Pyramid Encoding for Fast Additive Quantization

Anton Muravev[1], Ezgi Can Ozan[1], Alexandros Iosifidis[1,2], Moncef Gabbouj[1]
[1]Laboratory of Signal Processing, Tampere University of Technology, Tampere, Finland
[2]Department of Engineering, Aarhus University, Aarhus, Denmark
[1]{anton.muravev, ezgi.ozan, alexandros.iosifidis, moncef.gabbouj}@tut.fi, [2]alexandros.iosifidis@eng.au.dk

***Abstract*—The problem of approximate nearest neighbor (ANN) search in Big Data has been tackled with a variety of recent methods. Vector quantization based solutions have been maintaining the dominant position, as they operate in the original data space, better preserving inter-point distances. Additive quantization (AQ) in particular has pushed the state–of–the-art in search accuracy, but high computational costs of encoding discourage the practical application of the method. This paper proposes pyramid encoding, a novel technique, which can replace the original beam search to provide a significant complexity reduction at the cost of a slight decrease in retrieval performance. AQ with pyramid encoding is experimentally shown to obtain results comparable with the baseline method in accuracy, while offering significant computational benefits.**

Keywords—compact encoding; image retrieval; nearest neighbor search; vector quantization

I. INTRODUCTION

The nearest neighbor (NN) search is a ubiquitous problem encountered in many application areas. The recent emergence of Big Data [1] has rendered many existing solutions obsolete. As the dimensionality of the data increases, the computational costs of traditional NN search techniques grow exponentially, making them infeasible in practice [2]. This factor led to the introduction of approximate nearest neighbor (ANN) search techniques, which trades the accuracy of the solutions in favor of a manageable complexity. *Locality Sensitive Hashing* (LSH) [3] was among the first ANN approaches to achieve widespread adoption [4], resulting in emergence of a large family of hashing-based techniques [5][6]. Most recently quantization-based approaches, such as *Product Quantization* [7] and *Residual Vector Quantization* [8][9], have pushed the state-of-the-art in ANN search by learning lossy representations of the data, which allow for fast distance calculations.

Product Quantization (PQ) [7] performs a decomposition by splitting the original D-dimensional data into M subspaces of D/M dimensions each. Vector quantization [10] can then be applied independently within each subspace, using Lloyd's algorithm to learn K representative centroids. Thus M codebooks of K codewords (also called codevectors) are formed. The total number of possible representations is K^M, making PQ very space-efficient. The computational costs are also moderate: since the subspaces are by definition orthogonal, the codewords can be optimally assigned by M independent NN searches, requiring a total of $O(KD)$ operations. Learning the codebooks, as outlined above, can be performed independently in each subspace, resulting in the total cost of $O(NLKD)$, where N is the size of the training set and L is the number of iterations. PQ makes fast distance calculation possible by efficient use of table lookups: estimating the distance between a given query and an encoded vector takes $O(M)$ operations [7]. Unlike hashing-based methods, PQ operates within the original data space, retaining the original similarity measure (typically the squared Euclidean distance), which further improves the accuracy.

PQ assumes that every subspace is roughly equivalent in terms of information content, which is obviously not always the case in practice. *Optimized Product Quantization* (OPQ) [11], also known as *Cartesian K-means* (CKM) [12], addresses this drawback by adaptively allocating dimensions to subspaces instead of a simple splitting. This is achieved by learning a data-specific rotation, represented by an orthogonal matrix. In a general non-parametric approach, the rotation matrix is updated after each PQ learning iteration by solving the orthogonal Procrustes problem. Taking advantage of its formulation, OPQ achieves a significantly better performance with an additional cost. The computational complexity of learning an OPQ quantizer is $O(L(NKD + ND^2 + D^3))$ in the non-parametric case. Other suggested variants of product quantization include *Locally Optimized Product Quantization* (LOPQ) [13] and *Optimized Cartesian K-Means* (OCKM) [14].

Additive quantization (AQ) [15] is a generalization of both PQ and OPQ. As the name implies, the codevectors are added together instead of being concatenated to obtain the data representation. The codebooks are fully D-dimensional, and there is no subspace decomposition involved. The lack of constraints results in a more powerful representation, which, in turn, leads to a much more accurate search.

