Distance Encoded Product Quantization for Approximate K-Nearest Neighbor Search in High-Dimensional Space

Jae-Pil Heo, Zhe Lin, and Sung-Eui Yoon

Abstract—Approximate K-nearest neighbor search is a fundamental problem in computer science. The problem is especially important for high-dimensional and large-scale data. Recently, many techniques encoding high-dimensional data to compact codes have been proposed. The product quantization and its variations that encode the cluster index in each subspace have been shown to provide impressive accuracy. In this paper, we explore a simple question: is it best to use all the bit-budget for encoding a cluster index? We have found that as data points are located farther away from the cluster centers, the error of estimated distance becomes larger. To address this issue, we propose a novel compact code representation that encodes both the cluster index and quantized distance between a point and its cluster center in each subspace by distributing the bit-budget. We also propose two distance estimators tailored to our representation. We further extend our method to encode global residual distances in the original space. We have evaluated our proposed methods on benchmarks consisting of GIST, VLAD, and CNN features. Our extensive experiments show that the proposed methods significantly and consistently improve the search accuracy over other tested techniques. This result is achieved mainly because our methods accurately estimate distances.

Index Terms—Vector quantization, nearest neighbor search, image retrieval, compact code, high-dimensional search

1 Introduction

Approximate K-Nearest Neighbor (ANN) search has been an active research problem across many fields in computer science including computer vision, machine learning, computational geometry, data mining, and information retrieval. 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 compute vision applications such as image/video retrieval, feature matching, and image classification. Thus we restrict our discussion on approximate nearest neighbor techniques tailored to computer vision.

Conventional ANN search methods based on hierarchical structures [1], [2] can be inefficient in terms of both computational and memory costs, when the size of database is large and the dimensionality of the feature space is high, as is the case for web-scale image retrieval. Most of recent ANN search techniques for high-dimensional data in computer vision, machine learning, and related literatures are encoding data to compact codes based on hashing [3] or vector quantization [4]. Hashing approaches assign similarity-preserving binary codes to data. The major advantage of hashing techniques over vector quantization methods is that neighbors can be identified by accessing similar codes or near buckets from the query. However, hashing techniques do not accurately estimate the actual distances among data since they are based on the hamming distance. On the other hand, vector quantization techniques approximate data as codeword vectors (i.e. cluster centers). Those dictionary vectors are utilized for distance estimation. In vector quantization methods, the estimated distances are more accurate as larger dictionary we use. However, using many codewords require higher computational and memory costs.

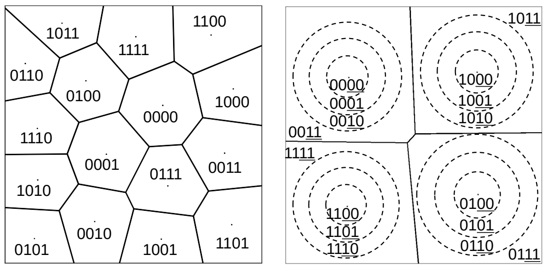
The product quantization (PQ) [5] can efficiently define exponentially large number of codewords with small memory footprint and computational costs. PQ divides the feature space into several disjoint subspaces, and each subspace is partitioned into clusters independently. Specifically, PQ can define k^M codewords when we utilize M subspaces and quantize each subspace into k clusters. The space complexity to store such codewords is $O(kD)$ where D stands for the dimensionality of data. Furthermore, distance between two data points can be computed in $O(M)$ via precomputed look-up tables. PQ and its improvements [6], [7] have shown the state-of-the-art performance in ANN search problem.

PQ can effectively generate many codewords based on the Cartesian product of subspaces 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. Furthermore, we have observed that the error of estimated distances tend to be larger as the data points are farther from the codewords (Sec. 3.2). This is mainly because they encode only the clusters containing data points, but are not designed to consider how far data points are located away from cluster centers.

In this paper we take account of the residual distance between a data point and its corresponding cluster center. Those residual distances are encoded in the compact codes

- *Jae-Pil Heo is with Sungkyunkwan University, South Korea. E-mail: jaepilheo@gmail.com*
- *Zhe Lin is with Adobe Research. E-mail: zlin@adobe.com*
- *Sung-Eui Yoon is with Korea Advanced Institute of Science and Technology (KAIST), South Korea. E-mail: sungeui@kaist.edu*

Manuscript received mm/dd/yyyy.



(a) PQ (b) DPQ

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

and utilized when estimating distances among data points. The contributions of this paper are summarized as follows:

1) We propose a new compact code representation for high-dimensional data, Distance encoded Product Quantization (DPQ). We follow exactly the same procedure as in PQ 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 residual distance from the data point to its closest cluster center. (Sec. 4.1)
2) We provide two different distance estimators, statistics and geometry based distance estimators, tailored to our encoding scheme. Especially, our geometry based distance estimator is based on novel geometric reasoning for high-dimensional space. (Sec. 4.2)
3) We further extend DPQ to Global Distance encoded Product Quantization (GDPQ) that encodes the global residual distances in the original feature space instead of subspaces. (Sec. 5)

Our method is independent from subspace and codeword optimization algorithms. In other words, we can take advantage of advances in PQ-like vector quantization techniques including PQ itself, OPQ [6], and LOPQ [8]. In this paper, we specifically apply our method to PQ and OPQ to validate merits of encoding residual distances and utilizing them in the distance estimation. We have evaluated our method on four benchmarks consisting of popular and recent image descriptors, GIST, VLAD, and CNN features (Sec. 6). Experimental results show that our DPQ with new distance estimators significantly and consistently outperforms the baseline methods on tested benchmarks. Furthermore, GDPQ provides even better results than the baselines and DPQ. These improvements are mainly caused by more accurately estimated distances. Overall our method is simple, but results in significant performance improvements over baseline PQ-like vector quantization methods.

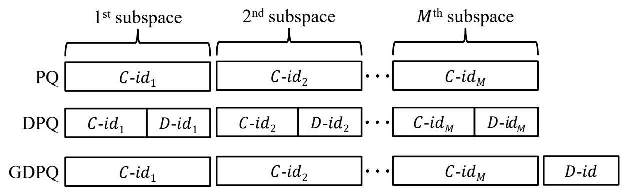


Fig. 2. This figure shows compact codes computed by Product Quantization (PQ) [5], Distance encoded Product Quantization (DPQ, Sec. 4), and Global Distance encoded Product Quantization (GDPQ, Sec. 5). In each subspace, DPQ encodes the cluster index ($C\text{-}id_i$) and quantized distance ($D\text{-}id_i$) between a data point and its corresponding cluster center, while PQ encodes only the cluster index. On the other hands, GDPQ encodes quantized global residual distances in the original space instead of subspaces.

2 Related Work

There have been many approaches that utilize hierarchical data structures for ANN search. Notable examples of such tree structures based on the recursive space partitioning include kd-trees [1], [9], R-trees [2], Vantage Point trees [10] and hierarchical k-means trees [11]. However, most of those techniques can provide even slower search performance compared to exhaustive scan especially for high-dimensional data. Furthermore, they do not provide compact data representation, and thus are less effective for handling billion-scale problems.

Recently compact code representations have been actively studied, since they provide a high compression rate by encoding high-dimensional data into compact codes, and fast distance (i.e. similarity) computation by simple bit-string operations or a precomputed lookup table. We categorize those compact code representation techniques into two categories: hashing and vector quantization based approaches.

Hashing methods project high-dimensional data to the Hamming space. Data points are encoded as similarity-preserving binary codes and neighbors can be identified by traversing similar binary codes or near hash buckets. These techniques can be categorized further into data-independent and data-dependent methods. The popular data-independent hashing techniques are the Locality Sensitive Hashing (LSH) [12] and its variations [13], [14], [15]. On the other hand, data-dependent hashing methods consider the data distribution for achieving higher accuracy. Notable examples include [16], [17], [18], [19]. Most of the hashing methods mentioned above estimate the distance among data by the Hamming distance which can be efficiently computed. The hamming distances, however, only give the discretized ranking but do not provide precise distances by nature.

The quantization based methods are closely related to clustering. In these approaches, a compact code of a data point encodes the index of a cluster (i.e. codeword index) containing the data point [20], [21]. Product Quantization (PQ) [5] decomposes the original data space into lower-dimensional subspaces and quantizes each subspace separately using k-means clustering. It then computes a compact code as a concatenation of cluster indices, encoded in subspaces. Ge et al. have proposed Optimized PQ (OPQ) [6] that optimizes PQ by minimizing quantization distortions with respect to the space decomposition and codebook vectors. Norouzi and Fleet have presented Cartesian k-means [7] that also reduces the quantization distortions of PQ in a similar manner to OPQ. Locally Optimized OPQ (LOPQ) [8] has been proposed to optimize a OPQ for each region defined by a coarse quantizer.

The residual quantization (RQ) [22], [23] is a recursive quantization method working in the original data space. The residual vector produced in the previous quantization step is further quantized in the next step. A data point is then approximated as a sum of codeword vectors. The main difference between our proposed method and RQ is that we only encode the residual norms, which are scalar values, and utilize them in the estimation, while RQ recursively quantizes the residual vectors. Furthermore, our method can be combined with RQ to encode the distance between a data point and its approximation.

Hamming embedding [24] 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 contained in different clusters.

PQ and its recent improvements achieved state-of-the-art results in approximate K-nearest neighbor search by significantly outperforming those projection-based methods. This is partially due to fact that PQ leverages the product space and estimates distances more accurately. However, they have the problem of diminishing returns with an increasing number of clusters in each subspace, and they also have large bias and variances in distance estimation (as discussed in Sec. 3.2).

