Takahiko Furuya, Ryutarou Ohbuchi, Accurate Aggregation of Local Features by using K-sparse Autoencoder for 3D Model Retrieval, *short paper*, ACM International Conference on Multimedia Retrieval 2016 (ICMR 2016), June 6-9, New York, NY, USA. (2016)

Accurate Aggregation of Local Features by using K-sparse Autoencoder for 3D Model Retrieval

Takahiko Furuya
University of Yamanashi
4-3-11 Takeda, Kofu-shi
Yamanashi-ken, 400-8511, Japan
+81-55-220-8570
takahikof AT yamanashi.ac.jp

Ryutarou Ohbuchi
University of Yamanashi
4-3-11 Takeda, Kofu-shi
Yamanashi-ken, 400-8511, Japan
+81-55-220-8570
ohbuchi AT yamanashi.ac.jp

ABSTRACT

Aggregating a set of local features has been used widely to realize recognition or retrieval of multimedia data including 2D images and 3D models. A number of feature aggregation algorithms (e.g., Bag-of-Features [5], Locality-constrained Linear coding [26], or Fisher Vector coding [23]) have been proposed. They first learn a codebook, or a set of codewords, by clustering the local features and then encode these local features by using the learned codebook. Despite the great success of these feature aggregation algorithms, we argue that they are not necessarily optimal in terms of accuracy since their codebook learning and feature encoding are computed separately. In this paper, we propose two novel feature aggregation algorithms based on *k*-Sparse Autoencoder (*k*SA) [20] that realize more accurate local feature aggregation. Our proposed algorithms, called *Database-adaptive kSA* (*DkSA*) aggregation and *Per-data-adaptive kSA* (*PkSA*) aggregation, *jointly* optimize codebook learning and feature encoding. In addition, the *k*SA-based feature encoding enhances saliency of local features due to *k*-sparseness constraints and non-negativity constraints. Of the two proposed algorithms, the P*k*SA aggregation exploits reconstruction error of a local feature derived from the *k*SA for more accurate aggregated feature. Experimental evaluation using a shape-based 3D model retrieval scenario showed that the retrieval accuracy of our proposed algorithms are superior to the existing feature aggregation algorithms we have compared against.

Keywords

Feature aggregation, feature encoding, autoencoder, sparse coding, local feature, bag-of-features, content-based retrieval, 3D shape.

1. INTRODUCTION

Aggregation of local features is an important technological component for effective and efficient comparison among multimedia data objects, including 2D images, 3D models, etc. Recently, for the task of classification or retrieval of 2D images, "end-to-end" deep learning has achieved higher accuracy than the conventional approach which combines hand-crafted local features and their aggregation. On the other hand, recent studies (e.g., [10]) have shown that aggregating features derived from a deep neural network is also effective for 2D image retrieval. Also, for shape-based 3D model retrieval (3DMR) on which we focus in this paper, extracting local features from a 3D model and aggregating them per 3D model remains as one of the most promising approaches. We thus think that improving feature aggregation is still quite meaningful for progress of multimedia information retrieval.

Goal of local feature aggregation is to generate a feature vector that reflects, as well as possible, distribution of local features in their feature space. Feature aggregation generally consists of two steps, i.e., *encoding* and *pooling*. A set of codewords are chosen, typically, by clustering the set of local features. A local feature is encoded by using codewords around the feature and statistics associated with the codewords. Such statistics as density, mean, and/or variance, can be used for the encoding. For pooling, encoded local features are accumulated into a single feature per 3D model. For efficient inter-feature comparison, aggregated feature better be compact.

