MULTI-VIEW ANCHOR GRAPH HASHING

Saehoon Kim[1] *and Seungjin Choi*[1,2]

[1] Department of Computer Science and Engineering, POSTECH, Korea
[2] Division of IT Convergence Engineering, POSTECH, Korea
{kshkawa, seungjin}@postech.ac.kr

ABSTRACT

Multi-view hashing seeks compact integrated binary codes which preserve similarities averaged over multiple representations of objects. Most of existing multi-view hashing methods resort to linear hash functions where data manifold is not considered. In this paper we present *multi-view anchor graph hashing* (MVAGH), where non-linear integrated binary codes are efficiently determined by a subset of eigenvectors of an averaged similarity matrix. The efficiency behind MVAGH is due to a low-rank form of the averaged similarity matrix induced by multi-view anchor graph, where the similarity between two points is measured by two-step transition probability through view-specific anchor (i.e. landmark) points. In addition, we observe that MVAGH suffers from the performance degradation when the high recall is required. To overcome this drawback, we propose a simple heuristic to combine MVAGH with locality sensitive hashing (LSH). Numerical experiments on CIFAR-10 dataset confirms that MVAGH(+LSH) outperforms the existing multi- and single-view hashing methods.

Index Terms— Anchor graphs, hashing, multi-view learning

1. INTRODUCTION

Hashing seeks a hash function to embed high-dimensional data into a similarity-preserving low-dimensional Hamming space such that an approximate nearest neighbor of a given query can be found with sub-linear time complexity [2, 15]. Classic approach on hashing (locality sensitive hashing [2]) is to compute a hash function purely in randomized manner, where random projections followed by rounding are used to generate binary codes such that the two similar examples have the same binary codes. Since the performance is not satisfied when short binary codes are used [12], multiple hash tables or longer binary codes should be employed in practice.

Learning to hash seeks to the compact similarity-preserving binary codes, which can be categorized into unsupervised and (semi-)supervised paradigms. Spectral hashing (SH) [14, 15] is a representative unsupervised hashing method, where the Laplace-Beltrami eigenfunctions of manifolds are used to determine binary codes. Iterative quantization (ITQ) [3] tries to rotate PCA-embedded data by an orthogonal matrix in order to minimize the quantization error caused by mapping the embedded data into a binary hypercube. Semantic hashing [11] is the earliest supervised hashing, exploiting deep networks to learn a non-linear mapping between input data and binary codes. Practical usefulness of semantic hashing is limited due to time-consuming training time. Supervised hashing with a reasonable training time has been proposed, where a hash function is determined sequentially yielding very short discrminative codes [8]. Semi-supervised hashing [13] minimizes the empirical error induced by the violation of pairwise constraints (must-link and cannot-link), and prevents from over-fitted hash functions using unlabeled data.

Recently, learning to hash for multi-view (or multi-modal) data has been developed. [4, 6, 17] seeks an integrated binary codes which preserves an averaged similarity, extending spectral hashing for multi-view data. [6] exploits a pre-defined averaged similarity (or identity matrix) to compute view-specific binary codes which concatenates into an integrated code. [17] uses a linear sum of view-specific similarity matrices as an averaged similarity, where an integrated binary code is directly computed. [4] seeks an integrated binary code in a sequential manner in order to de-correlate binary codes, where an averaged similarity is computed by α-average of view-specific distance matrices. [18] proposes a generative model, where intra-view and inter-view similarities are generated given by view-specific binary codes. Besides [4, 6, 17], [18] can adopt non-vectorial inputs, but the algorithm is not scalable and the performance is degraded when code length is increased.

