Improved learning of I2C distance and accelerating the neighborhood search for image classification

Zhengxiang Wang\textsuperscript{a}, Yiqun Hu\textsuperscript{b}, Liang-Tien Chia\textsuperscript{a}

\textsuperscript{a}Center for Multimedia and Network Technology, School of Computer Engineering
Nanyang Technological University, 639798, Singapore
\textsuperscript{b}School of Computer Science and Software Engineering
The University of Western Australia, Australia
wang0460@ntu.edu.sg, yiqun.hu@uwa.edu.au, asltchia@ntu.edu.sg

Abstract

Image-To-Class (I2C) distance is a novel measure for image classification and has successfully handled datasets with large intra-class variances. However, due to the lack of a training phase, the performance of this distance is easily affected by irrelevant local features that may hurt the classification accuracy. Besides, the success of this I2C distance relies heavily on the large number of local features in the training set, which requires expensive computation cost for classifying test images. On the other hand, if there are small number of local features in the training set, it may result in poor performance.

In this paper, we propose a distance learning method to improve the classification accuracy of this I2C distance as well as two strategies for accelerating its NN search. We first propose a large margin optimization framework to learn the I2C distance function, which is modeled as a weighted combination of the distance from every local feature in an image to its nearest neighbor (NN) in a candidate class. We learn these weights associated with local features in the training set by constraining the optimization such that the I2C distance from image to its belonging class should be less than that to any other class. We evaluate the proposed method on several publicly available image datasets and show that the performance of I2C distance for classification can significantly be improved by learning a weighted I2C distance function. To improve the computation cost, we also propose two methods based on spatial division and hubness score to accelerate the NN search, which is able to largely reduce the on-line testing time while still preserving or even achieving a better classification accuracy.
1. Introduction

Image classification is an active research topic in computer vision community due to the large intra-class variances and ambiguities of images. Many efforts have been investigated for dealing with this problem. Among recent works, nearest-neighbor (NN) based methods [1–8] have been attractive for handling the classification task due to its simple implementation and effective performance. While most studies focus on measuring the distance between images, e.g., learned local Image-To-Image (I2I) distance function [2, 3], a new NN based Image-To-Class (I2C) distance is proposed by Boiman et al. [1] in their Naive-Bayes Nearest-Neighbor (NBNN) method, which achieves state-of-the-art performance in several challenging datasets despite the simplicity of its algorithm. Compared to previous works using NN based methods, this new distance similarity measure directly deals with each image represented by a set of patch based local features, e.g., SIFT features [9], while most previous studies require quantizing these features into a fixed-length vector for representation, which may lose the discriminate information from the original image. The training feature set of each class is constructed by gathering features in every training image belonging to that class. The I2C distance from a test image to a candidate class is formulated as the sum of Euclidean distance between each feature in this test image and its NN feature searched from the training feature set of the candidate class. This is also different from most previous studies that only measure the distance between images. They attribute the success of NBNN to the avoidance of descriptor quantization and the use of I2C distance instead of I2I distance, and they have shown that descriptor quantization and I2I distance lead to significant degradation for classification.

The effectiveness of this I2C distance attracts many recent studies. For example, Huang et al. [10] applied it in face and human gait recognition, Wang et al. [11] learn a distance metric for this I2C distance, Behmo et al. [12] learn an optimal NBNN by hinge-loss minimization to further enhance its generalization ability, etc.

However, in the formulation of this I2C distance, each local feature in the training set is given equal importance. This makes the I2C distance sensitive
Find NN for each feature
Weight learnt during the training procedure and associated with each feature
Weighted I2C distance

Feature Collection of Class 1
Feature Collection of Class N

Class 1 Beach
Dist of Class 1
Dist of Class 2
Dist of Class 3
Dist of Class N

Figure 1: The whole procedure for classifying a test image. The I2C distances to different classes are denoted as different length of blue bars for expressing the relative size.

To irrelevant features, which are useless for measuring the distance and may hurt the classification accuracy. Besides, the performance of this I2C distance relies heavily on the large number of local features in the training set, which requires expensive computation cost during the NN search when classifying test images. On the other hand, a small training feature set may result in poor performance, although it requires less time for the NN search.

In this paper, we propose a novel NN based classification method for learning a weighted I2C distance. For each local feature in the training set, we learn a weight associated with it during the training phase, thus decreasing the impacts of irrelevant features that are useless for measuring the distance. These weights are learned by formulating a weighted I2C distance function in the training phase, which is achieved by constraining the weighted I2C distance to the belonging class to be the shortest for each training image among its I2C distances to all classes. We adopt the large margin idea in Frome et al. [3] and formulate the triplet constraint in our optimization problem that the I2C distance for each training image to its belonging class should be less than the distance to any other class with a large margin. Therefore, our method avoids the shortcomings of both non-parametric methods and most learning-based methods involving I2I distance and descriptor quantization. This leads to a better classification accuracy than NBNN or those learning-based methods while requiring relatively smaller number of local features in
With the weighted I2C distance from a test image to each candidate class, we predict the class label of this test image using a simple nearest-neighbor classifier as in [1], which selects the class with the shortest I2C distance as its predicted label. The whole procedure of classifying the test image is shown in Figure 1. First a set of local features are extracted from a given test image (denoted as crosses with different colors). Then for each feature, its NN feature in each class’s feature set is searched (denoted as crosses with the same color). The weighted I2C distance to each class is formulated as the sum of Euclidean distance between each individual feature and its NN feature in the class weighted by its associated weight, and the class with the shortest I2C distance is selected as the predicted class.