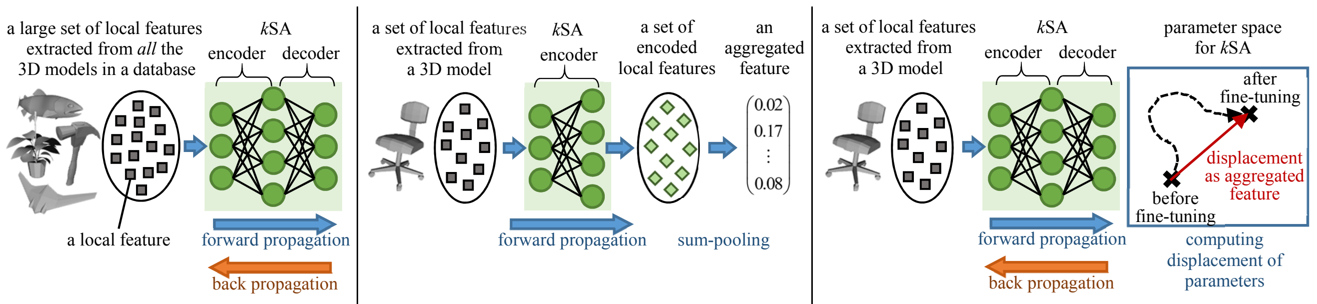
Many algorithms for feature aggregation have been proposed [12]. They can be roughly classified into two groups; sparse coding (SC)-based approach and higher-order statistics (HS)-based approach. The SC-based approach (e.g., [5][19][26]) sparsely encodes a local feature by using weighted linear sum of its neighboring codewords. Locality-constrained Linear (LL) coding [26] is one of the most representative SC-based methods. Bag-of-Features (BF) [5] is a special case of the SC-based approach in which a local feature is assigned to its nearest single codeword by vector quantization. On the other hand, the HS-based approach (e.g., [14][23][27]) tries to encode local features by using higher-order statistics around the codewords. Fisher Vector (FV) coding [23] employs mean and variance of local features around the codeword. Vector of Locally Aggregated Descriptor (VLAD) [14] approximates the FV by using only mean around the codeword.

These feature aggregation algorithms described above have been successfully utilized for recognition or retrieval of 2D images or 3D models. However, they are not necessarily optimal in terms of accuracy since the codebook learning and the feature encoding steps are processed separately. That is, as a pre-processing, a codebook (i.e., a set of codewords) is first learned by clustering the set of local features. Then, each local feature is encoded by using the learned codebook. This "greedy" framework is adopted by most of the feature aggregation algorithms including BF, LL[1], FV, VLAD, and Triangular Encoding (TE) [4]. The greedy approach could reduce expressive power of the aggregated features.

[1] Although joint optimization framework is proposed in [26], LL is typically used along with a codebook learned by *k*-means clustering in the literature.

ICMR'16, June 06-09, 2016, New York, NY, USA
© 2016 ACM. ISBN 978-1-4503-4359-6/16/06…$15.00
DOI: http://dx.doi.org/10.1145/2911996.2912054

Takahiko Furuya, Ryutarou Ohbuchi, Accurate Aggregation of Local Features by using K-sparse Autoencoder for 3D Model Retrieval, *short paper*, ACM International Conference on Multimedia Retrieval 2016 (ICMR 2016), June 6-9, New York, NY, USA. (2016)


a large set of local features extracted from *all* the 3D models in a database
*k*SA
encoder
decoder
a local feature
forward propagation
back propagation
a set of local features extracted from a 3D model
*k*SA
encoder
a set of encoded local features
an aggregated feature
0.02
0.17
⋮
0.08
forward propagation
sum-pooling
a set of local features extracted from a 3D model
*k*SA
encoder
decoder
parameter space for *k*SA
after fine-tuning
displacement as aggregated feature
before fine-tuning
forward propagation
back propagation
computing displacement of parameters

(a) training *k*SA (pre-processing). (b) D*k*SA aggregation. (c) P*k*SA aggregation.

Figure 1. Overview of the proposed feature aggregation algorithms using *k*-Sparse Autoencoder.

