IJCSE International Journal of Computer Sciences and Engineering Open Access

Survey Paper Vol.-7, Special Issue, 4, Feb 2019 E-ISSN: 2347-2693

K-Subspaces Quantization for Approximate Nearest Neighbour Search

A. Ramya[1*], **S. Sangeetha**[2]

[1,2]M.Sc Computer Science, Idhaya College for Women, Kumbakonam, Tamilnadu, India

Corresponding Author: ramya89priya@gmail.com

Available online at: www.ijcseonline.org

Abstract**—**Approximate Nearest Neighbour (ANN) search has become a popular approach for performing fast and efficient retrieval on very large-scale datasets in recent years, as the size and dimension of data grow continuously. In this paper, we propose a novel vector quantization method for ANN search which enables faster and more accurate retrieval on publicly available datasets. We define vector quantization as a multiple affine subspace learning problem and explore the quantization centroids on multiple affine subspaces. We propose an iterative approach to minimize the quantization error in order to create a novel quantization scheme, which outperforms the state-of-the-art algorithms. The computational cost of our method is also comparable to that of the competing methods.

Keywords**—**Approximate Nearest Neighbour Search, Binary Codes, Large-Scale Retrieval, Subspace Clustering, Vector Quantization.

I. INTRODUCTION

The Nearest Neighbor (NN) search aims to find a sample in a given dataset that is closest to a given query, which is called the nearest neighbor. It is widely used in different areas of signal processing such as information retrieval, computer vision, machine learning, pattern recognition and recommendation systems. However, the traditional NN search is not tractable for today’s very large-scale datasets. Both the search on the dataset and the distance calculation between sample pairs are computationally costly, considering the number of samples and the dimension of the feature space. In order to overcome these limitations of NN search and make it feasible for large-scale problems, Approximate Nearest Neighbor (ANN) search has been proposed [1]. ANN search uses com-pact representations in order to approximate pair-wise dis-tances between pairs of data points. It has proved to be a via-ble alternative and has so far achieved promising results [2].

Many of the existing algorithms in this field rely on the concept of “hashing”. Hashing methods aim to create binary strings from sample vectors and compare those strings using the Hamming distance representing the proximity neighborhood or some given similarity [3]–[7]. The binary string com-parison has also evolved from the simple Hamming distance to asymmetrical distance measures [8], [9] and the usage of look-up tables has enabled more accurate approximations, di-recting the research on ANN search towards vector quantiza-tion [2]. The focus of this paper is on vector quantization based approaches, and the reader is referred to [2] for a more detailed review on hashing.

The idea of quantization goes back to 1980’s. Lloyd de-fined the concept of “good quantization” [10], which is closely related to the K-Means algorithm [11]. Yet Lloyd’s quantization method, or K-Means is not directly applicable to large-scale data, for very large number of centroids. For example, considering a quantization using a binary string of 64-bits, the desired number of centroids is 264. Obviously, it is neither possible to find nor to store such amount of data.

A great improvement on Lloyd's approach for quantization has been proposed by Jegou *et al.* [12] for ANN. In their method called Product Quantization (PQ), the authors divide the sample vector into subvectors and quantize each of them independently using subquantizers. This makes the quantization codebook a Cartesian product, where each centroid in this codebook is represented as a concatenation of the correspond-ing centroids from the subcodebooks. Therefore, for a small number of subquantizers, while each of them having a feasi-ble number of centroids, obtaining the desired total number of centroids is made possible. Referring to the example above, thanks to the Cartesian product, selecting the number of subquantizers as 8 and the number of centroids for each subquantizer as 256 would be enough to reach 264. This ap-proach however suffers from the statistical dependency of subvectors, since they are quantized independently.

Another approach for efficient coding on high dimensional vectors has been proposed by Jegou *et al.* [6] and later by

Gordo *et al*. [8]. In both papers, the data is decorrelated by applying the Principal Component Analysis (PCA), and its di-mension is reduced to a desired value. Then, quantization is applied independently on each of the remaining principal components. This, with a Gaussian distribution assumption, solves the statistical dependency problem among the dimensions. As a result of PCA, the principal components are ranked in a decreasing order according to the corresponding variances, yet each component is quantized by two centroids only. Gong *et al.* [13] propose a two-step method called Iterative Quanti-zation (ITQ). In the first step, a PCA transformation is applied for dimension reduction as in [6], while in the second step, an orthogonal rotation on the data is applied iteratively for a bal-anced distribution of the variance among the principal com-ponents. The data are quantized on the principal components of the rotated space independently. While orthogonal rota-tions preserve the Euclidean distance between pairs of sam-ples, here again, the problem of statistical dependency reap-pears, since the dimensions are no longer decorrelated.

