Distance Encoded Product Quantization

Jae-Pil Heo[1], Zhe Lin[2], Sung-Eui Yoon[1]
[1] KAIST [2] Adobe Research

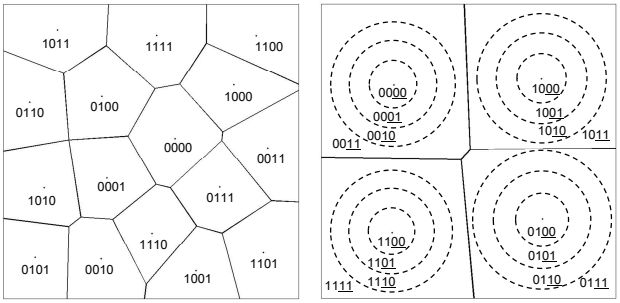
Abstract

Many binary code embedding techniques have been proposed for large-scale approximate nearest neighbor search in computer vision. Recently, product quantization that encodes the cluster index in each subspace has been shown to provide impressive accuracy for nearest neighbor search. In this paper, we explore a simple question: is it best to use all the bit budget for encoding a cluster index in each subspace? We have found that as data points are located farther away from the centers of their clusters, the error of estimated distances among those points becomes larger. To address this issue, we propose a novel encoding scheme that distributes the available bit budget to encoding both the cluster index and the quantized distance between a point and its cluster center. We also propose two different distance metrics tailored to our encoding scheme. We have tested our method against the-state-of-the-art techniques on several well-known benchmarks, and found that our method consistently improves the accuracy over other tested methods. This result is achieved mainly because our method accurately estimates distances between two data points with the new binary codes and distance metric.

1. Introduction

Approximate Nearest Neighbor (ANN) search has been an active research problem across many fields in computer science. The problem is especially important for high-dimensional and large-scale cases due to the efficiency requirement by many practical applications. In this paper we are mainly interested in applications for computer vision and thus restrict our discussion on approximate nearest neighbor techniques tailored to computer vision.

Among prior ANN techniques, Product Quantization (PQ) [13] and its recent improvement, Optimized Product Quantization (OPQ) [5], have shown the-state-of-the-art performance. Its high accuracy and efficiency is mainly because (1) quantization in subspaces offers a strong representational power and (2) distances between two data points can be computed via a look-up table. Its input feature space is divided into several disjoint subspaces, and each subspace is partitioned into clusters independently. A data point is then represented by an index of its corresponding cluster in each subspace. The distance between two data points is approximated by the sum of distances of their cluster centers computed in each subspace.



(a) PQ (b) Ours

Figure 1. The left and right figures show toy examples of partitioned space and assigned binary codes for PQ and ours, respectively, when using 4 bit binary codes. Our method allocates first two bits for encoding a cluster index and another underlined two bits for quantized distances, while PQ allocates all the bits for encoding a cluster index.

PQ and OPQ can effectively generate many clusters in each subspace and thereby reduce the quantization distortion. Nonetheless, we have found that their approach shows marginal accuracy improvement in practice, as we increase the number of clusters in each subspace. This is mainly because they encode only clusters containing data points, but are not designed to consider how far data points are located away from cluster centers.

Main contributions. To address aforementioned problems, we propose a new binary code encoding scheme, Distance-encoded Product Quantization (DPQ), and two different distance metrics tailored to the scheme. We follow exactly the same procedure as in PQ and OPQ to generate subspaces, quantize them with unsupervised clustering (i.e. k-means), and encode each data point with the index of its nearest cluster center in each subspace. The novelty of our method lies in that in addition to encoding the cluster index, we use additional bits to quantize the distance from the data point to its closest cluster center. Based on the new

binary encoding scheme, we propose two distance metrics, statistics and geometry based distance metrics, for symmetric and asymmetric cases. Especially, our geometry based distance metric is based on novel geometric reasoning for high-dimensional data spaces.

We have applied our method to three standard benchmarks consisting of GIST and BoW descriptors. Results show that our encoding scheme with our distance metrics consistently outperforms the existing state of the art methods across tested benchmarks. Specifically, combined with PQ our method improves results of PQ significantly, and combined with OPQ, the improvement is even larger. This indicates that our subspace encoding scheme is more useful, when the subspace is constructed more optimally. These improvements are mainly caused by both quantizing distances of points from cluster centers and well estimated distance metrics. Overall our method is simple, but results in meaningful performance improvement over PQ and OPQ.

2. Related Work

In this section we discuss prior work related to large-scale ANN search in a high dimensional space.

2.1. Hierarchical Methods

There have been many approaches that utilize hierarchical data structures for ANN search. Notable examples of such tree structures include kd-trees [4]. In the computer vision community, there have been a lot of efforts to optimize kd-trees to efficiently index high-dimensional image features or descriptors such as randomized kd-trees [20].