In this paper, aiming for more accurate aggregation of local features, we propose a pair of novel SC-based feature aggregation algorithms. The algorithm is based on *k*-Sparse Autoencoder (*k*SA) proposed by Makhzani et al. [20]. A *k*SA-based feature aggregation can be more accurate than the existing ones for two reasons. Firstly, *k*SA can jointly optimize codebook learning and feature encoding as *k*SA is trained by alternating encoding of local features and updating of codewords (i.e., updating of connection weights of *k*SA neurons) so that reconstruction error of the local features is minimized. Similar joint optimization framework is employed by Sparse Autoencoder (SA) [21] and KSVD [2], which are popular sparse coding algorithms. Secondly, feature encoding by *k*SA could enhance saliency of local features since the *k*SA has two constraints considered to aid accurate feature encoding, that are, *k*-sparseness and non-negativity. *K*-sparseness constrains feature encoding to use a small number (*k*) of codewords correlated to the local feature. Non-negativity forces coefficients weighting the codewords to be non-negative. Several state-of-the-art SC-based algorithms, e.g., LL coding and Localized Soft-assignment (LS) coding [19], showed these constraints improve accuracy of encoded features.

Our two *k*SA-based feature aggregation algorithms are called *Database-adaptive kSA* (*DkSA*) aggregation and *Per-data-adaptive kSA* (*PkSA*) aggregation. D*k*SA (Figure 1b) is a straight-forward approach for feature aggregation using *k*SA. As a pre-processing (Figure 1a), *k*SA is trained by using numerous and diverse local features sampled from the database (hence *database-adaptive*). After training, each of a set local features extracted from a 3D model is encoded by using the database-adapted *k*SA. The set of encoded local features are accumulated into a single feature vector per 3D model by sum-pooling for efficient comparison. P*k*SA (Figure 1c), on the other hand, aims at generating aggregated features more expressive than D*k*SA by exploiting *reconstruction error* of local features derived from the *k*SA. *k*SA is first trained in the same manner as D*k*SA (Figure 1a). Then, the database-adapted *k*SA is additionally trained, or *fine-tuned,* by using a set of local features extracted from each 3D model (hence *per-data-adaptive*). An aggregated feature for the 3D model is a set of displacements of parameters, or weights, caused by the fine-tuning. Table 1 compares characteristics of SC-based feature aggregation algorithms we will evaluate in the experimental section. The D*k*SA and the P*k*SA have potential to yield expressive aggregated features since both of them satisfy all the criteria essential for accurate feature aggregation, i.e., joint optimization of codebook and feature encoding, *k*-sparse constraint, and non-negative constraint. Exploiting reconstruction error of local features for P*k*SA is novel for the group of SC-based feature aggregation algorithms.

Table 1. Comparison of sparse coding-based feature aggregation algorithms.

algorithms	*k*-sparseness	non-negativity	joint optimization of codebook and encoding	exploiting reconstruction error
SA [21]	No	**Yes**	**Yes**	No
TE [4]	No	**Yes**	No	No
KSVD [2]	**Yes**	No	**Yes**	No
LL [26]	**Yes**	**Yes**	No[1]	No
D*k*SA (ours)	**Yes**	**Yes**	**Yes**	No
P*k*SA (ours)	**Yes**	**Yes**	**Yes**	**Yes**

Experimental evaluation under a 3DMR scenario shows that the proposed *k*SA-based feature aggregation algorithms significantly outperform existing SC-based and HS-based counterparts.

Contributions of this paper can be summarized as follows.

- Proposition of two *k*-Sparse Autoencoder based feature aggregation algorithms, that are, D*k*SA and P*k*SA, for more accurate aggregation of local features.
- Quantitative evaluation of D*k*SA and P*k*SA using multiple local features and multiple benchmark datasets for 3DMR.