Brandt in [14] proposes a method called Transform Coding (TC), to balance the variance corresponding to each code sep-arately after PCA. TC is a special case of PQ, where each di-mension itself is a subvector. However, in TC, each dimen-sion is allocated a variable number of bits, and a scalar quan-tization is performed on each principal component inde-pendently. Following the rotation idea in [13], Optimized Product Quantization [15] (OPQ) and Cartesian K-Means (CKM) [16] both produce an improvement over PQ by apply-ing an iterative optimization process in order to balance the dimension variances. In [17] Heo *et al.* improve OPQ by en-coding the distances to centroids separately in their algorithm called Distance Encoded Product Quantization (DEPQ). Lo-cally Optimized Quantization (LOPQ) [18] introduces local optimization before OPQ and further improves the perfor-mance.

Recently summation based multi-stage vector quantization methods such as Optimized Cartesian K-Means (OCK) [19], Additive Quantization (AQ) [20], Composite Quantization (CQ) [21] and (Optimized) Tree Quantization (OTQ) [22] which aim to use the summation of several dictionary items to represent the approximation of a vector, have been proposed. These methods produce the state-of-the-art results although the theory behind such quantization methods has been well studied in the past [23].

II. METHODOLOGY

Many of the proposed methods so far transform or project the data into a new (sub)space, where vector dimensions are reduced, reordered or rotated using PCA. Decorrelating the data using a single PCA step may not bring the desired statistical independency among dimensions, especially if the data do not follow a Gaussian distribution, which is the core inher-ent assumption of PCA. A better transformation however, may be designed by representing the data with more than one subspace. Local-PCA [24], [25], K-Means Projective Cluster-ing [26] and Bayesian PCA [27] all propose different solu-tions to this problem based on PCA. In our recent study enti-tled M-PCA Binary Embedding (MPCA-E) [28], we have also shown that using traditional PCA based embedding ap-proaches, such as [6], [8], [13], [14], with multiple PCAs in-stead of only a single PCA, the performance is improved sig-nificantly. In this paper this result is taken one step further, by developing an iterative approach to obtain the affine sub-spaces and codebooks at the same time. In this way, the pro-posed method achieves lower quantization error, which leads to a better encoding scheme with state-of-the-art perfor-mance. The main contributions of the proposed method are the following:

- Vector quantization is defined as a multiple sub-space learning problem, where the objective is to minimize the quantization error of the training samples in the learnt subspaces, while also mini-mizing the projection error of the samples to the corresponding subspaces.
- An optimization problem that jointly minimizes both errors defined above is formulated as an iter-ative process.
- A simple, yet effective scheme for faster sample encoding is proposed, by efficiently selecting a limited number of subspaces, thus decreasing the computational cost required for evaluating all possible encodings.

  The proposed approach is evaluated on publicly available datasets, and shown to achieve state-of-the-art performance

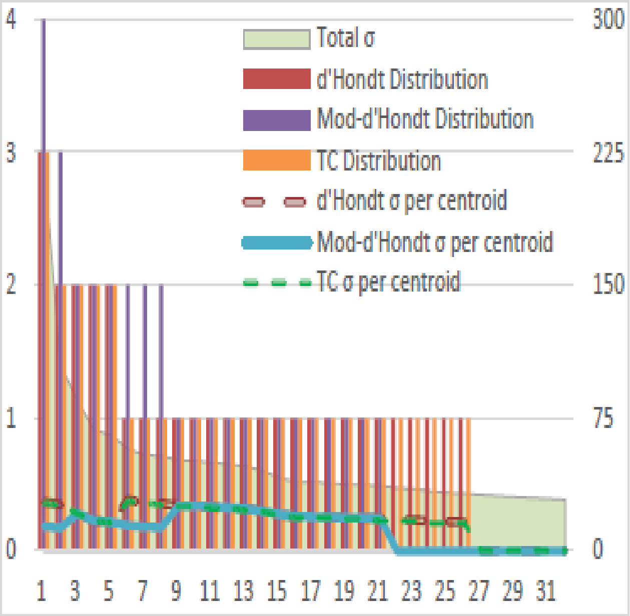
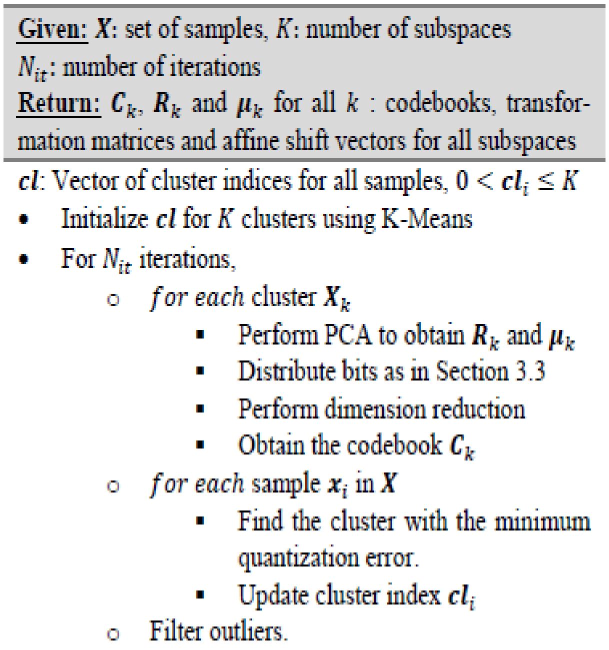