The hierarchical k-means tree [18] has been proposed to overcome the problem of the curse of dimensionality by using recursive k-means clustering. An automatic parameter selection technique [17] for such hierarchical data structures has been proposed for more accurate and faster search.

These hierarchical methods show high efficiency while keeping reasonable search accuracy. However, they do not provide compact data representations, and thus are less effective for handling large-scale image databases consisting of billions of items.

2.2. Binary Code Embedding Methods

Recently binary code embedding methods have been actively studied, since they provide a high compression rate by encoding high-dimensional data into compact binary codes, and fast distance (i.e., similarity) computation with simple bit-string operations or a pre-computed lookup table. We categorize binary code embedding methods into two categories: projection and clustering based methods.

Projection based methods. These techniques map high-dimensional data to the Hamming space by using projection functions. They can be categorized further into data-independent and data-dependent methods. In data-independent methods, the projection functions are defined independently from the data. One well-known method is Locality Sensitive Hashing (LSH) [11], for which projection functions are random vectors drawn from a specific distribution. Many extensions of LSH have been proposed for min-hash [2] and kernelized version [14].

Data-dependent projection based methods consider the data distribution for achieving higher accuracy. Some of them applied spectral graph partitioning [22] and graph Laplacian [16]. Gong and Lazebnik [6] have proposed a binary code embedding method called 'iterative quantization', which directly minimizes the quantization error by computing an optimal rotation of PCA directions. Lee et al. [15] used multiple bits for each projection to reduce the quantization error. He et al. [7] have presented a joint optimization framework of projection functions for both search accuracy and time.

The distances among binary codes obtained with most of the projection based methods mentioned above can be efficiently computed by the Hamming distance, which can be efficiently computed.

Quantization based methods. These techniques are closely related to clustering. In these methods, a binary code of a data point encodes the index of a cluster containing the data point. Product Quantization (PQ) [13] decomposes the original data space into lower-dimensional subspaces and quantizes each subspace separately using k-means clustering. It then computes a binary code as a concatenation of cluster indices, encoded in subspaces.

He et al. [8] have proposed k-means hashing, which optimizes cluster centers and their cluster indices in a way that the Hamming distance between encoded cluster indices reflects distances between cluster centers. Recently, Ge et al. have proposed Optimized PQ (OPQ) [5] that optimizes PQ by minimizing quantization distortions with respect to the space decomposition and code books. OPQ shows the state-of-the-art results over other quantization and projection based methods. Norouzi and Fleet have presented Cartesian k-means [19] that also reduces the quantization distortions of PQ in a similar manner to OPQ.

Our encoding scheme is based on the product space that PQ and OPQ are based on. Unlike PQ and OPQ, our method quantizes distances from data points to their cluster center for more accurately estimating distances between points based on our encoded binary codes.

Hamming embedding [12] uses an orthogonal projection and thresholding projected values for computing binary codes only within a cluster. This approach provides higher accuracy within each cluster and works for image retrieval. On the other hand, this method is not designed for accurately measuring distances between points that are contained in different clusters.

3. Background and Motivations

Let us define notations that we will use throughout the paper. We use $X = \{x_1, ..., x_n\}, x_i \in \mathbb{R}^D$ to denote a set of n data points in a D-dimensional space, A binary code corresponding to each data point x_i is defined by $b_i = \{0, 1\}^L$, where L is the length of the code. We denote $d(x, y)$ as the Euclidean distance $\| x - y \|$.

We first briefly review Product Quantization (PQ) [13] that our work is built upon and its two distance measures. Let us denote a point $x \in \mathbb{R}^D$ as the concatenation of M subvectors, $x = [x^1, ..., x^M]$. For simplicity, we assume that the dimensionality of data D is divisible by the number of subspace M. Each i^{th} subspace is encoded by L/M bits and we thus have $k(= 2^{L/M})$ codebook vectors, $\{c_1^i, ..., c_k^i\}$.

A vector quantizer $q^i(x^i)$ given i^{th} subspace is defined as following:

$$q^i(x^i) = \arg\min_{c_j^i} d(x^i, c_j^i).$$

The sub-binary code, b^i, computed from i^{th} subspace elements of x is computed by encoding an codebook index of the nearest cluster:

$$b^i = B\big(\arg\min_j d(x^i, c_j^i), \frac{L}{M}\big),$$

where the function $B(v, l)$ converts an integer $v - 1$ to a binary string with a length l; e.g., $B(6, 4) = 0101$. PQ then maps x to the concatenation of sub-binary codes, $b = [b^1, ..., b^M]$.

