EFFICIENT MANIFOLD LEARNING FOR 3D MODEL RETRIEVAL BY USING CLUSTERING-BASED TRAINING SAMPLE REDUCTION

Megumi Endoh, Tomohiro Yanagimachi, and Ryutarou Ohbuchi,

Dept. of Computer Science and Engineering
University of Yamanashi
Kofu, Yamanashi-ken, Japan
g10mk006AT gmail.com, tomohiro.yanagimachiAT gmail.com, ohbuchiAT yamanashi.ac.jp

ABSTRACT

Retrieval accuracy in content-based multimedia retrieval can be improved by using distance metric learned from distribution of features in input feature space. One way to achieve this is by dimension reduction via manifold-learning, such as Locally Linear Embedding [8]. While effective in improving retrieval accuracy, these algorithms have high computational cost that depends on feature dimensionality d and number of training samples N. In this paper, we explore a clustering-based approach to reduce number of training samples; it uses L cluster centers ($L<<N$) computed from N input features as training samples. We propose to use extremely randomized clustering tree [3] for clustering. Experiments showed that the proposed approach produces better retrieval performance than random sampling, and that the randomized tree is much faster than the k-means algorithm.

Index Terms— Content-based 3D model retrieval, distance metric learning, manifold learning, randomized tree clustering.

1. INTRODUCTION

Three-dimensional (3D) shape model has gained its status as a multi-media data type. It is used in designing and manufacturing mechanical parts, architectural design, in medical diagnosis, and in entertainment. User-generated contents are also flourishing. The Google 3D Warehouse now holds hundreds millions of 3D shape models. Accordingly, effective and efficient management of 3D models, especially via content based retrieval by their shape, has become quite important (e.g., [9]).

Generic processing pipeline for shape-based retrieval of 3D models starts with extraction of shape descriptors, or feature, from query and database 3D models. Then, distance, or dissimilarity among features of the query model and database models are computed to rank retrieval results.

Earlier 3D model retrieval algorithms used "fixed" distance, e.g., $L2$- or $L1$-norm. However, learned distance metric adapted to the feature distribution could significantly improve retrieval performance. In [6], so-called manifold learning algorithm such as *Laplacian Eigenmaps* (LE) [1] or *Locally Linear Embedding* (*LLE*) [8] is used for unsupervised non-linear dimension reduction. Given large enough number of feature samples in an ambient, or input, feature space, these algorithms find non-linear mapping onto lower dimensional subspace or "manifold" on which samples lie. By computing distance on the manifold, instead of the ambient feature space, better distance can be obtained. In addition to improved distance and thus improved retrieval accuracy, lower feature dimension also leads to lower temporal and spatial computational cost during the search through the database.

However, these manifold learning algorithms are computationally expensive. For example, during the training stage, LE and LLE constructs a graph connecting feature points, followed by Singular Value Decomposition (SVD) of its graph Laplacian. Using a naïve algorithm, graph construction requires $O(N^2)$ distance computations. If we use $L2$-norm, each distance requires $O(d)$ multiplications and additions. SVD of a $d \times N$ graph Laplacian matrix takes $O(dN^2)$ time assuming using a standard algorithm. We have to find a way to efficiently learn a manifold from a training set having a large number of samples, e.g., $N>10^4$ and a high feature dimensionality, e.g., $d=10^2$ or even $d=10^6$.

Assuming that the dimensionality of feature is given, approaches to deal with large number of samples have been explored. The simplest of the approaches is to pseudo-randomly or quasi-randomly [6, 7] down sample the training set. Another approach is to use Nystrom low-rank approximation [12, 2].