Fig. 1: Bit allocation as given in TC [14], by d'Hondt and Modified d'Hondt methods. (left vertical axis: bits per

dimension, right vertical axis: standard deviation, horizontal axis: dimensions).

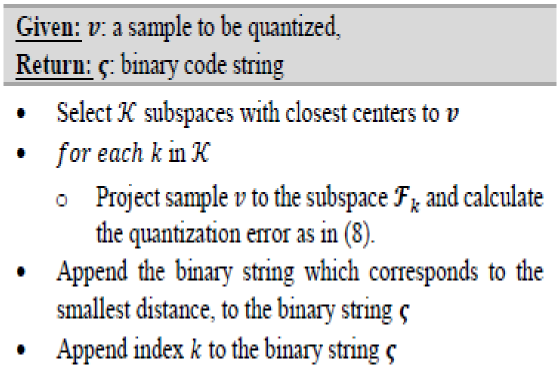
As it can be seen in **Fig. 1**, the *Modified d'Hondt* method emphasizes more the dimensions with higher variances. Since there is a limited number of bits, assigning multiple bits to one dimension means that another dimension is discarded from quantization, so this dimension can also be removed from the subspace. In other words, the reduced number of dimensions *Lk* for a subspace is the number of dimensions which has at least one allocated bit.

TRAINING FOR K-SUBSPACE QUANTIZATION



```
Given: $X$: set of samples, $K$: number of subspaces
$N_{it}$: number of iterations
Return: $C_k$, $R_k$ and $\mu_k$ for all $k$ : codebooks, transformation matrices and affine shift vectors for all subspaces
$cl$: Vector of cluster indices for all samples, $0 < cl_i \leq K$
•  Initialize $cl$ for $K$ clusters using K-Means
•  For $N_{it}$ iterations,
     o  for each cluster $X_k$
          ▪  Perform PCA to obtain $R_k$ and $\mu_k$
          ▪  Distribute bits as in Section 3.3
          ▪  Perform dimension reduction
          ▪  Obtain the codebook $C_k$
     o  for each sample $x_i$ in $X$
          ▪  Find the cluster with the minimum quantization error.
          ▪  Update cluster index $cl_i$
     o  Filter outliers.
```

ENCODING A NEW SAMPLE



```
Given: $v$: a sample to be quantized,
Return: $\varsigma$: binary code string
•  Select $\mathcal{K}$ subspaces with closest centers to $v$
•  for each $k$ in $\mathcal{K}$
     o  Project sample $v$ to the subspace $\mathcal{F}_k$ and calculate the quantization error as in (8).
•  Append the binary string which corresponds to the smallest distance, to the binary string $\varsigma$
•  Append index $k$ to the binary string $\varsigma$
```

The main difference between such methods and the proposed method is, while summation based methods require the addition of the code vectors from subcode books, in KSSQ, the most suitable one among them is selected, i.e., the one with the lowest quantization error, as given in (8). This is first of all, computationally advantageous because the summation based algorithms are usually computationally expensive. This is due to the limited constraints put on the generation and selection of code vectors in order to obtain better quantization performance.

In the proposed method however, the constraints of many Cartesian product based approaches such as [12], [14], [16] are retained, which require much less computations. Thanks to the jointly optimized subspace generation approach, the assumptions of the given constraints are much more realistic for the given dataset, resulting in a quantization scheme with improved performance.