The main computational bottleneck in I2C distance is the NN feature search due to the large number of features in the training feature set for each class. In most real world applications the training set is extremely large, which makes the NN search time consuming. Our training cost for learning the weight would be negligible compared to this heavy NN search. So in this paper we propose two methods for accelerating the NN search, which make the I2C distance more practical for real world problems. These two methods reduce the candidate features set for the NN search from different aspects. The first method uses spatial division to split the image into several spatial subregions and restrict each feature to find its NN only in the same spatial subregion, while the second method ranks all local features in the training set by hubness score [13] and removes those features with low hubness score. Both methods are able to accelerate the NN search significantly while maintaining or even achieving better classification accuracy.

This paper is an extension work of our published conference paper in [14]. The main extension work includes: 1. We provide a more complete description on the technique detail. 2. We propose two new methods for accelerating the NN search due to the heavy computation cost of the NN search. 3. We add five new publicly available datasets in the experiment section to validate our proposed methods.

The paper is organized as follows. In Section 2 we present a large margin optimization framework for learning the weight. Two NN search acceleration methods are present in Section 3. We validate our approach through experiment in Section 4. Finally, the conclusion is written in Section 5.
2. Learning Image-To-Class Distance

In this section, we propose a large margin optimization framework to construct a weighted I2C distance by learning the weight associated with each feature in the training set. These features are extracted from patches around each keypoint and represented by some local descriptor such as SIFT [9]. We first explain some notations for clarity. Let $F_i = \{f_{i,1}, f_{i,2}, \ldots, f_{i,m_i}\}$ denote local features belonging to an image $X_i$, where $m_i$ represents the number of features in $X_i$ and each feature is denoted as $f_{i,j} \in \mathbb{R}^d, \forall j \in \{1, \ldots, m_i\}$. The feature set of each class $c$ is composed of features from all training images belonging to that class and is denoted as $F_c = \{f_{c,1}, f_{c,2}, \ldots, f_{c,m_c}\}$. Similarly, here $m_c$ represents the number of features in class $c$. The original unweighted I2C distance from image $X_i$ to a class $c$ is formulated as the sum of L2 distance (Euclidean distance) between every feature $f_{i,j}$ in $X_i$ and its NN feature in class $c$ denoted as $f_{c,k}$, which is shown in the left part of Figure 2. Here the NN feature $f_{c,k}$ is searched over the feature set of class $c$ that has the shortest L2 distance to feature $f_{i,j}$ in image $X_i$ and the distance between them is denoted as $d_{j,k}$. This NN search is time consuming when the training feature set is large, so we will discuss some acceleration methods in Section 3. The formulation of this I2C distance is given as:

$$\text{Dist}(X_i, c) = \sum_{j=1}^{m_i} \| f_{i,j} - f_{c,k} \|^2 = \sum_{j=1}^{m_i} d_{j,k} \quad (1)$$

Where

$$k = \arg \min_{k' = \{1, \ldots, m_c\}} \| f_{i,j} - f_{c,k'} \|^2 \quad (2)$$

Figure 2: The original unweighted I2C distance in NBNN (left) and our proposed weighted I2C distance (right).
However, in this formulation each local feature in the training set is given equal importance. This makes the I2C distance sensitive to irrelevant features, which are useless for measuring the distance and may hurt the classification accuracy. To discriminate relevant features from irrelevant ones, we associate each feature in the training set with a weight, which is learned through the training phase. Therefore our new weighted I2C distance (as shown in the right part of Figure 2) is represented as follows with $k$ defined in equation (2):

$$\text{Dist}(X_i, c) = \sum_{j=1}^{m_i} w_{c,k} \cdot \| f_{i,j} - f_{c,k} \|^2 = \sum_{j=1}^{m_i} w_{c,k} \cdot d_{j,k}$$ \hspace{1cm} (3)$$

For each local feature $f_{i,j}$, the L2 distance $d_{j,k}$ between this feature and its NN $f_{c,k}$ is multiplied with the weight $w_{c,k}$ learned for this NN feature $f_{c,k}$. In fact, the original I2C distance can be viewed as every weight equating to 1 in this formula. Since all these weights in the training set are globally consistent and can be learned simultaneously, we concatenate all the weights to a weight vector $W$ during the learning procedure. For consistency, a distance vector $D^c_i$ is also constructed from image $X_i$ to class $c$ with the same length as $W$. The construction of these vectors is illustrated in Figure 3. Each component in the vector belongs to a feature in the training set and the length of the vector is equal to the number of features in the training set. The component of the weight vector reflects the weight associated with the corresponding feature. The L2 distances between features in the image $X_i$ and their NN features in class $c$ contribute as components of the distance vector $D^c_i$ at the locations of these NN features. In this way the weighted I2C distance can be formulated as:

$$\text{Dist}(X_i, c) = \sum_{j=1}^{m_i} w_{c,k} \cdot d_{j,k} = W^T \cdot D^c_i$$ \hspace{1cm} (4)$$

