MULTI-CODEBOOK QUANTIZATION

[1]**P.V.B.Subramanyam,**
M.Tech , Department of CSE
JNTUA College of Engineering, Ananthapuram,
Andhra Pradesh, India.

[2]**Dr. A. Sureshbabu M.Tech., Ph.D.**
Professor, Department of CSE
JNTUA College of Engineering, Ananthapuram,
Andhra Pradesh, India.

Abstract: Approximates Nearest Neighbour search(ANN search) deliver successful results in several fields. ANN Search uses Hashing method and quantization method with static database only. They cannot manage properly in a database with information sharing dynamically because of the high computational cost for remodelling a new database. The existing online Product Quantization(OPQ) method is an issue in the dynamic database environment. Furthermore, it needs high scale computation for OPQ updates in partial PQ codebooks. This paper proposes a Multi-Codebook Quantization(MCQ) method for online updates to the dynamic database platform to develop the search implementation. This method uses the K- mean vector algorithm and iterative approaches for reducing the quantization errors between the original input data and their relating codewords. These codewords will represent the set of sub-codes selecting from multiple codebooks. The experimental results shows that online MCQ model is time efficient and more successful for OPQ in dynamic databases with ANN search.

Keywords : Products retrieval, Multi codebook quantization, Online product quantization, ANN, Online Hashing.

1.INTRODUCTION :

Approximate Nearest Neighbour(ANN) search in a fixed database has achieved excellent success in many areas. It can be used to perform the detection of objects, sampling, information retrieval, and image classification. In this era of big data, there is a huge quantity of this data generation at high speed every day. Handling dynamically growing data is not easy with the existing methods. ANN search is presented in a dynamic database in so many apps. For example, a large number of news articles are produced daily. It requires the updating of data continuously. So that the news search system[2] supports the tracking of news topics and the retrieval of the regularly changing news database.

For objects detection in video surveillance[3], videos data is recording continuously. This recording data may contain the dissimilar or similar objects which rapidly changes over time and the relevant images are retrieved on same image query. The collected and stored database is the only source in real time to answer the queries for image retrieval.

Recently many studies on online hashing[4][5], shows that hash-based ANN approaches can be adopted for updating dynamic databases. A new streaming data and then updating hash codes for existing stored data is a method of new hash functions. The searching is conducted in Hamming space, creates low computational cost.

However, the maintenance of the hash code is a major problem that has not been addressed in these works. The hash functions must be frequently updated to handle the streaming fashion of data, resulting in a constant hash code recomputation of all existing data in the reference database. This will certainly lead to a growing quantity of update time as information quantity rises. In fact, these online hashing methods require a scheme to maintain the past information in location so that the new hash code of the prior information can be accessed at any moment, to overcome storage inefficiency and computational expenditure. Therefore, computational complexity and storage costs remain a major concern in the development of an online indexing model.

Herve Jegou et al. proposed the Product Quantization (PQ)[6] is an efficient ANN search method solution. PQ partitions the original space in the Cartesian product of low dimensional subspaces and quantifies each subspace in a number of sub-code words. PQ is capable of producing a number of sub-code words with least amount of storage and performs an ANN search at a reduced cost of computing. In addition, it reduces the quantization error and achieves satisfactory recall performance. Hash based methods are used to represent each data instance by hash codes that depend on the set of hash functions. For every Quantization[10] method it consists of an index value for every data instance which are associated with a code words in the same vector space for data instance.

The online PQ[1] method is used to store streaming data. In addition, we set plan constraints to make partial updates of the codebook easier in order to reduce the time value of the updates. A relative loss bound has been derived to ensure the performance of our online PQ model over a sliding window approach, with emphasis on real-time data. It shows that our method is significantly faster to accommodate streaming data and outperforms competing online hashing methods in terms of search accuracy and time frame updates.

2. LITERATURE SURVEY:

Hashing strategies generate hard and fast hash functions to map the facts in order to facilitate the fast search for the nearest Neighbour. Current Hashing methods are split into data-dependent hashing and data-independent hashing in databases. The locality sensitive Hashing (LSH)[7] is one of the most distinguished tasks for data-independent hashing. Where its hashing features are generated randomly. LSH has a theoretical performance guarantee that compares instances of information could be mapped to comparable hash codes with a certain possibility. Because information with online hashing strategies is independent of the entered records, it can be followed in an online fashion without any problems. However, are derived from the hash functions of the given statistics. This could attain stronger general efficiency than the records of autonomous hashing techniques.