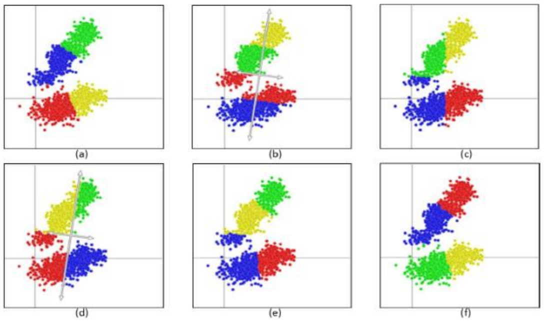


Fig. 2: A comparison of the methods (a) K-Means, (b) TC, (c) PQ, (d) OPQ Parametric, (e) LOPQ and our method (f) KSSQ for 2-bit quantization, obtained by running the algorithms on a 2-D toy example. For LOPQ and KSSQ there are 2 sub quantizers and for K-Means obviously the number of centroids is 4. Gray arrows are the principal components.

III. EXPERIMENTS

The proposed approach is tested on two publicly available datasets, SIFT1M and GIST1M [12]. SIFT1M consists of 1 Million samples of 128-dimensional SIFT vectors for test, 100,000 vectors for training and 10,000 for queries. GIST1M consists of 1 Million samples of 960-dimensional GIST vectors for test, 500,000 vectors for training and 1,000 queries.
The proposed method is trained using the given training sets and exhaustive search is performed on both datasets for all queries. K=256 and $\mathcal{K}$=16 for SIFT1M, and K=32 and $\mathcal{K}$=8 for GIST1M are selected, as later justified in **Sec-tion 5**. The proposed method (**KSSQ**) is compared with the recent state-of-the-art methods from the literature such as, *Transform Coding* (**TC***) [14], *Product Quantization* (**PQ**) [12], *Cartesian K-Means/Optimized Product Quantization* (**CKM/OPQ**) [15], [16], Distance Encoded Product Quantization (**DEPQ**) [17], an exhaustive implementation of Locally Optimized Product Quantization (**E-LOPQ***) [18], *Optimized Cartesian K-Means* (**OCK**) [19], *Additive Quantization* (**AQ/APQ**) [20], *Composite Quantization* (**CQ**) [21], *Optimized Tree Quantization* (**OTQ**) [22] and *MPCA Binary Embedding* (**MPCA-E***) [28]. The results for most of the

competing methods are obtained from the figures in the original publications while our own implementations of TC*, DEPQ*, E-LOPQ* and MPCA-E* are used. For DEPQ*, K=128 is selected and 1 bit is allo-cated for distance encoding as suggested in [17]. For E-LOPQ* an exhaustive version of LOPQ is developed for fair comparison with other exhaustive methods. K=256 is se-lected, allocating 8 bits for cluster index overhead. For MPCA-E, the multiple PCA version of the Transform Coding is selected as it provides the best retrieval performance [28]. For AQ/APQ, AQ is compared for 32-bits coding and APQ for 64-bits as suggested by the authors. *NA* indicates that the corresponding results are not presented in the original publi-cation for the corresponding method.

IV. CONCLUSION

In this study a novel vector quantization algorithm is proposed for the approximate nearest neighbor search problem. The proposed method explores the quantization centers in affine subspaces through an iterative technique, which jointly attempts to minimize the quantization error of the training samples in the learnt subspaces, while minimizing the projection error of the samples to the corresponding subspaces. The proposed method has proven to outperform the state-of-the-art-methods, with comparable computational cost and additional storage. In this paper it is also shown that, dimension reduction is an important source of quantization error, and by exploiting subspace clustering techniques the quantization error can be reduced, leading to a better quantization performance.

REFERENCES

[1] P. Indyk and R. Motwani, “Approximate nearest neighbors: towards removing the curse of dimensionality,” Proc. thirtieth Annu. ACM Symp. Theory Comput., pp. 604–613, 1998.

[2] J. Wang, H. T. Shen, J. Song, and J. Ji, “Hashing for Similarity Search: A Survey,” in arXiv preprint, 2014, p. :1408.2927.

[3] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, “Locality-Sensitive Hashing Scheme Based on P-stable Distributions,” in SCG, 2004, p. 253.

[4] K. Terasawa and Y. Tanaka, “Spherical LSH for Approximate Nearest Neighbor Search on Unit Hypersphere,” in WADS, 2007, pp. 27–38.

