and Van Gool, L. "SURF: Speeded Up Robust Features", 9th European Conference on Computer Vision, 2006

[6] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. CVPR, 2005.

[7] Engin Tola, Vincent Lepetit, Pascal Fua. IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. 32, Nr. 5, pp. 815 - 830, May 2010.

[8] Oncel Tuzel, Fatih Porikli, and Peter Meer, Pedestrian Detection via Classification on Riemannian Manifolds, IEEE Transactions on Pattern Analysis and Machine Intelligence, Oct. 2008.