2. PROPOSED ALGORITHM

2.1 Database-adaptive *k*SA Aggregation

As with the previous feature aggregation methods, D*k*SA (Figure 1b) aggregation consists of two processing steps, that are, encoding and pooling of local features. At the encoding stage, a set of local features extracted from a 3D model is first normalized by using ZCA-whitening [3] so that local features have zero means and all the pair of dimensions are uncorrelated. Then, each whitened local feature is fed into the trained *k*SA to obtain unit activations in the hidden layer. Feature encoding is performed as $\mathbf{h} = \sigma(\mathbf{y}) = \sigma(\mathbf{x}\mathbf{W}_e)$, where $\mathbf{h}$ is an encoded local feature (or unit activation in the hidden layer), $\mathbf{x}$ is a input (whitened) local feature, $\mathbf{W}_e$ is a weight matrix for encoder, and σ is a non-linear activation function. By using hidden layer with n units, d-dimensional local feature $\mathbf{x}$ is encoded into n-dimensional feature ($n > d$). Note that our *k*SA includes bias terms for both encoder and decoder, although the bias term is omitted from the equation for notation simplicity.

Activation function σ strongly influences saliency of encoded local features. Our activation function consists of Rectified Linear Units (ReLU) [16] and *k*-largest value selection. Specifically, y_i, which is i-th element of $\mathbf{y}$, is first rectified by using ReLU, i.e., $z_i = \max(0, y_i)$. ReLU ensures non-negativity of feature encoding. Then, k largest elements in $\{z_i \mid i = 1, \ldots, n\}$ are kept while the rest is set to 0.

At the pooling stage, the set of encoded local features is aggregated into a single feature vector per 3D model by summing. Therefore, the number of dimensions for aggregated feature is also n. Finally, the aggregated feature vector is normalized by using power normalization followed by L2 normalization as with [23]. Cosine similarity is used to compare a pair of D*k*SA-aggregated features.

D*k*SA has two hyper-parameters that require manual tuning, i.e., the number of units in the hidden layer *n* and the number of non-zero elements in the hidden layer *k*. In the experiments, *n* is typically set to 8,000 and small *k* (e.g.,*k*=3) is used.

Training of the *k*SA is performed as follows. The *k*SA is trained by minimizing mean squared reconstruction error of local features. We use *T*=250k local features randomly sampled from the database for training. We use Stochastic Gradient Descent (SGD) with mini-batch. Each mini-batch contains 200 local features randomly selected from the set of *T* training local features. To adaptively assign a learning rate to each parameter (i.e., each connection weight between neurons), we use AdaGrad algorithm [6] with an initial learning rate = 0.2. Training is regularized with weight decay coefficient $\lambda = 10^{-4}$. The optimization is iterated for 50 epochs.

2.2 Per-data-adaptive *k*SA Aggregation

P*k*SA (Figure 1c) aims at generating more accurate aggregated features than D*k*SA by exploiting reconstruction error of local features. For pre-processing, *k*SA is trained in the same manner as D*k*SA except for the initial learning rate = 0.05 for AdaGrad. We use this initial learning rate since it yielded slightly higher retrieval accuracy in the preliminary experiments.

To aggregate a set of local features extracted from a 3D model *M*, the database-adapted *k*SA is additionally trained, or fine-tuned, by using the local features of the 3D model *M*. The *k*SA parameters $\mathbf{W}_e$ for encoder and $\mathbf{W}_d$ for decoder are fine-tuned to reconstruct the local features of the 3D model *M* well. Then, we compute displacement, or differential, of parameters due to fine-tuning as $\Delta\mathbf{W}_d = \mathbf{W}^*_d - \mathbf{W}_d$, where $\mathbf{W}^*_d$ and $\mathbf{W}_d$ indicate weight matrices of the decoder after and before fine-tuning, respectively. The displacement matrix $\Delta\mathbf{W}_d$ is then vectorized to form a P*k*SA-aggregated feature for the 3D model *M*. The P*k*SA-aggregated feature vector has $D=(n+1)\times d$ dimensions where "+1" means that our *k*SA has bias terms. We can also use parameter displacement of the encoder, i.e., $\Delta\mathbf{W}_e = \mathbf{W}^*_e - \mathbf{W}_e$, for aggregation. However, preliminary experiments showed that decoder displacement $\Delta\mathbf{W}_d$ consistently outperformed encoder displacement $\Delta\mathbf{W}_e$.