[5] X. He, D. Cai, S. Yan, and H. Zhang, “Neighborhood Preserving Embedding,” in ICCV, 2005.

[6] H. Jegou, M. Douze, C. Schmid, and P. Perez, “Aggregating local descriptors into a compact image representation,” in CVPR, 2010, pp. 3304–3311.

[7] J. Heo, Y. Lee, and J. He, “Spherical hashing,” in CVPR, 2012.

[8] A. Gordo, F. Perronnin, Y. Gong, and S. Lazebnik, “Asymmetric distances for binary embeddings.,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 36, no. 1, pp. 33–47, Jan. 2014.

[9] W. Dong, M. Charikar, and K. Li, “Asymmetric distance estimation with sketches for similarity search in high-dimensional spaces,” in SIGIR, 2008, p. 123.

[10] S. Lloyd, “Least squares quantization in PCM,” IEEE Trans. Inf. Theory, vol. 28, no. 2, pp. 129–137, 1982.

[11] A. K. Jain, “Data clustering: 50 years beyond K-means,” Pattern Recognit. Lett., vol. 31, no. 8, pp. 651–666, 2010.

[12] H. Jégou, M. Douze, and C. Schmid, “Product quantization for nearest neighbor search.,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 33, no. 1, pp. 117–28, Jan. 2011.

[13] Y. Gong and S. Lazebnik, “Iterative quantization: A procrustean approach to learning binary codes,” in CVPR, 2011, pp. 817–824.

[14] J. Brandt, “Transform coding for fast approximate nearest neighbor search in high dimensions,” in CVPR, 2010, pp. 1815–1822.

[15] T. Ge, K. He, Q. Ke, and J. Sun, “Optimized Product Quantization.,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 36, pp. 1–12, Dec. 2014.

[16] M. Norouzi and D. J. Fleet, “Cartesian K-Means,” in CVPR, 2013, pp. 3017–3024.

[17] J.-P. Heo, Z. Lin, and S.-E. Yoon, “Distance Encoded Product Quantization,” in CVPR, 2014, pp. 2139–2146.

[18] Y. Kalantidis and Y. Avrithis, “Locally Optimized Product Quantization for Approximate Nearest Neighbor Search,” in CVPR, 2014.

[19] J. Wang, J. Wang, J. Song, X.-S. Xu, H. T. Shen, and S. Li, “Optimized Cartesian K-Means,” IEEE Trans. Knowl. Data Eng., vol. 27, no. 1, pp. 180–192, Jan. 2015.

[20] A. Babenko and V. Lempitsky, “Additive Quantization for Extreme Vector Compression,” in CVPR, 2014, pp. 931–938.

[21] T. Zhang, D. Chao, and J. Wang, “Composite Quantization for Approximate Nearest Neighbor Search,” in ICML, 2014.

[22] A. Babenko and V. Lempitsky, “Tree Quantization for Large-Scale Similarity Search and Classification,” in CVPR, 2015.

[23] R. M. Gray and D. L. Neuhoff, “Quantization,” IEEE Trans. Inf. Theory, vol. 44, no. 6, pp. 2325–2383, 1998.

[24] N. Kambhatla and T. K. Leen, “Dimension Reduction by Local Principal Component Analysis,” Neural Comput., vol. 9, no. 7, pp. 1493–1516, Oct. 1997.

[25] V. Gassenbauer, J. Křivánek, K. Bouatouch, C. Bouville, and M. Ribardière, “Improving Performance and Accuracy of Local PCA,” Comput. Graph. Forum, vol. 30, no. 7, pp. 1903–1910, Sep. 2011.

[26] P. Agarwal and N. Mustafa, “k-Means Projective Clustering,” in SIGMOD, 2004, pp. 155–165.

[27] C. M. Bishop, “Bayesian PCA,” in NIPS, 1999, vol. 11, pp. 382–388.

[28] E. C. Ozan, S. Kiranyaz, and M. Gabbouj, “M-PCA Binary Embedding For Approximate Nearest Neighbor Search,” in BigDataSE, 2015.

[29] M. Gallagher, “Proportionality, disproportionality and electoral systems,” Electoral Studies, vol. 10. pp. 33–51, 1991.

[30] A. Babenko and V. Lempitsky, “The inverted multi-index,” in CVPR, 2012, vol. 14, no. 1–3, pp. 3069–3076.

[31] H. Jégou, R. Tavenard, M. Douze, and L. Amsaleg, “Searching in one billion vectors: Re-rank with source coding,” ICASSP, no. 3, pp. 861–864, 2011.