In this paper, we present multi-view anchor graph hashing (MVAGH), where an non-linear integrated binary code is determined by a subset of eigenvectors of an averaged similarity induced by multi-view anchor graph. Multi-view anchor graph keeps the averaged similarity in a low-rank form, where the eigenvectors can be computed efficiently. Note that hashing with an anchor graph [9] has been already popular for unsupervised hashing, but the extension for multi-view data has never been studied. Since MVAGH computes non-linear binary code in an efficient way, it has a clear advantage than the previous multi-view hashing, where [4, 6, 17] tries to compute a linear hash function and [18] is not scalable and needs label information. More specifically, [4, 6, 17] considers a linear hash function to preserve the pair-wise similarity in Euclidean space. However, if the data lie on the embedded low-dimensional manifold, the linear hash function cannot well capture the data similarity, because the examples measured large distance in Euclidean distance can be similar in the manifold. [18] considers a non-linear hash function, but it is not scalable because the similarity matrix is kept explicitly. We observe that MVAGH suffers from the performance degradation when the high recall is required. To overcome this drawback, we propose a simple heuristic to combine MVAGH with locality sensitive hashing (LSH).

2. MULTI-VIEW ANCHOR GRAPH HASHING

In this section, we describe multi-view anchor graph hashing (MVAGH), exploiting multi-view anchor graph to approximate the averaged similarity. Before discussing MVAGH, we clarify our notation. Suppose that $\{\boldsymbol{x}_i\}_{i=1}^N$ is a set of N objects, where each object $\boldsymbol{x}_i$ is represented by K different view-specific examples by $\{\boldsymbol{x}_i^{(1)}, \ldots, \boldsymbol{x}_i^{(K)}\}$. We define the integrated binary code matrix as

$\boldsymbol{Y}^* = \{\boldsymbol{y}_i^*\}_{i=1}^N \in \{-1,+1\}^{r\times N}$, where $\boldsymbol{y}_i^*$ is the binary code associated with $\boldsymbol{x}_i$ and r is the code length.

MVAGH borrows the spectral hashing formulation in order to seek an integrated binary code matrix $\boldsymbol{Y}^* \in \{-1,+1\}^{r\times N}$ which preserves the averaged similarity $\boldsymbol{S}^*$:

$$\begin{aligned} \underset{\boldsymbol{Y}^*}{\arg\min} \quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} S_{ij}^* \|\boldsymbol{y}_i^* - \boldsymbol{y}_j^*\|_2^2, \\ \text{subject to} \quad & \boldsymbol{Y}^*\mathbf{1}_N = \mathbf{0}, \frac{1}{N}\boldsymbol{Y}^*\boldsymbol{Y}^{*\top} = \boldsymbol{I}_r, \end{aligned} \tag{1}$$

where $||\boldsymbol{y}_i||$ is the Euclidean norm of $\boldsymbol{y}_i$ and $\mathbf{1}_N \in \mathbb{R}^N$ is the vector of all ones. Since $\boldsymbol{y}_i^{*\top}\boldsymbol{y}_i^*$ is always r, the objective function is equivalent to

$$\underset{\boldsymbol{Y}^*}{\arg\max} \quad \frac{1}{2}\mathrm{tr}(\boldsymbol{Y}^*\boldsymbol{S}^*\boldsymbol{Y}^{*\top}). \tag{2}$$

Ignoring the binary constraint, the solution of (2) with the constraints of (1) is the largest eigenvectors of the similarity matrix. As in anchor graph hashing [9], if the similarity matrix is low-rank approximated, the eigenvectors are efficiently computed and the generalized eigenfunctions are also easily computed for a novel input. Therefore, the natural question is how to construct the low-rank approximation of the averaged similarity induced by multi-view data.

2.1. Multi-View Anchor Graph

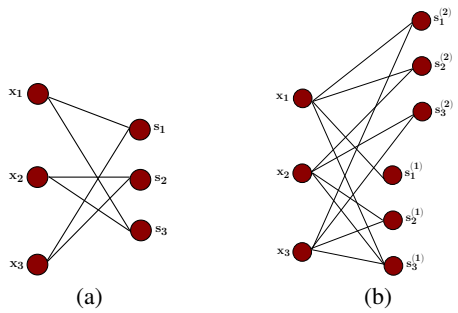


Fig. 1. Random walk view of similarity graph approximated by anchor graph (a) and multi-view anchor graph (b) when the number of views is two. If more than two views, the multi-view anchor graph can be developed like (b).