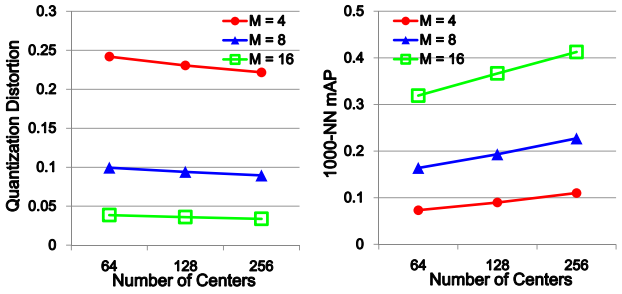
PQ uses two distance computation schemes: Symmetric Distance (SD) and Asymmetric Distance (AD). SD is used, when both vectors x and y are encoded, and is defined as following:

$$d_{SD}^{PQ}(x, y) = \sqrt{\sum_{i=1}^{M} d\big(q^i(x), q^i(y)\big)^2}. \quad (1)$$

On the other hand, AD is used, when only data point x is encoded, but query y is not, and is defined as following:

$$d_{AD}^{PQ}(x, y) = \sqrt{\sum_{i=1}^{M} d\big(q^i(x), y\big)^2}. \quad (2)$$

Recently proposed Optimized PQ (OPQ) [5] has the same underlying scheme as PQ, but operating on a transformed feature space obtained with an optimized projection matrix. OPQ shows the state of the art performance in the field of approximate nearest neighbor search. Cartesian k-means (ck-means) and OPQ are based on a very similar generalization of PQ, as stated in [19]. Our method is independent from clustering and construction methods for subspaces. Our method, therefore, can be built on the subspaces created by ck-means in the same manner to what we do for OPQ. In this paper we explain our concept and its benefits of our method on top of PQ and OPQ for the sake of succinct explanations.



(a) (b)

Figure 2. The left figure shows the empirical quantization distortions as a function of the number of clusters in each subspace. M indicates the number of subspaces used. The right figure shows mAP curves of 1000-nearest neighbor search with varying numbers of clusters for each subspace. We use OPQ on the GIST-960D dataset for the experiments.

3.1. Motivations

Quantization distortion has been identified to be closely related to the search accuracy [5]. OPQ directly aims to reduce the quantization distortion of PQ. In general we can reduce the quantization distortion by allocating longer binary codes, i.e., having more clusters. In particular, we have studied the relationship between the number of clusters and quantization distortion, ξ, which is defined as follows [13, 5]:

$$\xi = \frac{1}{M}\sum_{i=1}^{M}\frac{1}{n}\sum_{j=1}^{n} d\big(x_j^i, q^i(x_j^i)\big)^2.$$

We experimentally measure quantization distortions as a function of the number of clusters (Fig. 2(a)). As expected, the quantization distortion reduces as we have more bits. However we observe that the decreasing rate of the quantization distortion is marginal with respect to the number of centers. Similarly we observe the same diminishing return of having more clusters for the search accuracy, as shown in Fig. 2(b).

Once a data point is encoded as a compact code, a reconstructed position from the code is set as the center of the corresponding cluster of the code. Distances between encoded binary codes at the search phase are estimated only with such center positions. One can easily see that the error of estimated distances depends on the quantization distortion. Specifically, it has been shown previously that the distance is biased and the error is statistically bounded by two times of the quantization distortion [13]. It is also observed that error-corrected versions of distance measures can reduce the bias, but increase the distance variance, resulting in even worse search accuracy.

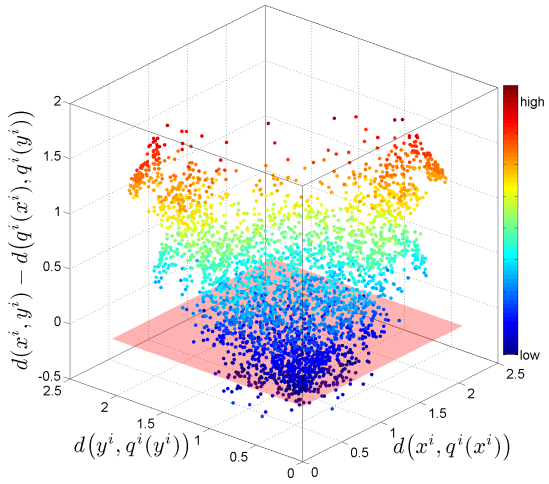


Figure 3. This figure visualizes errors of the symmetric distances. We sample two random points, x and y, in a randomly selected subspace. The x-axis indicates the distance between x^i and its corresponding cluster's center $q^i(x^i)$, and y-axis shows similar information for y. The vertical axis is the difference between the actual distance $d(x^i, y^i)$ and its estimated distance $d(q^i(x^i), q^i(y^i))$. The errors of estimated distances tend to be higher as the distance between data points and their corresponding clusters becomes larger. We use OPQ to define subspaces and clusters with the GIST-960D dataset.

