Multiclass Classification on 3D Shapes

Honghua Li (301141580) *
School of Computing Science
Simon Fraser University
Burnaby, BC V5A 1S6
honghual@cs.sfu.ca

Abstract

I present a multiclass classification method for 3D shapes in this project. First, D2 shape distribution is used to generate discriminative representation of 3D shapes, which is developed to be invariant to rigid transformations and scalings. Next I employ the *all-versus-all* approach to decompose the multiclass classification problem into several binary classification problems, which are solved later by Support Vector Machine. The experiments performed show that the proposed parameter-free method is successful on a large testing dataset.

1 Introduction

Shape classification is one type of classification problems where the studied objects are shapes. This problem has its own properties since shapes are usually complicated, especially 3D shapes. To classify 3D shapes, we first have to determine the similarity among 3D shapes, which itself is a very hard problem. Once the similarity measurement is determined, the traditional multiclass classification methods can be used to solve this problem.

Measuring similarity between two 3D shapes is a fundamental problem in computer graphics, computer vision, and varieties of other fields. An intuitive method is to use shape matching, which however turned out to be a very complicated problem, because it involves several difficult problems, including pose registration, feature correspondence, or model fitting. Alternatively, Osada et al.[1] presented a simpler and robust method to discriminate between classes of 3D shapes (eg. bottles versus chairs). Their basic idea was to reduce the shape matching problem to the comparison of probability distributions.

A classification problem could be either binary or multiclass classification, and the latter is always much complicated. There are several binary classification algorithms that can be naturally extended to solve multiclass classification problems, including neural networks [2], decision trees [3], k-Nearest Neighbor [4], Naive Bayes [5], and Support Vector Machine [6]. Another widely used strategy is to decompose the multiclass classification problem into several binary classification tasks which can be solved efficiently by binary classifier. Several decomposition algorithms have been proposed, including *one-versus-all* (OVA) [7] and *all-versus-all* (AVA) [8]. Support Vector Machine [9][6] is the most successful and widely used binary classifier so far, which is used to perform AVA in this project. Please refer to [10] for a complete survey on multi-class classification methods.

2 Approach

We have two tasks on hand: measuring similarity between 3D shapes and multiclass classification.

*Final Project of Machine Learning (CMPT 726)

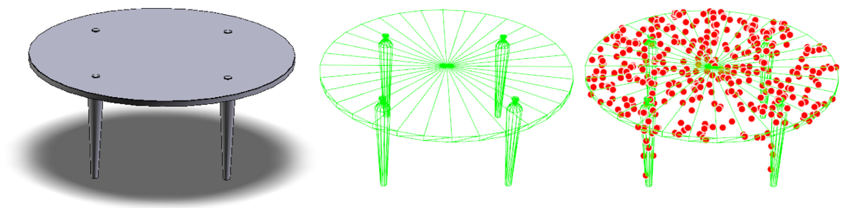

Figure 1: The 3D geometry (*left*), triangulations(*middle*) and uniform samples(*right*) of a table.

A 3D model can be presented as a probability distribution sampled from a shape function measuring geometry properties of the shape, which is called *shape distribution*. D2 shape function is used in this project, which measures the Euclidian distances between any pair of points on the shape surface. This shape function is chosen mostly for its computational efficiency and invariances to rigid transformations (rotations and translations) and scalings. A histogram is constructed for the samples of the D2 shape function, which is used as the signature of the original 3D shape.Sampling from this shape function is achieved via sampling from the shape surface first and then compute distances between any two random samples.

All-versus-all approach is used to decompose the multiclass classification problem to several binary classification problems, which are solved efficiently later by SVM.

Uniform Sampling on 3D Surface

Have chosen the D2 shape function, the next issue is to compute its distribution for any 3D shapes. The analytic calculation of D2 shape distribution exits only for certain geometrical primitives (eg. sphere, cylinder, cube). For arbitrary 3D shapes, stochastic methods can be employed to generate an approximation of the unknown underlying shape distribution. To do so, we need first generate uniform samples on the surface of the shape, and evaluate the Euclidian distances between each pair of samples.

A naive sampling method is to generate samples from mesh vertices. However, the irregular triangulation could make this method biased on certain regions of the shape. Take the table in Figure 1 for example, the triangulations are irregular and coarse in flat regions (eg. the top). In this case, simply sampling over all the mesh vertices can only generate lots of samples at the centroid of the table top.