Anchor graph [7] is a low-rank approximation of neighborhood graph (such as k-NN and ϵ-graph), where the similarity between data points are measured by a small number of anchor points. Fig. 1 (a) represents the bipartite graph, where the left vertices are data points and the right ones anchor points. Data points and their associated two-nearest anchor points are connected. The similarity of two points are measured by two-step transition probability through anchor points. We observe that the similarity of two points is greater than zero only when they share the same anchor point, which means that the similarity matrix approximated by anchor graph is empirically sparse and preserves the data locality. The anchor points should be selected to sufficiently cover the data distribution. In practice, the cluster centers by a few iteration of k-means are enough to be anchor points.

Though anchor graph is successfully applied to unsupervised hashing [9], the extension for multi-view data has not studied. We first define multi-view anchor graph, where the different contribution of each view is well combined to approximate the averaged neighborhood graph. Since the data distribution might be different for each view, we select anchor points for each view [1], and k-th view specific anchor points denoted as $\{\boldsymbol{\mu}_j^{(k)}\}_{j=1}^M$. Define the k-th view specific similarity between the example $\boldsymbol{x}_i^{(k)}$ and $\boldsymbol{\mu}_j^{(k)}$ as

$$Z_{ij}^{(k)} = \frac{k(\boldsymbol{x}_i^{(k)}, \boldsymbol{\mu}_j^{(k)})}{\sum_{k=1}^{K}\sum_{m\in[i]} k(\boldsymbol{x}_i^{(k)}, \boldsymbol{\mu}_m^{(k)})}, \qquad \forall j \in [i], \tag{3}$$

where $[i]$ contains the indices of the l-nearest anchors of $\boldsymbol{x}_i^{(k)}$ (usually $l = 3$), and $k(\cdot,\cdot)$ is any kernel function.

The average similarity between the objects $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is denoted as $\boldsymbol{S}_{ij}^* = p(\boldsymbol{x}_j|\boldsymbol{x}_i)$. When the number of views is two, we consider a tri-partite graph in Fig. 1 (b) to approximate the average similarity by multi-view anchor graph, where the similarity between the two objects, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, are two-step transition probability through view-specific anchor points:

$$p(\boldsymbol{x}_j|\boldsymbol{x}_i) = \sum_{k=1}^{K}\sum_{m=1}^{M} p(\boldsymbol{x}_j|\boldsymbol{\mu}_m^{(k)})p(\boldsymbol{\mu}_m^{(k)}|\boldsymbol{x}_i), \tag{4}$$

where $p(\boldsymbol{x}_j|\boldsymbol{\mu}_m^{(k)}) = \frac{Z_{jm}^{(k)}}{\sum_{j=1}^{N} Z_{jm}^{(k)}}$ and $p(\boldsymbol{\mu}_m^{(k)}|\boldsymbol{x}_i) = Z_{im}^{(k)}$. We define that $\boldsymbol{\Lambda}^{(k)}$ is a diagonal matrix whose m-th diagonal entry is $\sum_{j=1}^{N} Z_{jm}^{(k)}$ and $[\boldsymbol{Z}^{(k)}]_{im} = Z_{im}^{(k)}$. Using this notation, $p(\boldsymbol{x}_j|\boldsymbol{x}_i) = [\sum_{k=1}^{K}\boldsymbol{Z}^{(k)}{\boldsymbol{\Lambda}^{(k)}}^{-1}\boldsymbol{Z}^{(k)\top}]_{ij}$. Letting $\boldsymbol{Z}^* = [\boldsymbol{Z}^{(1)}\cdots\boldsymbol{Z}^{(K)}]$ and $\boldsymbol{\Lambda}^*$ is $\mathrm{diag}(\boldsymbol{\Lambda}^{(1)}\ldots\boldsymbol{\Lambda}^{(K)})$, the averaged similarity can be low-rank approximated by multi-view anchor graph: $\boldsymbol{S}^* \approx \boldsymbol{Z}^*{\boldsymbol{\Lambda}^*}^{-1}\boldsymbol{Z}^{*\top}$. Multi-view anchor graph can be interpreted as linear sum of view-specific anchor graph, where the normalization term in (3) is the link between the view-specific ones.