In this paper, we propose to cluster the N input features so that the computed L ($L<<N$) cluster centers become new training samples. Our hope is that, cluster centers, themselves created by unsupervised learning of feature distribution, could better represent distribution of input features than a simple down-sampling. For the clustering, we propose to use *Extremely Randomized Clustering* tree (ERC-tree) of Guerts, et al [3] for the task. While k-means is a "standard" clustering algorithm, it is quite slow if the feature dimension, number of features, and/or the number of cluster is large. And if clustering is too slow, total cost of learning, which is a sum of the cost of clustering and

manifold learning, can't be reduced. In addition, $L2$-norm used at the core of k-means algorithm is known to suffer in quality if applied to high-dimensional features. Experimental evaluation has shown that retrieval performances due to k-means and ERC-tree clustering are about equal, both of which are significantly better than one due to quasi-random sampling. In terms of computational cost, the ERC-tree is much faster than k-means.

2. METHOD

Our retrieval algorithm consists of two phases, the learning phase and retrieval phase.

The learning phase:

(1) **Extract shape feature vector:** Extract d-dimensional feature vectors from the N 3D models in the training database (*i.e.*, corpus).

(2) **Select training samples:** To reduce computational costs, subsample the training set down to L ($L \leq N$) features vectors. We compare three down-sampling methods, Quasi-Random Sampling (QRS), K-Means Clustering (kMC), and ERC-tree clustering (ERC).

(3) **Learn manifold:** Perform unsupervised learning of the m-manifold ($m \leq n$) spanned by the d-dimensional training samples by using LLE [8].

(4) **Approximate the manifold:** Construct a continuous approximation $\tilde{g}$ of the manifold by using the RBF kernel regression.

(5) **Project features of the models in the database:** Project features of all the models in the database onto the m-manifold using the approximation, and store the results together with the corresponding 3D models.

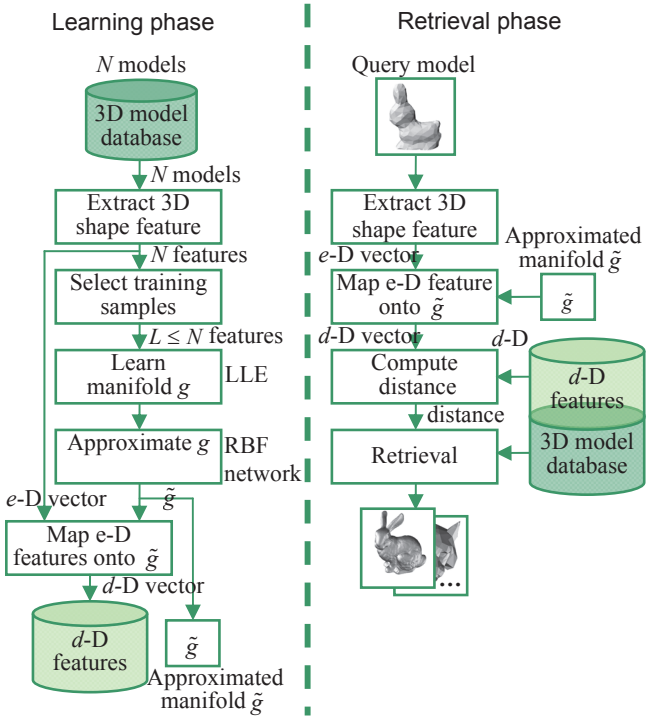
The retrieval phase:

(1) **Extract shape feature vector**: Extract a d-dimensional feature vector from the query model.

(2) **Project the feature onto the manifold**: Map the d-dimensional feature vector onto the approximated m-manifold $\tilde{g}$.

(3) **Compute distance on the manifold**: Compute distances from the query model to all the models in the database on $\tilde{g}$.

(4) **Retrieve and present the top p matches**: Retrieve the models in the database having the p-smallest distances from the query model.



g : Mapping from e-D feature vector to d-D feature vector, defined only at training samples.

$\tilde{g}$: Approximate, continuous mapping from e-D feature vector to d-D feature vector.

Figure 1. Retrieval pipeline using manifold learning for distance metric learning

2.1. Training Sample Selection

We will compare three different algorithms for sample reduction; one is the standard pseudo- (or quasi-) random sampling approach, and the other two are clustering based approach. We compared two clustering algorithms, k-means clustering and ERC-tree clustering.