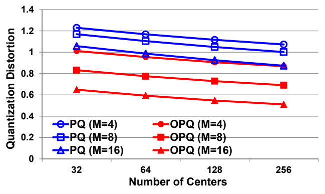
Our encoding scheme is based on the product space that PQ and OPQ are also based on. Unlike PQ and OPQ, our method encodes quantized distances from data points to their cluster center for more accurately estimating distances among points. Note that, our approach is orthogonal to PQ or its variations that optimize subspaces and codeword vectors. In other words, our method can be applied to such PQ-like techniques to provide additional performance improvements.

3 Background and Motivations

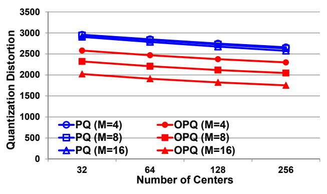
3.1 Background

Let us define notations that we will use throughout this 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 compact 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 $\parallel x - y \parallel$.

We first briefly review Product Quantization (PQ) [5] that our work is built upon and its two distance estimators. Let us denote a point $x \in \mathbb{R}^D$ as the concatenation of M sub-vectors, $x = [x^1, ..., x^M]$. For simplicity, we assume that the dimensionality of data D is divisible by the number of subspaces M. In PQ, points in each i^{th} subspace is encoded by L/M bits code and we thus have $N_K (= 2^{L/M})$ codebook vectors, $\{c_1^i, ..., c_{N_K}^i\}$, where those codebook vectors are cluster centroids computed by k-means clustering [25], [26].

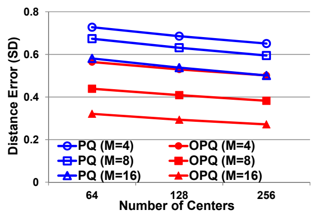


(a) GIST-1M-960D

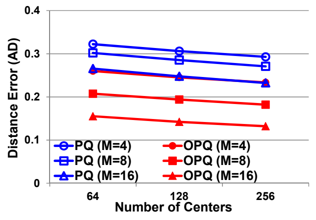


(b) CNN-1M-4096D

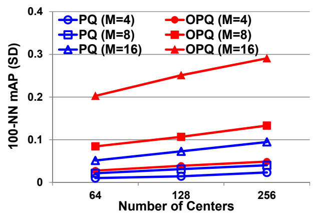
Fig. 3. This figure shows the empirical quantization distortions as a function of the number of clusters (=N_K) in each subspace on (a) 960-dimensional GIST descriptors and (b) 4096-dimensional CNN features. M indicates the number of subspaces used. Details about the datasets are given in Sec. 6.



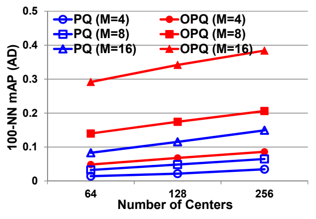
(a) Distance Error (SD)



(b) Distance Error (AD)



(c) 100-NN mAP (SD)



(d) 100-NN mAP (AD)

Fig. 4. (a) and (b) show the error of estimated symmetric distance (SD) and asymmetric distance (AD) with various number of subspaces, respectively. We randomly sample $100K$ pairs from the GIST-1M-960D dataset in the experiment. (c) and (d) similarly show mAP curves of 100-nearest neighbor search with SD and AD, respectively. M is the number of subspaces used.

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-code, b^i, computed from i^{th} subspace elements of x encodes an index of the nearest cluster:

$$b^i = B\big(\arg\min_j d(x^i, c_j^i), \frac{L}{M}\big), \tag{1}$$

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-codes, $b = [b^1, ..., b^M]$.

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

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

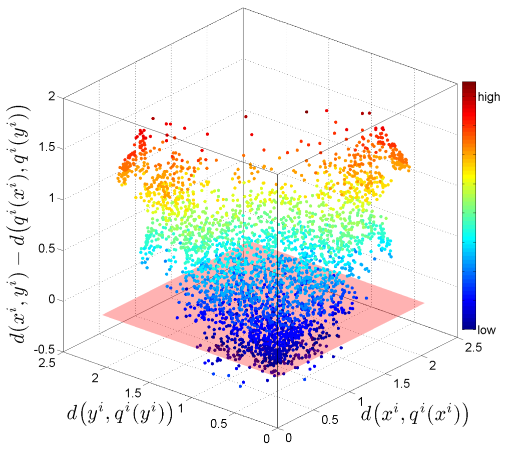


Fig. 5. This figure visualizes errors of the symmetric distances (SD). We sample two random points, x and y, in a randomly selected subspace. The x-axis indicates the distance between x^i and its corresponding cluster center $q^i(x^i)$, and y-axis shows similar information for y. The vertical axis is 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 cluster centers becomes larger. We use OPQ with $M = 8$ and $N_K = 256$ to define subspaces and clusters on the GIST-960D dataset.

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

$$d_{AD}^{PQ}(x, y) = \sqrt{\sum_{i=1}^{M} d(q^i(x), y)^2}. \quad (3)$$

Both SD and AD computations are accelerated by utilizing look-up tables. In case of SD, we construct a three dimensional table $T_{SD}(i, j, k) = d(c_j^i, c_k^i)^2$. Once we have the look-up table T_{SD} we can compute a squared SD by accessing and adding M elements in the table. Similarly, a two dimensional look-up table $T_{AD}(y, i, j)$ for a particular query y can computed as $T_{AD}(y, i, j) = d(y^i, c_j^i)^2$ and utilized in distance estimation.

3.2 Motivations

Quantization distortion has been identified to be closely related to the search accuracy [6]. OPQ directly aims to reduce the quantization distortion of PQ by solving Orthogonal Procrustes problem [17], [27]. In general we can reduce the quantization distortion by utilizing longer codes, i.e., having more codewords. In particular, we have studied the relationship between the number of clusters and quantization distortion, ξ, which is defined as follows [5], [6]:

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

We experimentally measure quantization distortions as a function of the number of clusters (Fig. 3). As expected, the quantization distortion reduces as we have more clusters. 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 distance estimation and search accuracy, as shown in Fig. 4.

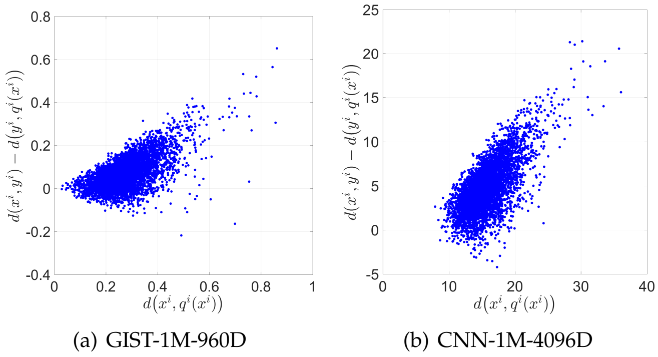


Fig. 6. The same protocol with Fig. 5 is used, however, the asymmetric distances are investigated on two different datasets. We treat that x is encoded but y is not. The y-axis indicates the error of asymmetric distance. Similar to the case of SD, the error of estimated distances tends to be higher when x is farther from its corresponding cluster center. The linear correlation coefficient of (a) and (b) are 0.68 and 0.70, respectively. In this experiment, we have used OPQ with $M = 8$ and $N_K = 256$ to define subspaces and clusters.

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 compact 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 [5]. An error-corrected version of distance measure which directly takes account of quantization distortions has also been proposed in [5], however, it is stated that the error-corrected distance estimator results in even worse search accuracy.

We have empirically studied a functional relationship between the errors of estimated distances and the distance of data points from their corresponding cluster centers (Fig. 5 and 6). 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. 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 Distance Encoded Product Quantization

In this section we introduce our compact code representation, Distance-encoded Product Quantization (DPQ), and two distance estimators tailored to our scheme.

4.1 Our Encoding Scheme

We explain our method and its benefit on top of PQ for the sake of succinct explanation. Our method can be used with any vector quantization techniques and can take advantage of any advanced PQ-like techniques since our method is independent from the subspace and cordword optimization algorithms. For instance, combining our method with OPQ is straightforward, since we only need to apply an optimized rotation to the input feature space. We use the term, Distance-encoded OPQ (DOPQ), to call our method that utilizes subspaces and clusters optimized by OPQ.

Suppose that the code length assigned for encoding the information in each subspace is L/M bits, where L and M indicate the overall 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. The distance is identical to the L_2-norm of residual vectors $|x^i - q^i(x^i)|$. Fig. 1 illustrates a visual example of our encoding method.

Specifically, we allocate L_C bits for encoding the cluster index to have $N'_K = 2^{L_C}$ clusters, and L_D bits for the encoding quantized distance from its cluster center. In order to quantize the residual norms, we define $N_D (= 2^{L_D})$ different distance thresholds, $t^i_{j,1}, ..., t^i_{j,N_D}$, for c^i_j, the center of the cluster j in i^{th} subspace. The compact 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:

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

$t^i_{j,0}$ and t^i_{j,N_D} 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, as follows:

$$P^i_{j,k} = \{x^i | j = \arg\min_v d(x^i, c^i_v) \text{ and } t^i_{j,k-1} \le d(x^i, c^i_j) < t^i_{j,k}\}$$

We use P^i_j to denote the union of $P^i_{j,1}$, ... , P^i_{j,N_D}:

$$P^i_j = \bigcup_{k=1}^{N_D} P^i_{j,k}$$

Computing thresholds

In order to choose distance thresholds determining N_D 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 N_D 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 $(|P^i_j|/N_D - |P^i_j|/N^2_D)$ and $(|P^i_j|/N_D + |P^i_j|/N^2_D)$; in this equation we use N^2_D 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. We can also use balanced-clustering techniques such as [28] for accelerating the aforementioned process of computing thresholds.

