# An evaluation of Nearest Neighbor Images-to-Classes versus Nearest Neighbor Images-to-Images

**Instructed by Professor David Jacobs**
**Phil Huynh**

## Abstract

*In 2008, Boiman et al. announced in their work "In defense of Nearest-Neighbor based image classification" an interesting point that the Nearest-Neighbor-based classifiers have been considerably undervalued. The inferior performance of most NN-based classifiers was not due to the nature of the classifier, but was mainly the consequence of using it in a mistakenly perceived context. The authors claimed that instead of finding NN-from-images-to-images, finding NN-from-images-to-classes significantly improves the accuracy, in their experiments up to 17%. A simple fix to the way NN was used has brought it to be among the best classifiers.*

*In the process of implementing the Leaf Recognition project, we had a chance to revalidate the above finding. We implemented both methods and tested on a big dataset of about 5,000 images. Against our expectation that the recommended NN-images-to-classes would greatly outperform the other, it actually insignificantly improves the accuracy from 70.2% to 70.4%. Our results versus their results raises the need for more investigation before confirming the robustness of the above finding.*

## 1. Introduction

Nearest-Neighbor (NN) based image classifier is believed to be the simplest multi-classes classifier that exists. Its simple nature of requiring no training phase, using a non-parametric model, and ease of implementation, makes NN favorable enough to be often the first method that gets implemented in a new experiment's setup or as a baseline in a test-bench to compare with much more complicated classifiers. In 2008, in [1], Boiman et al. claimed that NN-based classifier had been underestimated and mistakenly used. They pointed out that by finding nearest distances from images to classes, instead of from images to images, they could improve their experiment's accuracy more than 17%. This result was among the top 3, compared with the most current and complicated classifiers like Varma, Bosch Trees, SPM, Bosch SVM, etc…

In the on-going Field Guide research project carried out by the University of Maryland and Columbia University, we have about 5,000 leaf images in 143 species collected and described by Inner Distance Shape Context (IDSC) descriptors fed for training. An input leaf image is recognized by its best matched species. Naïve-Bayes NN classifier was chosen as the baseline classifier due to its simplicity. We have implemented both ways of using NN including finding the nearest distances from images to images and the recommended technique in [1] suggesting finding the nearest distances from images to species. Against our expectation that the recommended way would greatly outperform the first traditional one, it actually insignificantly improves the accuracy from 70.2% to 70.4% in our experiments. This finding versus the results in [1] raises an interesting question: in which conditions will NN-to-classes greatly outperform NN-to-images?

For comprehension, section 2 gives some background about bag-of-words model for image representation, IDSC shape descriptor and how IDSC is applied in our experiments. Naïve-Bayes Nearest Neighbor algorithm and how NN Images-to-images method differs from NN Images-to-Classes method will also be introduced. Section 3 discusses about the implementation details and results.

# 2. Background

## 2.1 Bag-of-words modeling

The term “bag-of-words” was originally a terminology in Natural Language Processing, which refers to a class of document classification techniques that consider documents as unordered set of words taken from a dictionary. Documents are classified by analyzing the frequencies of word occurrences. Analogous to this, in object recognition, an image can be treated as a document and words are image features. “Word” in images is not off-the-shelf like words in text in document, but the process of generating words involves three stages: feature detection, feature description, and codebook generation.

Feature detection finds the “best locations” in the image to sample. The features can be constructed around interest points such as scale-space extrema (e.g. SIFT keypoints [9]), or simply on windows extracted from the image at regular positions and various scales (e.g. HOG grids [11]), or even as simple as uniform spatially distributed points like in Shape Context [4] and IDSC [2]. The feature descriptors can be image patches, histograms of gradient orientations or color histograms; but to be useful they should be able to remain invariant to intensity, rotation, scale and affine variations to some extent. As these features are sensitive to noise and are represented in high dimension spaces, they are not directly used as words, but are categorized using a vector quantization technique such as k-means. The output of this discretization is the dictionary.

Based on the words, a classifier is then trained to recognize the categories of the images. Different techniques can be used such as Support Vector Machines (SVM), or Naive Bayes Classifiers [7]. Categorizing an image then simply entails extracting features, finding the corresponding words and applying the classifier to the set of words representing the image.

The bag-of-words representation of images is simple to use in a classifier, but it has one importation limitation. It ignores the spatial relationship among patches of the image. A global feature descriptor that captures information of a big region or the whole image (e.g. HOG, Shape Context, IDSC, etc…) may compensate to some extent.