Note that fine-tuning for P*k*SA aggregation is performed per 3D model. That is, the database-adapted *k*SA is duplicated for each 3D model, and each duplicated *k*SA is fine-tuned by using the set of local features extracted from the 3D model. For fine-tuning, we employ Steepest Descent (SD) algorithm; all the local features of the 3D model are used to compute gradients of the parameters. We use SD instead of SGD (with mini-batch) so as to exclude randomness from the result of fine-tuning. We use AdaGrad with initial learning rate = 0.05 and fine-tuning is iterated for 50 epochs.

Since P*k*SA aggregation produces higher-dimensional aggregated feature vectors than D*k*SA aggregation, we perform dimension reduction of the P*k*SA-aggregated features for efficient inter-feature comparison. We use Sparse Random Projection (SRP) [1] algorithm to compress the P*k*SA-aggregated features down to 8,000 dimensions. Preliminary experiments showed that the SRP down to 8,000 dimensions did not degrade retrieval accuracy. Comparison among the dimension-reduced P*k*SA-aggregated feature vectors is done in a manner identical to that of D*k*SA; aggregated features are power-normalized and then L2-normalized for Cosine similarity.

P*k*SA has the same hyper-parameters as D*k*SA, i.e., the number of units in the hidden layer *n* and the number of non-zero elements in the hidden layer *k*. In the experiments, *n* is to 500 and *k* is set to around 10, which is slightly larger than *k* for D*k*SA.

3. EXPERIMENTS AND RESULTS

3.1 Experimental Setup

To validate efficacy of the proposed feature aggregation algorithms, we use four benchmark databases for 3DMR; the Princeton Shape Benchmark (PSB) [24], the Engineering Shape Benchmark (ESB) [13], the SHREC 2011 Non-Rigid watertight meshes dataset (SH11NR) [17], and the SHREC 2014 Large-scale Comprehensive 3D shape retrieval dataset (SH14LC) [18]. For all the benchmarks, one of 3D models in the database is used as a query and remaining 3D models are used as retrieval targets. We use Mean Average Precision (MAP) [%] as an index of retrieval accuracy. Since optimization of the proposed algorithms using SGD have randomness, all the experiments for D*k*SA and P*k*SA are performed three times and their average MAP are reported. Table 2 summarizes the number of non-zero units *k* used for D*k*SA and P*k*SA aggregation. For another hyper-parameter, i.e., the number of hidden units, we use *n*=8,000 for D*k*SA and *n*=500 for P*k*SA unless otherwise stated.

Table 2. Number of non-zero units *k* used for experiments.

	POD	LSF	SI	RoPS	DSIFT	MO1SIFT
D*k*SA	3	3	3	3	1	3
P*k*SA	20	10	10	20	5	20

We compare the proposed feature aggregation algorithms against six existing feature aggregation algorithms, i.e., BF [5], LL [26], FV [23], VLAD (VL) [14], SV [27], and DM [9]. We also apply three sparse coding algorithms to local feature aggregation, i.e., Sparse Autoencoder (SA) [21], Triangular Encoding (TE) [4], and KSVD [2]. For sparse coding-based aggregation methods including BF, LL, SA, TE, and KSVD, we learn 8k codewords. On the other hand, for higher-order statistics-based methods including VL, FV, and SV, the number of codewords are determined so that dimensionality of aggregated feature vector becomes about 300k. For DM, we use 250k local features to generate a manifold graph for aggregation. All the aggregated feature vectors are power-normalized and are then L2-normalized. We use Cosine similarity for comparison among feature vectors.

We use six local features for 3DMR, i.e., Position and Orientation Distribution (POD) [9], Spin Image (SI) [15], Local Statistical Feature (LSF) [22], Rotational Projection Statistics (RoPS) [10], Dense SIFT (DSIFT) [7], and Multi-Orientation One SIFT (MO1SIFT) [8]. They are classified into two groups; local 3D geometric feature including POD, SI, LSF, RoPS, and local 2D visual feature including DSIFT and MO1SIFT. Each local feature extraction method has hyper-parameters, e.g., the number of local features per 3D model or the number of rendering views. In this paper, we follow the experimental settings by Furuya et al. [9].