The eigenvectors of $\boldsymbol{S}^*$ are obtained by the eigen-problem solution of small matrix $\boldsymbol{A} = {\boldsymbol{\Lambda}^*}^{-1/2}\boldsymbol{Z}^{*\top}\boldsymbol{Z}^*{\boldsymbol{\Lambda}^*}^{-1/2}$. Let the r-largest eigenvalues $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1,\ldots,\sigma_r)$ associated with the eigenvectors $\boldsymbol{V} = [\boldsymbol{v}_1,\ldots,\boldsymbol{v}_r]$ of $\boldsymbol{A}$. Now, the transpose of eigenvectors of $\boldsymbol{S}^*$ denoted as $\widetilde{\boldsymbol{Y}}$ is computed by

$$\widetilde{\boldsymbol{Y}} = \sqrt{N}{\boldsymbol{\Sigma}^*}^{-1/2}\boldsymbol{V}^\top\boldsymbol{\Lambda}^{-1/2}\boldsymbol{Z}^{*\top} = \sqrt{N}\boldsymbol{W}^\top\boldsymbol{Z}^{*\top}, \tag{5}$$

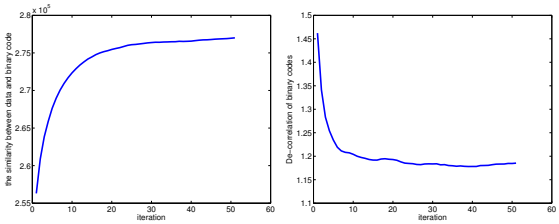
where $\boldsymbol{W} = {\boldsymbol{\Lambda}^*}^{-1/2}\boldsymbol{V}\boldsymbol{\Sigma}^{-1/2}$. The binary codes can be computed by rounding at zero: $\boldsymbol{Y}^* = \mathrm{sgn}(\widetilde{\boldsymbol{Y}})$, where sgn function operates on each element. However, the relaxed solution is not satisfied, and we adapt several heuristics to find the better solution (since the original solution is NP-hard, we have to use some reasonable heuristics for the better solution).

In spectral clustering, spectral rounding [16] has been widely used to refine the solution, where a rotation matrix is estimated to minimize the quantization error induced by discretizing the continuous relaxed solution. Recently, the similar idea has been successfully applied to learn a hash function, which is known as iterative quantization (ITQ) [3]. The objective function of ITQ is presented as

$$\begin{aligned} \underset{\boldsymbol{R},\boldsymbol{Y}^*}{\arg\min} \quad & ||\boldsymbol{Y}^* - \boldsymbol{R}^\top\widetilde{\boldsymbol{Y}}||_F^2, \\ \text{s.t.} \quad & \boldsymbol{Y}^*\mathbf{1}_N = \mathbf{0}_N, \boldsymbol{R}^\top\boldsymbol{R} = \boldsymbol{R}\boldsymbol{R}^\top = \boldsymbol{I}_r, \end{aligned} \tag{6}$$

[1] For view-specific anchor points, we apply k-means (10 iterations) into each view

where $\boldsymbol{R}$ and $\boldsymbol{Y}^*$ are iteratively estimated to minimize the quantization error as in [3, 16]. ITQ originally considers the PCA embedded space, but spectral embedded space is also well suited to ITQ, because we empirically observe that the binary codes become more compact and reflect the better data similarity (Fig. 2). Fig. 2 (a) represents $\text{tr}(\boldsymbol{Y}^*\boldsymbol{S}^*\boldsymbol{Y}^{*\top})$ over iterations, where we observe that the similarity between binary codes more reflects the original data similarity by ITQ. Fig. 2 (b) means $||\frac{1}{n}\boldsymbol{Y}^*\boldsymbol{Y}^{*\top} - \boldsymbol{I}_r||_F^2$ over iterations, where we observe that the binary codes are more de-correlated by ITQ.