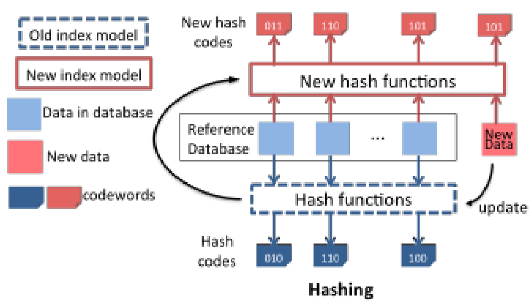

Fig.1: hashing

As shown in figure 1, the hash codes of the data points in the reference database will get updated if the hash functions get updated by new data.

A high-dimensional similarity search[8] is based on a similarity search for object data (e.g. images, documents) which can be categorized by a group of relevant features and represented as factors in a high dimensional characteristic area in view of the queries. The shape of the factors in that area, the closest (maximum comparable) items to the queries is required. The d-dimensional Euclidean area is a particularly interesting and also well researched situation. The statistical databases and data mining are retrieval of information, image and video databases, knowledge-raising system, reputation patterns, statistics and evaluation of records. The capabilities of the gadgets of interest are usually represented as points in the distance matrix used to measure the similarity of the gadgets. The fundamental problem then is to perform an index, which can be ignored by the similarity of trying to find question items.

Product quantization based on the ANN Search approach. PQ decomposes large space into a Cartesian low-dimensional subspace product to be quantified separately. Quantization indices can be used for this subspace. The vector used to represent this short Euclidean code distance between two vectors can be computed from the codes. The combination with the inverted file system. This method was validated on a data set of two billion vectors.

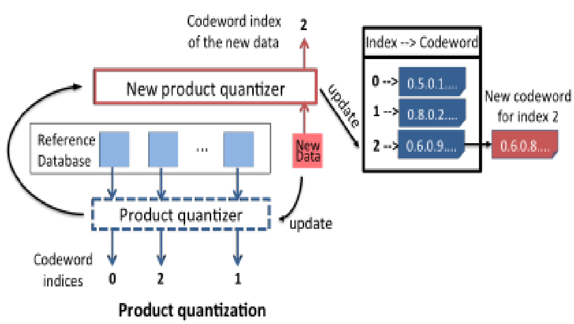

Fig.2 : Online product Quantization

As shown in Figure 2, The online PQ method assumes that the static database is not suitable for non-stationary data settings, especially when the data time is different. The online version of PQ was therefore used to deal with dynamic databases. Online product quantization is used as a codebook, every time streaming data is updated without retraining all data sets collected. Online PQ updates codebooks only and the codeword index of the existing data remains the same, but the codebook contains limited data information that can be received from a single product and similar data from large data sets. This allows for low performance and high cost of time and more memory space.

Additive quantization for extreme vector compression[9] is a new technique for high-dimensional vector compression. All these vectors are approximate using sum of M code words coming from different codebooks. The efficient distance can be shown by using Scalar product calculations between separated and uncompressed vectors. In order to minimize coding error, vector encoding and codebook learning method have been used in the novel approach. AQ can be used instead of online product quantization.

3. PROPOSED SYSTEM

Multi Codebook quantization is an effective and successful alternative to ANN search. Multi-codebook quantization (MCQ) methods will increase the search performance. Quantization method is proposed to minimize quantization error and to reduce computational costs. The idea of MCQ is closely linked to the K-means algorithm. The K-means algorithm is not directly applicable to large-scale data, i.e. to a very large number of centroids.

Product Quantization divides the sample vector into sub-vectors and quantifies each of them independently using sub-quantifiers. To overcome the limitation of ANN search, an iterative approach[11] is defined to minimize computational costs and to implement state-of-the-art .

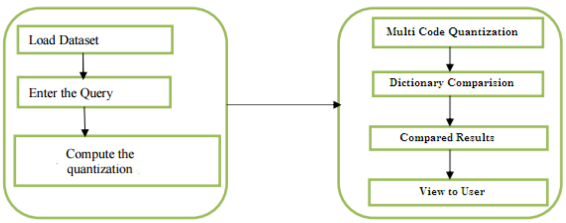

Fig.3 : Multi codebook Quantization

.

The proposed method is divided into some modules. In Load Dictionary Module, the admin loads the data dictionary related to products and ranks dataset. The user sends the query request for ANN search. Admin will apply the iterative approach to minimize the quantization. In the mean squared quantization module, the admin compute the mean squared quantization.