3.2 Experimental Results

3.2.1 Comparison with Other Feature Aggregation Algorithms

Table 3 compares the proposed feature aggregation algorithms against the other feature aggregation algorithms for the PSB dataset. Part of the table is due to [9]. Table 3 shows our *k*SA-based feature aggregation algorithms outperform the existing SC-based methods, i.e., BF, SA, TE, KSVD, and LL. We try to explain the results by using Table 1. TE has non-negativity but it lacks *k*-sparseness for feature encoding. KSVD has *k*-sparseness (we used *k*=5 for the experiment) and it learns codebook and feature encoding jointly.

However, feature encoding by KSVD could produce negative coefficients. LL has both *k*-sparseness (we used *k*=15) and non-negativity. But feature encoding would be not optimal because it uses a codebook learned by *k*-means clustering. In comparison, the D*k*SA and P*k*SA satisfy both *k*-sparseness and non-negativity of feature encoding. Also, D*k*SA and P*k*SA jointly optimize codebook learning and feature encoding. P*k*SA outperforms D*k*SA for five out of six features in Table 3 as the former exploits reconstruction error of local features for aggregation.

Table 3 also shows D*k*SA and P*k*SA outperform the state-of-the-art HS-based aggregation methods for most cases. Joint optimization framework of the proposed algorithms would produce more accurate aggregated features than the "greedy" approach adopted by the HS-based methods in Table 3.

Table 3. Comparison of MAP [%] (PSB dataset).

	POD	DSIFT	MO1SIFT	LSF	SI	RoPS
BF	47.8	54.0	51.9	33.0	38.1	40.5
SA	37.2	46.9	42.3			
TE	40.2	45.5	41.3			
KSVD	50.0	55.7	57.4			
LL	53.1	57.6	56.5	33.8	41.0	46.4
FV	52.9	61.7	54.2	36.7	44.9	46.3
VL	52.9	60.9	50.6	38.3	45.5	42.7
SV	55.3	63.8	53.2	40.2	**49.6**	48.4
DM	60.1	**64.7**	61.1	41.4	47.3	50.4
D*k*SA	57.1	56.7	**63.7**	35.3	37.4	52.1
P*k*SA	**61.5**	63.8	62.7	**43.0**	48.8	**55.1**

Figure 2 plots retrieval accuracies against the number of dimensions for aggregated features. We extracted POD features from the 3D models. Both D*k*SA and P*k*SA show higher retrieval accuracy than the state-of-the-art feature aggregation methods including LL, VL, FV, and SV. Also, P*k*SA significantly outperforms the DM which is one of the most accurate feature aggregation algorithm for 3DMR [9].

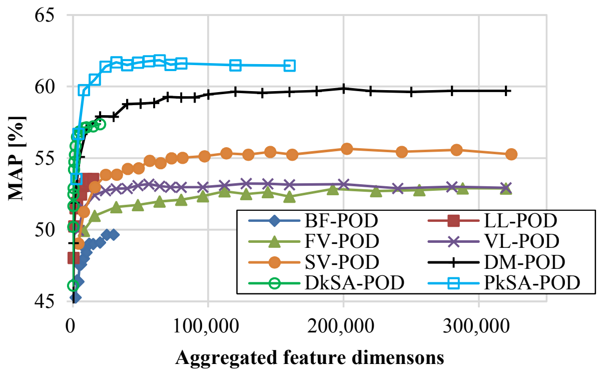


Figure 2. Comparison of feature aggregation methods (PSB).

In the experiments above, we used the only one benchmark database, i.e., the PSB. Table 4 shows generalization capability of the proposed algorithms for a variety of 3D model databases described in Section 3.1. We use POD local feature for the experiment. Table 4 shows the P*k*SA aggregation performs well, but not always the best, among the previous state-of-the-art feature aggregation algorithms.

