Kornana Manasa* et al.
(IJITR) INTERNATIONAL JOURNAL OF INNOVATIVE TECHNOLOGY AND RESEARCH
Volume No.5, Issue No.3, April – May 2017, 6505-6510.

Proficient Product Quantization for Large-Scale High Dimensional Data Using Approximate Nearest Neighbor Search

KORNANA MANASA
M.Tech Scholar, Department of Computer Science Engineering, AITAM.

D.T.V DHARMAJEE RAO
Prof, Department of Computer Science and Engineering, AITAM, Tekkali, Srikakulam, A.P, India.

Abstract: **K-nearest neighbor's classification and regression is broadly utilized as a part of data mining because of its easiness and precision. At the point when a prediction is required for an inconspicuous data case, the KNN algorithm will search through the preparation dataset for the k most comparable occasions. Finding the esteem k is application subordinate, thus a nearby esteem is set which expands the exactness of the issue. Grouping the question the lion's share class of its k neighbors is called K-nearest neighbors classification. In this paper the occurrence or question be grouped is known as the issue protest or venture in short. Worldwide KNN approach utilizes the entire data for searching the k-nearest neighbors of the venture. For data KNN approach is utilized where test objects are arbitrarily chosen from the preparation data space. Keeping in mind the end goal to enhance the exactness of finding the correct k-neighbors of nearby KNN, among various ANN approaches proposed in the current years, the ones in light of vector quantization emerge, accomplishing best in class comes about. Product quantization (PQ) decays vectors into subspaces for independent handling, taking into account quick query based separation estimations. This postulation work intends to lessen the intricacy of AQ by changing a solitary most costly stride in the process – that of vector encoding. Both the remarkable search execution and high expenses of AQ originated from its all-inclusive statement, along these lines by forcing some novel outside imperatives it is conceivable to accomplish a superior trade off: lessen many-sided quality while holding the precision advantage over other ANN strategies.**

Keywords: — **High-Dimensional Indexing; Image Indexing; Very Large Databases; Approximate Search; ANN; K- Nearest Neighbors;**

I. INTRODUCTION

Approximate nearest neighbor (ANN) search in high dimensional spaces is a repeating issue in computer vision, as well as experiencing noteworthy advance. A vast group of techniques keep up all data focuses in memory and depend on effective data structures to figure just a set number of correct separations that is in a perfect world settled. At the other extraordinary, mapping data focuses to smaller paired codes is proficient in space as well as may accomplish quick thorough search in hamming space. Product quantization (PQ) is an option conservative encoding strategy that is discrete however not parallel and can be utilized for comprehensive or non-thorough search through transformed ordering or multi-ordering. As is valid for most hashing techniques, better fitting to the hidden conveyance is basic in search execution, and one such approach for PQ is upgraded product quantization (OPQ) and its proportional Cartesian k-means. How are such preparing techniques gainful? Distinctive criteria are pertinent, however the basic rule is that all bits dispensed to data focuses ought to be utilized sparingly. Since search can be made quickly, such strategies ought to be eventually (a) k-means (b) PQ (c) OPQ (d) LOPQ. Thusly, k-means, By compelling centroids on a pivot adjusted, m-dimensional lattice, PQ accomplishes k m centroids keeping search at O(dk), a considerable lot of these centroids stay without data bolster e.g. on the off chance that the appropriations on m subspaces are not autonomous. OPQ enables the network to experience discretionary revolution and reordering of measurements to better adjust to data and adjust their change crosswise over subspaces to coordinate piece portion that is additionally adjusted. The nearest neighbor (NN) search, or the issue of coordinating the offered protest the most comparable one from the given gathering, is to a great degree normal; it is particular to science or building, as well as penetrates the regular daily existence in many structures, for example, acknowledgment. Its specialized definition can fluctuate in light of the idea of the products and how their likeness is built up. The last is generally characterized as a capacity, giving a numeric yield esteem. The objects of search are usually vectors, taking into account an assortment of similitude measures to be utilized, both metric and nonmetric. While the search issue is paltry when the quantity of articles to consider is little, the advances in software engineering and innovation prompt a reliable development of data sizes. Hence the data structures permitting productive search were widely contemplated since 1970s. Branch-and-bound approach specifically brought about various sorts of search trees, taking into account inquiries to be of logarithmic many-sided quality concerning