Alternatively, a simpler approach that equally distributes the distances into bins ($|P^i_{j,1}| = ... = |P^i_{j,N_D}|$) can be utilized. We found that this simple method provides a slightly lower search accuracy than the aforementioned variance based optimization with small N_D, but provides comparable performance with large N_D.

The time complexity to encode a data point of PQ is $O(M\frac{D}{M}N_K) = O(DN_K)$ [29]. The typical parameter of N_K is 256 [5], [6]. On the other hand, DPQ has a time complexity of $O\big(M(\frac{D}{M}N'_K + N_D)\big) = O(DN'_K + MN_D)$. Since DPQ produces the same lengthly compact code with PQ when $N'_K N_D = N_K$, we typically set two parameters (N'_K, N_D) as $(128, 2)$ or $(64, 4)$. As a result, in such typical settings our encoding method has a lower complexity compared to PQ (e.g. PQ: $O(256D)$, DPQ: $O(128D + 2M)$). In practice, DPQ has shown about two times faster encoding speed compared to PQ (Sec. 6).

4.2 Distance Estimators

We propose two distance estimators, statistics and geometry based estimators specialized to our compact code representation.

4.2.1 Statistics based distance estimator

Jegou et al. [5] have discussed the quantization distortion of each cluster and suggested error corrected versions of Symmetric Distance (SD) and Asymmetric Distance (AD). Those distance estimators use the quantization distortion of each cluster for the error correction.

For AD, we start with the following distance estimator, Error-Corrected AD (ECAD), considering the quantization distortion of the clusters containing x:

$$d^{PQ}_{ECAD}(x,y)^2 = d^{PQ}_{AD}(x,y)^2 + \sum_{i=1}^{M} \xi^i_j(x^i), \quad (4)$$

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:

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

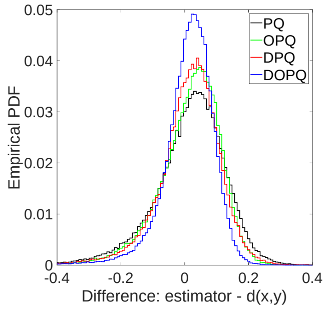
where p_w is w^{th} data point of P^i_j.

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:

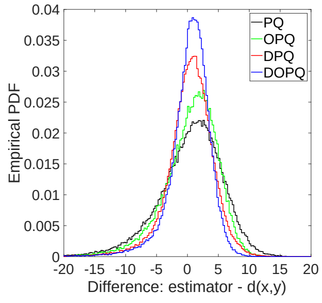
$$\xi^i_{j,k}(x^i) = \frac{1}{|P^i_{j,k}|} \sum_{w=1}^{|P^i_{j,k}|} d(p_w, c^i_j)^2. \quad (5)$$

We can similarly define an Error-Corrected distance estimator for SD (ECSD) and AD (ECAD) with the error correcting terms:

$$d^{DPQ}_{ECSD}(x,y)^2 = d^{PQ}_{AD}(x,y)^2 + \sum_{i=1}^{M} \xi^i_{j,k}(x^i) + \sum_{i=1}^{M} \xi^i_{j,k}(y^i) \quad (6)$$

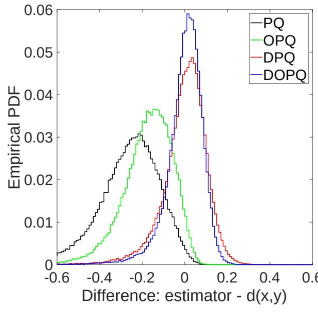


(a) GIST-1M-960D

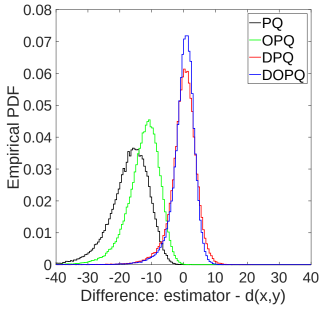


(b) CNN-1M-4096D

Fig. 7. This figure shows the distribution of differences from the ground-truth distances to results estimated by the error corrected version of asymmetric distance estimators (ECAD) on two different datasets. Specifically, PQ and OPQ used Eq. 4), and DPQ and DOPQ used Eq. 7. Our encoding scheme and distance estimator provide lower bias and variance in distance estimation. The values of bias and variance are available in Table. 1. We draw $100K$ pairs of samples randomly chosen from each dataset. All the tested methods utilized 64 bit compact codes. PQ and OPQ used $M = 8$ and $N_K = 256$ while DPQ and DOPQ used $M = 8$, $N'_K = 128$, and $N_D = 2$. (Best viewed in color).



(a) GIST-1M-960D



(b) CNN-1M-4096D

Fig. 8. The same protocol with Fig. 7 but different distance estimators are investigated. PQ/OPQ used the asymmetric distance estimator (Eq. 3), and DPQ/DOPQ used our geometry based asymmetric distance estimator (Eq. 12). PQ and OPQ with AD have significantly high biases and variances in distance estimation. In contrast, DPQ and DOPQ with GMAD provide a negligible bias with much smaller variances compuared to PQ or OPQ. Such bias and variance values are available in Table. 1. (Best viewed in color).

$$d_{ECAD}^{DPQ}(x,y)^2 = d_{AD}^{PQ}(x,y)^2 + \sum_{i=1}^{M} \xi_{j,k}^i(x^i). \quad (7)$$

Interestingly, [5] reported that the error-corrected distance estimators did not improve accuracy over estimator 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. 5) shows higher accuracy over ours without the terms. 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. 7 and Table. 1, the variance and bias of our error-corrected distance metric are reduced from those of PQ and OPQ. Since our encoding scheme quantizes the distance of a data point from its corresponding cluster, our error correcting term (Eq. 5) reduces the bias and, more importantly, the variance of the distance estimates effectively.

Dataset	GIST-1M-960D			
Distance	ECAD		AD/GMAD	
	Bias	Variance	Bias	Variance
PQ	0.0091	0.0143	-0.2707	0.0238
OPQ	0.0107	0.0107	-0.1805	0.0151
DPQ	0.0079	0.0109	-0.0061	0.0120
DOPQ	0.0075	0.0074	-0.0055	0.0085
GDPQ	-	-	0.0062	0.0074
GDOPQ	-	-	0.0037	0.0045

Dataset	CNN-1M-4096D			
Distance	ECAD		AD/GMAD	
	Bias	Variance	Bias	Variance
PQ	0.3265	27.9472	-17.2540	38.5955
OPQ	0.4155	20.3692	-12.8942	27.2969
DPQ	0.2448	15.4019	-0.2670	17.3223
DOPQ	0.2793	11.5191	-0.2474	13.3204
GDPQ	-	-	0.1528	8.4015
GDOPQ	-	-	0.2028	5.7056

TABLE 1
This table shows empirically measured biases and variances of differences between exact distances and results estimated by various distance estimators. Note that, GDPQ and GDOPQ stand for our new compact code representations that encode global residual norms built upon subspaces and codeword vectors computed by PQ and OPQ, respectively (Sec. 5).The empirical distributions regarding to this table are plotted in Fig. 7, Fig. 8, and Fig. 12.

4.2.2 Geometry based distance estimator

We now propose a novel geometric approach to develop a distance estimator 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. 9). Given these geometric configurations, the vector $x - y$ is reformulated as:

$$x - y = (c_x - c_y) + (x - c_x) + (c_y - y). \quad (8)$$

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

As the dimension of a space 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 [30], [31]. 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

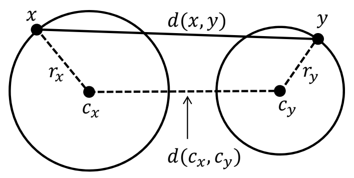


Fig. 9. This figure shows two points x and y on the hyper-spheres centered at c_x and c_y respectively. r_x and r_y represent the radii of hyper-spheres.

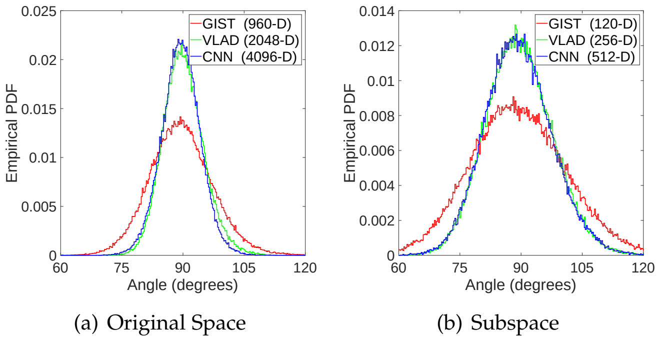


Fig. 10. (a) shows empirical distributions of an angle between two vectors $x - q(x)$ and $y - q(x)$ in three different benchmarks, where $100K$ (x,y) pairs are randomly sampled and $q(\cdot)$ is a vector quantizer. (b) shows the distribution in the subspaces constructed by OPQ with $M = 8, N_K = 256$.