- **Quasi-Random Sampling (QRS):** Instead of popular pseudo-random sampling, we used quasi-random sampling, which produced slightly better retrieval accuracy than pseudo-random sampling [6]. The set of experiments that follow uses *Niederreiter's* quasi-random sequence.
- ***k*-means Clustering (kMC):** Well known k-means clustering produces k clusters given the cluster number k as a parameter. We use standard, or Lloyds' algorithm.
- **Extremely Randomized Clustering tree (ERC):** The ERC-tree [3] is a randomized, tree-based clustering algorithm which subdivides the feature space into a half by each added tree node. Each subdivision is due to a hyper plane perpendicular to one of the coordinate axes. For each subdivision, the algorithm first randomly picks the dimension (or axis) to subdivide, and then randomly chooses the point (a scalar value) on the selected axis at which a separating hyperplane is placed. Subdivision of the feature space continues until the number of data points per subspace is below a set parameter S_{min}. While the $S_{\min}$ affects the number of clusters L, that is, number of reduced samples, exact value of L differ from a run to another due to the randomized nature of the algorithm. The L can also be controlled by pruning the tree, but we don't perform any pruning.

Since the feature space is subdivided into subspaces by a set of orthogonal hyperplanes, the quality of the clusters produced by ERC-tree may not seem as good as those produced by *k*-means. This is true in a low-dimensional feature space. However, *k*-means algorithm is disadvantaged in a high-dimensional feature space, as the algorithm uses *L*2-norm and that the *L*2-norm is known to behave poorly in a high-dimensional feature space. Consequently, clusters produced by ERC-tree could be better than *k*-means clusters. In terms of computational cost, ERC-tree is much faster than *k*-means.

2.2. Unsupervised Dimension Reduction

To obtain learned distance metric, we used manifold-based, unsupervised dimension reduction by LLE [8]. Given a set of N unlabeled e-dimensional ambient (original) features, LLE algorithm learns the d-dimensional subspace S spanned by the ambient feature set. The resulting S maps an input e-dimensional feature onto the d-dimensional feature in which $d < e$. For LLE, the manifold S is defined only for the trained features points. Thus, to handle out-of-sample features, e.g., that of query, manifold S must be approximated so that it is defined everywhere in the input space. Following [4], we used RBF-network algorithm for the approximation.

The retrieval algorithm stores, in the database, the (projected) d-dimensional feature together with the corresponding 3D model for later retrieval. As the dimension d is much smaller than the dimension e of the corresponding ambient feature, cost of distance computation and feature storage are significantly reduced.

We used the *LLE* implementation available in the *MatLab Statistical Learning Toolbox*. Mesh of the features for the LLE is computed by proximity in *L*2-norm, and the weight for the edges of the mesh is 0/1 (as opposed to the heat kernel). The *MatLab Neural Network Toolkit* is used for RBF-network. Parameters for the LLE and RBF-network, such as neighbourhood size for the LLE, reduced dimension d, bandwidth of the RBF, are found experimentally to have the best retrieval performance among the tested.

3. EXPERIMENTS AND RESULTS

We experimentally evaluated the efficacy of sample selection methods, both in terms of retrieval accuracy and computational cost, using LLE in 3D model retrieval setting. As shape feature, the experiment uses *Surflet-Pair Relation Histograms* (SPRH) [11] having d=625. To evaluate, we use SHREC 2006 [10] 3D model retrieval benchmark containing 1,814 of diverse 3D shape models. All the models in the SHREC 2006 benchmark database are classified into ground truth classes. The SHREC 2006 query set contains 30 out-of-sample models. To train LLE, we need large number of 3D models. Ideally, we'd like to have a large dataset, e.g., a database having >100k 3D models. However, as we can't obtain such a large benchmark dataset, we settled for the *National Taiwan University* (NTU) database [5] containing 10,908 unlabeled models. From the NTU database, we created/selected smaller number of features for LLE to learn by using the method described in Section 2.1. As numerical indices of retrieval performance, we use Mean Average Precision (MAP).