## 2.2 Shape Context and Inner Distance Shape Context

*Shape Context (SC)*

SC describes the relative spatial distribution (distance and orientation) of landmark points around feature points. The idea is to pick n sample points $p_1$, $p_2$ ... $p_n$ on a shape contour. Consider the n−1 vectors obtained by connecting $p_i$ to all other points. The set of all these vectors is a rich description of the shape localized at that point, but this set is far too detailed since the number of points can be arbitrarily large. In addition, the set itself is sensitive to noise, distortion and articulation. A quantization step to reduce the details and errors is therefore taken. By definition, the shape context at point $p_i$ is defined as the coarse histogram $h_i$ of the relative coordinates of the remaining n-1 points:

$$\mathbf{h_i(k) = \# \{p_j : j \neq i,\ (x_j - x_i) \in bin(k)\}}$$

The bins are normally taken to be uniform in log-polar space. *(figure 1)*

The distance between two shape context histograms is defined using the Chi-squared statistic.

$$c(i,j) \equiv \frac{1}{2} \sum\nolimits_{1 \le k \le K} \frac{[h_{A,i}(k) - h_{B,j}(k)]^2}{h_{A,i}(k) + h_{B,j}(k)}$$

*Inner Distance Shape Context (IDSC)*

The shape context uses the Euclidean distance to measure the spatial relation between landmark points. This causes less discriminability power for complex shapes or shapes deformed by articulations *(figure 2&3)*. IDSC extends the SC feature by replacing Euclidean distance by the inner distance. The inner distance is defined as the length of the shortest path going through a subset of sampled points that does not exit the shape boundaries. The inner distance naturally captures the shape structure better than Euclidean distance. [2]

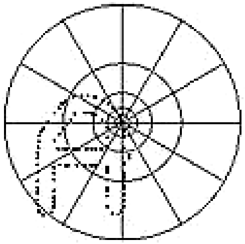

**Figure 1: log-polar space formed by 5 levels of distance and 12 levels of angle. The corresponding histogram has 60 bins.**

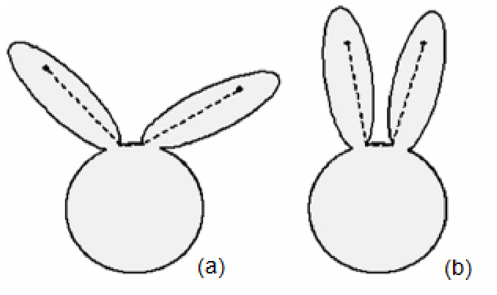

**Figure 2: The dashed lines denote inner distance shortest paths within the shape boundary that connect landmark points. (b) was articulated from (a) but the length of the inner distance shortest path unchanged.**

*Source: Haibin Ling , David W. Jacobs, "Shape Classification Using the Inner-Distance", 2007*

## 2.3 Naïve-Bayes NN (NBNN) algorithm

The detailed mathematical explanation of NBNN can be found in [1]. The key idea here is to make the Naïve-Bayes assumption of conditional independence of the features given the class membership. All descriptors of an image Q are i.i.d. given the class of Q. Then the cost to assign Q to a class C is modeled as the sum of the lowest costs of assigning each of descriptors $d_i$ of Q to C. A NN search algorithm finds the closest descriptor of each class $C_k$ associating with its distance (cost) to $d_i$. A class $\hat{C}$ with the lowest total cost is chosen as the classified class for Q. The algorithm can be summarized as follow:

$$\hat{C} := \mathrm{argmin}_C \sum_{i=1}^{n} \|d_i - \mathrm{NN}_C(d_i)\|^2$$

where:
$d_i$ *i=1..n* are descriptors of image Q
$NN_C(d_i)$ is the nearest neighbor descriptor of $d_i$ that belongs to class C.

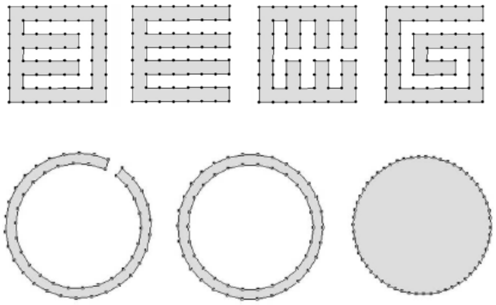

**Figure 3: With the same sample points, the distributions of Euclidean distances between all pair of points are indistinguishable for shapes on the rows, while the distributions of the inner-distances are quite different.**

*Source: Haibin Ling , David W. Jacobs, "Shape Classification Using the Inner-Distance 2007*

## 2.4 NN Images-to-Images vs. NN Images-to-Classes

Both methods refer to the Nearest-Neighbor-based classification techniques whose the goal is to find the class that best matches a queried image. In NN Images-to-Images approach, each queried image is compared to all known images and the class of the closest image is chosen and assigned to queried image. On the other hand, NN Images-to-Classes approach first pools all descriptors of all the images belonging to each class to form a single representation of that class. The queried image is then compared to all the classes and the closest class is chosen. Figure 4 summarizes algorithms of these two approaches.