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 three benchmarks consisting of 960 dimensional GIST, 2048 dimensional VLAD, and 4096 dimensional CNN features. For this we have used $100K$ pairs of samples (x_i, y_i) from each dataset and measured the angles between $x_i - q(x_i)$ and $y_i - q(x_i)$. The empirical probability distribution functions are shown in Fig. 10(a). The average angles and standard deviations on GIST, VLAD, and deep descriptors are $(89.5° \pm 7.7°)$, $(89.9° \pm 5.0°)$, and $(89.4° \pm 4.7°)$, respectively. We also have performed the same experiment on the subspace defined by OPQ with $(M = 8, N_K = 256)$ and its results is shown in Fig. 10(b). In subspaces whose dimensionality is 1/8 of the original space, the average angles and standard deviations on GIST, VLAD, and deep descriptors are $(89.7° \pm 11.7°)$, $(89.5° \pm 8.1°)$, and $(89.2° \pm 8.0°)$, respectively. Note that, standard deviations on synthetic points from uniform distributions in 3, 10, 100, and 1000 dimensional spaces are 39.7, 17.5, 5.6, and 1.8, respectively.

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 approximated as follows:

$$\begin{aligned} \| x - y \|^2 &= \| (c_x - c_y) + (x - c_x) + (c_y - y) \|^2 \\ &\approx \|c_x - c_y\|^2 + \|x - c_x\|^2 + \|c_y - y\|^2 \ . \end{aligned} \quad (9)$$

The first term, $\|c_x - c_y\|^2$, is precomputed as the distance between different cluster centers. The second and third

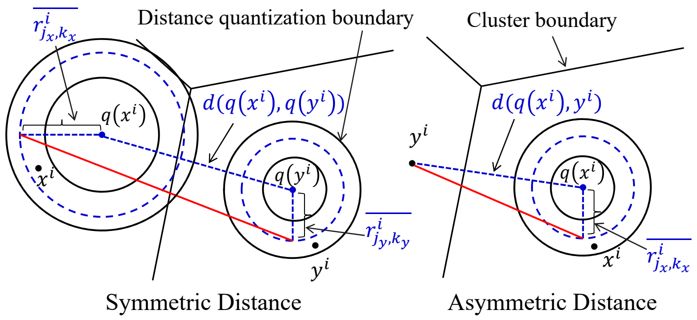


Fig. 11. This figure illustrates our Geometry-based Symmetric and Asymmetric distances (GMSD and GMAD).

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 $p_w \in P^i_{j_x,k_x}$:

$$\overline{r^i_{j_x,k_x}} = \frac{1}{|P^i_{j_x,k_x}|} \sum_{w=1}^{|P^i_{j_x,k_x}|} d(p_w, c^i_{j_x}). \quad (10)$$

The third term of Eq. 9 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 estimator based on Eq. 9 and Eq. 10. Our GeoMetry based squared Symmetric Distance (GMSD) between two points x and y is defined as:

$$d^{DPQ}_{GMSD}(x,y)^2 = \sum_{i=1}^{M} \Big(d\big(q(x^i), q(y^i)\big)^2 + \overline{r^i_{j_x,k_x}}^2 + \overline{r^i_{j_y,k_y}}^2 \Big). \quad (11)$$

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

$$d^{DPQ}_{GMAD}(x,y)^2 = \sum_{i=1}^{M} \Big(d\big(q(x^i), y\big)^2 + \overline{r^i_{j_x,k_x}}^2 \Big). \quad (12)$$

Note that $\overline{r^i_{j,k}}$ is precomputed and stored as a lookup table as a function of i, j, and k values. Fig. 11 illustrates our GMSD and GMAD.

One may find that our geometry based distance estimator using the average distance (Eq. 10) of points from their cluster have a similar form to our statistics based distance estimator using the error correcting term (Eq. 5). In order to compare two distance estimators, let us first denote the distance between p_w which is w^{th} data point of P^i_j and its corresponding cluster center c^i_j, $d(p_w, c^i_j)$, as a random variable R. The error correcting term (Eq. 5) can be expressed as:

$$\xi^i_{j,k}(x^i) = \frac{1}{|P^i_{j,k}|} \sum_{w=1}^{|P^i_{j,k}|} d(p_w, c^i_j)^2 = \mathrm{E}(R^2). \quad (13)$$

Code Length	16	32	64	128	256
DPQ (ECSD)	0.0518	0.0825	0.1258	0.2112	0.3245
DPQ (GMSD)	0.0551	0.0839	0.1296	0.2166	0.3440
DPQ (ECAD)	0.0795	0.1264	0.2003	0.3354	0.4825
DPQ (GMAD)	0.0817	0.1295	0.2080	0.3448	0.5011

TABLE 2
This table shows the mAPs of DPQ with different distance estimators. Our geometry based distance estimators, DPQ with GMSD/AD, improves our statistics based estimator, DPQ with ECSD/AD. The results are obtained with GIST-1M-960D dataset and the number of ground truth K=1000.

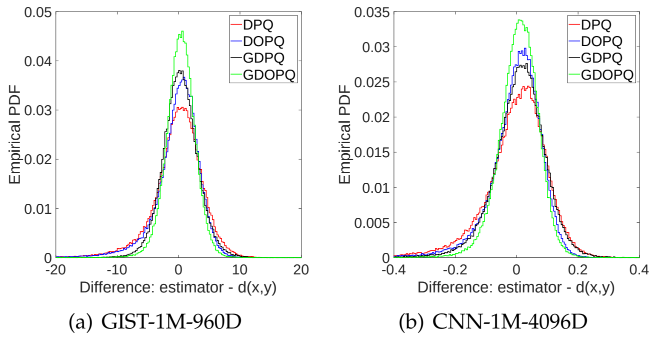


Fig. 12. This figure shows the empirical distribution of differences from the exact distances to results of the geometry based distance estimator; graphs are best viewed in colors. GDPQ and GDOPQ provide lower bias and variances compared to DPQ and DOPQ, respectively. All tested methods used 64-bit compact codes. Specifically, DPQ and DOPQ used the same parameters used in Fig. 8, and GDPQ and GDOPQ used $M = 8$, $N''_K = 128$, and $N^G_D = 256$. The values of biases and variances are given in Table. 1.

On the other hand, the squared average distance (Eq. 10) is written as the follows:

$$\overline{r^i_{j_x,k_x}}^2 = \left(\frac{1}{|P^i_{j_x,k_x}|} \sum_{w=1}^{|P^i_{j_x,k_x}|} d(p_w, c^i_{j_x}) \right)^2 = \{\mathrm{E}(R)\}^2. \quad (14)$$

The difference between our statistics and geometry based distance estimator is $\mathrm{Var}(R)$, since $\mathrm{Var}(R) = \mathrm{E}(R^2) - \{\mathrm{E}(R)\}^2$. It is hard to theoretically tell which approach is better, but these two different estimators consider different aspects of input data points; the statistics based estimator considers the variance of distances, while the geometry based one does not.

Empirically we, however, have observed that the geometry based estimator shows better performance, 3.5%, on average over our statistics based estimator (Table. 2). We, thus, consistently use the geometry based distance estimators in the following evaluations in Sec. 6.

Let us analyze time complexities of distance estimation. For symmetric distances, all the aforementioned distance estimators have the same time complexity of $O(M)$ needed to refer the lookup table. In case of asymmetric distance estimators, PQ and DPQ have time complexities of $O(D)$ and $O(D + M)$, respectively. Since the dimensionality of data D is much larger than the number of subspaces M (i.e. $D \gg M$), such additional computation is negligible. We also need to construct a lookup table for asymmetric distance estimation when estimating distances from a query to many data points. In such case, PQ and DPQ requires the complexity of $O(DN_K)$ and $O(DN'_K + N'_K N_D)$, respectively. DPQ has a lower computational overhead in constructing a lookup table for asymmetric distance computation since we typically set $N'_K N_D = K_N$ and $2N'_K \leq N_K$.

5 Global Distance Encoded Product Quantization

Even though DPQ (and DOPQ) utilizing the residual distance information outperforms the accuracy of PQ (and OPQ), there is still a room for improvement. We introduce the following two cues to further extend DPQ.

First, since we quantize the distance from the nearest centroid (i.e. residual norm) and use it for distance estimation in each subspace, the error of estimated distances based on such residual norm quantization is accumulated over the subspaces. This accumulated error produces inaccuracy in search results.

Second, the orthogonality assumption on our geometry based distance estimation better holds in the original space compared to the subspace since the dimensionality of the original space is M times higher than one of the subspace. We experimentally validated this claim regarding to angles (Fig. 10 and Sec. 4.2).

Based on the aforementioned motivations, we introduce Global Distance encoded Product Quantization (GDPQ), which encodes the residual distances in the original data space instead of subspaces. Given a product quantizer $q(x)$, we have an approximated reconstruction $q(x)$ of encoded x by concatenating the corresponding codeword vectors:

$$q(x) = [\ q^1(x^1)\ \ q^2(x^2)\ \ \ldots\ \ q^M(x^M)\ ]. \quad (15)$$

GDPQ encodes the distance between a data point x and its reconstruction, $d(x, q(x))$, to L'_D bits. The global residual norm, $d(x, q(x))$, is quantized with $N^G_D = 2^{L'_D}$ different distance ranges, $t^G_1, \ldots, t^G_{N^G_D}$. Such residual distance is encoded by an index $\hat{k}$ that satisfies the following:

$$t^G_{\hat{k}-1} \leq d(x, q(x)) < t^G_{\hat{k}},$$

where $t^G_1 = 0$ and $t^G_{N^G_D} = \infty$. In order to compute distance thresholds, we follow the procedure used for DPQ described in Sec. 4.1.