Figure 1 shows, for three sample reduction methods tested, retrieval performance in MAP. Due to the randomized nature of the ERC-tree (ERC), retrieval performance varies. Note also that, for ERC-tree, the number of clusters L also scatters a bit, as it can only be controlled indirectly by the parameter S_{min}. For *k*-means (kMC), on the other hand, the number of samples can be set exactly by the parameter k. In cases of QRS and kMC the number of training samples can be controlled precisely, and retrieval performances indicated are averages of 5 trials.

The figure shows that distance metric learning in the form of unsupervised dimension reduction by using LLE does improve retrieval performance, compared to the case using fixed distance as well as the case using LLE combined with QRS. It also shows that, in terms of retrieval performance, for the number of samples tested, ERC is equal or better than the much slower kMC. "Raw" performance of MAP=0.248 using fixed distance is pushed up to MAP=0.32~0.33 for the ERC-tree clustering.

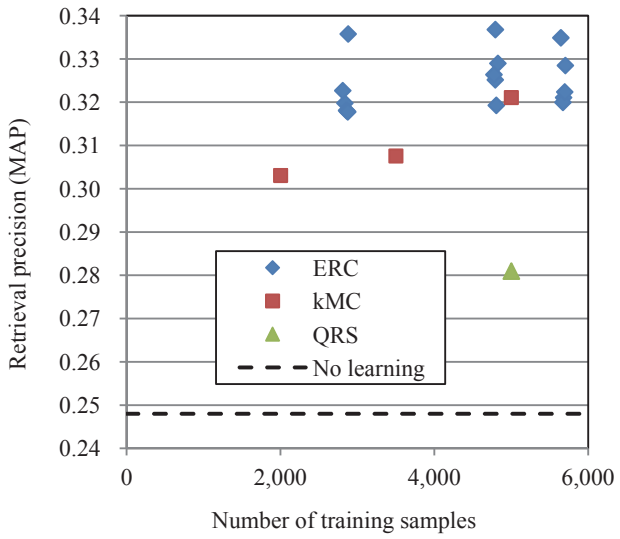


Figure 1. Training sample reduction methods and retrieval performance.

Figure 2 compares total cost of distance metric learning, that is, sum of time required for sample reduction and LLE learning. Broken lines are for clustering only, while solid lines are for total learning time. Total learning time includes clustering for sample reduction and LLE learning. The latter includes LLE learning, RBF approximation, and projection of 1,814 database features.

It is clear from the graph that kMC is too expensive; sample reduction by kMC took more time than manifold learning itself. As the number of output samples (i.e., clusters) increases, computation time for *k*-means increases very rapidly. On the other hand, temporal cost of ERC is significantly smaller than LLE learning, efficiently reducing number of samples. Temporal cost of ERC clustering is only a fraction of that of kMC.

Overall, sample reduction by ERC combined with LLE-based dimension reduction performed well both in terms of retrieval accuracy and computational cost.

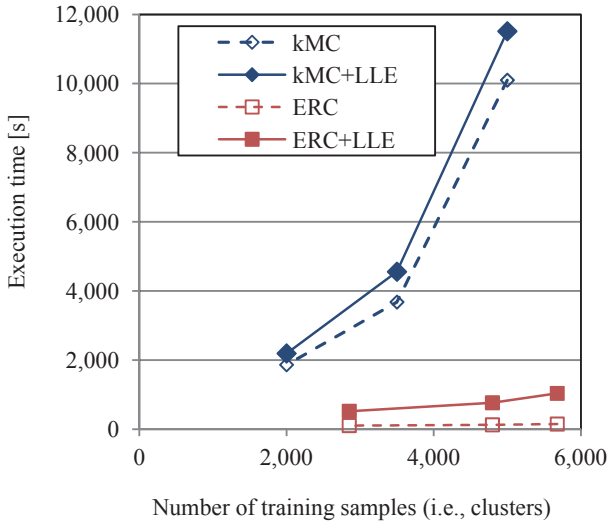


Figure 2. Number of training samples (i.e., clusters) after sample reduction to time needed for sample reduction.