(a) (b)

Fig. 2. During ITQ iteration, the binary codes more preserve the data similarity (a) and are more de-correlated (b). CIFAR-10 dataset is used.

2.2. Out-of-Sample Extension

A subset of eigenvectors of multi-view anchor graph is used to determine the binary codes of training data. We can compute analytically the binary code of a novel point by using Nyström method as in [9]. Lemma 1 represents the analytic hash function of a novel point.

Lemma 1. *Given m view-specific anchor points $\{\mu_j^{(k)}\}_{j=1}^M$ for $k = 1, ..., K$ and any example $\boldsymbol{x}$, define a mapping $\boldsymbol{z} : \mathcal{R}^D \rightarrow \mathcal{R}^M$ as follows*

$$\boldsymbol{z}(\boldsymbol{x}) = \frac{[\boldsymbol{z}(\boldsymbol{x}^{(1)}), \ldots, \boldsymbol{z}(\boldsymbol{x}^{(K)})]^\top}{\sum_{k=1}^K \boldsymbol{z}(\boldsymbol{x}^{(k)})},$$

where $\boldsymbol{z}(\boldsymbol{x}^{(k)}) = [\delta_1 k(\boldsymbol{x}^{(k)}, \boldsymbol{\mu}_1^{(k)}), \ldots, \delta_m k(\boldsymbol{x}^{(k)}, \boldsymbol{\mu}_m^{(k)})]$ and $\delta_i \in \{1, 0\}$. $\delta_j = 1$ iff anchor $\mu_j^{(k)}$ is one of s-nearest anchors of sample $\boldsymbol{x}^{(k)}$ according to the kernel function $k(\cdot, \cdot)$. The binary code of a novel point, $\boldsymbol{y}^*$, is expressed as

$$\boldsymbol{y}^* = sgn(\boldsymbol{R}^\top \boldsymbol{W}^\top \boldsymbol{z}(\boldsymbol{x})),$$

where $\boldsymbol{W}$ is defined in (5).

The proof of this lemma is straightforward from Theorem 1 in [9]. Algorithm 1 represents a pseudo-code for MVAGH.

2.3. Multi-View Anchor Graph Hashing + LSH

Multi-view anchor graph hashing consists of the two steps: (1) project the data onto the largest eigenvectors of multi-view anchor graph, and (2) rounding at zero to produce binary codes. Since the intrinsic dimension is usually low, the similarity between binary codes with large code size might not well reflect the data similarity. We observe that the precision is decreased with large code size when the high recall is required.

As in Fig. 3, when the high recall is required (the number of returned example is large), the similarity induced by binary codes turns

Algorithm 1 Multi-View Anchor Graph Hashing (MVAGH)

Input: For each k-th view, training data are $\boldsymbol{X}^{(k)} = [\boldsymbol{x}_1^{(k)} \cdots \boldsymbol{x}_n^{(k)}] \in \mathbb{R}^{m \times n}$, anchor points are $\{\boldsymbol{s}_j^{(k)}\}_{j=1}^M$, and test data is $\boldsymbol{x}^{(k)} \in \mathbb{R}^m$ for $k = 1, \cdots, K$. Binary code length is r.

Output: Binary code $\boldsymbol{y}$ associated with the test data $\{\boldsymbol{x}^{(k)}\}_{k=1}^K$.