A compact code of a data point x encoded by GDPQ is a concatenation of cluster indices $b^1, \ldots, b^M$ over subspaces (Eq. 1) computed by a product quantizer and the quantized global residual distance index $\hat{k}$ as follows:

$$b^{GDPQ} = [\ b^1\ b^2\ \ldots\ b^M\ B(\hat{k}, L'_D)\ ].$$

We define a representative distance $\overline{r_i}$ for a residual norm range $[t^G_i, t^G_{i+1})$ by computing the average value within the range:

$$\overline{r^i} = \frac{1}{|S_i|} \sum_{s \in S_i} d(s, q(s)), \quad (16)$$

where

$$S_i = \{x | t^G_i \leq d(x, q(x)) < t^G_{i+1} \text{ and } x \in X\}. \quad (17)$$

The symmetric and asymmetric distance estimators for GDPQ are then defined by extended from our geometry based estimators:

$$d_{SD}^{GDPQ}(x,y)^2 = \sum_{i=1}^{M} d\big(q(x^i), q(y^i)\big)^2 + \overline{r^{i_x}}^2 + \overline{r^{i_y}}^2, \quad (18)$$

$$d_{AD}^{GDPQ}(x,y)^2 = \sum_{i=1}^{M} d\big(q(x^i), y\big)^2 + \overline{r^{i_x}}^2, \quad (19)$$

where i_x and i_y denote the global residual norm index of x and y, respectively.

We have empirically studied the error of estimated distances produced by GDPQ and GDOPQ. As shown in Fig. 12 and Table. 1, encoding global residual norms and utilizing in distance estimation significantly reduced both biases and variances of errors of the estimated distances. GDOPQ, especially, provided the lowest variance and bias among all the tested methods. Note that, our statistics based distance estimator is not able to be extended for GDPQ, since it is impractical to use the error correcting terms in the original data space. Specifically, we need to precompute and store ${N''_K}^M N_D^G = 2^L$ different error correcting terms.

The time complexity of GDPQ to encode a data in D-dimensional space is $O(M\frac{D}{M}N''_K + N_D^G) = O(DN''_K + N_D^G)$. For 64 bits encoding, we suggest to use $M = 8$, $N''_K = 128$, and $N_D^G = 256$ (Table. 3). In this setting, GDPQ requires $O(128D + 256)$ time which is about two times smaller than PQ whose complexity is $O(256D)$. A distance estimation via precomputed lookup table has $O(M + 1)$ time complexity which requires $O(1)$ additional but negligible overhead compared to PQ. On the other hands, computing a lookup table for asymmetric distance estimation for a particular query takes $O(DN''_K)$ time which is twice faster than PQ in typical settings of 64 bits encoding.

6 Evaluation

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

6.1 Protocol

We evaluate our method on the following benchmarks:

- **GIST-1M-960D**: One million of 960-dimensional GIST descriptors [5].
- **VLAD-1M-2048D** and **VLAD-1M-4096D**: One million of 2048 and 4096-dimensional VLAD descriptors [32]. In order to compute 2048D and 4096D VLAD descriptors, we have aggregated 128-dimensional SIFT features [33] with 16 and 32 codeword vectors, respectively.
- **CNN-1M-4096D**: One million of 4096-dimensional deep features extracted from the last fully connected layer (fc7) in the CNN [34].

For all the experiments, we use 1000 queries that do not have any overlap with the data points. Tested methods are evaluated on accuracy of K-nearest neighbor search when K = 100 and K = 1000. The ground truth for each query is computed by performing the linear scan. The performance is measured by the mean Average Precision (mAP). To verify benefits of our method we compare the following methods:

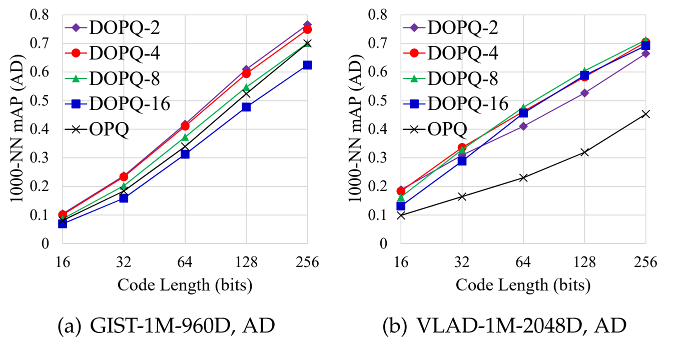


Fig. 13. Experimental results with respect to the number of distance quantizations (N_D) for DOPQ. The postfix numbers in the method names indicate the number of distance quantizations N_D in each subspaces.

- **PQ**: Product Quantization [32]
- **OPQ**: Optimized Product Quantization [6], [29].
- **RQ**: Residual Quantization [22].
- **DPQ**: Our method that encodes residual distances in subspaces (Sec. 4) and using the same subspaces and clustering method as used for **PQ**.
- **DOPQ**: Similar to **DPQ** but using the subspaces and clusters optimized by **OPQ**.
- **GDPQ**: Our method that encodes global residual distances in the original space (Sec. 5) and using subspaces and codeword vectors optimized by **PQ**.
- **GDOPQ**: Similar to **GDPQ** but using subspaces and codeword vectors optimized by **OPQ**.

All the methods are implemented in the C++ (Intel Compiler and MKL library are used). The experiments are conducted on a machine consisting of 2 Xeon E5-2698 processors and 256GB main memory. When measuring times we use a single thread and the sequential version of MKL.

For all the methods, $100K$ data points randomly sampled from the dataset are used in the training stage, and we allow 100 iterations for each k-means clustering. Note that, in order to train OPQ we perform 50 iterations for parametric version and remaining 50 iterations for non-parametric version as suggested in [6], [29].

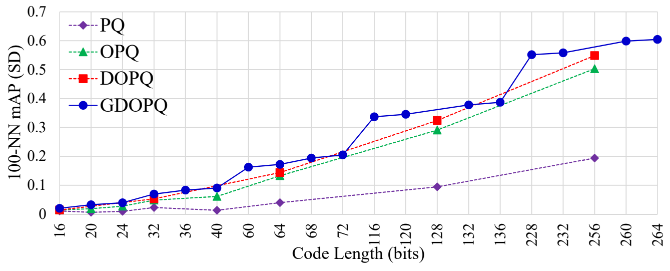
For PQ and OPQ we use symmetric and asymmetric distances (Eq. 2 and Eq. 3), which achieved the best accuracy according to [5], [6]. On the other hand, DPQ and DOPQ use our geometry based distance estimators (Eq. 11 and Eq. 12). GDPQ and GDOPQ also use geometry based estimator but defined in the original feature space (Eq. 18 and Eq. 19).

We assign 8 bits for each subspace for PQ, OPQ, DPQ, and DOPQ for the code lengths of 16, 32, 64, 128, and 256. In this configuration PQ and OPQ then have 256 clusters in each subspace as suggested in [5], [6], [29], and the number of subspaces M is $L/8$, where L is the tested code length. Similarly, we use 8 bits for each quantization step for RQ to have $N_K = 256$ codewords, and the number of quantization steps is set to $M(= L/8)$ as did in [22]. RQ additionally requires memory footprint to store L_2 norms of the data which is necessary for the distance estimation. We allocate

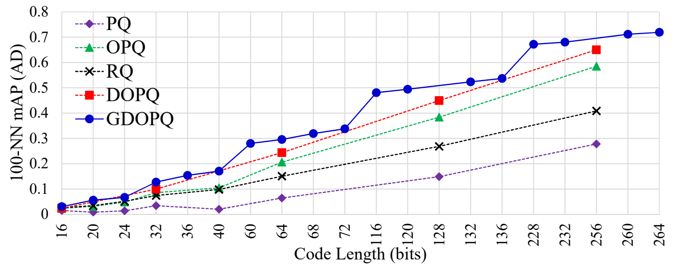
CodeLen	16	20	24	32	36	40	60	64	68	72
(O)PQ, RQ	2, 256	4, 32	4, 64	4, 256	-	8, 32	-	8, 256	-	-
D(O)PQ	2, 128, 2	-	-	4, 128, 2	-	-	-	8, 128, 2	-	-
GD(O)PQ	2, 64, 16	2, 256, 16	2, 256, 256	4, 128, 16	4, 256, 16	4, 256, 256	8, 128, 16	8, 128, 256	8, 256, 16	8, 256, 256

CodeLen	116	120	128	132	136	228	232	256	260	264
(O)PQ, RQ	-	-	16, 256	-	-	-	-	32, 256	-	-
D(O)PQ	-	-	16, 128, 2	-	-	-	-	32, 128, 2	-	-
GD(O)PQ	16, 128, 16	16, 128, 256	-	16, 256, 16	16, 256, 256	32, 128, 16	32, 128, 256	-	32, 256, 16	32, 256, 256

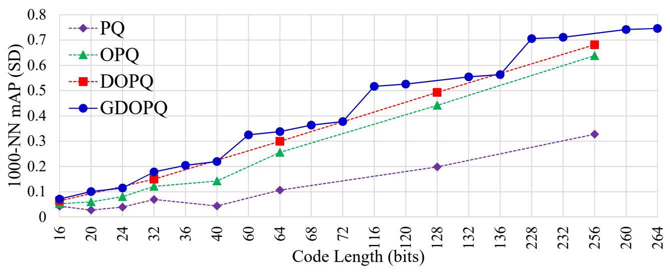
TABLE 3
Parameters used in the experiments. Each pair of numbers for (O)PQ denotes (M, N_K). In case of RQ, each pair is the number of quantization steps and the number of codewords in each step. Each tuple for D(O)PQ and GD(O)PQ represents (M, N'_K, N_D) and (M, N''_K, N^G_D), respectively.