data estimate. Space parceling would stay overwhelming for a considerable length of time from that point. The development of Big Data has prompted reexamination of many already settled arrangements. This new condition displays various difficulties, one of which is the sheer volume of the databases. Conventional algorithms and techniques generally turn out to be computationally infeasible in such situations, once in a while to the point of finish inapplicability. The nearest neighbor search procedures were no special case from this; some settled data structures were found to lose their points of interest totally in higher-dimensional spaces, getting to be noticeably substandard compared to comprehensive figurings. Numerous flow down to earth applications including the search don't require superbly precise outcomes. For example, when data recovery is played out, the returned records or pictures are regularly regarded satisfactory in the event that they are significant; the correct meaning of pertinence fluctuates on case by case premise, however can by and large be interpreted as meaning "sufficiently comparative to the ideal result". This variable, joined with already specified computational costs issue, prompts the idea of the rough nearest neighbor (ANN) search, which has been the concentration of much late research.

II. RELATED WORK

To empower KNN coordinating effective on enormous data, the researchers of KNN were settled to locate the estimated nearest neighbors instead of the correct neighbors. Can Ozan et al. [1] proposed a novel vector quantization technique for surmised nearest neighbors (ANN) search which empowers quicker and more precise recovery on freely accessible huge datasets. They characterized vector quantization as a various relative subspace learning issue and investigated the quantization centroids on different relative subspaces. An iterative approach to limit the quantization blunder is utilized. The computational cost of this strategy is practically identical to that of the contending strategies. Preceding the procedure of KNN C/R, large portions of the proposed techniques changed or anticipated the data into another subspace, where vector measurements are decreased, reordered or pivoted utilizing COMPUTERA [3]. Decorrelating the data utilizing a solitary computer step may not bring the coveted factual independency among measurements, particularly if the data don't take after a Gaussian dissemination, which is the center innate supposition of computer [2]. Highlight choice is required in preprocessing stage as it diminishes algorithm overhead. Dawen et. al. [2] proposed a nearest neighbor approach utilizing connection investigation under a MapReduce system on a Hadoop stage, to address the troublesome issue of continuous prediction with substantial preparing data. It was executed by an ongoing prediction framework (RPS) including disconnected circulated preparing (ODT) and online parallel prediction (OPP), in view of a parallel k-nearest neighbors streamlining (ParKNNO) classifier. Parceling trees like the kd-tree [6], [7] is one of the best known nearest neighbors algorithms. While exceptionally powerful in low dimensionality spaces, its execution rapidly diminishes for high dimensional data. Trees are regularly utilized for nearest neighbors search as the search multifaceted nature is logarithmic. Most KNN strategies utilize k-d trees to store k-nearest neighbors and various hash tables are utilized for ordering. Muja et al. [8] proposed algorithms for surmised nearest neighbor coordinating to defeat the restriction of finding nearest neighbors matches to high dimensional vectors that speak to the preparation data. For coordinating high dimensional elements the randomized k-d timberland and the need search k-means tree were assembled and discharged as open source library called Fast Library for Approximate Nearest Neighbors (FLANN). Arya et al. [9] proposed a variety of the k-d tree to be utilized for rough search by considering (1+ Ɛ)- surmised nearest neighbors, focuses for which dist(i, p-question) $\leq$ (1+ε)dist(i*, p-protest) where i* is the correct nearest neighbor. The creators additionally proposed the utilization of a need line to support search speed. This strategy is otherwise called "mistake bound" Approximate search. The KNN technique, albeit basic beats more advanced and complex strategies as far as speculation mistake. Be that as it may, the key test with this classifier is settling the fitting estimation of k. García et al. [10] introduced a basic approach to set a nearby estimation of k. A possibly extraordinary k to each model is related to acquire the best estimation of k by enhancing a standard comprising of the nearby and worldwide impacts of the distinctive k esteems in the neighborhood of the model. Twofold implanting of highlight vectors for speedier separation computations has turned into a profoundly prominent research theme lately and the basic approach is to encode the vectors as parallel strings and pack vast datasets in significantly littler sizes, diminishing the capacity cost [9]. Moreover, the guess of the separation between two vectors by utilizing pre-figured separation esteems gives a noteworthy lift as far as the search speed. Can Ozan et al. [1] actualized an aggressive quantization for surmised nearest neighbors search in [10].