1: Compute the similarity between the example and anchor point as $[\boldsymbol{Z}^{(k)}]_{im}$ using the equation 3.
2: Let $\boldsymbol{Z}^* = [\boldsymbol{Z}^{(1)}, \cdots, \boldsymbol{Z}^{(K)}]$ and $\boldsymbol{\Sigma}^* = [\boldsymbol{\Sigma}^{(1)}, \cdots, \boldsymbol{\Sigma}^{(K)}]$.
3: Apply eigenvalue decomposition to $\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where $\boldsymbol{A} = {\boldsymbol{\Lambda}^*}^{-1/2} \boldsymbol{Z}^{*\top} \boldsymbol{Z}^* {\boldsymbol{\Lambda}^*}^{-1/2}$.
4: $\widehat{\boldsymbol{Y}} = \sqrt{N}\boldsymbol{W}^\top \boldsymbol{Z}^{*\top}$, where $\boldsymbol{W} = {\boldsymbol{\Lambda}^*}^{-1/2} \boldsymbol{V}\boldsymbol{\Sigma}^{-1/2}$.
5: $\boldsymbol{R}_0$ is a random rotation matrix.
6: **for** $i = 1, \ldots, 50$ **do**
7: $\quad \boldsymbol{Y}_i = \text{sgn}(\boldsymbol{R}_{i-1}^\top \widehat{\boldsymbol{Y}})$.
8: $\quad \boldsymbol{R}_i = \widehat{\boldsymbol{M}}\boldsymbol{M}^\top$, where $\boldsymbol{Y}_i \widehat{\boldsymbol{Y}} = \boldsymbol{M}\boldsymbol{\Omega}\widehat{\boldsymbol{M}}^\top$ by SVD.
9: **end for**
10: Return k-bit binary code: $\boldsymbol{y} = \text{sgn}(\boldsymbol{R}_{50}^\top \boldsymbol{W}^\top \boldsymbol{z}(\boldsymbol{x}))$, where $\boldsymbol{z}(\boldsymbol{x})$ is defined in Lemma 1.

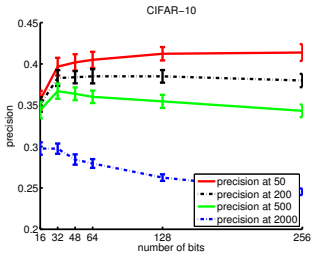
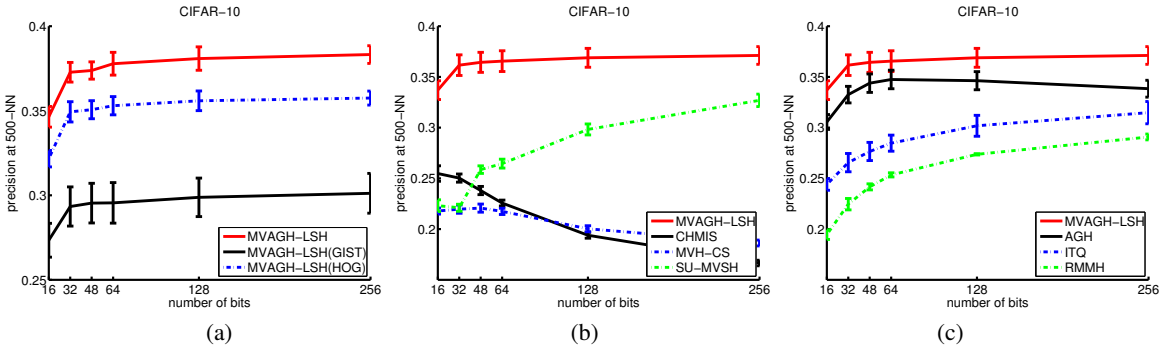


Fig. 3. Precision of MVAGH over the code size when the number of returned examples is changed.

to be incorrect. This observation leads us to propose a simple heuristic to increase retrieval performance when code size is large, where the binary codes from MVAGH and locality-sensitive hashing (LSH) are concatenated. We compute the meaningful binary code from MVAGH by choosing r_a largest eigenvectors, where r_a should be small (in practice $r_a = 32$ is sufficient). If we want to use $r_1$bits binary code and $r_1 > r_a$, we first compute $\boldsymbol{Y}_{r_a}^* = \text{sgn}(\boldsymbol{R}^\top \widetilde{\boldsymbol{Y}})$, where $\widetilde{\boldsymbol{Y}}$ is transpose of r_a largest eigenvectors from multi-view anchor graph. The remaining $(r_1 - r_a)$bits are generated by LSH, where $\boldsymbol{Y}_{remain}^* = \text{sgn}(\boldsymbol{M}^\top \boldsymbol{R}^\top \widetilde{\boldsymbol{Y}})$ and M is generated from $N(\boldsymbol{0}, \boldsymbol{I})$. The final binary code is obtained by concatenating into $\boldsymbol{Y}_{r_a}^*$ and $\boldsymbol{Y}_{remain}^*$, yielding $\boldsymbol{Y}^* = [\boldsymbol{Y}_{r_a}^*; \boldsymbol{Y}_{remain}^*]$. Since the r_a largest eigenvectors reveal the intrinsic data structure, we expect that the bits generated by LSH are meaningful. We denote this heuristic as MVAGH + LSH, and section 3 shows that this method outperforms state-of-the art methods over all bits.

