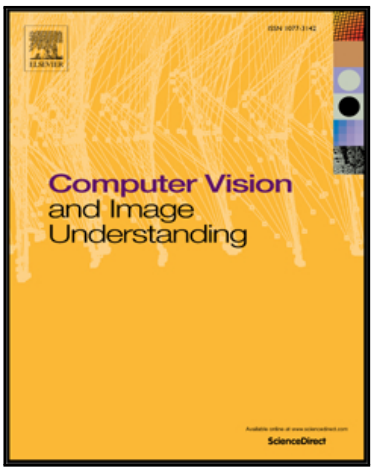
Accepted Manuscript



Efficient Large-Scale Multi-Class Image Classification By Learning Balanced Trees

Tien-Dung Mai, Thanh Duc Ngo, Duy-Dinh Le, Duc Anh Duong, Kiem Hoang, Shin'ichi Satoh

PII: S1077-3142(16)30163-1
DOI: 10.1016/j.cviu.2016.10.008
Reference: YCVIU 2496

To appear in: *Computer Vision and Image Understanding*

Received date: 15 December 2015
Revised date: 28 September 2016
Accepted date: 19 October 2016

Please cite this article as: Tien-Dung Mai, Thanh Duc Ngo, Duy-Dinh Le, Duc Anh Duong, Kiem Hoang, Shin'ichi Satoh, Efficient Large-Scale Multi-Class Image Classification By Learning Balanced Trees, *Computer Vision and Image Understanding* (2016), doi: 10.1016/j.cviu.2016.10.008

This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

ACCEPTED MANUSCRIPT

Highlights

- We propose a label-tree based method for efficient large-scale multi-class image classification.
- The proposed method learns effective and balanced trees by jointly optimizing balance and confusion constraints.
- We carried out extensive experiments to compare our method to other state-of-the-art methods on four large-scale datasets.

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

Efficient Large-Scale Multi-Class Image Classification By Learning Balanced Trees

Tien-Dung Mai[a], Thanh Duc Ngo[a], Duy-Dinh Le[a,b,*], Duc Anh Duong[a], Kiem Hoang[a], Shin'ichi Satoh[b]

[a] *University of Information Technology, VNU-HCM, Ho Chi Minh City, Vietnam*
[b] *National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan*

Abstract

Large-scale multi-class image classification is essential for big data applications. One of the challenges is to deal with situations in which the number of classes is very large and for which the standard one-versus-all method is not appropriate because its computational complexity is linear in the number of classes. Using a label tree is a popular way to reduce complexity. By organizing classes into a hierarchical structure, the number of classifier evaluations of a test sample when traveling from the root node to a leaf node is significantly reduced. Having a balanced learned tree is essential to this approach. The current methods for learning the tree structure use clustering techniques, such as k-means or spectral clustering, to group confusing classes into clusters associated with the nodes. However, the output tree in such cases might not be balanced. In this paper, we propose a method for learning effective and balanced trees by jointly optimizing balance and confusion constraints. Experimental results on large-scale datasets including Caltech-256, SUN-397, ILSVRC2010-1K, and ImageNet-10K, show that our method outperforms other state-of-the-art methods.

Keywords: Large-Scale Image Classification; Multi-Class Classification; Label Tree Learning; Balanced Tree; Hierarchical Classification; Clustering

*Corresponding author
Email addresses: dungmt@uit.edu.vn (Tien-Dung Mai), thanhnd@uit.edu.vn (Thanh Duc Ngo), ledduy@nii.ac.jp (Duy-Dinh Le), ducda@uit.edu.vn (Duc Anh Duong), kiemhv@uit.edu.vn (Kiem Hoang), satoh@nii.ac.jp (Shin'ichi Satoh)

ACCEPTED MANUSCRIPT

1. Introduction

The generation, storage, and sharing of multimedia data are increasing at explosive rates. Research on image and video understanding is becoming crucial for many applications such as object categorization, scene classification, and semantic image retrieval [1–15]. In this paper, we study the problem of multi-class image classification regarding the task of classifying an image into one of a set of pre-defined classes. The challenge is how to handle a situation in which there is a very large number of classes.

One standard method of multi-class classification is to use multiple binary one-versus-all (OvA) classifiers [10, 16]. However, this method is not scalable to large-scale datasets (e.g., ImageNet [12] includes 21,841 concepts) because all classifiers have to be evaluated at runtime for every test image.

One popular method for reducing testing complexity is to use hierarchical structures of classes (i.e., label trees) [8, 14, 17–20]. In a label tree, the label of a test sample is assigned by traversing the tree from the root node to a leaf node. At each node visited, several classifiers (the number of classifiers is usually equal to the number of child nodes) are used to compute scores for determining which branch to follow. The tree structure causes the classification complexity to increase logarithmically, rather than linearly, with the number of classes.

Testing efficiency of the label tree methods, measured in terms of the average number of operations needed for one test sample, is maximized when the tree is balanced. This is because, in a balanced tree, a path from the root to a leaf node is shorter than those of the unbalanced tree structures on average [15, 17–19, 21].

The label tree method entails two tasks: learning the tree structure and learning classifiers at nodes. The current methods [17–19, 21] either separate or combine these two tasks in an optimization framework. Although the combining methods usually have higher classification accuracy, they are too costly because the node classifiers have to be trained multiple times until the solutions converge. Hence, in this study, we chose to separate the two tasks, as Bengio et al. discussed [17], and focus on the first task, i.e., learning the tree structure.

ACCEPTED MANUSCRIPT

Given a set of class labels at each node (the root node contains all class labels, and each leaf node contains a single class label) and a number of branches k, the task is to split these labels into k groups. There are two important constraints: *(i) a confusion constraint such that confusing class labels (i.e., highly similar classes) should be in the same group and (ii) a balance constraint such that the numbers of class labels of the branches should be equal.* The first constraint is for reducing the tree loss and easily learning node classifiers. The second constraint is for balancing the tree, which might help to reduce computational complexity in the testing stage.

One solution is to use clustering methods such as k-means and spectral clustering [22]. For example, Bengio et al. [17] used the confusion matrix to measure confusion and spectral clustering to recursively split a set of class labels into disjoint groups. Because the objective function of spectral clustering penalizes unbalanced partitions, balanced trees are expected to be obtained, resulting in sub-optimal test efficiency. However, this method is not reliable because it is assumed that the confusion matrix can be estimated using OvA classifiers as surrogate classifiers. In practice, this assumption does not always hold; it may be violated when the binary OvA classifiers have poor accuracy due to a small number of available training samples and the curse of dimensionality. Another method involves performing k-means clustering on training samples [23]. With this method, the mean of all feature vectors of the training samples of a class is used as a representative of that class. Such a representation implicitly enforces the balance constraint when it is used with k-means clustering. However, using the mean is not effective for classes with large variations, and the resulting tree is not always balanced.

In this paper, we propose a method for learning effective and balanced trees by jointly optimizing the balance and confusion constraints. We formulate the learning tree structure problem within an optimization framework in which the balance constraint is solved using integer linear programming and the confusion constraint is solved using a variant of k-means clustering. Specifically, for the confusion constraint, we introduce a new minimized function that takes into

ACCEPTED MANUSCRIPT

account advantages of using only one mean feature vector and all feature vectors for each class. The resulting tree is balanced and its classification accuracy is improved. We carried out experiments on several benchmark datasets such as Caltech-256, SUN-397, ILSVRC2010-1K, and ImageNet-10K, and the results show the superiority of our method over other state-of-the-art methods.

The rest of the paper is organized as follows. We discuss related work in Section 2 and describe the proposed method in Section 3. We explain the experimental results in Section 4 and conclude the paper in Section 5.