III. PRODUCT QUANTIZATION (PQ)

Product quantization (PQ) is one of the most important methods in the vector quantization family. To allow for more powerful representations with small number of codevectors, PQ splits the

data space into M subspaces, each of D/M dimensions. Vector quantization (learned with Lloyd's algorithm) is then separately applied on eachsubspace, resulting in M codebooks. Every database vector can be subsequently reconstructed by concatenating M corresponding codevectors. Assuming that each codebook has K codevectors, the total number of possible representations is KM. Any quantized database vector is stored as a sequence of M codes, indexing into K elements each, resulting in a total code length of M log2 K bits. The amount of memory required for codebook storage (in terms of scalar values) is $M \cdot K \cdot DM = KD$, which is small in practice, as $K \ll N$. Figure 7 shows an example of a product quantizer applied to 128-dimensional vector. It is typical to use parameter values which are powers of 2, as indices are stored in a binary computer memory. In this case M = 8 codebooks are used, and each 16-dimensional subspace is quantized to K = 256 centroids (codevectors). A single subspace can then be indexed with an 8-bit (log2 256 = 8) code. The whole vector is subsequently stored as a concatenation of sixteen 8-bit codes for a total of 64 bits.

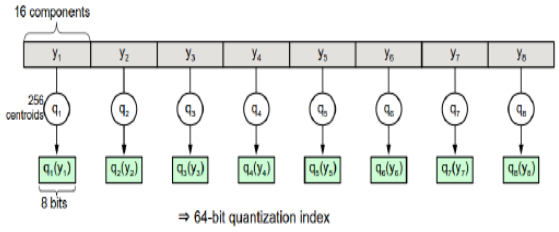

Figure 1. Product quantizer for 128-dimensional vectors.

IV. OPTIMIZED PRODUCT QUANTIZATION (OPQ)

Optimized product quantization (OPQ), also known as Cartesian k-means (CKM), is a variant of product quantization that makes an attempt to optimize the allocations of dimensions to subspaces. In the original PQ paper it was noted that the search performance varies greatly based on the contents (semantics) of the data. Product quantization considers all the subspaces to be equal in terms of information content, which is quite likely to be incorrect for the real sets of vectors. In fact, the choice of dimensions for a particular subspace has been shown to have a major effect on the quantization performance. Since the domain knowledge is not always available, the quantization algorithm can benefit from an internal subspace optimization. Rotation is a linear transformation that preserves vector norms and pairwise Euclidean distances; for any orthogonal $D \times D$ matrix R and D-dimensional vectors x and y the following expression holds: $\|x - y\|2\ 2 = \|Rx - Ry\|2\ 2$. (10) Due to (10) any centroid assignments (codes) of a vector quantizer with codebook C on dataset X remain valid, if both code vectors and data vectors are transformed with the same matrix R. This property naturally generalizes to PQ. The benefit of rotation lies in the fact that it can represent any reordering of the vector dimensions. Optimizing rotation of PQ quantizer is thus equivalent to finding better allocation of dimensions to subspaces. Random rotation has been explored and found to improve the quantization error and search performance. Further gains can be expected if the matrix R is fine-tuned with respect to the data. OPQ differs from PQ in that it learns not only codebooks and codes, but also an orthogonal transformation matrix R. Any data to be encoded is first rotated via multiplication by R, and then the product quantizer is applied. Computational complexity of encoding a single vector thus increases by the multiplication cost, resulting in a total estimate of $O(KD + D\ 2\ )$. Two formulations of OPQ exist. Parametric formulation is derived from the assumption that the data is generated by Gaussian distribution. If the assumption holds, the solution is provably optimal and achieves the theoretical lower bound of quantization error, which is derived to be $E \geq DMK - 2MD \sum |\Sigma m|$ $MMDm$=1 , (11) where Σm is the covariance sub matrix of m-th subspace of rotated data Rx. Further theoretical analysis shows that for a parametric solution to achieve optimality, two conditions must hold: subspace independence and balance of subspace variance. These conclusions exactly correspond to the ones drawn in ITQ – a hashing-based method described earlier. To construct a matrix R satisfying the above requirements for subspaces, a simple greedy algorithm – eigenvalue allocation – was proposed. The database is first processed with PCA, which takes $O(ND\ 2 + D\ 3\ )$ operations. The eigenvalues are then sorted in a descending order and sequentially allocated to M bins of capacity D/M, such that the product of values inside each bin would be roughly equal (thus balancing the variance). When the eigenvectors are reordered according to allocation of their respective eigenvalues, the matrix R is formed. Then the database is rotated and PQ is applied on the result. Total training complexity of parametric OPQ is thus $O(ND\ 2 + D\ 3 + NLKD)$. Naturally, the Gaussian assumption rarely holds in practice. Nonparametric OPQ does not construct a single rotation matrix, but instead optimizes it during the training phase. At the beginning of each iteration the data is rotated with the current estimate of R. Then the codebook adaptation and encoding are performed on the rotated data, with no differences from PQ. Finally, the current estimate of R is updated so as to minimize the quantization error criterion: 2 min , F R RX Y– (12) where X