3. NUMERICAL EXPERIMENTS

In this section, we use CIFAR-10 [5] contains 60,000 images with 10 labels. We form a query set by randomly choosing 1,000 images and construct a training set using the rest of images. All experiments are repeated five times to compute the mean and standard deviation for error bars. We use GIST [10] and HOG [1] descriptors to produce two views of each image. For GIST, we use Gabor filter with 8 orientations and 3 scales, leading to 384-dimensional vector. For HOG,



(a) (b) (c)

Fig. 4. Precision of MVAGH+LSH when single feature is used, or two features combined by multi-view anchor graph (a). Comparison between MVAGH+LSH into multi-view (b) and single-view (c) hashing methods.

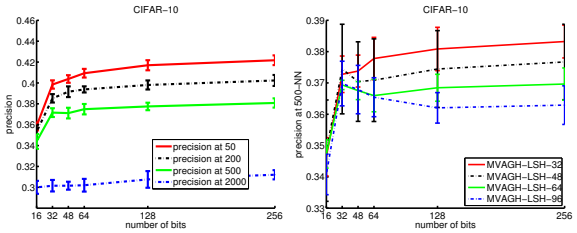


Fig. 5. Precision of MVAGH-LSH over the number of returned image (left), and the effects of the parameter r_a (right).

we compute the image gradients of non-overlapping windows, where the orientation of gradients is quantized into 8 bins and normalized with 4 different metrics, and 36 4-by-4 non-overlapping windows yield 1152-dimensional vectors.

We compare our proposed method (MVAGH + LSH) and the recent multi-view hashing methods: multi-view hashing [6], CHMIS [17], and SU-MVAGH [4] (Fig. 4 (b)). We also compare our method into the state-of-the art single-view hashing methods: anchor graph hashing (AGH), iterative quantization (ITQ), random maximum margin hashing (RMMH) (Fig. 4 (c)). For single-view hashing methods, multi-view data are concatenated into single representation. As a performance measure, we use Hamming ranking [9, 13] where the rankings between a query and data points are decided by Hamming distance. If there exists a tie in the Hamming distance, we break it randomly. We calculate the precision at the top 500 examples.

For the parameter of MVAGH, we select 500 anchor points for each view, $s = 3$, and $r_a = 32$. We use Gaussian kernel: $\exp(-\frac{1}{\sigma^2}||\boldsymbol{x}-\boldsymbol{y}||_2^2)$, where σ is set the median of pairwise distances of data points. We choose the best parameters for the compared methods. Fig. 4 summarizes the comparison between MVAGH + LSH into various state-of-the-art hashing methods, where (a) means multi-view hashing clearly improves the precision than using single descriptor, (b) and (c) suggest that our MVAGH + LSH outperforms the existing multi and single-view hashing methods. Fig. 5 shows that the performance of MVAGH-LSH is not decreased when code length is large, and the effects of the parameter r_a, which means that if we choose longer r_a than 32, the performance is decreased. Fig. 6 represents the example of retrieval result, where the leftmost image is a query and the 20-nearest neighbors are displayed. The incorrect retrieved images are marked by red rectangle. We can see that the precision is increased when the two features are used together, and MVAGH is superior to the single-view hashing methods.