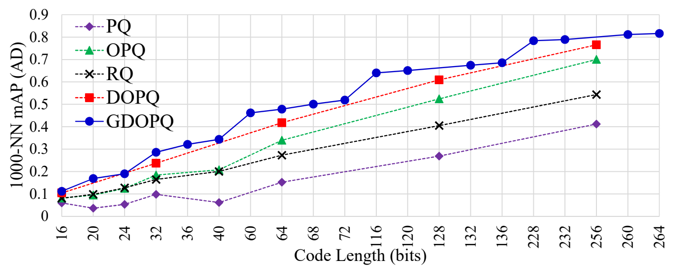
(a) GIST-1M-960D, K=100, Symmetric Distance



(b) GIST-1M-960D, K=100, Asymmetric Distance



(c) GIST-1M-960D, K=1000, Symmetric Distance



(d) GIST-1M-960D, K=1000, Asymmetric Distance

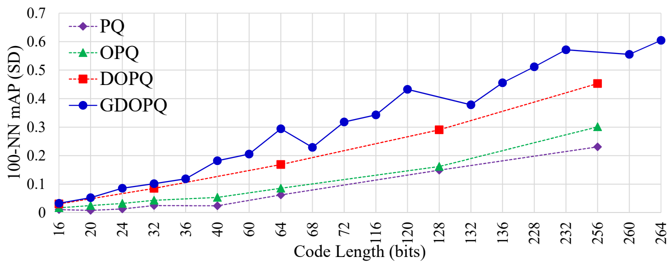
Fig. 14. Comparisons on GIST-1M-960D

32 bits for RQ to store such norms, so RQ utilizes $L + 32$ bits in total. However, we did not include the additional bits that RQ uses in the experimental results.

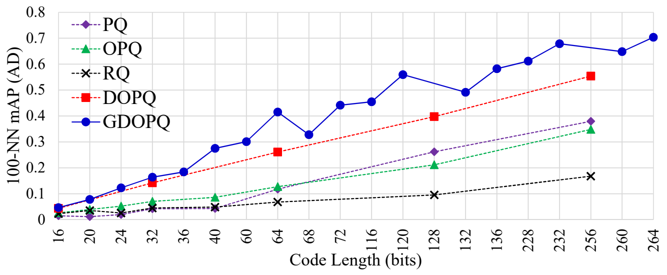
In DPQ and DOPQ, we have the parameter L_D (the number of bits) for the distance quantization in each subspace. We empirically studied about this parameter and the experimental results are given in Fig. 13. We observe that $L_D = 1$ ($N_D = 2$) or $L_D = 2$ ($N_D = 4$) gives reasonable performance on GIST-1M-960D benchmark. On the other hand, on VLAD-1M-2048D dataset the results are insensitive to choice of the parameter L_D. We, however, pick $L_D = 1$ for all the following tests for the consistency. In other words, we use $N'_K = 128$ centers in each subspaces and $N_D = 2$ distance quantizations for each cluster. Since we have higher degree of freedom in code length selection with GDPQ and GDOPQ, we test GDPQ and GDOPQ with varying code lengths. For instance, GDPQ and GDOPQ are able to define 60 bits compact code by utilizing $M = 8$ subspaces, $N''_K = 128$ centers in each subspace, and $N'_D = 16$ global residual distance quantizations. The parameters used in the experiments are available in Table. 3. Note that we also additionally report the results of PQ and OPQ with code lengths of 20, 24, and 40 by adjusting M and N_K to compare with GDPQ and GDOPQ. We, however, observe that PQ and OPQ are most effective among tested configurations when $N_K = 256$ which is standard setting used in [5], [6], [29].

6.2 Results

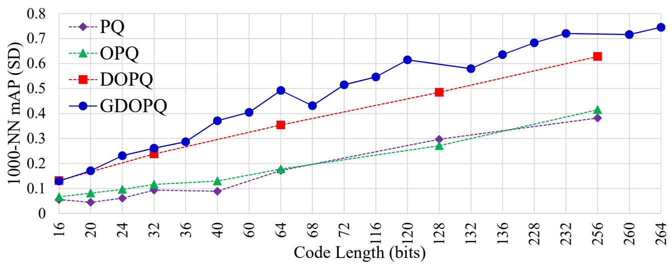
Fig. 14, Fig. 15, Fig. 16, and Fig. 17 show mAP curves of K-nearest neighbor search for the tested methods PQ, OPQ, DOPQ, and GDOPQ on GIST-1M-960D, VLAD-1M-2048D, VLAD-1M-4096D and CNN-1M-4096D datasets, respectively. DOPQ consistently shows better results over OPQ across the code lengths 16, 32, 64, 128, and 256 bits in all the tested benchmarks. Specifically when we use 64 bits compact codes and asymmetric distances, DOPQ provides 18%, 104%, 84%, and 16% higher mAPs than OPQ in 100-nearest neighbor search on GIST-1M-960D, VLAD-1M-2048D, VLAD-1M-4096D and CNN-1M-4096D, respectively. Similarly such improvements by DOPQ over OPQ are 23%, 100%, 295%, and 34% in 1000-nearest neighbor search. GDOPQ shows the highest performance among tested methods. Specifically GDOPQ provides 56%, 120%, 173%, and 59% higher mAPs over OPQ with 32 bits compact



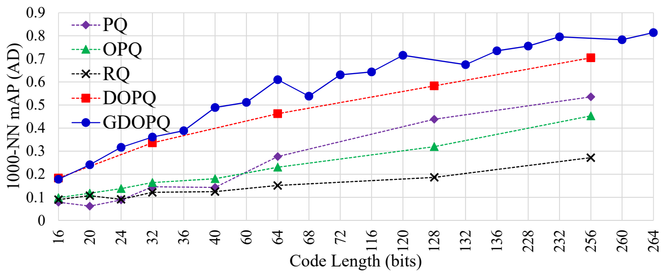
(a) VLAD-1M-2048D, K=100, Symmetric Distance



(b) VLAD-1M-2048D, K=100, Asymmetric Distance

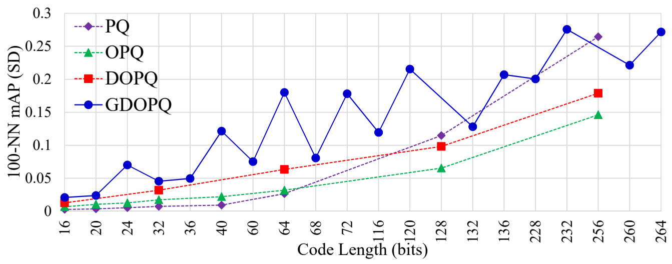


(c) VLAD-1M-2048D, K=1000, Symmetric Distance

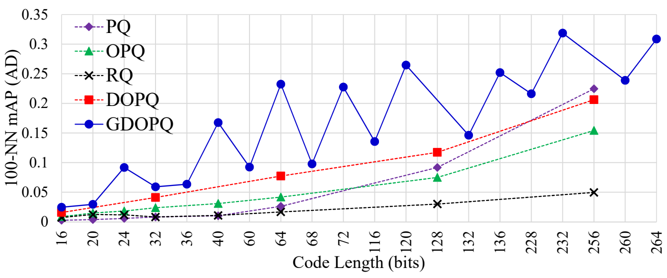


(d) VLAD-1M-2048D, K=1000, Asymmetric Distance

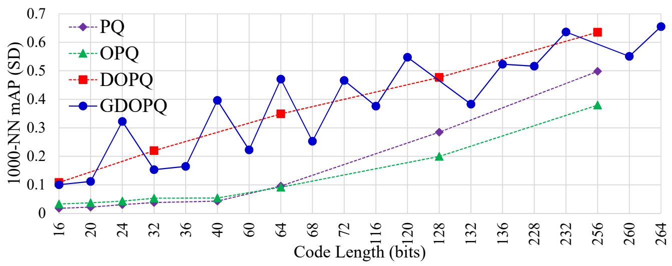
Fig. 15. Comparisons on VLAD-1M-2048D



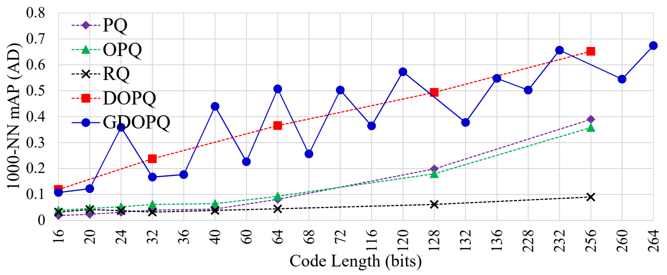
(a) VLAD-1M-4096D, K=100, Symmetric Distance



(b) VLAD-1M-4096D, K=100, Asymmetric Distance



(c) VLAD-1M-4096D, K=1000, Symmetric Distance



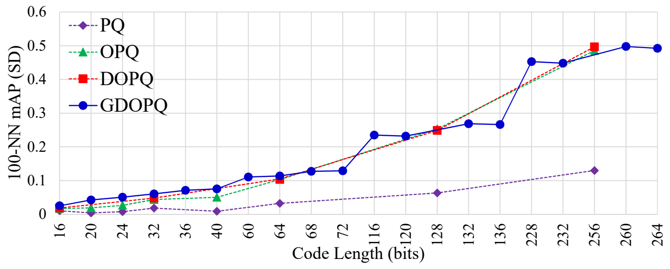
(d) VLAD-1M-4096D, K=1000, Asymmetric Distance

Fig. 16. Comparisons on VLAD-1M-4096D

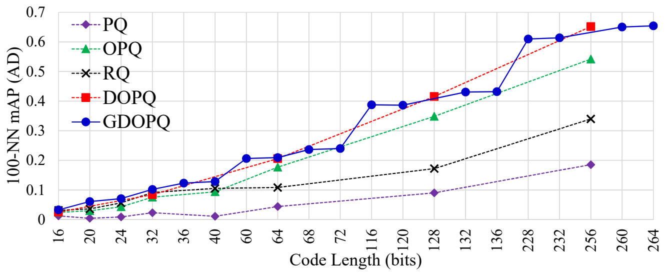
codes. Moreover, GDOPQ shows 15%, 32%, 39%, and 7% higher accuracy compared to DOPQ with 64 bits codes on four different benchmarks. The experimental results confirm that encoding the residual distances, especially in the original feature space, reduces the errors of distance estimation and thus significantly improves the search accuracy. The recall curves for 64 bits encoding are given in Fig. 18.