We have empirically studied a functional relationship between the errors of estimated distances and the actual distance of data points from centers of their corresponding clusters (Fig. 3). We have found that the estimated distances tend to have higher errors, as data points are further away from centers of their corresponding clusters.

These results suggest that by reducing quantization distortions, we can predict the distances between data points more reliably, i.e. lower variance. Motivated by this, we allocate additional bits to directly encode the distances of data points from their corresponding cluster centers in each subspace, instead of constructing more clusters and encoding data with them.

4. Our Approach

In this section we explain our binary code encoding scheme, Distance-encoded Product Quantization (DPQ), and two distance metrics tailored to the scheme.

4.1. Our Binary Code Encoding Scheme

Our encoding scheme can be used with any hashing techniques that encode cluster indices in computed binary codes. For simplicity we explain our method by following the PQ framework. Combining our method with OPQ is straightforward, since we only need to apply an estimated rotation projection to the input feature space.

Suppose that the binary code length assigned for encoding the information in each subspace is L/M bits, where L and M indicate the overall binary code length and the number of subspaces, respectively. In each subspace, our method encodes the distance of a data point from the center of its cluster containing the point as well as the index of the cluster. Fig. 1 shows a visual example of our encoding method. Specifically, we allocate l_c bits for encoding the cluster index, and l_d bits for the distance from its cluster center. We define $h(= 2^{l_d})$ different distance thresholds, $t^i_{j,1}, ..., t^i_{j,h}$, for c^i_j, the center of the cluster j in i^{th} subspace. The binary code of a data point, x, for the i^{th} subspace is then the concatenation of the nearest center index, $\hat{j}$, and the quantized distance index, $\hat{k}$, as follows:

$$b^i(x^i) = [B(\hat{j}, l_c), B(\hat{k}, l_d)],$$

where

$$\hat{j} = \arg\min_j d(x^i, c^i_j),$$

and $\hat{k}$ is the value satisfying the following:

$$t^i_{\hat{j},\hat{k}-1} \leq d(x^i, c^i_{\hat{j}}) < t^i_{\hat{j},\hat{k}}.$$

$t^i_{j,0}$ and $t^i_{j,h}$ are defined as 0 and ∞, respectively. We also use $P^i_{j,k}$ to denote a set of data points that are encoded by the cluster j with threshold k in i^{th} subspace. We use P^i_j to denote all the data points of the union of $P^i_{j,1}$, ... , $P^i_{j,h}$.

Computing thresholds. In order to choose distance thresholds determining h disjoint regions within each cluster, we identify points $|P^i_j|$ contained in the cluster j in i^{th} subspace. We then construct distances of those points from the cluster center, c^i_j. For choosing thresholds, we first compute h different regions in a way that we minimize the variances of distances of points contained in each region, i.e., minimizing the within-region variance.

It is also important to balance the number of points contained in each region. To achieve this, we enforce the number of points in each $P^i_{j,k}$ to be between $(|P^i_j|/h - |P^i_j|/h^2)$ and $(|P^i_j|/h + |P^i_j|/h^2)$; in this equation we use h^2 to achieve a near-balance among the numbers of points allocated to regions. Each cluster has a small number of points, and the search space of the candidate set for computing thresholds given the balancing criterion are small. As a result, we can efficiently find thresholds that minimize the within-region variance even by exhaustively searching the optimal one. Alternatively, we can also use balanced-clustering techniques such as [1] for accelerating the aforementioned process of computing thresholds.

4.2. Distance Metrics

We propose two distance metrics, statistics and geometry based metrics, that consider quantization distortions for achieving higher accuracy.

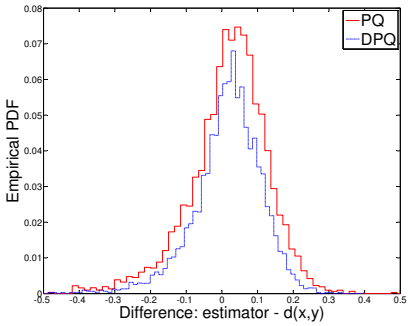


Figure 4. This figure shows the distribution of differences from the ground-truth distances to estimated results from statistics based distance metrics used for our method and PQ. We draw 100 K pairs of samples randomly chosen from the GIST-1M-960D dataset. Our method shows the bias of 0.0091, which is reduced from 0.0096 of PQ. Similarly, the variance of our method is 0.0099, while PQ has 0.0133.

Statistics based distance metric. Jegou et al. [13] have discussed the quantization distortion of each cluster and suggested error corrected versions for Symmetric Distance (SD) and Asymmetric Distance (AD).