3.2.2 *Efficiency*

We evaluate efficiency of the proposed *k*SA-based feature aggregation algorithms. We use a PC having two *Intel Xeon E5-2650V2* CPUs, 256GB DRAM, and a *GeForce GTX 980* GPU. Implementations of VL and SV are based on the VLFeat library [25]. They employ parallel computing on the multi-core CPUs for

Table 4. Comparison of MAP [%] (POD feature).

algorithms	PSB	ESB	SH11NR	SH14LC
BF	47.8	52.1	87.3	39.6
LL	53.1	53.0	94.7	44.7
FV	52.9	54.4	**96.7**	44.8
VL	52.9	49.8	95.9	44.0
SV	55.3	54.0	96.5	45.3
DM	60.1	57.2	95.8	50.7
D*k*SA	57.1	57.4	96.0	47.7
P*k*SA	**61.5**	**59.4**	96.4	**50.9**

Table 5. Computation time [s] for feature aggregation.

algorithms	pre-processing	feature aggregation per 3D model
VL	41.5	0.04
SV	342.6	0.08
D*k*SA	798.6	0.29
P*k*SA	226.9	0.75

both codebook learning and feature encoding. Computations of D*k*SA and P*k*SA are accelerated by using a GPU. We use our own implementations based on the CUDA library for D*k*SA and P*k*SA.

Table 5 compares efficiency of feature aggregation algorithms. In the table, pre-processing for VL and SV mean learning a codebook with 3k codewords. Meanwhile, pre-processing for D*k*SA and P*k*SA includes whitening of local features and training *k*SA. Number of hidden units *n* for D*k*SA and P*k*SA are set to 8k and 500 respectively. As for feature aggregation, we measured computation time for aggregating 3k POD features per 3D model. D*k*SA and P*k*SA are slower than VL and SV in aggregating local features. P*k*SA is the slowest among the four algorithms since it requires fine-tuning of *k*SA for aggregation. However, by the help of GPU, computation time of aggregation for both D*k*SA and P*k*SA are within 1 second, which is acceptable for 3DMR.

Querying a 3D model database by using the proposed algorithms is also efficient. We measured computation time per query for the SH14LC database which includes 8,987 3D models. A retrieval consists of three processes, i.e., extracting local features from the given query 3D model, aggregating the local features of the query, and computing similarities among the query and the 3D models in the database. 3DMR using D*k*SA took 0.61s per query and 3DMR using P*k*SA took 1.12s per query.

4. CONCLUSION AND FUTURE WORK

In this paper, for more accurate aggregation of local features, we proposed two novel feature aggregation algorithms D*k*SA and P*k*SA that employ *k*-Sparse Autoencoder (*k*SA). These algorithms jointly optimize codebook learning and feature encoding. Also, feature encoding using the *k*SA can enhance saliency of local features due to two constraints, i.e., *k*-sparseness and non-negativity, on unit activation in the hidden layer of the *k*SA. Furthermore, P*k*SA aggregation exploits reconstruction error of local features per data object for better retrieval accuracy.

Quantitative evaluation using multiple local features and multiple benchmarks for 3DMR showed that our proposed algorithms perform equal or better than existing state-of-the-art feature aggregation algorithms. As a future work, we will evaluate effectiveness of the *k*SA-based aggregation under 2D image retrieval or recognition setting.

5. ACKNOWLEDGMENTS

This research is supported by *JSPS Grant-in-Aid for Young Scientists* (*B*) #16K16055 and *JSPS Grant-in-Aid for Scientific Research on Innovative Areas* #26120517.

6. REFERENCES

[1] Achlioptas, D. 2003. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *Journal of Computer and System Sciences*, **66**(4), 671–687.

[2] Aharon, M., Elad, M., Bruckstein, A. 2006. K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation. *IEEE Transactions on Signal Processing*, **54**(11), 4311–4322.

[3] Bell, A., Sejnowski, T.J. 1996. Edges are the `Independent Components' of Natural Scenes. *Proc. NIPS 1996.*

[4] Coates, A., Ng, A.Y., Lee, H. 2011. An analysis of single-layer networks in unsupervised feature learning. *Proc. AISTATS 2011*, 215–223.