2. Related Work

One of the critical challenges in large-scale classification is how to balance classification accuracy and testing efficiency. Flat methods, e.g., OvA [16] or directed acyclic graph [24], have good classification accuracy. However, the computational cost is prohibitive for practical applications because all OvA classifiers have to be evaluated for each test sample.

Error-Correcting Output Coding (ECOC)-based methods are alternative ways of reducing testing complexity. These methods mainly involve designing an optimal coding matrix that requires a small number of bits for efficiency, good row and column separation for robustness, and highly accurate bit predictors. Sparse random codes and dense random codes [25, 26] usually require a large number of bit predictors (respectively $15.log(N)$ and $10.log(N)$, where N is the number of classes) to achieve reasonable accuracy. However, the accuracy of these methods is worse than that of the OvA method [16]. Passerini et al. [27] proposed a new decoding method that combines the margins through an estimate of their class conditional probabilities. Spectral ECOC [28] is based on a spectral decomposition on the normalized Laplacians of the similarity graph of the classes. The resulting eigenvectors are used to define partitions. Because it uses one-versus-one classifiers to generate the similarity matrix, it is not scalable to a large number of classes. Sparse Output Coding (SpOC) [29] is a new encoding and decoding scheme that learns a coding matrix and bit predictor

ACCEPTED MANUSCRIPT

separately but strikes a good balance between error-correcting ability and bit prediction accuracy. However, it requires a predefined class taxonomy to build a semantic relatedness matrix for both stages; such a taxonomy may not always be available.

Hierarchical methods, e.g., ones using a label tree [15, 17–19, 23, 30], are more popular because they are able to lower testing complexity sub-linearly with the number of classes while maintaining reasonable classification accuracy. The key idea of the label tree method is to organize classes into a tree in which each leaf node is associated with a class and each internal is associated with a subset of classes. In the testing process, test examples traverse the tree from the root node until a leaf node is reached. In the ideal case, the number of classifier evaluations on the path is logarithmic with the number of classes. The label tree methods involve two tasks, i.e., learning the tree structure and learning node classifiers.

Bengio et al. [17] proposed a method that uses spectral clustering for learning a tree structure. It has a confusion matrix that is generated by applying OvA classifiers to a validation set as an affinity matrix to split classes into disjoint subsets. Each subset corresponds to a child node of the tree. Because spectral clustering does not encourage unbalanced partitions, the result is implicitly a balanced tree.

Deng et al. [18] proposed a method that jointly performs class partitioning and classifier learning for each child node. The OvA training step is no longer needed. Learning the classifier weights and determining the partitions are formulated as an optimization problem. The problem is solved by alternating between two optimization steps. However, by allowing overlapping of classes among child nodes to reduce false navigation, it increases the testing cost; thus, it cannot be ensured to produce the desired speed increase.

The relaxed hierarchy method proposed by Gao and Koller [21] is an alternative based on max-margin optimization in which a subset of confusing classes is allowed to be ignored at each node. This method shares the idea of the method proposed by Marszalek and Schmid [31], but has significant improvements over

ACCEPTED MANUSCRIPT

it. However, learning complexity increases if there are more than two branches at each node.

Liu et al. [19] recently proposed a probabilistic approach for learning a tree structure. Each node of the probabilistic label tree is associated with a categorical probability distribution and a maximum likelihood classifier defined as a multinomial logistic regression model. The training process at each node is formulated as a maximum optimization of a log likelihood function, which is then solved by using alternating convex optimization.

Zhu et al. [20] proposed a strategy for selecting the branch to follow. Instead of taking the branch with the highest classification response, a new error recovery scheme is used to verify the consistency of the responses of the candidate nodes and their child nodes. It helps to increase classification accuracy, but adds to the cost of training and testing.

Of the other methods proposed for solving the problem of large-scale image classification [32–36], most use the OvA method because of its competitive performance and easy parallelization on multicores or machines. For example, Lin et al. [33] described a method for fast feature extraction and support vector machine (SVM) learning for OvA classifiers. Several studies tried to achieve higher accuracy by simultaneously learning shared characteristics common to the classes and minimizing the classification loss by using the trace-norm [10, 32, 34, 35]. The method called Sparselets [36] learns a shared intermediate representation for multi-class object detection with deformable part models using sparse reconstruction of object parts.

3. Our Learning Framework

3.1. Overview of Label Trees

A label tree is a hierarchical structure composed of a set of class labels $L = \{1, .., N\}$. In a Q-way label tree, each node v has at most Q children and a subset of the class labels $\ell(v)$, where $\ell(v) \subseteq L$. The set of children of v is denoted as $\sigma(v) = \{\zeta_1, .., \zeta_Q\}$.

ACCEPTED MANUSCRIPT

The tree is usually built by recursively applying a clustering algorithm to split a set of class labels at each node into clusters. One cluster of class labels corresponds to a child node. Highly confusing classes (i.e., very similar classes such as Pug dog and Hunting dog) are grouped into the same child node. This process is done in a top-down fashion from the root of the tree to its leaves. The root includes all class labels. However, one leaf contains only one class label.

Given a label tree T, a new sample x is classified by traversing the tree from its root to a leaf node whose label is assigned to x as the classification result. At a v along the path, there are $|\sigma(v)|$ possible branches to traverse in the next step. A branching decision at a v is made by applying $|\sigma(v)|$ OvA binary classifiers of v to the feature vector of x. The branch with the largest classification response value is selected. Because only classifiers of nodes along the path are evaluated, the testing complexity is sub-linear to the total number of class labels, $|L|$. Particularly, this complexity is $log(|L|)$ if T is a balanced tree.

3.2. Optimization

As discussed in Sections 1 and 2, balance and confusion constraints impact the final performance. A balanced tree leads to a lower computational complexity in testing. Meanwhile, classification accuracy is strongly affected by the grouping of confusing classes in nodes. Here, we incorporate both constraints in one optimization problem in order to learn an accurate and balanced tree.

We denote a Q-way balanced tree with a maximum depth H as $T_{Q,H}$. The depth of a node is defined as the number of nodes along the path starting from the root of the tree to that node. The depth of the root is zero. Given $T_{Q,H}$ and N classes, we have that H is at most $log_Q(N)$ and the number of classes of each child of a v is at most $P(v)_{max}$, where $P(v)_{max} = Q^{h(v)-1}$ with $h(v)$ is the maximum depth of a subtree having a root at v, $h(v) = log_Q(|\ell(v)|)$. For example, a 32-way balanced tree for $N = 1,000$ classes should have a maximum depth of $H = log_{32}(1,000) \approx 2$ and a maximum of $P(root)_{max} = 32$ classes at any child node of the root.

ACCEPTED MANUSCRIPT

Let $S(v)_{|\ell(v)|\times|\sigma(v)|}$ denote a matrix whose element $S(v)_{i,j}$ represents whether the ith class is associated with the jth child node. We have $S(v)_{i,j} = 1$ if $c_i \in \ell(v)$ and $c_i \in \ell(\zeta_j), \zeta_j \in \sigma(v)$; otherwise, $S(v)_{i,j} = 0$. Then, the constraint that each class label only belongs to one child of v can be described as

$$\sum_{j=1}^{|\sigma(v)|} S(v)_{i,j} = 1, \tag{1}$$

and the constraint that each child node of v has the maximum $P(v)_{max}$ class labels can be stated as:

$$\sum_{i=1}^{|\ell(v)|} S(v)_{i,j} \leq P(v)_{max}. \tag{2}$$

The constraints (1) and (2) lead to a balanced tree. It is also important to have a confusion constraint to ensure that confusing classes are grouped in the same nodes. This reduces tree loss and improves classification accuracy.