and Y are matrices composed of original (not rotated) data vectors and their reconstructions, respectively, and $\|\cdot\|F$ is the Frobenius norm: $\|A\|F = \sqrt{trace(A*A)}$. To solve this optimization problem, first the singular value decomposition (SVD) of XYT is calculated: $XYT = USVT$. The matrix R is then calculated as follows: $R = VUT$. (13) The complexities of this procedure are as follows: XYT multiplication is $O(ND\ 2\ )$, SVD is $O(D\ 3\ )$, and VUT multiplication is also $O(D\ 3\ )$. Since rotation is adapted in each iteration, the total cost of non-parametric OPQ training becomes $O(L(NKD + ND\ 2 + D\ 3\ ))$. Nonparametric OPQ typically performs better on real datasets, as it does not rely on Gaussian assumption. OPQ training can be initialized similarly to PQ, with an identity matrix as a first estimate of *R*. Better results are attained if parametric solution is used for initialization, although that increases the amount of computation required.

V. VECTOR QUANTIZATION FOR APPROXIMATE SEARCH

Vector quantization (VQ) is a technique that represents a given set of NDdimensional vectors with another set of K*centroids* of the same dimensionality

(*K*<*N*). The set of centroids C is called a *codebook*, while centroids themselves are alternatively called *code vectors*. Each vector from the original set is represented by one and only one code vector. Needless to say, VQ representation of the data is loss. The quantization loss is typically measured by mean squared error (MSE) between the data vectors and their reproductions:

$$E = \frac{1}{N}\sum_{i=1}^{N}\|q(x) - x\|_2^2, \quad (5)$$

Where x is the data vector and $q(x)$ is the corresponding code vector.

Any optimal (having minimal quantization loss) VQ quantizer is subject to two necessary optimality conditions, known as *Lloyd conditions* [17]. First condition states that each vector must be assigned to a centroid which is the closest in Euclidean distance terms:

$$q(x) = \arg\min_{c_i} \|c_i - x\|_2. \quad (6)$$

ci C

Second condition limits the position of each centroid to the mean value of all the vectors it represents:

$$c_i = \frac{1}{n_i}\sum_{j=1}^{n_i} x_j\ ,\ s.t.\ q(x_j) = c_i \quad (7)$$

These conditions are greatly reminiscent of well-known k-means clustering. Indeed, a common way to construct a vector quantizer is with a *Lloyd's algorithm*:

1. Randomly initialize centroid positions.
2. Repeat for a predetermined number of iterations:
   a. Assign every data vector to the nearest centroid (6).
   b. Replace every centroid with the mean of vectors assigned to it (7).

The total computational complexity of Lloyd's algorithm is $O(LNKD)$, where L is the number of iterations. While Lloyd's algorithm is intuitive, simple and widely used, it only converges to a locally optimal solution. The results may vary wildly with different initializations, and some starting points may lead to very poor representations.

A set of vectors assigned to a particular centroid is known as *Voronoi cell* or just a *cell*. Any vector quantizer therefore defines a space partitioning – a *Voronoi tessellation*, with each cell defining a separate subspace. An example of such is shown on Figure 2.

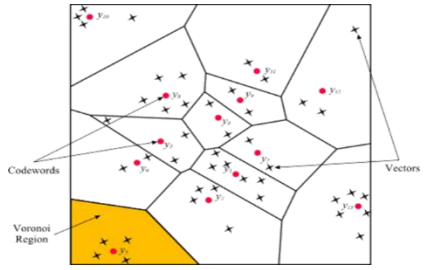

Figure 2. An example of a vector quantizer and corresponding Voronoi tesselation

Evidently, the Voronoi tessellation can be used as a foundation for non-exhaustive distance calculation. One strong advantage of such an approach, when compared to hashing, would be the fact that the original data space is preserved, and the Euclidean distances remain meaningful instead of being approximated with Hamming distances. Better preservation of pairwise vector dissimilarities, in turn, leads to better search performance.

Since vector quantization does not change the distance function, there are two possible approaches to the nearest neighbor search. If *symmetric distance computation* (SDC) is used, the query vector is quantized to the nearest centroid, and then the distances from that centroid to all the others are estimated. In case of *asymmetric distance computation* (ADC), the distances between the query and all the centroids are calculated directly. Both scenarios are shown on Figure 2.