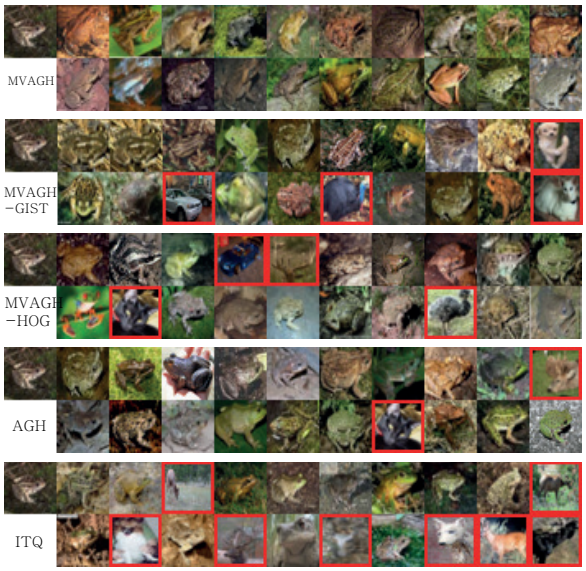


Fig. 6. Retrieval results on CIFAR-10. Leftmost image is a query and the 20-nearest images are displayed, and the incorrect ones are marked by the red rectangle.

4. CONCLUSIONS

We have presented multi-view anchor graph hashing (MVAGH) where non-linear integrated binary codes are determined by a subset of eigenvectors of an averaged similarity matrix. The underlying idea was based on multi-view anchor graph to keep the averaged similarity in a low-rank form, where the similarity between two points is measured by two-step transition probability through view-specific anchor points. We have also presented a heuristic to improve the performance of MVAGH by combining it with LSH, demonstrating its high performance over existing methods.

Acknowledgments: This work was supported by NIPA-MSRA Creative IT/SW Research Project, ITRC Program (NIPA-2012-H0301-12-3002), POSTECH Rising Star Program, and NRF WCU Program (R31-10100).

5. REFERENCES

[1] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, San Diego, CA, 2005.

[2] A. Gionis, P. Indyk, and R. Motawani, "Similarity search in high dimensions via hashing," in *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, 1999.

[3] Y. Gong and S. Lazebnik, "Iterative quantization: A procrustean approach to learning binary codes," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, Colorado Springs, CO, 2011.

[4] S. Kim, Y. Kang, and S. Choi, "Sequential spectral learning to hash with multiple representation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Firenze, Italy, 2012.

[5] A. Krizhevsky and G. E. Hinton, "Learning multiple layers of features from tiny images," Computer Science Department, University of Toronto, Tech. Rep., 2009.

[6] S. Kumar and R. Udupa, "Learning hash functions for cross-view similarity search," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Barcelona, Spain, 2011.

[7] W. Liu, J. He, and S. F. Chang, "Large graph construction for scalable semi-supervised learning," in *Proceedings of the International Conference on Machine Learning (ICML)*, Hiafa, Israel, 2010.

[8] W. Liu, J. Wang, R. Ji, Y. G. Jiang, and S. F. Chang, "Supervised hashing with kernels," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, Providence, Rhode Island, USA, 2012.

[9] W. Liu, J. Wang, S. Kumar, and S. F. Chang, "Hashing with graphs," in *Proceedings of the International Conference on Machine Learning (ICML)*, Bellevue, WA, 2011.

[10] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.

[11] R. Salakhutdinov and G. Hinton, "Semantic hashing," in *Proceeding of the SIGIR Workshop on Information Retrieval and Applications of Graphical Models*, 2007.

[12] A. Torralba, R. Fergus, and Y. Weiss, "Small codes and large image databases for recognition," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, Anchorage, Alaska, 2008.

[13] J. Wang, S. Kumar, and S. F. Chang, "Semi-supervised hashing for scalable image retrieval," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, 2010.

[14] Y. Weiss, R. Fergus, and A. Torralba, "Multidimensional spectral hashing," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Firenze, Italy, 2012.

[15] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 20. MIT Press, 2008.

[16] S. X. Yu and J. Shi, "Multiclass spectral clustering," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2003.

[17] D. Zhang, F. Wang, and L. Si, "Composite hashing with multiple information sources," in *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, Beijing, China, 2011.

[18] Y. Zhen and D. Y. Yeung, "A probabilistic model for multimodal hash function learning," in *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, Beijing, China, 2012.