One simple way to model a confusion constraint is to use the sum of distances between each sample and the cluster center and solve the problem by using algorithms such as k-means clustering [15]. However, if there are large variations in the samples, a situation may occur in which certain samples of one class belong to several clusters. A voting strategy can be used to select the most appropriate cluster that contains the majority of samples of one class.

We extend the aforementioned confusion constraint by using prior knowledge, i.e., the class label of the samples within a cluster. Specifically, we propose to use a combinatory distance based on two principle distances: *i) the distance from each sample of a class to the cluster center and ii) the distance from the mean sample (i.e., the mean of the feature vectors of all samples) of a class to the cluster center.* Here, by taking into account the distances from the samples to the cluster centers, we could preserve the key characteristics and individual samples of the class at the expense of sensitivity to outliers. On the other hand, while using the sample mean can deal with noise, it ignores the key characteristics and samples of the class. Combining these complementary principle distances is expected to result in a distance that is robust against variations in the distribution of the classes. An illustration is shown in Figure 1.

ACCEPTED MANUSCRIPT

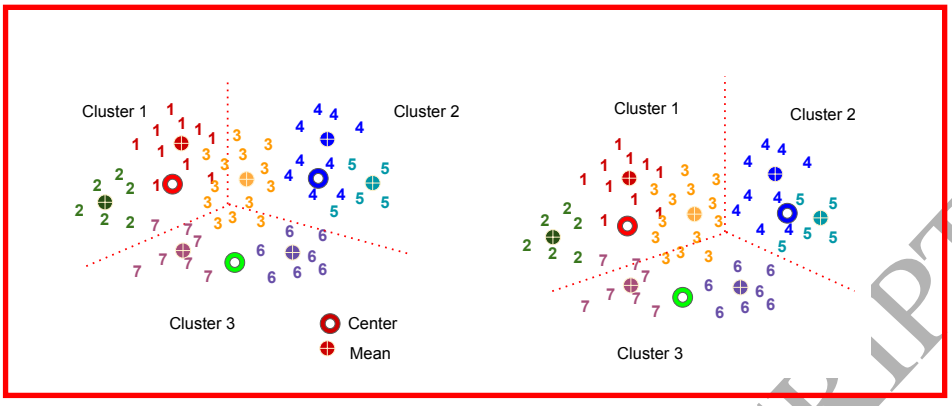


Figure 1: Example in which class 3 is erroneously assigned to cluster 2 (left) when the prior knowledge of the class labels of the samples is not used as in [15]. In contrast, our combinatory distance for the confusion constraint properly assigns class 3 to cluster 1 (right).

Let $\tilde{C}_j$ be the center of the jth cluster and $d(f_{i,k}, \tilde{C}_j)$ be a function that measures the distance from a feature vector $f_{i,k}$ of an image $x_{i,k}$ to a center $\tilde{C}_j$, where i and k are the index of a class and that of an image of that class, respectively.

Furthermore, let $F(v)_{|\ell(v)|\times|\sigma(v)|}$ denote a matrix whose element $F(v)_{i,j}$ is the combinatory distance between the ith class and j^{th} center. The distance is computed as

$$F(v)_{i,j} = \frac{1}{n_i} \sum_{k=1}^{n_i} d(f_{i,k}, \tilde{C}_j) + d(\tilde{f}_i, \tilde{C}_j), \tag{3}$$

where n_i is the total number of feature vectors of images (i.e., samples) of the ith class and the mean feature vector (i.e., mean sample) $\tilde{f}_i = \frac{1}{n_i} \sum_{k=1}^{n_i} f_{i,k}$.

Learning a balanced tree is equivalent to finding $S(v)_{|\ell(v)|\times|\sigma(v)|}$ and $F(v)_{|\ell(v)|\times|\sigma(v)|}$ such that the sum of $F(v)$s is minimized. This problem can be formulated as a minimum optimization problem with two variables $S(v)$ and $F(v)$:

$$\min_{S(v),F(v)} \sum_{i=1}^{|\ell(v)|} \sum_{j=1}^{|\sigma(v)|} S(v)_{i,j} \cdot F(v)_{i,j}, \tag{4}$$

subject to the balance constraint (1) and confusion constraint (2).

Problem (4) can be solved using an alternating optimization strategy. First, fix $F(v)$; problem (4) is an integer linear programming problem for the integer

ACCEPTED MANUSCRIPT

variable $S(v)$ subject to constraints (1) and (2) and coefficients $F(v)$. Next, update the cluster centers using the non-zero values in the columns of $S(v)$ with fixed $S(v)$. Accordingly, obtain $F(v)$ by applying Equation (3). This optimization is repeated for a fixed number of iterations t or until the solution is converged (i.e., when the assignments no longer change).

Algorithm 1 is the procedure for splitting the set of class labels and balancing the number of class labels for each v, starting from the root node.

Algorithm 1 $[A] = SplittingBalancing(\ell(v), X, Q, P(v)_{max}, t)$: for splitting the set of class labels $\ell(v)$ into Q child nodes and balancing the number of class labels of children

Input:

1: $\ell(v)$: the set of class labels of v;

2: $X = \{(x_i, y_i)\}$: the set of feature vectors of images in training set with $\cup y_i = \ell(v)$;

3: Q: the desired number of children of v;

4: $P(v)_{max}$: the maximum number of class labels in a child node;

5: t: the number of iterations;

Output:

6: Set $A = \{a_1, .., a_N\}$ contains information about the assignment of class labels to Q child nodes. $a_i = k$ means that class label $c_i \in \ell(v)$ is assigned to child node k.

7: **Step 1:** Initialize Q cluster centers: $C_Q = k\text{-means}(X, Q)$.

8: **Step 2:** Obtain $F(v)$ from Equation (3).

9: **Step 3:** Fix $F(v)$, solve (4) for $S(v)$, and update centers C_Q

10: **Step 4:** Repeat Step 2 and 3 until the solution of (4) is converged or t is reached.

Several techniques can be used for the initialization (i.e., Step 1 of Algorithm 1). The simplest one is to randomly select Q initial cluster centers. Another is to use the outputs of a clustering algorithm. For a stable initialization, we decided to use k-means instead of random selection. Given multiple training

ACCEPTED MANUSCRIPT

images for each class, k-means can be applied to all samples of the classes or to only the mean samples of the classes. The experimental results indicated that using all samples yielded higher accuracy than using the means.

4. Experiments

The experiments described below compared our method with other state-of-the-art methods.

4.1. Datasets

We conducted experiments on several benchmark datasets including Caltech-256 [37], SUN-397 [5], ILSVRC2010-1K [12], and ImageNet-10K [7]. These datasets are widely used for large-scale image classification.

- **Caltech-256** [37] is a popular dataset for multi-class object classification. It contains 29,780 images in 256 object classes and has at least 80 images per class. There are many images in which objects appear in different illumination conditions, poses, and backgrounds. The sizes and resolutions of the images also vary.
- **SUN-397** [5] is a subset of the SUN dataset. It was selected from 908 scene classes of a large-scale scene recognition problem. There are 108,754 images in 397 classes, at least 100 images per class. The collection covers a large variety of environmental scenes, places, and objects therein.
- **ILSVRC2010-1K** [12] is used in ILSVRC 2010, a large-scale image classification challenge. It includes 1,461,406 images in 1,000 classes that cover a wide range of visual concepts including animals, plants, artifacts, people, activities, and events. Each class contains at least 668 images.
- **ImageNet-10K** [7] contains 9M images in 10,184 classes. Each class contains at least 200 images.