We adopt the idea of large margin to learn these weights. This idea is popular due to its success in SVM classifier, which simultaneously minimizes the empirical classification error and maximizes the geometric margin. For the binary classification problem, a hyperplane is optimized to create the largest separation between two classes. In our large margin framework for learning the weight, we optimize the triplet constraint in a way different from [3]. In [3], for each input image a triplet is constructed by selecting an image from the same class and an image from a different class, and the constraint
Figure 3: The construction of distance vector $D^c_i$ and weight vector $W$. The I2C distance between image $X_i$ and class $c$ is represented by the distance vector $D^c_i$. $x_1, x_2, x_3$ represent the training images in class $c$. The crosses in each image represent features extracted for that image. For the test image $X_i$, its local features $f_{i,1}, f_{i,2}, f_{i,3}$ find their NN $f_{c,1}, f_{c,3}, f_{c,9}$ in class $c$. $d_{1,1}, d_{2,3}$ and $d_{3,9}$ are the L2 distances between features in $X_i$ and their NN features in class $c$, and they contribute as components of $D^c_i$ at the locations of their NN features. In this way the weighted I2C distance can be formulated as $W^T \cdot D^c_i$.

is formulated that the I2I distance between images in the same class should be less than that in different classes with a margin. Besides the limitation incurred by I2I distance as described in [1], this triplet formulating method will cause too many triplet constraints in the distance learning, especially when there are large number of training images in each class. However, by using I2C distance, we construct our triplet for each input image just by selecting two classes, one as positive class that the image belongs to, and the other from any other class as negative class, hereby reducing the number of triplet constraints significantly. Our triplet constraint is therefore formulated by keeping the I2C distance to the positive class to be less than that to the negative class with a margin. This is illustrated in Figure 4. For each input image $X_i$, the triplet with positive class $p$ and negative class $n$ should be constrained as:

$$W^T \cdot (D^n_i - D^p_i) \geq 1 - \xi_{ipm} \tag{5}$$

Here the slack variable $\xi_{ipm}$ is used for soft-margin as in the standard SVM form.

We formulate our large margin optimization problem in the form similar to SVM. Since the initial weight of each feature is 1 as in the original I2C
Figure 4: The triplet constraint for distance learning. For an image $X_i$, its L2C distance to the positive class $p$ should be less than that to the negative class $n$ with a margin.

distance, we regularize the learned weights according to a prior weight vector $W_0$, whose elements are all equal to 1. This is to penalize those weights that are too far away from this prior in the optimization problem and keep consistency between training and testing phase. The optimization problem is given as:

$$\arg \min_{W, \xi} \frac{1}{2} ||W - W_0||^2 + C \sum_{i, p, n} \xi_{ipn}$$

s.t. $\forall i, p, n : W \cdot (D_i^n - D_i^p) \geq 1 - \xi_{ipn}$

$\xi_{ipn} \geq 0$

$\forall k : W(k) \geq 0$

Here the parameter $C$ controls the trade-off between the regularization and error terms as in SVM optimization problem. Each element of the optimizing weight vector is enforced to be non-negative as distances are always non-negative. For a dataset with $N_c$ classes, the number of triplet constraints for each training image is $N_c - 1$ since there are $N_c - 1$ different negative classes, and the total number of triplet constraints in the optimization problem should be $(N_c - 1) \times N_c \times N_{tr}$, where $N_{tr}$ stands for the number of training
images in each class. This is a significant reduction compared to the number of triplets in I2I distance [3], which needs $O(N^2 \times N_\ell^3)$ triplets for learning the weight. Such reduction on the number of triplets will result in a faster speed during the weight updating procedure.

We solve the optimization problem of equation (6) in the dual form using the method in [3] as:

$$\arg \max_{\alpha, \mu} -\frac{1}{2} \left\| \sum_{i,p,n} \alpha_{ipn} \cdot Q_{ipn} + \mu \right\|^2 + \sum_{ipn} \alpha_{ipn}$$

$$- \sum_{ipn} \alpha_{ijk} \cdot Q_{ipn} + \mu \right] \cdot W_0$$

s.t. \quad \forall i, p, n : 0 \leq \alpha_{ipn} \leq C

\forall j : \mu(j) \geq 0

Where $Q_{ipn} = (D^n_i - D^n_p)$ for simplicity. This dual form is solved by iteratively updating the dual variable $\alpha$ and $\mu$ alternatively. Each time we update the $\alpha$ variables that violate the Karush Kuhn Tucker (KKT) [15] conditions by taking the derivative of equation (7) with respect to $\alpha$ and then update $\mu$ to ensure the positiveness of weight vector $W$ in each iteration. The updated formula of $\alpha$ and $\mu$ is given as:

$$\alpha_{ipn} = \left[ 1 - \sum_{\{i',p',n'\} \neq \{i,p,n\}} \alpha_{i'p'n'} \cdot (Q_{i'p'n'} \cdot Q_{ipn}) - \langle (\mu + W_0) \cdot Q_{ipn} \rangle / \| Q_{ipn} \|^2 \right]_{[0,C]}$$

$$\mu = \max \{ 0, -\sum_{i,p,n} \alpha_{ipn} \cdot Q_{ipn} - W_0 \}$$

And the KKT conditions are:

$$\alpha_{ipn} = 0 \implies W \cdot Q_{ipn} \geq 1$$

$$0 < \alpha_{ipn} < C \implies W \cdot Q_{ipn} = 1$$

$$\alpha_{ipn} = C \implies W \cdot Q_{ipn} \leq 1$$

The operation of $[f(x)]_{[0,C]}$ is to clip the value of $f(x)$ in the region $[0,C]$, and $\{i',p',n'\}$ represent any triplet in constraint set that is different from $\{i,p,n\}$. Since the optimization problem is convex, this objective function is guaranteed to reach the global minimum after iterative updating. The
number of $\alpha$ variable is equal to the number of triplet constraints, which is much less than that in the I2I distance framework that was mentioned before. After convergence, the weight vector $W$ in the primer form can be calculated by:

$$W = \sum_{i,p,n} \alpha_{i,p,n} \cdot Q_{i,p,n} + \mu + W_0 \quad (10)$$

As all weights are optimized simultaneously in the weight vector, this consistency will ensure global ranking of the weighted I2C distances to all classes.

3. Efficiency Improvement

The main computational bottleneck in the I2C distance calculation is the NN feature search, as the number of training features in each class is usually very large especially when using dense sampling strategy for feature extraction. The large amount of training features also requires large memory cost during the NN search. In [1], they used KD-Tree to approximate the finding of NN feature for acceleration, which is commonly used in the NN search. However, since the training set would be extremely large in most real world applications, the running time for the NN search is still very expensive even when using KD-Tree. Besides that, the large memory cost cannot be alleviated by using KD-Tree. Other NN approximation algorithms such as locality-sensitive hashing (LSH) also face similar problems. In this section, we propose two methods to reduce candidate feature set for the NN search in addition to these NN approximation algorithms. They can accelerate the NN search and thus make our I2C distance more practical for real world problems.

3.1. Feature Reduction by Spatial Division

The first method for reducing candidate feature set to accelerate the NN search is dividing the image into subregions of equal area. So the NN of each feature in each subregion is restricted to those features in the same subregion from candidate feature set of each class. Figure 5 illustrates this spatial division. At level 1, images are divided into $2 \times 2$ subregions, so the candidate feature set is reduced to $1/4$ and at level 2 it is reduced to $1/16$. This spatial division is inspired by Lazebnik et al. [16] who divided the image into subregions at increasingly fine resolutions for each level and then merge them together, which is known as spatial pyramid match (SPM). Since each feature is more likely to find its NN in the same subregion, using spatial...
Figure 5: The I2C NN feature search with spatial division. We adopt the idea of spatial pyramid by restricting the NN of each feature in the same subregion at each spatial level.

division to reduce the candidate set will preserve the classification accuracy of weighted I2C distance while accelerating the NN search. For scene images that usually have geometric structures and classes that manifest common spatial layout, this spatial division will not only accelerate the NN search, but also improve classification accuracy, as it can reduce the rate of false NN match. Even for datasets with less spatial aligned images, this spatial division can still work. This is because the NN of each local feature in a test image is searched over every training image in the same subregion and it is less likely that none of the training images contain the object in the same subregion as in the test image.

Besides accelerating the NN search for improving efficiency, the spatial division can also be used to further improve the classification accuracy. This can be achieved by adding the weighted I2C distances from different spatial levels to formulate the I2C distance with SPM as shown in Figure 5, which is analogous to the spatial pyramid [16] and pyramid match [17]. Although using spatial division at a single spatial level can accelerate the NN search, the SPM of combining multiple spatial levels for improving the classification accuracy requires additional running time. Note that the spatial subregions cannot be divided too finely, as it will make the NN search too sensitive to the spatial layout and increase the number of false matches. For the experiments in this paper, we use spatial division for learning I2C distance only at spatial level 1 (2 × 2 subregions) and level 2 (4 × 4 subregions).
3.2. Feature Reduction by Hubness Score

Spatial division is a simple but effective method for reducing candidate feature set. However, this method does not remove any feature in the training set and therefore still requires large memory cost during the NN search. For such situation, we develop another method which explores the hubness of features in candidate set for feature reduction. This is inspired by the work of [13], which measured data points using hubness score and explored its importance in many NN problems. Specifically, the hubness of a data point is reflecting the number of times this point is matched by other points as their NN. A hubness score is therefore introduced to measure the hubness of each data point. For those points with high occurrences as NN points by others, they should have high hubness score and thus are recognized as hub points.

In our problem where the weighted I2C distance is used for classification, we define a new hubness score for measuring the importance of each feature (here a data point is represented as a feature) in candidate set. Those unimportant features with low hubness scores are removed for improvement in speed. Inspired by the triplet in our learning framework which is composed of positive and negative classes, the counting of occurrence as matched by other features should also be split into positive and negative parts, since the occurrence of being matched by features in the belonging class plays a different role to that in other classes. Intuitively, we would want to remove those features frequently matched by features from other class while less likely to be matched from the belonging class. Let \( N_p(x) \) denotes the number of times a feature is matched by other features from the belonging class, which we also call positive match. Similarly, we denote the negative match \( N_n(x) \) as the number of times a feature is matched by features from other classes. We define the hubness score of each feature \( x \) by:

\[
h(x) = \frac{N_p(x)}{N_n(x)} \tag{11}
\]

For all features in the training set, we rank them in the order of descending hubness score, and only preserve features with higher hubness score. Other features are removed permanently and no longer put into the candidate feature set. We only learn the weights for these preserved features and use them for NN match during the testing phase.
3.3. Discussion

We discuss the advantages and disadvantages of the two proposed methods for accelerating the NN search. The first method uses spatial division to reduce candidate set for each feature in a test image during the NN search. Since determining which subregion a feature in test image belongs to is simply calculated by its pixel location, the selection of reduced candidate feature set for the NN search will require little time during the testing phase. For geometric structured images, this method performs even better than without spatial division. However, this method does not remove any feature in the training set, therefore it still requires large memory cost during the NN search. The number of weights for learning is not reduced either, which may result in a long running time during learning when the training set is very large.