For AD, we start with the following distance metric, Error-Corrected AD (ECAD), considering the quantization distortion of x [13] :

$$d^{PQ}_{ECAD}(x,y)^2 = d^{PQ}_{AD}(x,y)^2 + \sum_{i=1}^{M} \xi^i_j(x^i), \tag{3}$$

where $\xi^i_j(x^i)$ is a pre-computed error correcting term for the cluster j containing x^i. The error correcting term $\xi^i_j(\cdot)$ is defined as the average distortion of the cluster j in the i^{th} subspace:

$$\xi^i_j(x^i) = \frac{1}{|P^i_j|} \sum_{w=1}^{|P^i_j|} d(p_w, c^i_j)^2,$$

where p_w is w^{th} point of P^i_j. We can similarly define an Error-Corrected distance metric for SD (ECSD) considering quantization distortions of both x and y.

We can easily extend these error-corrected distance metrics to our encoding scheme. For our method we define a new error correcting term, $\xi^i_{j,k}(x^i)$, with $x^i \in P^i_{j,k}$, which contains points in the k^{th} region of the cluster j in the i^{th} subspace:

$$\xi^i_{j,k}(x^i) = \frac{1}{|P^i_{j,k}|} \sum_{w=1}^{|P^i_{j,k}|} d(p_w, c^i_j)^2. \tag{4}$$

Interestingly, [13] reported that the error-corrected distance metrics did not improve accuracy over metrics without the error correcting terms, mainly because the error-corrected distance metrics have higher variances. In contrast, our encoding scheme with our error-correcting terms (Eq. 4) shows higher accuracy over ours without the terms.

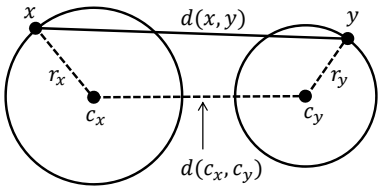


Figure 5. This figure shows two points x and y on the hyper-spheres centered at c_x and c_y respectively. r_x and r_y represent the radii of hyper-spheres.

In order to identify reasons why the similar error-correcting terms result in contrasting results between our encoding scheme and PQ, we have measured the bias and variance of these two distance estimators. As can be seen in Fig. 4, the variance and bias of our error-corrected distance metric are reduced from those of PQ. Since our encoding scheme quantizes the distance of a data point from its corresponding cluster, our error correcting term (Eq. 4) reduces the bias and, more importantly, the variance of the distance estimates effectively.

Geometry based distance metric. We now propose a novel geometric approach to develop distance metrics for our encoding scheme. Suppose that two high dimensional points x and y are randomly chosen on the surfaces of two hyper-spheres centered at c_x and c_y, respectively, with r_x and r_y radii (Fig. 5). Given these geometric configurations, the vector $x - y$ is reformulated as:

$$x - y = (c_x - c_y) + (x - c_x) + (c_y - y). \tag{5}$$

Our goal is to estimate the length of the vector $x - y$ with available information within our encoding scheme.

As the dimension of data points goes higher, the surface area of the hyper-sphere becomes closer to the length of its equator. One may find this is counter-intuitive, but this has been proved for high dimensional spaces [10]. Given a D dimensional hyper-sphere, a cross section of the hyper-sphere against a horizontal hyperplane is $D-1$ dimensional hyper-sphere. The length of the cross section is longest in the equator. It then exponentially decreases with a function of $D-1$ degree, as the cross section gets closer to the north pole. As a result, as we have a higher dimensional space, the length of the equator takes a more dominant factor in the surface area of the hyper-sphere.

Given those x and y points, we rotate our randomly chosen points such that x is located at the north pole. By applying the above theorem, we have a higher probability that another point y is located on the equator of the rotated hyper-sphere, as we have higher dimensional space. As a result, we can conclude that it is highly likely that two vectors $x - c_x$ and $y - c_y$ are orthogonal, when these data points are in a high-dimensional space. Similarly, we can show

that these two vectors are also orthogonal to another vector $c_y - c_x$. We have also experimentally checked its validity with a benchmark consisting of 960 dimensional GIST descriptors. For this we have measured the average angle between two randomly chosen points (i.e. 100K pairs) from a random cluster. On average their average angle is $89.81°$ with the standard deviation of $\pm 7.1°$.

Since $c_x - c_y$, $x - c_x$, and $c_y - y$ are highly likely to be mutually orthogonal in a high-dimensional space, the squared magnitude of the vector $x - y$ can be computed as follows:

$$\begin{aligned} \| x - y \|^2 &= \| (c_x - c_y) + (x - c_x) + (c_y - y) \|^2 \\ &\approx \|c_x - c_y\|^2 + \|x - c_x\|^2 + \|c_y - y\|^2 \quad (6) \end{aligned}$$

The first term, $\| c_x - c_y \|^2$, is pre-computed as the distance between different clusters. The second and third terms indicate how far x and y are located from the centers of their corresponding clusters.