ACCEPTED MANUSCRIPT

4.2. Experimental Setting

The original SUN-397 and Caltech-256 datasets were split into three disjoint subsets as follows: 50% of the images were randomly selected for training, 25% were for validation, and the remaining were for testing. The experiments on ILSVRC2010-1K used the settings provided by the organizer of the challenge; the development set consisted of one validation set with 50,000 images, 50 images for each class, and a test set with 150,000 images, 150 images for each class. The training set with 100,000 images was built by randomly picking 100 images from each class. In our experiments, the validation set was used to compute the confusion matrix, similar to what was done in [17]. The training set was used to train the OvA classifiers as well as classifiers at nodes.

To compare our method with the previous ones, such as those of Bengio et al. [17] and Deng et al. [18], we used the same feature settings as Deng et al. described [18]. Specifically, we extracted dense SIFT features using the VLFeat toolbox [38] for each image. These features were then encoded using the locality-constrained linear coding (LLC) approach [39]. With a codebook having 10,000 visual words and two-level spatial pyramid matching (SPM) [1] (i.e., 1×1 and 2×2 grids), the resulting feature vector of each image had 50,000 dimensions.

4.3. Evaluated Methods

This section briefly describes the methods that were used for comparison.

- Regrading the method by Bengio et al. [17], we trained N binary OvA classifiers independently, where N is number of class labels. These classifiers were applied to a validation set to compute a confusion matrix C and an affinity matrix $A = \frac{1}{2}(C + C^T)$. The tree structure was built using spectral clustering [22] and the affinity matrix A. The resulting tree was similar to the label tree using $Relaxation(1)$ described by Bengio et al. [17] in which node classifiers are optimized independently.

ACCEPTED MANUSCRIPT

- The fast-balanced tree method proposed by Deng et al. [18] combines the task of learning the tree structure and the task learning the classifiers in an optimizing framework. With this method, we used their reported results for comparison without re-implementation. We only compared their results on ILSVRC2010-1K because they did not run experiments on the other datatsets (i.e., SUN-397 and Caltech-256).

- The SVM tree method of Liu et al. [23] uses the mean feature vectors of classes to build the SVM tree. In particular, each class is represented by its corresponding mean vector. In the experiments, the k-means algorithm was used for partitioning the set of mean vectors into Q clusters to build a Q-way tree instead of the original binary tree as presented in [23].

- The ECOC-based methods we tested included Random Sparse Output Coding ($RSOC$), Random Dense Output Coding ($RDOC$) [26], and Spectral ECOC [28]. We used the ECOC library provided by Escalera et al. [40].

- We evaluated our proposed method without prior knowledge of the class labels of the samples [15]. And, we also reported the performance of the OvA method.

We used the LIBLINEAR library [41] to train the classifiers.

We measured the classification accuracy and increase in test speed (i.e., test speedup). These measures have been used in related classification studies on label trees [18, 19].

- *Classification Accuracy* (Acc) is measured as the number of correct predictions among the total number of predictions on the testing set. It is calculated as

$$Acc = \frac{1}{m}\sum_{i=1}^{m} fi(\hat{y}_i = y_i), \tag{5}$$

where m is the total number of images in the testing set and $fi(\hat{y}_i = y_i)$ is an indicator function. $fi(\hat{y}_i = y_i) = 1$ if the predicted class $\hat{y}_i$ and the given assigned class y_i are the same; otherwise, $fi(\hat{y}_i = y_i) = 0$.

ACCEPTED MANUSCRIPT

- *Test speedup* (S_{te}) measures the speedup of a label tree-based method relative to the speed of the OvA-based method. It is estimated as

$$S_{te} = \frac{N * m}{M}, \tag{6}$$

where N is the number of class labels, m is the total number of images in the testing set, and M is the total number of dot-product operations performed for classifying m images. A higher S_{te} implies that there are fewer classifiers evaluated in the classification. It also means that this method is more computationally efficient. For example, if S_{te} of a tree $T_{16,2}$ is 8.0, only $256/8 = 32$ linear classifiers, rather than the 256 of the OvA-based method, are used for classifying a testing image.

To analyze the stability of the experimental results, we repeated each experiment at least five times for each dataset with different randomly selected subsets. The reported performance are the average values of the five experiments and their standard deviations.

4.4. Results

Tables 1, 2, and 3 list the experimental results on ILSVRC2010-1K, SUN-397, and Caltech-256. Each column of the tables is associated with a different tree configuration. Each row shows the performance of one method. As we can see, the classification performance of a tree-based method depends on the tree configuration. Specifically, when the number of children per node of a tree decreases, the depth of the tree increases; this means that the number of OvA binary classifiers of one node decreases, which in turn increases S_{te}; however, accuracy increases as a result.

The experimental results essentially show that our method outperformed the tree-based methods of Bengio et al. [17], Deng et al. [18], and Liu et al. [23]. At the same classification accuracy *Acc*, our method is usually more efficient (i.e., higher S_{te}) than the other tree-based methods. Moreover, at the same test speedup level (S_{te}), our method achieved higher accuracy in most cases.

ACCEPTED MANUSCRIPT

Furthermore, with the number of bit predictors being equal to the number of classifiers evaluated for classifying a test sample using the label tree, the accuracies of the ECOC methods were worse than those of the label tree methods.

4.4.1. Results on ILSVRC2010-1K

Table 1 and Figure 2 show the experimental results on ILSVRC2010-1K. Here, our method significantly outperformed the other methods. For example, in the case of the tree $T_{32,2}$, there were approximately $32 * 2$ classifiers evaluated for a test image, our method achieved $S_{te} = 15.76 \pm, 0.02$, and the highest $Acc = 13.28 \pm 0.06\%$. The highest Acc of the ECOC methods was $3.48 \pm 0.20\%$, and that of the label tree-based methods was $12.52 \pm 0.08\%$ with $S_{te} = 15.77 \pm 0.06$. As the S_{te} increases from 15.76 to 33.33, Acc drops from $13.28 \pm 0.06\%$ to $11.20 \pm 0.05\%$, a 15.66% decrease in accuracy in return for a speedup of two times.

Figure 2 illustrates the correlation between Acc and S_{te} of the methods. Compared with Bengio et al.'s method, our method was two times more accurate at the same computational cost.

If we directly compare our method with the OvA-based method, its advantage is clear. For example, our method achieved an $S_{te} \approx 16$ with $T_{32,2}$. This indicates that it used only 64 classifiers for classifying a new test image. With this tree, Acc was $13.28 \pm 0.06\%$. However, the OvA-based method required $1,000$ classifiers to reach $20.45 \pm 0.24\%$ accuracy. In addition, as we change the configuration of the trees, the performances of the methods change accordingly. As the number of child nodes of non-leaf nodes of the tree increases (i.e., depth of the tree decreases), the accuracies of all of the label tree-based methods increase. The larger the number of child nodes of non-leaf nodes, the smaller the number of classes associated with each child node. Thus, classes in different nodes become more discriminative or less confusing. This helps to improve Acc. However, S_{te} decreases because more classifiers need to be evaluated.

ACCEPTED MANUSCRIPT

Table 1: Performance of methods on ILSVRC2010-1K.