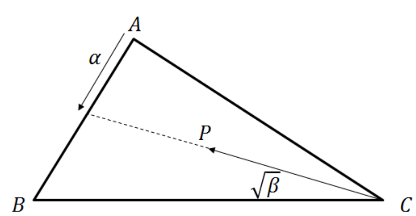

Figure 2: Sampling in a triangle.

To overcome this drawback, I generate uniform samples respect to the surface area of the shape. I first go through each triangle (this project focus on triangular models, but the proposed method can be easily extended to other 3D meshes), compute its area and store the cumulative area of all the triangles visited so far into an array. Next, a random number is generated between 0 and the total cumulative area, and the corresponding triangle is chosen by binary searching the array storing cumulative areas. To produce uniform samples on the selected triangle $T = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C})$, another two random number $\alpha, \beta \in [0, 1]$ are generated, and thus the sample $\boldsymbol{P}$ takes the form

$$\boldsymbol{P} = (1 - \alpha)\sqrt{\beta}\boldsymbol{A} + \alpha\sqrt{\beta}\boldsymbol{B} + (1 - \sqrt{\beta})\boldsymbol{C}.$$

This sampling method can also be interplaited in a *Sampling Importance Resampling* (SIR) manner. First, a uniform distribution within the range $[1, F]$, where F is the number of triangles on the mesh, is used as the proposal distribution to draw samples. Next, each sample is assigned a weight as the area of the corresponding triangle, then resample with respect to these weights. Note in this case the proposal distribution is trivial since it is defined on a discrete set. In practice, simply listing all the members in this discrete set can be used. Following SIR, another uniform sampling is performed on each sampled triangle to generate uniform samples on the entire shape surface, as showed in Figure 2.

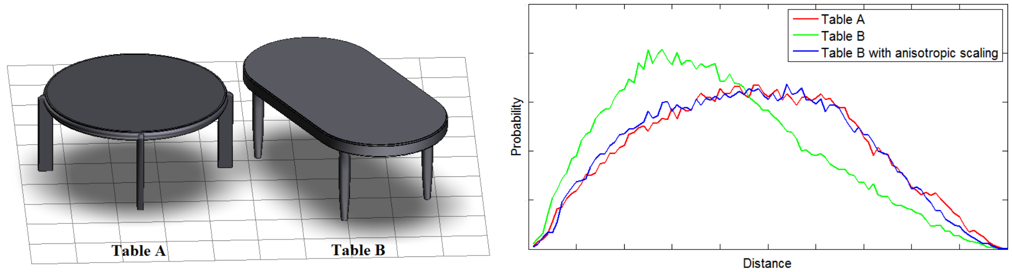

Figure 3: Anisotropic scaling is used to make D2 shape distribution invariant to anisotropic scaling.

D2 Shape Distribution

The N_p sample points generated in the last step can produce $N = N_p(N_p - 1)/2$ samples from the D2 shape distribution. A histogram with B fixed size of bins is constructed for the N samples, which is then used as a B dimensional presentation of the original 3D shape. In this project, I use $N_p = 500$, and $B = 100$.

It is notable that the B dimensional shape signature implies that the maximal sample values of D2 shape functions are scaled to be the same. In other words, all the shapes are isotropically scaled to have the same maximum of Euclidian distances among any points on the shape. So this shape signature is invariant to isotropic-scaling, which is a very good property in practice. However, isotropic-scaling invariance is not sufficient in certain cases. For example, table A and B in Figure 3 have very dissimilar shape distributions, which become very similar after anisotropic scaling. For implementation, a preprocessing is performed to scale the bounding boxes of all the shapes to unit cubes.

Multiclass Classification

Given a training set of the form (S_i, y_i), where S_i is the i-th example 3D shape and $y_i \in \{1, 2, \ldots, K\}$ is the corresponding label, I aim at learning a model $\mathbb{M}$ such that $\mathbb{M}(S_i) = y_i$ for new unseen 3D shapes. Once the 3D shapes are presented as shape signatures, the 3D shape classification becomes a traditional multiclass classification problem. In this project, I utilize OVA and AVA to decompose the multiclass classification problem into several binary classification problems, which are efficiently solved later using SVM.