In our encoding scheme, the second term can be estimated by using points $p_w \in P^i_{j_x,k_x}$, where j_x and k_x are encoded cluster and threshold indices for x, respectively. Specifically, the second term is estimated as the average distance, $\overline{r^i_{j_x,k_x}}$, from the center $c^i_{j_x}$ to $p_w \in P^i_{j_x,k_x}$:

$$\overline{r^i_{j_x,k_x}} = \frac{1}{|P^i_{j_x,k_x}|} \sum_{w=1}^{|P^i_{j_x,k_x}|} d(p_w, c^i_{j_x}). \quad (7)$$

The third term of Eq. 6 is estimated in the same manner with points in $P^i_{j_y,k_y}$, where j_y and k_y are chosen cluster and threshold indices for y.

We then formulate our distance metric based on Eq. 6 and Eq. 7. Our GeoMetry based squared Symmetric Distance (GMSD) between two points x and y is defined as:

$$d^{DPQ}_{SD}(x,y)^2 = \sum_{i=1}^{M} \Big(d\big(q(x^i), q(y^i)\big)^2 + \overline{r^i_{j_x,k_x}}^2 + \overline{r^i_{j_y,k_y}}^2 \Big). \quad (8)$$

Our GeoMetry based squared Asymmetric Distance (GMAD) between encoded data x and query y is defined similarly as:

$$d^{DPQ}_{AD}(x,y)^2 = \sum_{i=1}^{M} \Big(d\big(q(x^i), y\big)^2 + \overline{r^i_{j_x,k_x}}^2 \Big). \quad (9)$$

Note that $\overline{r^i_{j,k}}$ is precomputed and stored as a lookup table as a function of i, j, and k values.

One may find that our geometry based distance metric using the average distance (Eq. 7) of points from their cluster have a similar form to our statistics based distance metric using the error correcting term (Eq. 4). One can theoretically show that our statistics based metric generates a distance equal or larger than that of the geometry based metric, and their value difference is the variance of distances between data points and their corresponding clusters. It is hard to theoretically tell which approach is better, but these two different metrics consider different aspects of input data points; the statistics based metric considers the variance of distances, while the geometry based one does not.

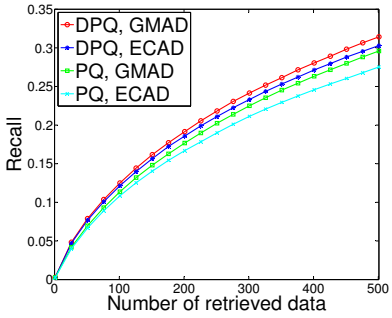


Figure 6. This figure shows the recall curves with respect to the number of retrieved data. Our geometry based distance metric, Ours w/ GMAD, improves our statistics based metric, Ours w/ ECAD, and the PQ method, PQ w/ ECAD. The results are obtained with GIST-1M-960D dataset and the number of ground truth K=100. We use 64 bits in this experiment and PQ uses 8 bits for each subspace to define 256 centers. Our method uses 7 bits for the center and 1 bit for the distance quantization.

Empirically we, however, have observed that the geometry based metric shows better performance, 6%, on average over our statistics based metric (Fig. 6). Furthermore, when we integrate our geometry based distance metric within PQ, we have observed that our geometry-based distance metric, PQ w/ GMAD, shows higher accuracy over the statistics based one proposed for PQ, PQ w/ ECAD. These results demonstrate benefits of our geometry-based distance metric.

5. Evaluation

In this section we evaluate our method for approximate nearest neighbor search and compare its results to the state-of-the-art techniques.

5.1. Protocol

We evaluate our method on the following public benchmarks:

- **GIST-1M-960D**: One million 960D GIST descriptors that are also used in [13, 5].
- **GIST-1M-384D**: One million 384D GIST descriptors, a subset of Tiny Images [21].
- **BoW-1M-1024D**: One million 1024D Bag-of-visual-Words (BoW) descriptors, which are subsets of the ImageNet database [3].

For all the experiments, we use 1000 queries that do not have any overlap with the data points. We compute $K = 1000$ nearest neighbors for each query point. Also

	Bits		Num. of Subspaces			
	l_c	l_d	2	4	8	16
OPQ baseline	6	0	0.045	0.117	0.238	0.408
Ours+OPQ	6	2	0.101	0.230	0.408	0.584
OPQ baseline	7	0	0.061	0.144	0.276	0.459
Ours+OPQ	7	1	0.106	0.236	0.415	0.595
OPQ baseline	8	0	0.077	0.171	0.315	0.500

Table 1. This table shows mAPs with different bit budgets. mAPs are measured with the asymmetric distances in **GIST-1M-960D** dataset. 'OPQ baseline' is provided here to see how much our scheme improves the performance by using quantized distances.

we compute the ground truth for each query by performing the linear scan. The performance of different methods is measured by the mean Average Precision (mAP). To verify benefits of our method we compare the following methods:

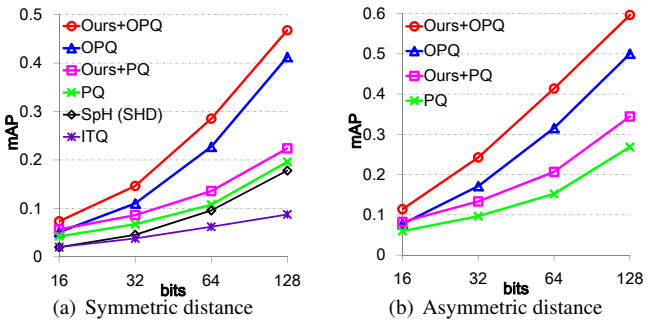
- **PQ:** Product Quantization [13]
- **OPQ:** Optimized Product Quantization [5]. We use the non-parametric version, because it shows better accuracy than the parametric version.
- **Ours+PQ:** Our method using the same subspaces and clustering method as used for **PQ**, and the geometry based distance metrics.
- **Ours+OPQ:** Our method using the same subspaces, projection matrix, and clustering method, as used for **OPQ**, and the geometry-based distance metric.

For all the methods, $100K$ data points randomly sampled from the dataset are used in the training stage, and we allow 100 iterations in the k-means clustering. We assign 8 bits for each subspace as suggested in [13, 5]. **PQ** and **OPQ** then have 256 centers in each subspace, and the number of subspace M is $L/8$, where L is the given code-length. We also use public source codes for all the compared methods including **PQ** and **OPQ**. For **PQ** and **OPQ** we use symmetric and asymmetric distances (Eq. 1 and Eq. 2), which achieved the best accuracy according to their original papers [13, 5]. On the other hand, the proposed binary encoding scheme and our geometry based distance metrics (Eq. 8 and Eq. 9) are used for **Ours+PQ** and **Ours+OPQ**.

In our method, we have parameter l_d (the number of bits) for the distance quantization. We observe that $l_d = 1$ or $l_d = 2$ gives reasonable performance across tested benchmarks. Experimental results with these parameter values are given in Tab. 1. Although the performances with $l_d = 1$ and $l_d = 2$ are both similarly good, we pick $l_d = 1$ for all the following tests. In other words, we use 128 centers in each subspace and 2 distance quantizations for each cluster.

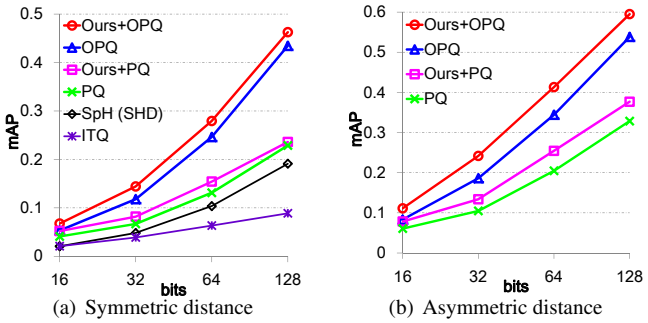
5.2. Results

Fig. 7 and Fig. 8 show mAPs of nearest neighbor search for all the tested methods on **GIST-1M-960D** and **GIST-1M-384D** datasets, respectively. **Ours+OPQ** shows better



(a) Symmetric distance (b) Asymmetric distance

Figure 7. Comparisons on **GIST-1M-960D**. The left and right graphs show the mAP curves with symmetric and asymmetric distances, respectively.



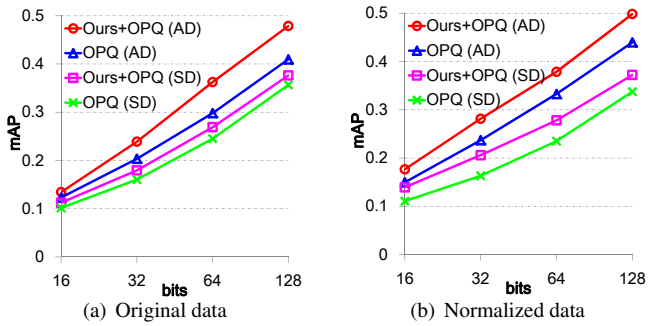
(a) Symmetric distance (b) Asymmetric distance

Figure 8. Comparisons on **GIST-1M-384D**. The left and right graphs show the mAP curves with symmetric distances and asymmetric distances, respectively.

results over all the tested methods across all the tested code lengths ranging from 16 bits to 128 bits. Moreover, our methods **Ours+OPQ** and **Ours+PQ** consistently improve both **PQ** and **OPQ** in all the tests, respectively. In addition, we show mAPs of some of well-known binary embedding techniques, iterative quantization (ITQ) [6] and spherical hashing (SpH) [9] with its distance metric, SHD, for symmetric distance cases. Since these results are lower than PQ, we do not show them in other tests. This improvement clearly demonstrates the merits of our proposed binary code encoding scheme and the new distance metrics.