# 3. Experiments

We tested both ways of using NN-based classifier as mentioned earlier on the 5-fold cross validation sets formed by the original Central Park dataset. The Central Park data set has about 5,000 normalized segmented stock leaf images in 143 species. Each crossed set was formed by round robin taking 20% of the images of each species used for testing and the remaining 80% used for validation. The results are averaged by the number of crossed datasets. Implementation details are mentioned in section 2.1. Test results are discussed in section 2.2.

## 3.1 Implementation

For leaf recognition, we have all the images segmented and normalized to eliminate scaling factor. In our first attempt, we used IDSC as the single feature descriptor. IDSC does very well in capturing the shape of silhouette images.  Though combining multiple descriptors may improve the accuracy, it does not affect the relative performance of the two approaches we are considering: NN images-to-classes and NN images-to-images. In image sampling process, 1024 points equally spread on the shape contour are chosen for each image to form sixteen uniform groups each has 64 points. Sixteen IDSC computations are done and the results are taken to average. For best performance, 5 inner-distance and 12 inner-angle levels are pre-configured for IDSC. Therefore, each image is described by 64 histograms, each has 60 slots. To enhance the speed of finding the nearest neighbors - the process which dominates the total cost, K-D tree search structures from Approximate Nearest Neighbor (ANN) software package [3] are used to answer the nearest distance request efficiently in O(log N) time. In our implementation, the choice of variance functions to measure the distance between two histograms does not really make a big difference. Though Chi-square or KL-divergence is usually a good choice, Euclidean distance works well in our experiments.  The implementation is summarized as given below.

```
NBNN Images-to-Classes algorithm

• Learning
        For all training images $I$:
        compute and add descriptors $d_1, d_2, ..., d_n$ to K-D tree $T_C$ , where
        $I \in class\ C$

• Recognition
    1.  Compute descriptors $d_1, d_2, ..., d_n$ of the query image
    2.  $\forall d_i\ \ \forall T_C$ compute the NN of $d_i$ to class $C$:
        $NN_C(d_i) := T_C \rightarrow findNN(d_i)$
    3.  $\hat{C} := \mathrm{argmin}_C \sum_{i=1}^{n} || d_i - \mathrm{NN}_C(d_i)||^2$
```

```
NBNN Images-to-Images algorithm

• Learning
        For all training images $I$:
        compute and add descriptors $d_1, d_2, ..., d_n$ to K-D tree $T_I$
• Recognition
    1.  Compute descriptors $d_1, d_2, ..., d_n$ of the query image
    2.  $\forall d_i\ \ \forall T_I$ compute the NN of $d_i$ to image $I$:
        $NN_I(d_i) := T_I \rightarrow findNN(d_i)$
    3.  $\hat{C} := class\_of(\mathrm{argmin}_I \sum_{i=1}^{n} || d_i - \mathrm{NN}_I(d_i)||^2 )$
```

**Figure4: Implementation routines for NN Images-to-Classes and NN Images-to-Images**

## 3.2 Result & Discussion

To our surprise, NN-Images-to-Classes method achieves 70.4% accuracy, just slightly better than NN-Images-to-Images by 0.2%. We have tested on our 5 crossed sets and all results seem consistent with each other. The mean accuracy and standard deviation of our experiments is given in the table 1.

| | Accuracy | Stdev |
|---|---|---|
| **Images-to-Images** | **70.2%** | **3.4%** |
| **Images-to-Classes** | **70.4%** | **1.8%** |

**Table1: Results on Central Park dataset**

Our experiments showed that there was not much difference in performance between the two approaches running on our datatset as compared to the huge performance boost achieved in [1]. One point worth noticing about the Central Park dataset we used, it contains 143 different species of the same class (leaf) whereas Caltech 101 or Caltech 256 used in [1] contains objects of different classes (e.g. airplane, book, elephant, etc…). We hold the hypothesis that the Images-to-Classes approach may obtain much better performance in inter-classes classification compared to the Images-to-Images approach, but there would not be that difference in case of intra-class classification.