Given a target accuracy, GDOPQ requires the shortest code lengths over all the other tested methods. For instance, GDOPQ requires 68 and 64 bits to achieve the mAP of 0.5 in 1000-nearest neighbor search on GIST-1M-960D and CNN-1M-4096D respectively, while OPQ needs 128 bits on both benchmarks. Moreover, GDOPQ achieves the mAP of 0.5 with 40 and 64 bits compact codes on VLAD-1M-2048D and VLAD-1M-4096D benchmark, respectively

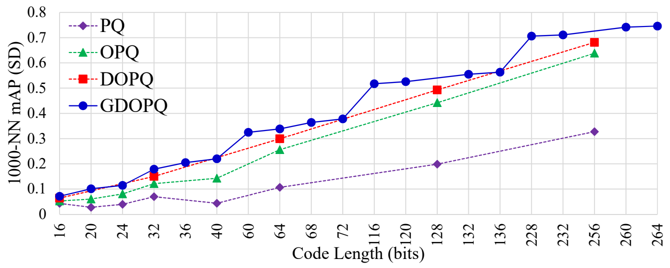
While our proposed methods significantly improve the performance of OPQ on VLAD-1M-2048D, VLAD-1M-4096D, and GIST-1M-960D, the improvements on CNN-1M-4096D are slightly lower than ones we achieved on the other benchmarks. Since the CNN features used in the experiments are extracted from a neural network trained for image classification tasks [34], the features are already



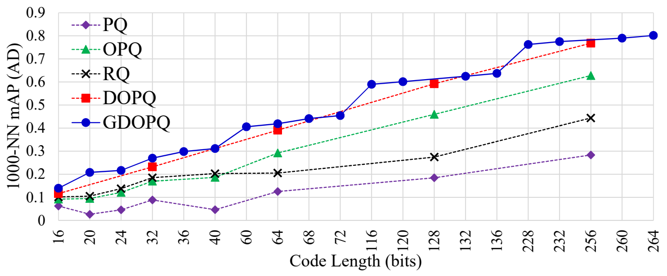
(a) CNN-1M-4096D, K=100, Symmetric Distance



(b) CNN-1M-4096D, K=100, Asymmetric Distance

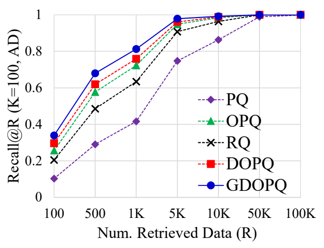


(c) CNN-1M-4096D, K=1000, Symmetric Distance

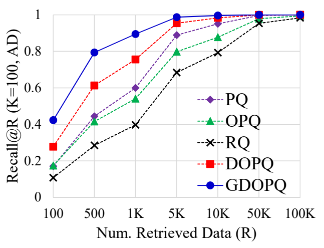


(d) CNN-1M-4096D, K=1000, Asymmetric Distance

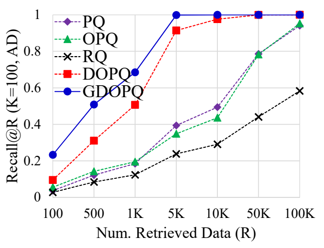
Fig. 17. Comparison on CNN-1M-4096D



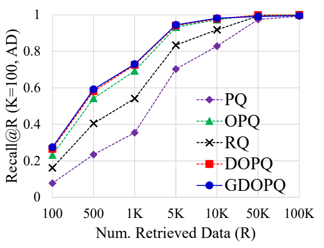
(a) GIST-1M-960D, K=100, AD



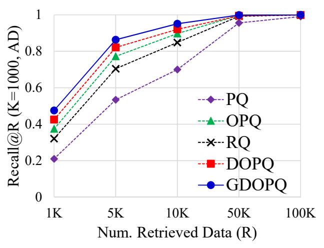
(b) VLAD-1M-2048D, K=100, AD



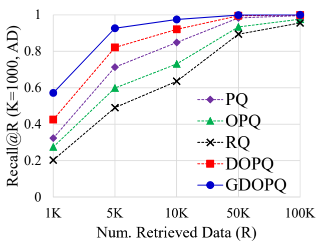
(c) VLAD-1M-4096D, K=100, AD



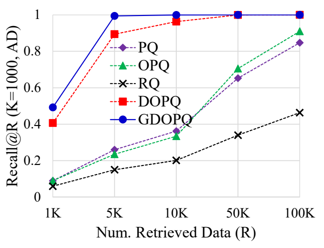
(d) CNN-1M-4096D, K=100, AD



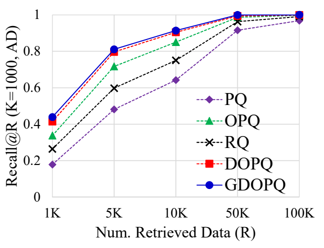
(e) GIST-1M-960D, K=1000, AD



(f) VLAD-1M-2048D, K=1000, AD



(g) VLAD-1M-4096D, K=1000, AD



(h) CNN-1M-4096D, K=1000, AD

Fig. 18. Recall curves with 64 bits compact codes.

well-clustered than GIST or VLAD vectors. We observe that using the residual distances is less effective in such data distribution compared to unstructured data. The performance improvements on CNN features are, however, still noticeable. Moreover, our method interestingly achieve higher improvements on CNN features when we retrieve more neighbors (i.e. larger K in K-nearest neighbor search). Specifically the accuracy improvements by using DOPQ and GDOPQ over OPQ on CNN-1M-4096D benchmark are 7% and 34% respectively when K = 100. These rates are increased to 30% and 56% as we increase the number of neighbors K to 1000.

In contrast to the case of deep features which are semantic and well-clustered, the residual distance quantization achieves significant accuracy improvement on VLAD vectors which represent more visual information and are thus unstructured. This can be validated by investigating the results of GDOPQ at code lengths of 60, 64, and 68 bits. According to parameters given in Table. 3, (M, N_K'', N_D^G) tuples for 60, 64, and 68 bits are $(8, 128, 16)$, $(8, 128, 256)$, and $(8, 256, 16)$, respectively. As shown in the Fig. 15 and 16 allocating 4 more bits for distance quantization (i.e. 60 $\rightarrow$ 64 bits) achieves a significant improvement of the search accuracy, however, using twice more centers by adding 8 bits (i.e. 60 $\rightarrow$ 68 bits) provides a marginal improvement. Moreover, the mAP at 64 bits is even higher than one at 68

	Encoding	SD	AD w/o T	AD w/ T
PQ	60.2 s	19.4 ms	391 ms	12.3 ms
RQ	372.8 s	-	1307 ms	14.8 ms
DPQ	31.2 s	19.5 ms	405 ms	12.2 ms
GDPQ	31.6 s	19.6 ms	382 ms	12.4 ms

TABLE 4
This table shows computational times for encoding 1M (million) data, 1M symmetric distance estimations, 1M asymmetric distance estimations with/without utilizing lookup tables. The timings are measured on GIST-1M-960D at 64 bits encoding. Note that, OPQ/DOPQ/GDOPQ are additionally required to perform orthogonal projection in common. The projection of 1M data takes about 28 and 2 seconds by using sequential and parallel version of MKL library.

bits. An opposite trend of results can be observed on CNN-1M-4096D benchmark. As a result, the residual distance quantization and distance estimation utilizing those information provide relatively higher accuracy improvement in unstructured data compared to clustered data. Furthermore, given a same underlying structure of the data but different dimensionality (VLAD-1M-2048D and VLAD-1M-4096D), having more distance quantizations significantly improves the search accuracy in higher dimensional data where the orthogonality assumption better holds.

We report experimental results of DPQ in Fig. 19. For symmetric distance estimation, we compare DPQ against PQ and some of well-known binary code embedding techniques (i.e. hashing methods), Iterative Quantization (ITQ) [17] and Spherical Hashing (SpH) [19] with its distance metric, SHD (Fig. 19(a)). Since the results of those hashing techniques are lower than PQ, we do not show them in other tests. In addition, we report the evaluation of DPQ with asymmetric distance against PQ on three benchmarks in Fig. 19(b). The improvement achieved by DPQ over PQ clearly demonstrates the merits of encoding residual distances and distance estimators utilizing them.