OVA converts the classification problem among K classes into K binary classification problems, each of which discriminants one class versus all the rest $K - 1$ classes. Given a new example, the binary classifier producing the maximal output wins and assigns a label to this example. While AVA builds $K(K-1)/2$ binary classifiers, where each pair of classes have their own classifier. To assign a label to a new example, a majority voting is performed and the class with the most votes wins. SVM [11] is used as the binary classifier in this project.

3 Experiments

Training The Model

The training data set contains 50 shapes from 5 classes (10 for each): airplane, chair, table, bottle, and goblet, as listed in Figure 4. The D2 shape distribution of all the training shapes are showed in Figure 4, in which the histograms are presented as piecewise linear functions. We can see that the shapes from the same class have more similar D2 shape distributions than those from different classes.

One disadvantage of using SVM is that it is not parameter free. Tuning the parameters is always painful for the user and becomes unmanageable if we have huge numbers of binary classification problems to solve. To make the performance comparison between OVA and AVA fair and make my

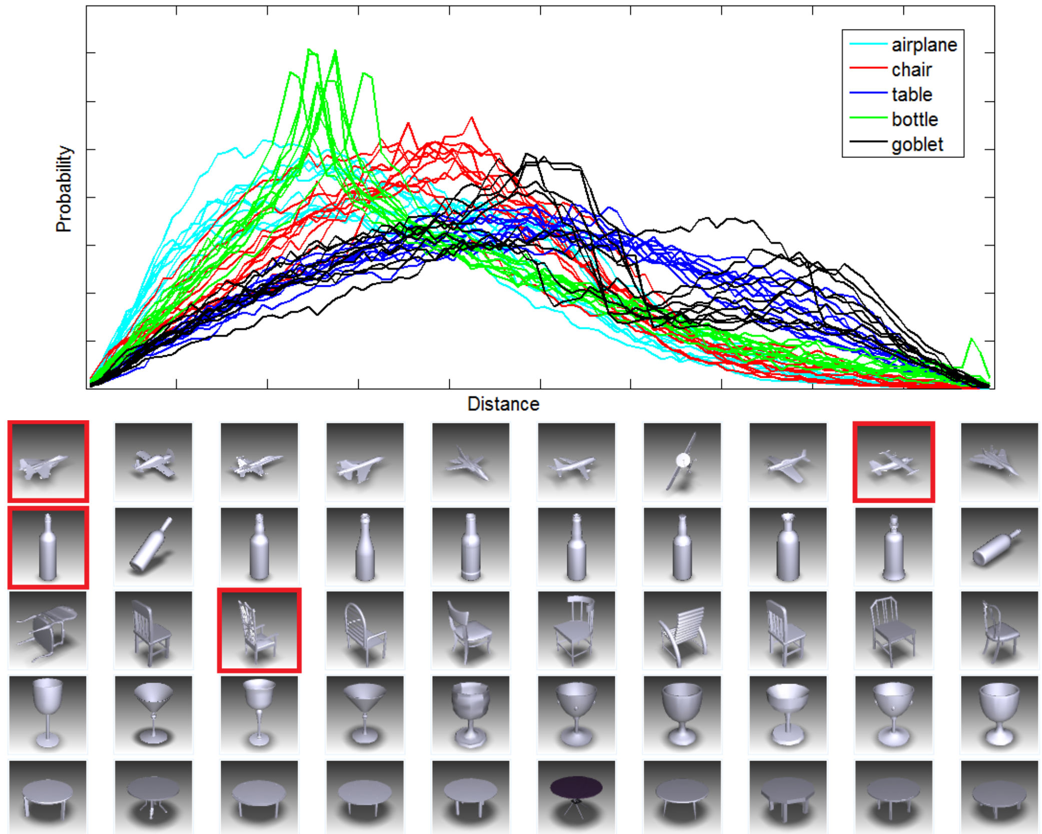

Figure 4: The training data set with 5 classes used in this project (*bottom*) and their D2 shape distributions (*top*). The proposed parameter-free all-versus-all approach obtains a 92% accuracy (the wrongly classified shaped are identified by red squares), while one-versus-one fails completely with a polynomial kernel of degree 3.

method parameter-free, I use polynomial kernels of degree 3 without bias term for as the model of all the binary classifiers.