The second method does not have the above mentioned problems. Since the features with low hubness scores are removed permanently after ranking all training features, it can reduce the memory cost during the NN search in the testing phase. It can also speed up the weight learning procedure, as lesser number of weights is required for learning. However, unlike spatial division that can improve classification accuracy for structured images, this method may lose some accuracy when most of the training features are reduced. In addition, every feature in the training set needs to count its positive and negative match for calculating hubness score, which increases the computation cost during the training phase, although faster NN search speed in the testing phase is gained.

To achieve the best efficiency, both methods can be combined together for the task of NN search during the I2C distance calculation.

4. Experiment

4.1. Datasets and Setup

We evaluate our proposed method on five publicly available datasets: Scene-15, Sports, Corel, Caltech 101 and Caltech 256 datasets. We describe them briefly as follows:

- **Scene-15**: Scene dataset consists of 15 scene categories, among which 8 were originally collected by Oliva et al. [18], 5 added by Li et al. [19] and 2 from Lazebnik et al. [16]. Each class has about 200 to 400 images, and the average image size is around 300 × 250 pixels.
Following [16], we randomly select 100 images per class for training, and test on the rest. The average of each per-class accuracy is reported for evaluation.

- **Sports**: Sports event dataset is firstly introduced in [20], consisting of 8 sports event categories. The number of images in each class ranges from 137 to 250, so we follow [20] to select 70 and 60 images per class for training and test respectively. Since images in this dataset are usually very large, they are first resized such that the largest x/y dimension is 500.

- **Corel**: Corel dataset contains 10 scene categories published by Corel Corporation. Each class contains 100 images, and we follow [21] to separate them randomly into two subsets of equal size to form the training and test set. All the images are of the size 384 × 256 or 256 × 384.

- **Caltech 101**: Caltech 101 dataset is a large scale dataset containing 101 categories [22]. The number of images in each class varies from about 30 to 800. This dataset is more challenging due to the large number of classes and intra-class variance. Following the widely used measurement by the community we randomly select 15 images per class for training. For testing, we also select 15 images for each class and report the mean accuracy.

- **Caltech 256**: Caltech 256 dataset [23] is even larger than Caltech 101. Each class contains at least 80 images. The total of 256 categories make this dataset much more challenging than the other four datasets. We use this dataset to validate the scalability of our methods. In the experiment we randomly select 15 images per class for training and 25 different images per class for test.

Since the training and test set are selected randomly, we repeat each dataset for 5 times and report the average result. For feature extraction, we use dense sampling strategy and SIFT features [9] as our descriptor, which are extracted over every 8 pixels for all datasets. Although this feature extraction is simple and even discard color information, encouraging performance can be achieved. For the parameter $C$ in our learning problem, we empirically fix it to 1 throughout the experiment. We name our method as LI2C, short for Learning Image-To-Class distance.
Table 1: Classification accuracy (%) of LI2C compared to NBNN for each dataset with its attributes.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Classes</th>
<th>Training Images</th>
<th>Test Images</th>
<th>LI2C (%)</th>
<th>NBNN (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scene-15</td>
<td>15</td>
<td>1500</td>
<td>2985</td>
<td>80.0±0.4</td>
<td>72.8±0.7</td>
</tr>
<tr>
<td>Sports</td>
<td>8</td>
<td>560</td>
<td>480</td>
<td>82.0±1.2</td>
<td>67.6±1.1</td>
</tr>
<tr>
<td>Corel</td>
<td>10</td>
<td>500</td>
<td>500</td>
<td>91.1±1.1</td>
<td>85.7±0.9</td>
</tr>
<tr>
<td>Caltech 101</td>
<td>101</td>
<td>1515</td>
<td>1515</td>
<td>52.1±1.4</td>
<td>37.1±1.2</td>
</tr>
<tr>
<td>Caltech 256</td>
<td>256</td>
<td>3840</td>
<td>6400</td>
<td>24.0±1.1</td>
<td>18.3±1.3</td>
</tr>
</tbody>
</table>

4.2. Classification Results

To validate the effectiveness of our learned weight, we first compare LI2C to NBNN [1], which uses unweighted I2C distance for classifying. Table 1 gives the classification accuracy of both methods, averaged over all classes. The number of classes in each dataset and the number of training and test images are also listed. Results in Table 1 show that in every dataset, our LI2C outperforms NBNN significantly, which confirms our expectation that the weight learned in training set is able to enhance the discriminate of I2C distance for classifying test images.

For further exploring the improvement over the original I2C distance, we also compare the Per-Class classification accuracy of LI2C to NBNN. We show the comparison for Scene-15, Sports and Corel datasets. The weighted I2C distance is able to outperform unweighted I2C distance in most classes for these three datasets as shown in Figure 6. We find that the improvement is more apparent in those challenging classes that NBNN is unable to classify well, e.g. the worst 3 classes in Scene-15, the worst 4 classes in Sports and the worst 2 classes in Corel. Therefore, we can conclude that our large margin framework for learning the weight is able to significantly improve the classification accuracy on those challenging classes, which are difficult to be classified well by the unweighted I2C distance.