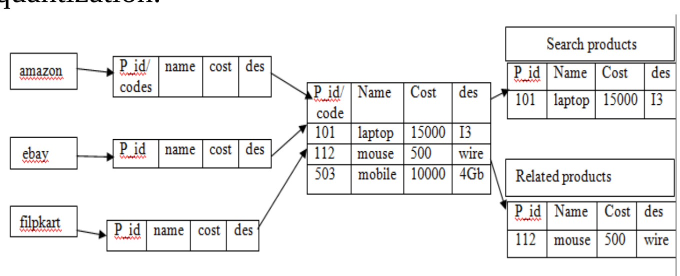

Fig.4 : multi codebook quantization module

In the MCQ module, multi-codebook quantization (MCQ) methods are obtained by minimizing the quantization error between the initial entry data and the respective codewords. The iterative approach defines the determination of subspaces, the allocation of bits, the quantization of multiple codes, and then provides cluster updates to show all ANN search.

It is intended to define the affine subspaces and then for each subspace to obtain the bits distributed between dimensions. Multiple PCAs are used to generate projection matrixes, statistical dependences between dimensions have been minimized. i.e. quantization can be performed independently on each dimension and the code vector can be obtained as a Cartesian product of quantified values.

Algorithm: The Multi-codebook quantization

1: function MCQ(Q);
2: subCQs={$subCQ^1[V]$,$subCQ^2[V]$,…,$subCQ^T[V]$};
3: while t <=T do
4: $t = t + 1$;
5: $NNVS^t$ = genNNVS($newTable[U][V]$,$subCQ^t[V]$);
6: for every *vector* ϵ *NNVS* do
7: Id_s←search *vectors* in $subCQTables^t$;
8: sc←obtained the vectors set through Id_s;
9: $S_C{}^t = S_C{}^t \cup sc$;
10: end for each
11: $S_C = S_C \cup S_C{}^t$;
12: end while
13: $S_{Ck} = S_C$;
14: size←Getsize of S_C;
15: if size > k then
16: S_{Ck}← analyze the similarity and sort;
17: end if

The input algorithm includes the products set of size k, the query vector is Q, the generator new table is new Table[U][V], and T subCQ tables, where subCQTables={$subCQTable^1$,$subCQTable^2$,...,$subCQ Table^T$}. The output is the products sets S_{Ck}, whose size is k.

Line 1 indicates that the query vector Q is quantized and compressed by the MCQ algorithm.

Line 2 indicates that the compressed vector MCQ(Q) is split into the T sub-compressed vectors, i.e. subCQs ={$subCQ^1[V]$, $subCQ^2[V]$, $subCQ^T[V]$}.

Line5 represents the generating NNVS for each sub-compressed vector $subCQ^t[V]$.

Lines 6–10 described that the algorithm searches for all vectors in the $\mathrm{subCQTable}^{t}$ and gets the corresponding codewords sets S^{t}, where $1 \le t \le T$.
Line 13 indicates that the algorithm obtained the final products set S_{Ck} through calculate similarity and sorting for their original vectors. The algorithm can also set of the similarity thresholds in advances and only adding the vectors whose similarity is greater than the threshold to the products set S_{Ck}.

User can enter queries to retrieve the data from dictionary module for selecting a particular product. Query is used for selecting the follow keyword on codeword or object name. It performs the query after computing the quantization. In MCQ module each query is compared with the follow keyword of the object name in data dictionary and shows the result to the user.

4. EXPERIMENT

We observationally assess the recommended system for design on the MCQ method with datasets. i.e e-commerce sites like flipkart, ebay, amazon, etc., collection of all data sets E-commerce site recommends many products to the admin, so that he can that add the product details in the site. Admin module maintains the site for transaction between user and e-commerce. He adds the product with all details, index as category. Each category contains the sub category with product name/code, manufacturer etc., user can perform the search product task with follow keywords for searching
Admin updates the ranks of a product and search history. A user can view the ratings to buy any product. An iterative approach defines the determination of subspaces, bit allocation. Multi Codes are quantization and then provides the cluster updates to show all the approximate nearest neighbours. Admin performs the tasks like view all products, view product search result and product rank result.

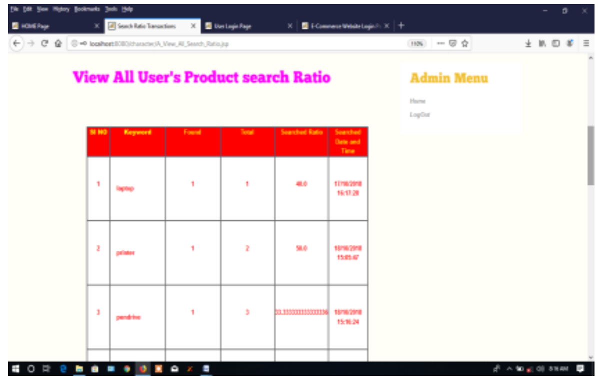

Fig 5 : user product search ratio

User can create the account with his details to register and to login. User performs searching for a product with follow keywords like product name/code to retrieve the product details. User can retrieve the view product with related products by using the MCQ method.