For OVA, this simple kernel fails completely in all the K binary classification problems, where SVM just assigns every shape the same label to obtain an 80% (40 out of 50) accuracy. The reason might be each class is ”surrounded” by other classes in the feature space, and thus there is no a binary boundary between one class and all the rest classes. On the contrary, AVA performs very well with this simple polynomial model of binary classifiers, which gains a 92% accuracy on the training data set.

Labeling New Shapes

The testing dataset in Figure 5 contains 40 shapes from the 5 classes as before. The models learned by AVA method are used to determine the labels for the testing shapes, and correctly predicts 34 shapes out of 40 (85% accuracy). The red boxes are highlighting the wrongly predicted shapes. Note that the wrongly classified models always have relatively larger variances to the shapes in the class to which they belong. Since I just use very simple kernel for the SVM, the results can of course be improved by properly tuning the SVM parameters.

All these 3D models used in this project come from the 3D database of Nanyang Technology University.

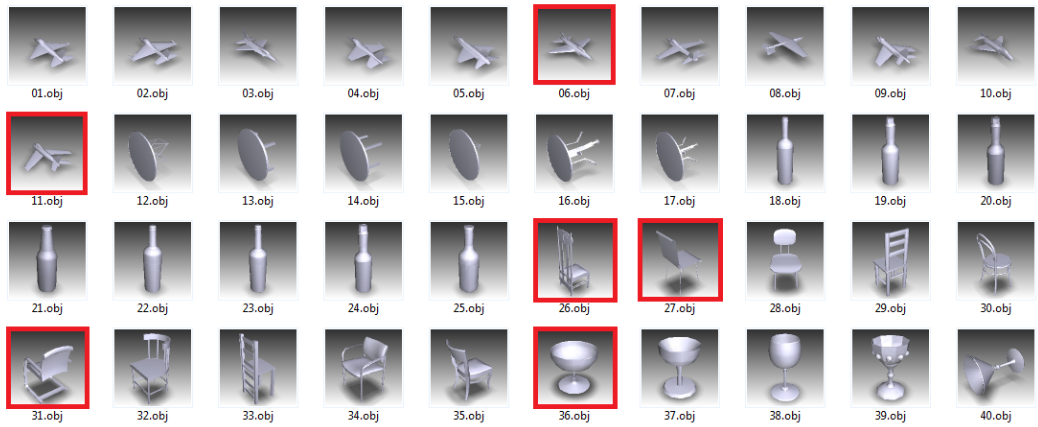

Figure 5: The testing data set. The proposed method correctly predicts 85% of these the shapes, and the wrongly classified shaped are highlighted by red squares.

4 Conclusion

In this project, a multiclass classification method is proposed, which is parameter free. First, a discriminative representation of 3D shapes is developed, which is invariant rigid transformations and scalings. Next, the multiclass classification problem is converted into binary classification problems using all-versus-all method, where SVM with polynomial kernel is applied as the the binary classifier.

The experiments performed on a training data set of size 50 and a testing data set of size 40 show that the proposed method is successful even with an easy binary classifier.

References

[1] Robert Osada, Thomas Funkhouser, Bernard Chazelle, and David Dobkin. Shape distributions. *ACM Transactions on Graphics*, 21(4):807–832, October 2002.

[2] Christopher M. Bishop. *Neural Networks for Pattern Recogntion*. Oxford University Press, 1995.

[3] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth International Group, Belmont, CA, 1984.

[4] Stephen D. Bay. Combining nearest neighbor classifiers through multiple feature subsets. In *Machine Learning, Proceedings of the Fifteenth International Conference (ICML '98)*, pages 37–45. Morgan Kaufmann, San Francisco, CA, 1998.

[5] Igor Kononenko. Successive naive bayesian classifier. *Informatica*, 17(2), 1993.

[6] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20:273, 1995.

[7] Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141, 2004.

[8] Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. In Michael I. Jordan, Michael J. Kearns, and Sara A. Solla, editors, *NIPS*. The MIT Press, 1997.

[9] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. In *Knowledge Discovery and Data Mining*, volume 2, pages 121–167. 1998.

[10] Mohamed Aly. Survey on multiclass classification methods, 2005.

[11] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.