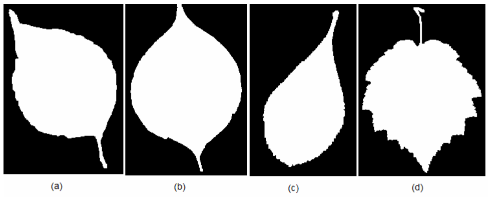

**Figure 5: Samples from Central Park dataset**
*(a) Cornus kousa* *(b)Cornus florida*
*(c) Crataegus crus-galli* *(d) Corylus colurna*

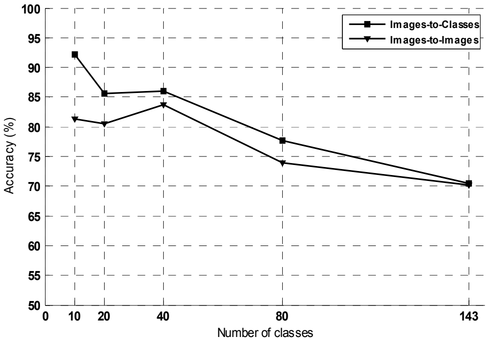

**Figure 6: Performance varies by the number of classes in both approaches**

However, there is little evidence that really explain why the NN-Images-to-Classes approach would greatly outperform NN-Images-to-Images. NN distances from images to classes are obviously smaller than NN distance from images to images. An image I of class C will have a smaller nearest distance to class C (found in the first approach) than the nearest distance to any of the other class-C images I' (found in second approach). This is good for the first approach, but the good is only half of the story. At the same time, the nearest distance from I to a different class C' also gets smaller than the nearest distance from I to its nearest class-C' image. Finally, when switching from NN-Images-to-Images to NN-Images-to-Classes, both the cost of getting I classified to C and not classified to C decrease. Without further investigation, it might be hard to convince one why one approach would greatly outperform the other.

On the practical side, with the support of a search structure like K-D tree, NN-Images-to-Classes does achieve faster running time. This approach maintains far fewer search structures than the other one so it reduces the time cost caused by cache missing and cache loading. In our experiment on a big dataset like Central Park, we have about 5,000 images versus only 143 species, the reduction in time cost is huge. (The number of search structures reduces from 5,000 to 143). Another thought would be, however, when the number feature descriptors describing an image are relatively small, like 64 in this case, it may not be worth to pay the high cost of building and maintaining search structures for finding distance between two images when implementing NN-Images-to-Images classification.

## 4. Conclusion

The result from this experiment suggests that the big performance gap between NN Images-to-Images and NN Images-to-Classes approach may not always hold, especially when classifying intra-class objects.

## 5. References

[1] Oren Boiman, Eli Shechtman, Michal Irani, "In defense of Nearest-Neighbor based image classification", CVPR, pp.1-8, 2008 IEEE Conference on Computer Vision and Pattern Recognition, 2008

[2] Haibin Ling , David W. Jacobs, “Shape Classification Using the Inner-Distance”, IEEE Transactions on Pattern Analysis and Machine Intelligence, v.29 n.2, p.286-299, February 2007

[3] D. Mount and S. Arya. “ANN: A library for approximate nearest neighbor searching”. In CGC 2nd Annual Workshop on Comp. Geometry, 1997.

[4] S. Belongie, J. Malik, and J. Puzicha. “Shape matching and object recognition usingshape contexts”. IEEE-PAMI, 24(4):509–522, April 2002

[5] Arya, S., D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions. *Journal of the ACM*, vol. 45, no. 6, pp. 891-923

[6] R.O. Duda, P.E. Hart, and D.G. Stork. Pattern Classification. Wiley, 2001.

[7] G. Csurka, C. Dance, L. Fan, J. Williamowski, and C. Bray, “Visual categorization with bags of keypoints,” in ECCV04 workshop on Statistical Learning in Computer Vision, 2004, pp. 59–74. [TBD]

[8] F. Jurie and B. Triggs, “Creating efficient codebooks for visual recognition,” in International Conference on Computer Vision, 2005.

[9] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” Int. Journal of Computer Vision, vol. 60, no. 2, pp. 91–110, 2004.

[10]J. Beis, D. G Lowe , "Shape indexing using approximate nearest-neighbour search in high-dimensional spaces". Conference on Computer Vision and Pattern Recognition, Puerto Rico: sn. pp. 1000–1006. doi: 10.1109/CVPR.1997

[11]N. Dalal and B. Triggs. “Histograms of oriented gradients for human detection”. InProc. CVPR, pages I:886–893, 2005

[12]J.H. Friedman, J.L. Bentley, and R.A. Finkel. “An Algorithm for Finding Best Matches in Logarithmic Expected Time”, ACM Trans, 3(3):209-226, 1977