To make this conclusion more convincing, we also show some examples for the distributions of weighted local features. We choose two most challenging classes (“owls” and “mountains”) in Corel dataset that LI2C improves significantly over NBNN and another easy class (“flowers”) that both methods can achieve high accuracy. For each class we select the top 1% high weighted features and top 1% low weighted features and show some example images in
Figure 6: Per-Class classification accuracy on Scene-15, Sports and Corel datasets for comparing our method with NBNN.

Figure 7. We can see that in the two challenging classes most high weighted features are locate on the object, while the location of low weighted features are more arbitrary, i.e. either on the object or the background. As in LI2C those features with high weight are given more importance, the weighted I2C distance is able to largely improve the classification accuracy over the original unweighted distance. However, in the class of “flowers”, both high and low weighted features are difficult to be distinguished from either the object or background. As both weighted and unweighted I2C distances can easily classify images in this class, this distribution shows that features located on both the object and background in this class are useful for the classification. Some predicted examples in this dataset are also shown in Figure 8. In the top row, both methods are able to correctly classify these images. In the middle row, images are wrongly classified by NBNN. However, after learning the weight, they can be correctly classified by our LI2C. In the bottom row, we also show images that are unable to be correctly classified even after learning the weight. We can see that some of these images are difficult to be classified as they has less connection to their ground truth labels.

Compared to the published result of NBNN [1] on Caltech 101 and Caltech 256 datasets, our implementation of NBNN achieves lower result than their
best reported performance. We notice that they have used multiple scales and more densely sampled features in their experiment, which is about 20 times more than the number of features in our experiment. This shows that the performance of I2C distance without training phase relies heavily on the large number of complex features, which require heavy computation cost for the NN search. In contrast, we use simpler and smaller number of features in our experiment and therefore achieve a much lower result for NBNN, which proves that using small number of features result in poor performance for I2C distance without training phase. However, by learning the weight using our proposed method, the performance of I2C distance can be significantly improved using the same, small number of features. In addition, even using a large number of local features in the training set same as that in [1], we are unable to reproduce the result reported in [1]. We guess this may be due to the different parameter settings in the SIFT feature extraction stage. However, the result of our implemented NBNN is comparable to that in [11]. Since both methods use the same feature descriptor as input in our experiment, the comparison of our method with NBNN is fair. This experiment also shows that our large margin framework for learning the weight is
Figure 8: Predicted examples in Corel dataset. Images in the top row are correctly classified by both LI2C and NBNN. The middle row shows images that are correctly classified by LI2C but wrongly classified by NBNN, and images in the bottom row are wrongly classified by either LI2C or NBNN.

capable of dealing with large-scale datasets, while the computation cost of weight updating procedure in training is nearly negligible compared to the NN search in the I2C distance calculation as mentioned before.

4.3. Results on Spatial Division

Next we analyze the performance of LI2C as well as NBNN at different spatial levels. Table 2 shows the results of both methods from level 0 to level 2 as well as their combinations (SPM). Compared to level 0, the classification accuracies of both methods are improved at either spatial level 1 or level 2. This proves that spatial division is able to not only accelerate the NN search, but also improve the classification accuracy. Since images in the Sports dataset do not have such common spatial layout like that in scene constraint structures of the other datasets, but still gain improvement when using spatial division, we suggest that the spatial division is a useful strategy to help improve I2C distance in most real-world images. We also notice
Table 2: Classification accuracy (%) of LI2C compared to NBNN on different spatial levels as well as SPM for Scene-15, Sports, Corel, Caltech 101 and Caltech 256 datasets. L0 denotes level 0 of 1×1 subregions, L1 denotes level 1 of 2×2 subregions and L2 denotes level 2 of 4×4 subregions. SPM combines three spatial levels and achieves the best accuracy, while the accuracy of spatial levels at L1 and L2 is better than L0.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>L0</th>
<th>L1</th>
<th>L2</th>
<th>SPM</th>
<th>L0</th>
<th>L1</th>
<th>L2</th>
<th>SPM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scene-15</td>
<td>80.0</td>
<td>82.2</td>
<td>81.7</td>
<td>83.8</td>
<td>72.8</td>
<td>75.6</td>
<td>77.6</td>
<td>78.7</td>
</tr>
<tr>
<td>Sports</td>
<td>82.0</td>
<td>83.5</td>
<td>84.0</td>
<td>84.1</td>
<td>67.6</td>
<td>71.3</td>
<td>71.2</td>
<td>71.4</td>
</tr>
<tr>
<td>Corel</td>
<td>91.1</td>
<td>91.2</td>
<td>91.6</td>
<td>92.2</td>
<td>85.7</td>
<td>87.4</td>
<td>87.3</td>
<td>88.2</td>
</tr>
<tr>
<td>Caltech 101</td>
<td>52.1</td>
<td>58.1</td>
<td>62.1</td>
<td>66.7</td>
<td>37.4</td>
<td>43.6</td>
<td>48.1</td>
<td>50.4</td>
</tr>
<tr>
<td>Caltech 256</td>
<td>24.0</td>
<td>25.5</td>
<td>29.2</td>
<td>30.2</td>
<td>18.3</td>
<td>20.0</td>
<td>21.0</td>
<td>22.2</td>
</tr>
</tbody>
</table>

that the largest improvement by spatial division is achieved in Caltech 101 dataset, where the performances of both methods at level 1 are improved over 6% compared to level 0 and further improved about 4% at level 2. This may be because that objects in most images of Caltech 101 dataset are often located at the center, so spatial division is able to maintain the common spatial layout in each subregion and reduce the rate of false NN match.