Methods	$T_{32,2}$		$T_{10,3}$		$T_{6,4}$		$T_{4,5}$	
	Acc	S_{te}	*Acc*	S_{te}	*Acc*	S_{te}	*Acc*	S_{te}
Bengio et al.[17]	6.56	15.91	4.76	31.31	4.11	37.88	3.70	43.02
	±0.10	±0.03	±0.08	±0.45	±0.09	±0.40	±0.01	±0.32
Deng et al. [18]	11.9	10.3	8.92	18.20	5.62	31.3	NA	NA
Liu et al. [23]	12.52	15.77	10.63	30.73	9.74	37.99	9.06	41.19
	±0.08	±0.06	±0.07	±0.06	±0.08	±0.18	±0.06	±0.28
RDOC [26]	3.35	15.63	2.15	33.33	1.28	40.00	0.99	50.00
	±0.23	±0.00	±0.06	±0.00	±0.03	±0.00	±0.02	±0.00
RSOC [26]	3.48	15.63	2.18	33.33	1.34	40.00	1.06	50.00
	±0.20	±0.00	±0.04	±0.00	±0.05	±0.00	±0.03	±0.00
SpectralECOC[28]	3.33	15.63	1.62	33.33	1.48	40.00	1.19	50.00
	±0.12	±0.00	±0.17	±0.00	±0.19	±0.00	±0.25	±0.00
Ours [15]	13.14	15.77	10.74	33.33	9.85	42.24	9.61	50.06
	±0.04	±0.00	±0.07	±0.00	±0.09	±0.9	±0.13	±0.02
Ours	**13.28**	15.76	**11.20**	33.33	**10.39**	40.58	**10.09**	50.05
	±0.06	±0.02	±0.05	±0.00	±0.16	±0.67	±0.01	±0.01
OvA	20.45 ± 0.24							

ACCEPTED MANUSCRIPT

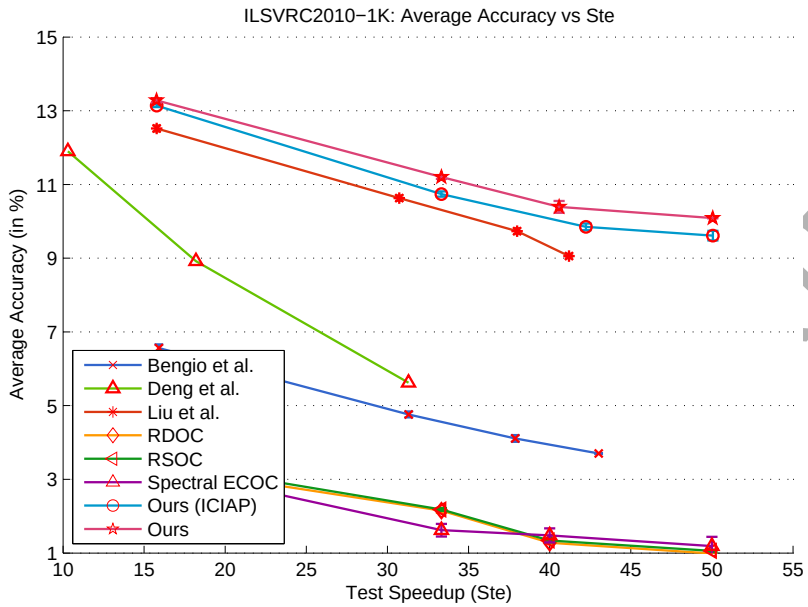


Figure 2: Performance of methods on ILSVRC2010-1K.

4.4.2. Results on SUN-397 and Caltech-256

Tables 2 and Figure 3 list the experimental results on the SUN-397 dataset. Similar to the results for ILSVRC2010-1K, the experimental results on SUN-397 demonstrate that our method achieved a higher *Acc* than those of the other methods for the same S_{te} in most cases. For example, consider the tree $T_{20,2}$; the best *Acc* of our method was $38.90 \pm 0.16\%$, whereas those of the methods of Bengio et al. [17] and Liu et al. [23] were $30.33 \pm 0.75\%$ and $38.29 \pm 0.01\%$. The other observations on ILSVRC2010-1K, SUN-397, and Caltech-256 (Table 3 and Figure 4) were consistent.

Figure 5 shows an example of $T_{4,4}$ in which the resulting tree generated by Bengio et al.'s method [17] is not balanced (the numbers of classes in the nodes are 82, 66, 52, and 56) (top). In addition, some classes (marked in red) such as class 248 (Node 1), class 142, and class 123 (Node 3) are incorrectly assigned (i.e., they are not visually similar to other classes in the nodes) and cause *Acc* to be poor. Meanwhile, the tree generated by our method is balanced

ACCEPTED MANUSCRIPT

Table 2: Performance of methods on SUN-397.

Methods	$T_{20,2}$		$T_{8,3}$		$T_{5,4}$		$T_{4,5}$	
	Acc	S_{te}	*Acc*	S_{te}	*Acc*	S_{te}	*Acc*	S_{te}
Bengio et al.[17]	30.33 ±0.75	9.90 ±0.18	24.19 ±0.89	16.67 ±0.48	23.07 ±0.52	20.13 ±0.32	21.72 ±0.70	20.98 ±0.56
Liu et al. [23]	38.29 ±0.01	9.79 ±0.11	35.04 ±0.11	15.62 ±0.25	33.61 ±0.44	18.41 ±0.15	32.63 ±0.12	19.33 ±0.36
RDOC [26]	12.08 ±0.36	9.93 ±0.00	7.58 ±0.29	15.88 ±0.00	5.47 ±0.31	20.89 ±0.00	5.15 ±0.32	22.06 ±0.00
RSOC [26]	11.77 ±0.42	9.93 ±0.00	7.35 ±0.29	15.88 ±0.00	5.42 ±0.32	20.89 ±0.00	4.98 ±0.23	22.06 ±0.00
SpectralECOC[28]	15.92 ±0.44	10.18 ±0.00	12.04 ±0.43	15.88 ±0.00	9.29 ±0.82	20.89 ±0.00	8.74 ±0.55	22.06 ±0.00
Ours [15]	38.43 ±0.45	9.97 ±0.01	35.58 ±0.40	17.11 ±0.04	33.20 ±0.20	20.97 ±0.14	33.04 ±0.56	22.27 ±0.21
Ours	**38.90** ±0.16	9.95 ±0.00	**36.30** ±0.20	16.99 ±0.03	**34.67** ±0.09	20.36 ±0.08	**33.91** ±0.02	21.12 ±0.13
OvA	50.89 ± 0.14							

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

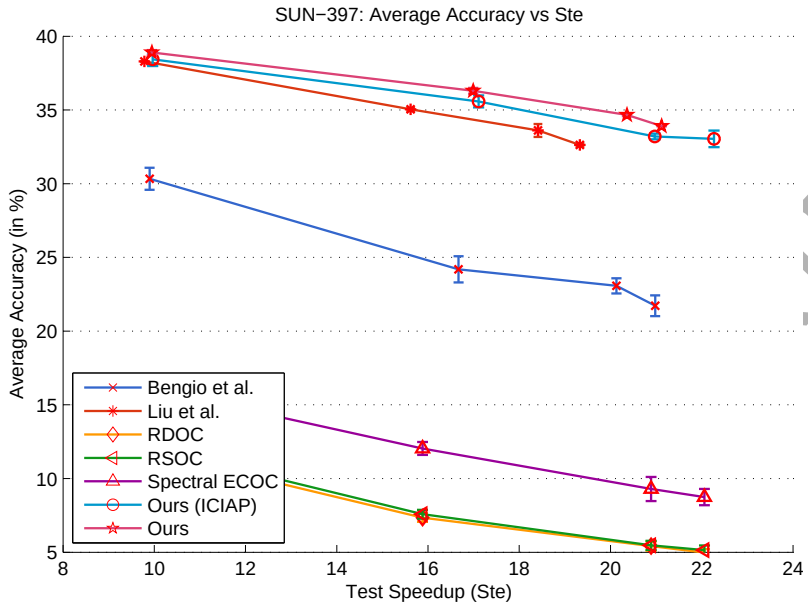


Figure 3: Performance of methods on SUN-397.