[5] Csurka, G. et al. 2004. Visual Categorization with Bags of Keypoints, *Proc. ECCV 2004 workshop on Statistical Learning in Computer Vision*, 59–74.

[6] Duchi, J., Hazan, E., Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, **12**, 2121–2159.

[7] Furuya T., Ohbuchi. R. 2009, Dense sampling and fast encoding for 3D model retrieval using bag-of-visual features, *Proc. ACM CIVR* 2009, Article No. 26.

[8] Furuya, T., Ohbuchi, R. 2014. Fusing Multiple Features for Shape-based 3D Model Retrieval, *Proc. BMVC 2014*.

[9] Furuya, T., Ohbuchi, R. 2015. Diffusion-on-Manifold Aggregation of Local Features for Shape-based 3D Model Retrieval. *Proc. ICMR 2015*, 171–178.

[10] Gong, Y., Wang, L., Guo, R., Lazebnik, S. 2014. Multi-Scale Orderless Pooling of Deep Convolutional Activation Features. *Proc. ECCV 2014*, 392–407

[11] Guo, Y. et al. 2013. Rotational Projection Statistics for 3D Local Surface Description and Object Recognition, *IJCV*, **105**(1), 63–86.

[12] Huang, Y., Wu, Z., Wang, L., Tan, T. 2014. Feature Coding in Image Classification, A Comprehensive Study. *IEEE TPAMI*, **36**(3), 493–506.

[13] Jayanti, S., Kalyanaraman, Y., Iyer, N., Ramani, K. 2006. Developing an engineering shape benchmark for CAD models, *Proc CAD*, **38**(9), 939–953.

[14] Jégou, H., Douze, M., Schmid, C., P. Perez. 2010. Aggregating local descriptors into a compact image representation, *Proc. CVPR 2010*, 3304–3311.

[15] Johnson, A.E., Hebert, M. 1999. Using spin images for efficient object recognition in cluttered 3D scenes, *Pattern Analysis and Machine Intelligence*, **21**(5), 433–449.

[16] Krizhevsky, A., Sutskever, I., Hinton, G.E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. *Proc. NIPS 2012*, 1097–1105.

[17] Lian, Z. et al. 2011. SHREC'11 Track: Shape Retrieval on Non-rigid 3D Watertight Meshes, *Proc. EG 3DOR 2011*, 79–88.

[18] Li, B. et al. 2014. Large Scale Comprehensive 3D Shape Retrieval, *Proc. EG 3DOR 2014*, 131–140.

[19] Liu, L., Wang, L., Liu, X. 2011. In defense of soft-assignment coding, *Proc. ICCV 2011*, 2486–2493.

[20] Makhzani, A., Frey B. 2012. *k*-Sparse Autoencoders, arXiv:1312.5663.

[21] Ng, A.Y. 2011. Sparse auto-encoder. CS294A Lecture notes.

[22] Ohkita, Y., Ohishi, Y., Furuya, T., Ohbuchi, R. 2012. Non-rigid 3D Model Retrieval Using Set of Local Statistical Features, *Proc. ICME 2012 Workshop on Hot Topics in 3D Multimedia*, 593–598.

[23] Perronnin, F., Sánchez, J., Mensink, T. 2010. Improving the fisher kernel for large-scale image classification, *Proc. ECCV 2010*, Part IV, 143–156.

[24] Shilane, P., Min, P., Kazhdan, M., Funkhouser, T. 2004. The Princeton Shape Benchmark, *Proc. SMI 2004*, 167–178.

[25] Vedaldi, A., Fulkerson, B. 2010. Vlfeat: an open and portable library of computer vision algorithms. *Proc. ACM MM 2010*, 1469–1472.

[26] Wang, J. et al. 2010, Locality-constrained Linear Coding for Image Classification, *Proc. CVPR 2010*, 3360–3367.

[27] Zhou, X., Yu, K., Zhang, T., Huang, T.S. 2010. Image Classification using Super-Vector Coding of Local Image Descriptors, *Proc. ECCV* 2010, 141–154.