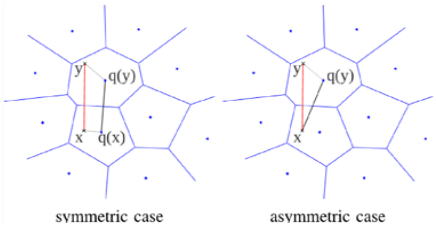

Figure 3. Distance computation with vector quantization:

symmetric (left) and asymmetric (right).

Since centroid positions are always known in advance when the search is performed, it is possible to precompute all the pairwise distances between centroids and store them. This seemingly makes SDC very quick, but to quantize the query, one needs to calculate the distance from it to every centroid, which is the exact same process as in ADC. As a result, the differences in the calculation speed between the two are minimal. However, the quantization of the query vector introduces additional distortion. It is thus reasonable to conclude that ADC is superior to SDC and should be preferred. In fact, availability of ADC is a direct consequence of space preservation and can be considered an additional advantage of vector quantization over hashing-based approaches.

Despite the aforementioned benefits, traditional vector quantization without any changes is not a practical solution for approximate nearest neighbor search, as it too suffers from the "curse of dimensionality". If the number of codevectors (centroids) is

K, the code (index) of each database vector has a length of $\log_2 K$ bits. This is not nearly enough for discrimination; for example, if a vector of dimensionality 960 would be represented by a 960-bit code (1 bit per dimension), the corresponding vector quantizer would have to have 2^{960} centroids. Evidently, it is impossible to even store a codebook of that size in a computer memory, let alone applying any search operations on it. A different approach is necessary to take advantage of VQ properties. A number of such approaches have emerged, proceeding to outperform hashing-based techniques by a large margin.

VI. PROPOSED METHOD

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.

VII. 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. So far we have focused mainly on exhaustive search but an index-based non-exhaustive extension for the proposed method can be further investigated. Our approach can also be extended to labeled datasets in order to test k-nearest neighbor classification performance. These will be the topics of our future work.

VIII. REFERENCES

[1] Dimitris Papadias. Hill climbing algorithms for content-based retrieval of similar configurations. In Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '00, pages 240–247, 2000.

[2] Rodrigo Paredes and Edgar Chvez. Using the k-nearest neighbor graph for proximity searching in metric spaces. In In Proc. SPIRE'05, LNCS 3772, pages 127–138, 2005.

[3] Yury Lifshits and Shengyu Zhang. Combinatorial algorithms for nearest neighbors, near-duplicates and small-world design. In Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'09, pages 318–326, 2009.

[4] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in Proc. 30th Annu. ACM Symp. Theory Comput., 1998, pp. 604–613.

[5] J. Wang, H. T. Shen, J. Song, and J. Ji, "Hashing for similarity search: A survey," arXiv preprint, 2014, p. 1408.2927.

[6] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Localitysensitive hashing scheme based on P-stable distributions," in

Proc. 20th Annu. Symp. Comput. Geom., 2004, pp. 253–262.

[7] K. Terasawa and Y. Tanaka, "Spherical LSH for approximate nearest neighbor search on unit hypersphere," in Proc. 10th Int. Conf. Algorithms Data Struct., 2007, pp. 27–38.

[8] X. He, D. Cai, S. Yan, and H. Zhang, "Neighborhood preserving embedding," in Proc. 10th IEEE Int. Conf. Comput. Vis., 2005, pp. 1208–1213.

[9] H. Jegou, M. Douze, C. Schmid, and P. Perez, "Aggregating local descriptors into a compact image representation," in Proc. IEEE Conf. Comput. Vis. Pattern Recog., 2010, pp. 3304–3311.

[10] W. Dong, M. Charikar, and K. Li, "Asymmetric distance estimation with sketches for similarity search in high-dimensional spaces," in Proc. 31st Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval, 2008, pp. 123–130.

AUTHOR's PROFILE

KORNANA.MANASA. Holds a B.Tech certificate in Computer Science Engineering from Aditya Institute of Technology And Management the University of Jntu Kakinada. She presently Pursuing M.Tech Department of computer science engineering from Aitam, Tekkali, Srikakulam, AP, India.

D.T.V DHARMAJEE RAO is a Professor in the Department of CSE at AITAM, Tekkali, Srikakulam, AP. He received the B.Tech degree in Computer Science and Engineering and M.Tech degree in Computer Science and Technology from Andhra University, Visakhapatnam, A.P and India. He is currently Pursuing PH.D degree in the Department of Computer Science Engineering, JNT University, Kakinada, A.P, India. His Current Research interests include Data Mining, Neural Networks and Linear Algebra Techniques.