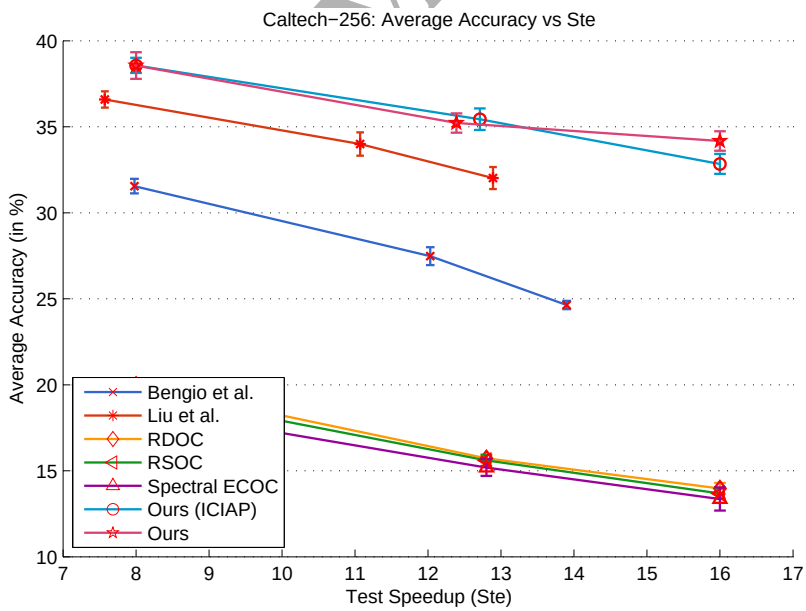


Figure 4: Performance of methods on Caltech-256.

ACCEPTED MANUSCRIPT

Table 3: Performance of methods on Caltech-256.

Methods	$T_{16,2}$		$T_{7,3}$		$T_{4,4}$		$T_{2,8}$	
	Acc	S_{te}	*Acc*	S_{te}	*Acc*	S_{te}	*Acc*	S_{te}
Bengio et al. [17]	31.55	7.98	27.48	12.03	24.64	13.90	22.60	14.21
	±0.43	±0.12	±0.52	±0.18	±0.24	±0.23	±0.48	±0.14
Liu et al. [23]	36.59	7.57	34.00	11.07	32.02	12.89	29.28	11.51
	±0.48	±0.10	±0.68	±0.37	±0.64	±0.60	±0.70	±0.46
RDOC [26]	20.00	8.00	15.72	12.80	13.97	16.00	13.97	16.00
	±0.15	±0.00	±0.23	±0.00	±0.32	±0.00	±0.32	±0.00
RSOC [26]	19.55	8.00	15.60	12.80	13.68	16.00	13.68	16.00
	±0.31	±0.00	±0.32	±0.00	±0.30	±0.00	±0.30	±0.00
SpectralECOC[28]	18.63	8.00	15.19	12.80	13.36	16.00	13.36	16.00
	±0.53	±0.00	±0.49	±0.00	±0.67	±0.00	±0.67	±0.00
Ours [15]	**38.57**	8.00	**35.44**	12.71	32.84	16.00	29.93	16.00
	±0.44	±0.00	±0.63	±0.09	±0.58	±0.00	±0.67	±0.00
Ours	38.56	8.00	35.22	12.39	**34.17**	16.00	**30.57**	16.00
	±0.77	±0.00	±0.56	±0.07	±0.57	±0.00	±0.33	±0.00
OvA	50.90 ± 0.35							

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

with equal numbers of classes in each node (i.e., 64 classes); classes such as 248, 142, and 123 are properly assigned, and classification performance improves by comparison.

Figure 6 shows the advantages of our method when using the confusion constraint with the combinatory distance in comparison with the simple one described in [15]. The classification accuracy of our method is higher because of the proper assignment of classes (marked in red) to the nodes.

4.4.3. Results on ImageNet-10K with Deep Features

Here, we describe experiments that evaluated the scalability of the methods on a large-scale dataset and the usefulness of deep features.

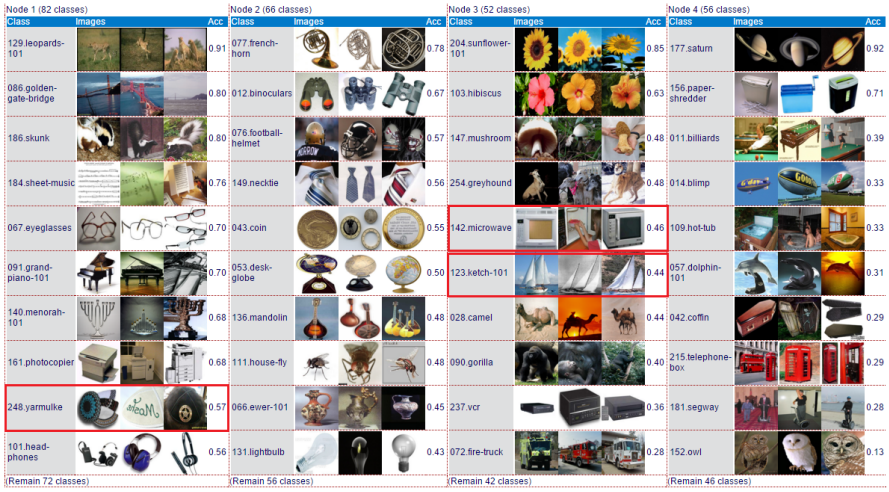
We used a state-of-the-art visual feature extracted by using the deep learning architecture introduced by Simonyan and Zisserman [42]. This architecture is a deep convolutional neural network designed for large-scale image recognition. The network is built from 16 weight layers including 13 convolutional layers and three fully-connected layers. We used MatConvNet toolbox [43] with the VERYDEEP-VGG16 model pre-trained on the ILSVRC-2012 dataset [42]. The output of the network at the layer fc7 with 4,096 channels was used as the feature vector of the input image.

We first tried the deep features on ILSVRC2010-1K. We fully used 1.2M training images of the dataset. The experimental results are listed in Table 4 and illustrated in Figure 7. Compared with SIFT-based features, the deep features significantly improved the classification accuracies of all methods. In addition, the use of deep features make it easier to solve a large-scale problem because of the lower number of dimensions of the representative vectors.

To evaluate the scalability of the methods, we carried out experiments on ImageNet-10K [7]. The dataset contained 9M images in 10,184 classes. For each class, we randomly picked 100 images for training, 50 images for validation, and 50 images for testing. The same settings were used by Liu et al. [19].

Table 5 lists the performance of the evaluated methods, and Figure 8 illustrates the correlation between Acc and S_{te}. In this experiment, we mainly eval-

ACCEPTED MANUSCRIPT

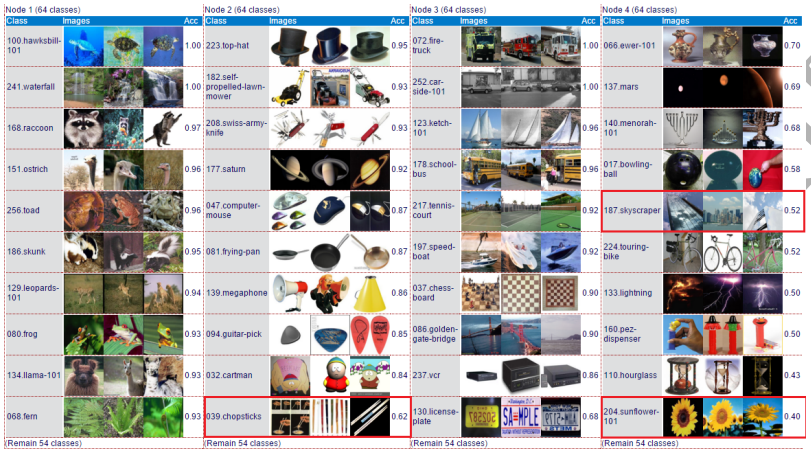


(a) Tree $T_{4,4}$ generated using Bengio et al.'s method [17]

(b) Tree $T_{4,4}$ generated with our method

Figure 5: Comparison of trees generated using Bengio et al.'s method [17] (a) and our method (b).