We also compare DPQ and GDPQ. The experimental results are shown in Fig. 20. Overall, GDPQ consistently shows higher accuracy compared to DPQ. For instance, the mAP achieved by GDPQ at 40 bits is even higher than mAP of DPQ with 64 bit codes in VLAD-1M-2048D benchmark. Similarly GDPQ reaches the mAP of 0.2 only with 40 bits in CNN-1M-4096D while DPQ requires 128 bits for the same accuracy.

Finally, we report the computational times with GIST-1M-960D dataset at 64 bits encoding. As reported in Table. 4, DPQ and GDPQ provide about two times faster encoding over PQ while those methods have insignificant speed difference in distance estimations. Since DPQ and GDPQ utilize smaller number of clusters than PQ at the same code length, they can provide faster encoding performance. Note that, the methods using subspaces and clusters optimized by OPQ, OPQ/DOPQ/GDOPQ, require orthogonal projection which takes about 28 and 2 seconds for 1 million data by using sequential and parallel versions of MKL library, respectively. RQ shows the slowest performance in both encoding and distance estimation, since it works on the original space but not the subspaces. RQ, specifically, has $O(MDN_K)$ time complexity to encode a data point which requires $O(M)$ times longer computational time compared to PQ. As discussed in Sec. 4.1 and 5, DPQ and GDPQ have even lower complexity than PQ.

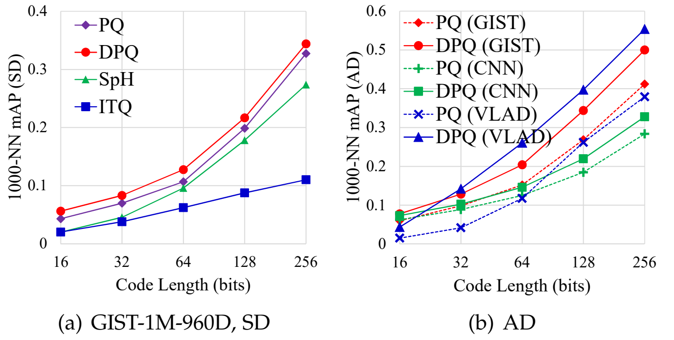


Fig. 19. (a) shows experimental results on GIST-1M-960D. We compare DPQ against PQ and two hashing methods SpH [19] with its distance estimation SHD, and ITQ [17] with the Hamming distance. Since hashing techniques assume that both query and data are encoded, we use symmetric distance estimators for PQ and DPQ. (b) shows comparison between PQ and DPQ on three benchmarks with asymmetric distance estimators. Note that, the VLAD in this figure indicates VLAD-1M-2048D.

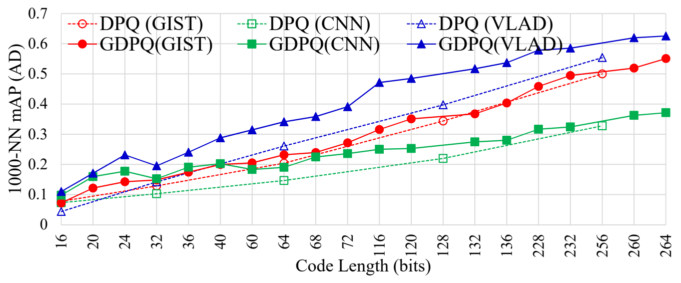


Fig. 20. This figure shows comparison of DPQ and GDPQ on three different benchmarks when $K = 1000$ and asymmetric distance estimators are used. Note that, the VLAD in this figure indicates VLAD-1M-2048D.

7 Conclusion

We have presented a novel compact code representation scheme DPQ 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 further extend DPQ to GDPQ which encodes global residual distances in the original feature space instead of subspaces. Since our encoding scheme is independent from the subspace and codeword vector construction algorithms, our method can be applied to any vector quantization techniques and we believe that our method can provide additional accuracy improvements.

We have applied our encoding schemes to PQ and OPQ, and extensively evaluated against those baseline techniques on popular high-dimensional image descriptors. We demonstrated that our proposed method significantly and consistently improves the baseline methods. These results confirm the merits of our method encoding residual distances and using them in distance estimations.

ACKNOWLEDGEMENTS

The authors would like to think anonymous reviewers for constructive comments. This work was supported in part by Basic Science Research Program (NRF-2017R1D1A1B03036330), SW Starlab (IITP-2015-0-00199), and Grand Information Technology Research Center support program (IITP-2018-2015-0-00742). This manuscript is extended from the conference paper version [35]. Jae-Pil Heo is the corresponding author of this paper.

REFERENCES

[1] J. H. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Transactions on Mathematical Software*, 1977.

[2] A. Guttman, "R-trees: a dynamic index structure for spatial searching," *ACM SIGMOD Record*, 1984.

[3] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *International Conference on Very Large Data Bases*, 1999.

[4] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, 1991.

[5] H. Jégou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[6] T. Ge, K. He, Q. Ke, and J. Sun, "Optimized product quantization for approximate nearest neighbor search," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[7] M. Norouzi and D. J. Fleet, "Cartesian k-means," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[8] Y. Kalantidis and Y. Avrithis, "Locally optimized product quantization for approximate nearest neighbor search," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[9] Y. Jia, J. Wang, G. Zeng, H. Zha, and X.-S. Hua, "Optimizing kd-trees for scalable visual descriptor indexing," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[10] T. cker Chiueh, "Content-based image indexing," in *International Conference on Very Large Data Bases*, 1994.

[11] D. Nistér and H. Stewénius, "Scalable recognition with a vocabulary tree," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[12] P. Indyk and R. Motwani, "Approximate nearest neighbors: toward removing the curse of dimensionality," in *Symposium on Computational Geometry*, 1998.

[13] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *ACM Symposium on the Theory of Computing*, 2002.

[14] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Symposium on Computational Geometry*, 2004.

[15] O. Chum, J. Philbin, and Z. Zisserman, "Near duplicate image detection: min-hash and tf-idf weighting," in *British Machine Vision Conference*, 2008.

[16] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Conference on Neural Information Processing Systems*, 2008.

[17] Y. Gong and S. Lazebnik, "Iterative quantization: a procrustean approach to learning binary codes," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[18] J. He, R. Radhakrishnan, S.-F. Chang, and C. Bauer, "Compact hashing with joint optimization of search accuracy and time," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[19] J.-P. Heo, Y. Lee, J. He, S.-F. Chang, and S.-E. Yoon, "Spherical hashing: Binary code embedding with hyperspheres," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015.

[20] R. M. Gray and D. L. Neuhoff, "Quantization," *IEEE Transactions on Information Theory*, 1998.

[21] J. Brandt, "Transform coding for fast approximate nearest neighbor search in high dimensions," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[22] Y. Chen, T. Guan, and C. Wang, "Approximate nearest neighbor search by residual vector quantization," *Sensors*, 2010.

[23] L. Ai, J. Yu, Z. Wu, Y. He, and T. Guan, "Optimized residual vector quantization for efficient approximate nearest neighbor search," *Multimedia Systems*, 2015.

[24] H. Jégou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large-scale image search," in *European Conference on Computer Vision*, 2008.

[25] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, 1982.

[26] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: analysis and implementation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002.

[27] P. H. Schönemann, "A generalized solution of the orthogonal procrustes problem," *Psychometrika*, 1966.

[28] A. Banerjee and J. Ghosh, "On scaling up balanced clustering algorithms," in *SIAM International Conference on Data Mining*, 2002.

[29] T. Ge, K. He, Q. Ke, and J. Sun, "Optimized product quantization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.

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

[31] T. Cai, J. Fan, and T. Jiang, "Distributions of angles in random packing on spheres," *Journal of Machine Learning Research*, 2013.

[32] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[33] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, 2004.

[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Conference on Neural Information Processing Systems*, 2012.

[35] J.-P. Heo, Z. Lin, and S.-E. Yoon, "Distance encoded product quantization," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

Jae-Pil Heo is an assistant professor at SKKU (Sungkyunkwan University), South Korea. Before joining SKKU, he was a researcher at ETRI (Electronics and Telecommunications Research Institute). He received his B.S., M.S., and Ph.D. degrees in computer science from KAIST (Korea Advanced Institute of Science and Technology) in 2008, 2010, and 2015, respectively. His research interests include scalable algorithms for high-dimensional data, computer vision, machine learning, and deep learning.

Zhe Lin received the BE degree in automatic control from the University of Science and Technology of China in 2002, the MS degree in electrical engineering from the Korea Advanced Institute of Science and Technology in 2004, and the PhD degree in electrical and computer engineering from the University of Maryland, College Park, in 2009. He has been working at Adobe Systems Inc., San Jose, California since 2009, and is currently a principal scientist at Adobe Research. His research interests include computer vision, image processing, machine learning, deep learning, artificial intelligence. He is a member of the IEEE.

Sung-Eui Yoon is an associate professor at Korea Advanced Institute of Science and Technology (KAIST). He received the B.S. and M.S. degrees in computer science from Seoul National University in 1999 and 2001 respectively. He received his Ph.D. degree in computer science from the University of North Carolina at Chapel Hill in 2005. His main research topic include image search, rendering, and motion planning. He has published more than 50 technical papers in top journals and conference related to his topics. He also gave numerous tutorials on rendering, collision detection, and image search in premier conferences like ACM SIGGRAPH and CVPR. He served as conf. co-chair and paper co-chair for ACM I3D 2012 and 2013 respectively. At 2008, we published a monograph on real-time massive model rendering. Some of his papers received a test-of-time award, and best paper awards.