Since SPM combines the I2C distances of all spatial levels, it achieves the best classification accuracy in each dataset as shown in Table 2. However, SPM requires calculating the I2C distance and learning weight for LI2C at each spatial level, so its computation cost is the sum of each spatial level and thus very expensive. Since the improvement of SPM is limited compared to single spatial level in these datasets, using only spatial division at level 1 or level 2 for classification is a better strategy when efficiency is the primary issue. For example, the Caltech 256 dataset contains much more images and classes than the others, so using spatial division at level 2 to learn its I2C distance is faster while its accuracy is close to that using SPM.

Although spatial division and SPM is able to improve the classification performances of both LI2C and NBNN, it should be emphasized that by learning the weight for each training feature, our LI2C is more discriminative than the unweighted I2C distance in NBNN in every spatial level and its combination, which again validate the effectiveness of our large margin framework for learning the weight.
4.4. Results on Hubness Score

The result of spatial division is encouraging for reducing the candidate feature set during the NN search. However, a large number of features in the training set are still stored in memory during the testing phase, because features in different subregions are required to be matched by features in the corresponding subregions for each test image. Such large number of training features not only occupy large memory space, but also require substantial computation cost for learning the weight associated with each feature. These problems can be overcome by our second method for improving the efficiency. This new method reduces the training features by the use of hubness score.
Figure 10: Comparing the classification accuracy of LI2C to NBNN for Caltech 101 in spatial level 2 with various percentages of preserved training features in the training set.

To evaluate the effectiveness of this method, we compare the performances of the training set with different sizes of preserved features. Here the amount of improvement in efficiency is represented by the percentage of preserved training features to the original, since less number of preserved features need less computation cost for the NN search and achieve better improvement on efficiency. Figure 9 shows the results of both LI2C and NBNN with percentages of preserved training features ranging from 1% to 100% for Scene-15, Sports and Corel datasets. In all the three datasets, the accuracy reduced by our LI2C method is nearly negligible even when only 5% training features are preserved, but this reduction starts to become obvious when the preserved training features are reduced from 5% to 1%. However, massive reduction of training features will not allow NBNN to maintain satisfactory result, as its accuracy drops quickly in every dataset. This result indicates that the performance of NBNN relies heavily on the size of candidate training set, while our method is able to maintain the classification accuracy when there are only a small number of training features due to the effectiveness of learned weight. It should be emphasized that this method for feature reduction not only accelerates the NN search, but also speeds up weight learning and requires less memory space.

We also evaluate the performance of combining both spatial division and hubness score. We use Caltech 101 dataset in spatial level 2 for this experiment with percentages of preserved features ranging from 1% to 100%. Results on both LI2C and NBNN are reported in Figure 10. Although only
15 images in each class are used for training in this dataset, which are much less than the training sets in the previous three datasets, we can see that the classification accuracy of our LI2C method is still close to the original until the percentage of preserved training features drops to below 10%, whereas the accuracy of NBNN drops much faster compared to LI2C.

4.5. Computation analysis

We also show the computation cost of our learning method as well as the accelerated NN search using our proposed two strategies. The average running time for updating the weight using our large margin optimization are shown in Table 3, where the running time for the NN search of each image without acceleration are also list as comparing. All of the experiments are running on **Intel x86 Xeon CPU E7320@2.13GHz**. We can see that in Scene-15, Sports and Corel datasets, due to the small number of classes, the weight learning procedure runs quickly and requires little computation cost. With the increased number of classes, more triplet constraints and more weights are required to be updated. Therefore the running time for updating the weight on Caltech 101 and 256 datasets are increased. However, such training cost is nearly negligible compared to the heavy computation cost for the NN search. As Table 3 shows, it requires about 1 to 4 minutes for the NN search of only one image. This means for a dataset like Caltech 101, which contains 1515 test images, the total running time for the NN search is about 32 hours. For the largest dataset of Caltech 256, which contains 6400 test images, the running time is increased to 17 days. Such large computation cost makes the I2C distance impractical for large scale dataset.

However, with our proposed acceleration method, the running time for the NN search can be significantly reduced. Taking Caltech 101 dataset as
example, using spatial division at level 2, the running time is reduced to 12.1 seconds as shown in Table 4. By combining both spatial division and hubness score, the computation cost can be further reduced. As Figure 10 in previous subsection shows, the classification accuracy of preserving 10% training features on spatial level 2 is comparable to the original for our LI2C. However, the computation cost is reduced to 4.3 seconds per image as shown in Table 4. So the total running time for the NN search of the whole test set is only 1.8 hours, which would make the I2C distance practical for applying to large scale dataset.