4. SUMMARY AND FUTURE WORK

In this paper, we proposed a clustering-based sample number reduction to improve efficiency of manifold-learning based unsupervised distance metric learning. Experimental evaluation showed that proposed reduction method using Extremely Randomized Clustering (ERC) trees [3] significantly outperforms (quasi-) random sampling in terms of retrieval accuracy. If compared to *k*-means based sample reduction, ERC-tree based approach is much faster, while attaining equal or better retrieval accuracy.

Further experiments are required to quantify the efficiency and efficacy of proposed sample reduction method. We need to significantly scale up the training as well as benchmark (retrieval target) database. As current 3D model retrieval benchmarks are limited in their size to about ~1k labeled 3D models, we probably need to use benchmarks in image and other media data type for the experiment. Effects of different features and different datasets having different feature dimensions and distributions must be evaluated.

Proposed method reduced number of training samples by using clustering to make LLE computationally reasonable. However, even after reducing number of samples, training LLE for a set of features having very high dimensionality (e.g. d>50k) is difficult, both in terms of time and storage cost. We are currently investigating ways to effectively and efficiently reduce feature dimensionality so that LLE can be applied, without losing retrieval accuracy, to a large number of features having a very high dimensionality.

5. ACKNOWLEDGEMENTS

The authors would like to thank those who carefully reviewed the paper. The authors also would like to thank those who created benchmark databases. This research has been funded in part by the Ministry of Education, Culture, Sports, Sciences, and Technology of Japan (No. 18300068).

6. REFERENCES

[1] M. Belkin, P. Niyogi, Laplacian eigenmaps for dimensionality reduction and data representation, *Neural Computation*, **15**(6), pp. 1373-1396, (2003).

[2] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the Nyström method, *IEEE PAMI*, **26**(2), pp. 214–225, 2004.

[3] P. Guerts, D. Ernst, L. Wehenkel, Extremely randomized trees, *Machine Learning*, 2006, **36**(1), 3-42, (2006).

[4] X. He, W-Y. Ma, H-J. Zhang, Learning an Image Manifold for Retrieval, *Proc. ACM Multimedia 2004*, pp. 17-23 (2004)

[5] NTU 3D Model Database ver.1 http://3d.csie.ntu.edu.tw

[6] R. Ohbuchi, J. Kobayashi, Unsupervised Learning from a Corpus for Shape-Based 3D Model Retrieval, *Poster paper, Proc. ACM MIR 2006*, pp. 163-172, (2006).

[7] R. Ohbuchi, A. Yamamoto, J. Kobayashi, Learning semantic categories for 3D Model Retrieval, *Proc. ACM MIR 2007*, pp. 31-40, (2007).

[8] S.T. Roweis, L.K. Saul, Nonlinear Dimensionality Reduction by Locally Linear *Embedding, Science*, **290**(5500), pp.2323-2326, (2000).

[9] J.W.H. Tangelder, R. C. Veltkamp: A survey of content based 3D shape retrieval methods. *Multimedia Tools Appl.* **39**(3): 441-471 (2008)

[10] R. C. Veltkamp, R. Ruijsenaars, Michela Spagnuolo, R. Van Zwol, F. ter Haar, SHREC2006 3D Shape Retrieval Contest, Utrecht University Dept. Information and Computing Sciences *Technical Report UU-CS-2006-030* (ISSN: 0924-3275)

[11] E. Wahl, U. Hillenbrand, G. Hirzinger, Surflet-Pair-Relation Histograms: A Statistical 3D-Shape Representation for Rapid Classification, *Proc. 3DIM 2003*, pp.474-481, 2003.

[12] C. Williams, M. Seeger, Using the Nyström method to speed up kernel machines. *Advances in Neural Information Processing Systems* **13**, pp. 682-688, (2001).