It also shows an interesting trend that in general the relative improvement of our method over the baseline **PQ** and **OPQ** is more significant for the higher dimensional cases, e.g. 960D vs 384D. Furthermore, performance improvement for **Ours+OPQ** is in general larger than **Ours+PQ**, which indicates that better subspaces would provide more benefit for our method.

We also perform the same experiments with another popular global image descriptor Bag-of-Words (BoW) on **BoW-1M-1024D**, and its results are shown in Fig. 9 with SD and AD cases. Since BoW descriptors are also used with the vector normalization according to L_2 norm, we test the normalized data too. Our method robustly provides better



(a) Original data (b) Normalized data

Figure 9. Comparisons on **BoW-1M-1024D**. The left and right graphs show results for original and L_2 normalized data with symmetric and asymmetric distance metrics.

performance compared to **OPQ**.

Finally, we measure the computational times with **GIST-1M-960D** dataset at 64 bits encoding. **Ours+PQ** takes 94.34s to encode one million data, 21ms for one million SD computations, and 819ms for one million AD computations, while **PQ** takes 187.45s, 21ms, and 829ms, respectively. Since our method uses less number of cluster centers, it has a faster encoding performance. Also, distance computation time of our methods are similar to that of PQ.

6. Conclusion

We have presented a novel compact code encoding scheme that quantizes distances of points from their corresponding cluster centers in each subspace. In addition, we have proposed two different distance metrics tailored for our encoding scheme: statistics and geometry based metrics. We have chosen the geometry based metrics, since it consistently show better accuracy over the statistics based one. We have tested our method against the-state-of-the-art techniques with three well known benchmark, and have demonstrated benefits of our method over them.

Many interesting research directions lie ahead. Our current encoding scheme uses a fixed bit length for quantizing distances from all the clusters. A clear improvement would be to use a variable bit length for different clusters depending on quantization distortions of them. The key challenge is to design an effective optimization for deciding the bit distributions between encoding clusters and quantizing distances. We leave it as future work.

Acknowledgements

We would like to thank anonymous reviewers for constructive comments. S.-E. Yoon is a corresponding author of the paper. This work was supported in part by NRF-2013R1A1A2058052, DAPA/ADD (UD110006MD), MEST/NRF (2013-067321), and IT R&D program of MOTIE/KEIT [10044970].

References

[1] A. Banerjee and J. Ghosh. On scaling up balanced clustering algorithms. In *SIAM Int. Conf. on Data Mining*, 2002.

[2] O. Chum, J. Philbin, and A. Zisserman. Near duplicate image detection: min-hash and tf-idf weighting. In *BMVC*, 2008.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248 –255, june 2009.

[4] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM TOMS*, 3(3):209–226, 1977.

[5] T. Ge, K. He, Q. Ke, and J. Sun. Optimized product quantization for approximate nearest neighbor search. In *CVPR*, 2013.

[6] Y. Gong and S. Lazebnik. Iterative quantization: a procrustean approach to learning binary codes. In *CVPR*, 2011.

[7] J. He, R. Radhakrishnan, S.-F. Chang, and C. Bauer. Compact hashing with joint optimization of search accuracy and time. In *CVPR*, 2011.

[8] K. He, F. Wen, and J. Sun. K-means hashing: an affinity-preserving quantization method for learning binary compact codes. In *CVPR*, 2013.

[9] J.-P. Heo, Y. Lee, J. He, S.-F. Chang, and S.-E. Yoon. Spherical hashing. In *CVPR*, 2012.

[10] J. Hopcroft. High dimensional data in course notes of mathematical foundations for the information age, 2010.

[11] P. Indyk and R. Motwani. Approximate nearest neighbors: toward removing the curse of dimensionality. In *STOC*, 1998.

[12] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large-scale image search. In *ECCV*, 2008.

[13] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE TPAMI*, 2011.

[14] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing for scalable image search. In *ICCV*, 2009.

[15] Y. Lee, J.-P. Heo, and S.-E. Yoon. Quadra-embedding: Binary code embedding with low quantization error. *Computer Vision and Image Understanding (CVIU)*, 2014. To appear.

[16] W. Liu, J. Wang, S. Kumar, and S.-F. Chang. Hashing with graphs. In *ICML*, 2011.

[17] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISAPP*, pages 331–340, 2009.

[18] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006.

[19] M. Norouzi and D. J. Fleet. Cartesian k-means. In *CVPR*, 2013.

[20] C. Silpa-Anan, R. Hartley, S. Machines, and A. Canberra. Optimised kd-trees for fast image descriptor matching. In *CVPR*, 2008.

[21] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *CVPR*, 2008.

[22] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *NIPS*, 2008.