5. EXPERIMENT RESULTS:

This section describes the results of experiment. In the first phase, the value of k (number of sampled codes) to be chosen for the data set and the given number of indices in order to minimize the effective error. It worked out that the minimum estimations of k are basically free of n and h and consequently we can utilize a similar value for the various estimations of these parameters.

Fig.6 : product keyword searched results

Fig.7 : product rank results

In the second phase, We predict the no. of error-indices. Finally, we measure the efficiency of MCQ by computing (for a variety of data sets) the minimum number of indices which are required to achieve the specified error value. We also compare this performance with online product quantization.

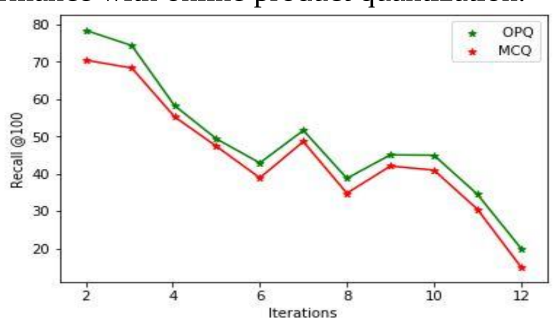

Fig.8: the number of products iteration by recall of search_history.

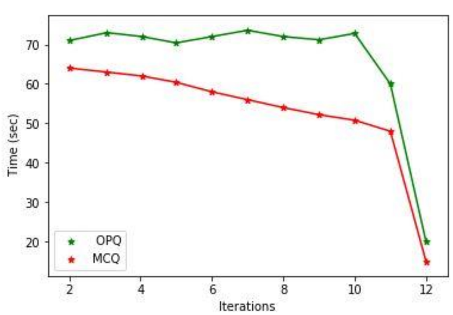

Fig.9: The number of iterations to the particular time period is the ratio of the products.

To compare with online product quantization, we implement that MCQ on random subsets of the total datasets. The average number of blocks are performed by MCQ query. The number of iterations to the particular time period is the ratio of the products and the comparison with products iteration by recall of search_history. So that the performance of MCQ will reduce the quantization error and computational cost to some extent.

6. CONCLUSION

In this paper, we have presented Multi codebook quantization methods for online updates and search performance in the original data. We generated a number of code indices by the vectors for each code. Vector consists of sub-code in the query. Vector reduces the quantization error rate and improve the recall product retrieval. We use the MCQ algorithms for products retrieval and design products similarity detection systems. The vector quantization algorithms are presented in PQ and OPQ can be obtained for less-dimensional codebooks by quantizing and compress with high-dimensional vectors. We achieved good performance in ANN search. But there are some quantization errors in the process of vector quantization and compression by the use of the *K*-means clustering algorithm. As for future work, we can combine deep learning with vector quantization algorithm to study the distribution of data and get more accurate codebooks.

REFERENCES:

[1]Donna Xu, Ivor W. Tsang, and Ying Zhang Online Product Quantization in IEEE ,2018.

[2]A. Moffat, J. Zobel, and N. Sharman, “Text compression for dynamic document databases,” TKDE, vol. 9, no. 2, pp. 302–313,1997.

[3]R. Popovici, A. Weiler, and M. Grossniklaus, “On-line clustering for real-time topic detection in social media streaming data,” in SNOW 2014 Data Challenge, 2014, pp. 57–63.

[4]L. Huang, Q. Yang, and W. Zheng, “Online hashing,” in IJCAI, 2013, pp. 1422–1428.
[5]F. Cakir and S. Sclaroff, “Adaptive hashing for fast similarity search,” in ICCV, 2015, pp. 1044–1052.
[6] H. J´egou, M. Douze, and C. Schmid, “Product quantization for nearest Neighbour search,” PAMI, vol. 33, no. 1, pp. 117–128, 2011.
[7] F. Cakir, S. A. Bargal, and S. Sclaroff, “Online supervised hashing,” CVIU, 2016.
[8]A. Gionis, P. Indyk, and R. Motwani, “Similarity search in high dimensions via hashing,” in VLDB, 1999, pp. 518–529.
[9] A. Babenko and V. S. Lempitsky, “Additive quantization for extreme vector compression,” in CVPR, 2014, pp. 931–938.
[10]R. M. Gray and D. L. Neuhoff, “Quantization,” IEEE Transactions on Information Theory, vol. 44, no. 6, pp. 2325–2383, 1998.
[11]Y. Gong and S. Lazebnik, “Iterative quantization: A procrustean approach to learning binary codes,” in CVPR, 2011, pp. 817–824.