ACCEPTED MANUSCRIPT



(a) The children of root node of tree $T_{4,4}$ using method proposed in [15]

(b) The children of root node of tree $T_{4,4}$ using our method

Figure 6: Comparison of the trees generated using the method described in [15] (a) and our method (b).

ACCEPTED MANUSCRIPT

uated the OvA and tree-based approaches. The ECOC-based approaches were not evaluated since their scalability is not under consideration. The proposed method achieved the best accuracy among the evaluated tree-based approaches. This result is consistent with the previous experimental results. More importantly, our method was very competitive in accuracy compared to OvA method, despite having 50-250 times fewer operations.

4.5. Discussion

In this study, we followed the same classification scenario as that of existing work on hierarchical classification [17–19], i.e., only one input image. We basically assumed that classes are uniformly distributed. The numbers of images in classes are equal (e.g., ILSVRC2010-1K and ImageNet-10K) or not significantly imbalanced (e.g., Caltech-256 and SUN-397). Experimental results in such a scenario demonstrated that the essence of the label tree approach is the balance of the learned tree. By jointly optimizing the balance and confusion constraints, our method outperformed the other label tree-based methods and the ECOC-based methods.

However, a balance tree may not be the optimal structure for datasets whose data is not uniformly distributed. With such datasets, the paths of majority classes in the tree may be shorter than those of minority classes to minimize the overall number of operations. Most of real-life datasets are of that type and the distribution information may be known at the time of training. Hence, how to use such information in tree construction for non-uniform distributed datasets is an interesting problem for future work.

Tree methods are more limited in parallelization ability than flat ones, i.e., OvA. Assuming that the classes are independent, one can use parallel computing (e.g., a GPU) to speed up the testing process of OvA by parallelizing all N classifier evaluations. Although parallelization can be used for making branching decisions at nodes along the traversal path, tree traversal is a sequential process; as such, the classification of new images cannot be fully parallelized when using tree-based methods.

ACCEPTED MANUSCRIPT

Table 4: Performance of methods using the VGG-VERYDEEP-16 feature on ILSVRC2010-1K.

Methods	$T_{32,2}$		$T_{10,3}$		$T_{6,4}$		$T_{4,5}$	
	Acc	S_{te}	Acc	S_{te}	Acc	S_{te}	Acc	S_{te}
Bengio et al. [17]	53.21	16.01	44.27	32.59	40.06	39.64	36.00	45.42
Liu et al. [23]	56.91	15.84	54.27	28.21	52.71	37.90	**51.66**	42.60
RDOC [26]	20.21	15.63	12.50	33.33	6.00	40.00	3.96	50.00
RSOC [26]	13.18	15.63	4.70	33.33	3.47	40.00	2.51	50.00
SpectralECOC[28]	23.06	15.63	12.62	33.33	10.04	40.00	7.33	50.00
Ours [15]	58.02	15.73	54.55	33.33	52.05	42.65	51.08	50.11
Ours	**58.13**	15.76	**54.84**	33.33	**52.91**	42.46	**51.54**	50.09

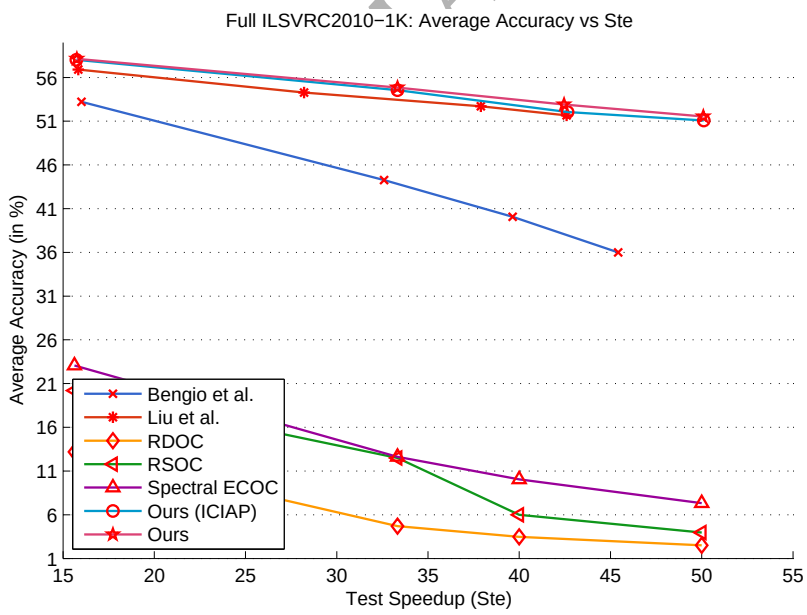


Figure 7: Performance of methods using the VGG-VERYDEEP-16 feature on ILSVRC2010-1K with 1.2M training images.

ACCEPTED MANUSCRIPT

Table 5: Performance of methods using the VGG-VERYDEEP-16 feature on ImageNet-10K.

Methods	$T_{101,2}$		$T_{10,4}$	
	Acc	S_{te}	Acc	S_{te}
Bengio et al. [17]	12.39	52.40	7.21	234.66
Liu et al. [23]	17.44	52.59	15.33	233.92
Ours [15]	18.19	50.45	15.22	254.08
Ours	**18.34**	50.45	**15.33**	254.14
OvA	20.05			

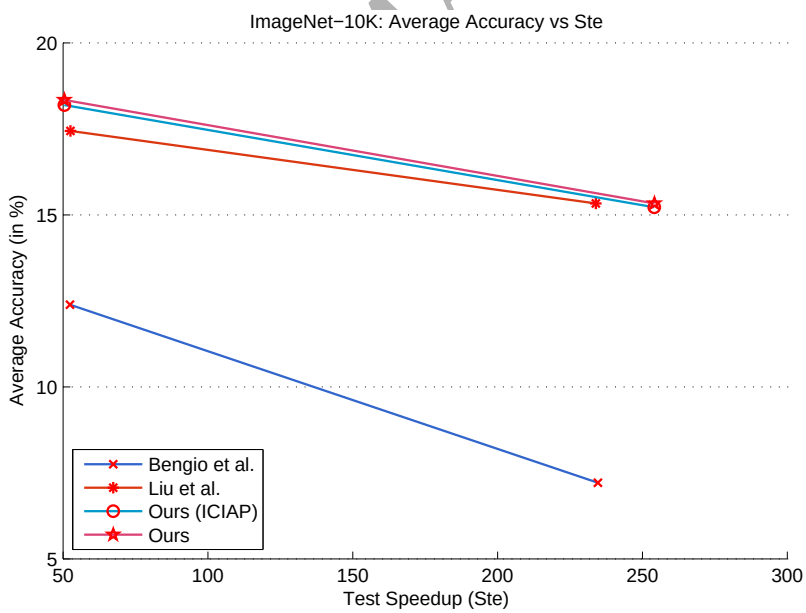


Figure 8: Performance of methods using the VGG-VERYDEEP-16 feature on ImageNet-10K.

ACCEPTED MANUSCRIPT

5. Conclusion

We proposed a method for learning an effective and balanced tree by jointly optimizing the balance and confusion constraints. We compared our method with other state-of-the-art methods in experiments on benchmark datasets including Caltech-256, SUN-397, ILSVRC2010-1K, and ImageNet-10K. The results show that the our method outperformed other state-of-the-art methods. Incorporation of the semantic relationship between classes and the distribution of classes would be promising direction of future work.