### 4.6. Comparing to Previous Results

We also compare our method with recent published results on these datasets. All the results are summarized in Table 5. The best classification accuracy in our experiment is achieved by combining LI2C with SPM in each dataset. Although we use single local feature in our experiment and do not even consider color information, our result is able to achieve state-of-the-art performance in most of these datasets. Specifically, in Scene-15 dataset, our result is able to outperform most of previous methods [16, 25, 26, 28]. The current state-of-the-art performance for this dataset is around 84% [29, 27, 24]. However, we find that the best result among these results reported in [29] use multi-scale and densely sampled keypoints for feature extraction as well as multiple feature combination, which is much too complex, while our method uses single local feature but is still comparable to most methods. Similar comparison is in Sports dataset, where our method is significantly outperform [20, 31] and is comparable to [29]. For Corel, Caltech 101 and Caltech 256 datasets, our method is also able to outperform most other methods that report their results recently.

### Table 4: The average running time for the NN search of each image on different spatial division level and various percentages of preserved training features in the training set. (second)

<table>
<thead>
<tr>
<th>Percentage of remaining training set</th>
<th>1%</th>
<th>2%</th>
<th>5%</th>
<th>10%</th>
<th>15%</th>
<th>20%</th>
<th>25%</th>
<th>30%</th>
<th>40%</th>
<th>50%</th>
<th>100%</th>
</tr>
</thead>
<tbody>
<tr>
<td>L0</td>
<td>1.9</td>
<td>2.8</td>
<td>5.2</td>
<td>9.0</td>
<td>12.8</td>
<td>16.8</td>
<td>20.8</td>
<td>24.8</td>
<td>30.4</td>
<td>33.1</td>
<td>78.0</td>
</tr>
<tr>
<td>L1</td>
<td>1.4</td>
<td>1.8</td>
<td>2.7</td>
<td>3.6</td>
<td>4.5</td>
<td>5.5</td>
<td>6.5</td>
<td>7.6</td>
<td>9.4</td>
<td>11.4</td>
<td>24.7</td>
</tr>
<tr>
<td>L2</td>
<td>1.6</td>
<td>2.2</td>
<td>3.3</td>
<td>4.3</td>
<td>5.1</td>
<td>5.6</td>
<td>6.1</td>
<td>6.6</td>
<td>7.3</td>
<td>7.9</td>
<td>12.1</td>
</tr>
</tbody>
</table>
Table 5: Comparing to recently published results for classification accuracy (%).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scene-15</td>
<td>Ours</td>
<td>83.8</td>
</tr>
<tr>
<td></td>
<td>Lazebnik et al. [16]</td>
<td>81.4</td>
</tr>
<tr>
<td></td>
<td>Liu et al. [24]</td>
<td>83.3</td>
</tr>
<tr>
<td></td>
<td>Rasiwasia et al. [25]</td>
<td>72.2</td>
</tr>
<tr>
<td></td>
<td>van Gemert et al. [26]</td>
<td>76.5</td>
</tr>
<tr>
<td></td>
<td>Bosch et al. [27]</td>
<td>83.7</td>
</tr>
<tr>
<td></td>
<td>Yang et al. [28]</td>
<td>80.3</td>
</tr>
<tr>
<td></td>
<td>Wu et al. [29]</td>
<td>84.1</td>
</tr>
<tr>
<td></td>
<td>Zhou et al. [30]</td>
<td>83.5</td>
</tr>
<tr>
<td>Sports</td>
<td>Ours</td>
<td>84.1</td>
</tr>
<tr>
<td></td>
<td>Li et al. [20]</td>
<td>73.4</td>
</tr>
<tr>
<td></td>
<td>Wu et al. [31]</td>
<td>78.5</td>
</tr>
<tr>
<td></td>
<td>Wu et al. [29]</td>
<td>84.2</td>
</tr>
<tr>
<td>Corel</td>
<td>Ours</td>
<td>92.2</td>
</tr>
<tr>
<td></td>
<td>Lu et al. [32]</td>
<td>77.9</td>
</tr>
<tr>
<td></td>
<td>Lu et al. [21]</td>
<td>90.0</td>
</tr>
<tr>
<td>Caltech 101</td>
<td>Ours</td>
<td>66.7</td>
</tr>
<tr>
<td></td>
<td>Yang et al. [28]</td>
<td>67.0</td>
</tr>
<tr>
<td></td>
<td>Wang et al. [33]</td>
<td>65.4</td>
</tr>
<tr>
<td></td>
<td>Wu et al. [29]</td>
<td>65.2</td>
</tr>
<tr>
<td></td>
<td>Gu et al. [34]</td>
<td>65.0</td>
</tr>
<tr>
<td>Caltech 256</td>
<td>Ours</td>
<td>30.2</td>
</tr>
<tr>
<td></td>
<td>Yang et al. [28]</td>
<td>27.7</td>
</tr>
<tr>
<td></td>
<td>Griffin et al. [23]</td>
<td>28.3</td>
</tr>
</tbody>
</table>
5. Conclusion and Future Work

In this paper, we introduced a novel method for nearest-neighbor classification by learning weighted Image-To-Class distance function. We reduced the effects of irrelevant patches e.g. cluttered background and enhanced relevant patches by learning their weights in a large margin framework. We also presented two approaches to reduce the heavy computation cost of the NN search. Experiments on several prevalent image datasets show promising performance of our method compared with other methods. Due to its success on image classification tasks, we would consider applying this framework to other object classification problems in future work where objects are represented as sets of features.

6. Reference


26