Acknowledgments

This research is funded by Vietnam National University Ho Chi Minh City (VNU-HCM) under grant number B2015-26-01.

References

[1] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in: CVPR, 2006, pp. 2169–2178.

[2] G. Carneiro, A. B. Chan, P. J. Moreno, N. Vasconcelos, Supervised learning of semantic classes for image annotation and retrieval, IEEE Trans. Pattern Anal. Mach. Intell. 29 (3) (2007) 394–410.

[3] R. Datta, D. Joshi, J. Li, J. Z. Wang, Image retrieval: Ideas, influences, and trends of the new age, ACM Comput. Surv. 40 (2) (2008) 5:1–5:60.

[4] L. Li, R. Socher, F. Li, Towards total scene understanding: Classification, annotation and segmentation in an automatic framework, in: CVPR, 2009, pp. 2036–2043.

[5] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, A. Torralba, SUN database: Large-scale scene recognition from abbey to zoo, in: CVPR, 2010, pp. 3485–3492.

ACCEPTED MANUSCRIPT

[6] L. Li, R. Socher, F. Li, Towards total scene understanding: Classification, annotation and segmentation in an automatic framework, in: CVPR, 2009, pp. 2036–2043.

[7] J. Deng, A. C. Berg, K. Li, F. Li, What does classifying more than 10, 000 image categories tell us?, in: ECCV, 2010, pp. 71–84.

[8] J. Deng, A. C. Berg, F. Li, Hierarchical semantic indexing for large scale image retrieval, in: CVPR, 2011, pp. 785–792.

[9] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: NIPS, 2012, pp. 1106–1114.

[10] Z. Akata, F. Perronnin, Z. Harchaoui, C. Schmid, Good practice in large-scale learning for image classification, TPAMI 36 (3) (2014) 507–520.

[11] M. Lapin, B. Schiele, M. Hein, Scalable multitask representation learning for scene classification, in: CVPR, 2014, pp. 1434–1441.

[12] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, L. Fei-Fei, ImageNet Large Scale Visual Recognition Challenge, International Journal of Computer Vision (IJCV) (2015) 1–42.

[13] A. Karpathy, F. Li, Deep visual-semantic alignments for generating image descriptions, in: CVPR, 2015, pp. 3128–3137.

[14] J. Fan, J. Zhang, K. Mei, J. Peng, L. Gao, Cost-sensitive learning of hierarchical tree classifiers for large-scale image classification and novel category detection, Pattern Recognition 48 (5) (2015) 16731687.

[15] T. Mai, T. D. Ngo, D. Le, D. A. Duong, K. Hoang, S. Satoh, Learning balanced trees for large scale image classification, in: ICIAP, 2015, pp. 3–13.

[16] R. M. Rifkin, A. Klautau, In defense of one-vs-all classification, JMLR 5 (2004) 101–141.

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

[17] S. Bengio, J. Weston, D. Grangier, Label embedding trees for large multi-class tasks, in: NIPS, 2010, pp. 163–171.

[18] J. Deng, S. Satheesh, A. C. Berg, F. Li, Fast and balanced: Efficient label tree learning for large scale object recognition, in: NIPS, 2011, pp. 567–575.

[19] B. Liu, F. Sadeghi, M. F. Tappen, O. Shamir, C. Liu, Probabilistic label trees for efficient large scale image classification, in: CVPR, 2013, pp. 843–850.

[20] S. Zhu, X. Wei, C. Ngo, Collaborative error reduction for hierarchical classification, Computer Vision and Image Understanding 124 (2014) 79–90.

[21] T. Gao, D. Koller, Discriminative learning of relaxed hierarchy for large-scale visual recognition, in: ICCV, 2011, pp. 2072–2079.

[22] A. Y. Ng, M. I. Jordan, Y. Weiss, On spectral clustering: Analysis and an algorithm, in: NIPS, 2001, pp. 849–856.

[23] S. Liu, H. Yi, L. Chia, D. Rajan, Adaptive hierarchical multi-class SVM classifier for texture-based image classification, in: ICME, 2005, pp. 1190–1193.

[24] J. C. Platt, N. Cristianini, J. Shawe-Taylor, Large margin dags for multi-class classification, in: NIPS, 1999, pp. 547–553.

[25] T. G. Dietterich, G. Bakiri, Solving multiclass learning problems via error-correcting output codes, J. Artif. Intell. Res. (JAIR) 2 (1995) 263–286.

[26] E. L. Allwein, R. E. Schapire, Y. Singer, Reducing multiclass to binary: A unifying approach for margin classifiers, Journal of Machine Learning Research 1 (2000) 113–141.

[27] A. Passerini, M. Pontil, P. Frasconi, New results on error correcting output codes of kernel machines, IEEE Trans. Neural Networks 15 (1) (2004) 45–54.

ACCEPTED MANUSCRIPT

[28] X. Zhang, L. Liang, H. Shum, Spectral error correcting output codes for efficient multiclass recognition, in: ICCV, 2009, pp. 1111–1118.

[29] B. Zhao, E. P. Xing, Sparse output coding for large-scale visual recognition, in: CVPR, 2013, pp. 3350–3357.

[30] Y. Wang, D. A. Forsyth, Large multi-class image categorization with ensembles of label trees, in: ICME, 2013, pp. 1–6.

[31] M. Marszalek, C. Schmid, Constructing category hierarchies for visual recognition, in: ECCV, 2008, pp. 479–491.

[32] Y. Amit, M. Fink, N. Srebro, S. Ullman, Uncovering shared structures in multiclass classification, in: ICML, 2007, pp. 17–24.

[33] Y. Lin, F. Lv, S. Zhu, M. Yang, T. Cour, K. Yu, L. Cao, T. S. Huang, Large-scale image classification: Fast feature extraction and SVM training, in: CVPR, 2011, pp. 1689–1696.

[34] N. Loeff, A. Farhadi, Scene discovery by matrix factorization, in: ECCV, 2008, pp. 451–464.

[35] Z. Harchaoui, M. Douze, M. Paulin, M. Dudík, J. Malick, Large-scale image classification with trace-norm regularization, in: CVPR, 2012, pp. 3386–3393.

[36] H. O. Song, R. B. Girshick, S. Zickler, C. Geyer, P. F. Felzenszwalb, T. Darrell, Generalized sparselet models for real-time multiclass object recognition, IEEE Trans. Pattern Anal. Mach. Intell. 37 (5) (2015) 1001–1012.

[37] G. Griffin, A. Holub, P. Perona, Caltech-256 object category dataset, Tech. Rep. 7694, California Institute of Technology (2007).

[38] A. Vedaldi, B. Fulkerson, Vlfeat: an open and portable library of computer vision algorithms, in: ACMMM, 2010, pp. 1469–1472.

ACCEPTED MANUSCRIPT

ACCEPTED MANUSCRIPT

[39] J. Wang, J. Yang, K. Yu, F. Lv, T. S. Huang, Y. Gong, Locality-constrained linear coding for image classification, in: CVPR, 2010, pp. 3360–3367.

[40] S. Escalera, O. Pujol, P. Radeva, Error-correcting ouput codes library, Journal of Machine Learning Research 11 (2010) 661–664.

[41] R. Fan, K. Chang, C. Hsieh, X. Wang, C. Lin, LIBLINEAR: A library for large linear classification, JMLR 9 (2008) 1871–1874.

[42] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, CoRR abs/1409.1556.

[43] A. Vedaldi, K. Lenc, Matconvnet: Convolutional neural networks for matlab, in: ACM Multimedia 2015, 2015, pp. 689–692.

ACCEPTED MANUSCRIPT