However, the higher generality also has its drawbacks. The lack of orthogonality between different codebooks complicates the distance computations, as the dot products between codewords are nonzero (unlike PQ or OPQ). In practice this means that one distance estimation takes $O(M^2)$ table lookups instead of $O(M)$. Another important consequence of non-orthogonality is the fact that learning cannot be split into several simpler problems, making the k-means algorithm unsuitable for codebook generation. The codebooks can instead be derived by solving the systems of linear equations, given the data encoding. Iterative learning is still possible by alternating between codebook adaptation and data encoding. The latter step is of

ISBN 978-0-9928626-7-1

utmost importance, as it drives both the representation accuracy and the computational costs of the quantizer.

These drawbacks can be addressed by imposing additional constraints on the representation. *Tree quantization* (TQ) enforces orthogonality between certain pairs of codebooks, as described by a dynamic graph structure [16]. *Composite quantization* (CQ) minimizes the dot product between codebook vectors, allowing for distance estimation with $O(M)$ required operations, as in PQ [17][18]. This work instead focuses on the optimization process in codebook encoding.

II. Encoding in Additive Quantization

Since additive quantization imposes no constraints on the codebooks, optimal encoding for AQ is equivalent to inference on a fully connected pairwise Markov Random Field (MRF), which is an NP-hard problem [15][19]. Thus, local heuristics are used instead of exhaustive search. The original AQ work [15] suggested two possible solutions – Iterated Conditional Modes (ICM) and Beam search. These are described in more detail before the proposed approach is presented.

A. Iterated Conditional Modes (ICM)

Iterated Conditional Modes (ICM) is an algorithm for approximate MRF inference that can be trivially adjusted to the problem of AQ encoding. It seeks a locally optimal representation by improving the current solution, one codebook at a time. Random assignment is used to initialize the encoding; then, one codebook is chosen and is searched through exhaustively to locate a codeword which can reduce the error. The other $M-1$ codewords are fixed during this search. Repeating this once for each codebook yields a full ICM iteration. The encoding can run for a fixed number of iterations or until convergence is achieved.

The major benefit of ICM is its mild computational costs – one iteration requires $O(MKD)$ operations in total. However, the algorithm is found to be unsuitable for AQ, as it was found to produce suboptimal encoding in most scenarios [15].

B. Beam Search

Beam search is proposed to be used for encoding by the authors of AQ in [15] to allow for better encoding by drastically expanding the search space. First, all codebooks are combined, resulting in a set of MK codewords. From this set H closest matches to the target vector are chosen (H is a search depth parameter). These are the initial solution candidates. Then H current residuals are computed, and H more codevectors *per residual* are chosen from yet unused codebooks. The resulting set of H^2 candidate solutions is sorted based on the quantization error, after which the top H are kept. The beam search continues until all M codebooks are utilized in the candidate solutions, at which point the single encoding with the smallest quantization error is retained as a final result. The authors of AQ recommend the values $H = 16$ during codebook learning and $H = 64$ for encoding the actual data.

The beam search encoding vastly outperformed ICM and allowed AQ to reach state-of-the-art ANN performance [15]. However, its prohibitive computational cost makes it undesirable for practical purposes, as encoding a single vector requires $O(M^3KH + MKD)$ operations. As the number of codebooks is directly related to the representation power, the cubic scaling of beam search with respect to M is also undesirable.

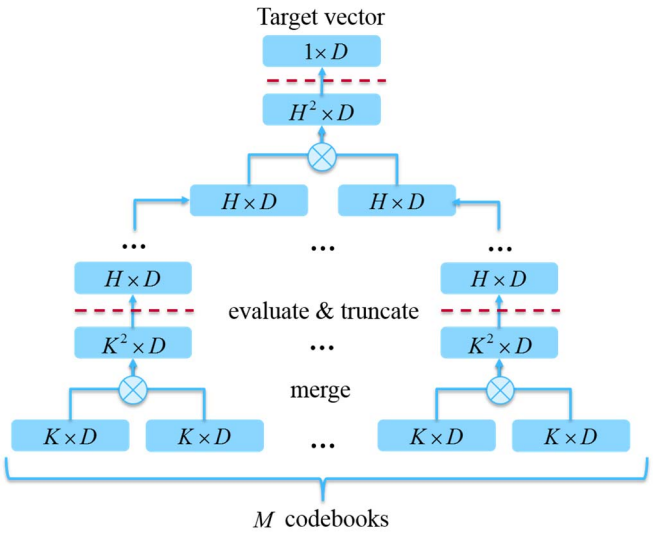

Fig. 1. The bottom-up approach as employed by pyramid encoding.

III. Pyramid Encoding

In this study, a novel encoding method called *Pyramid Encoding* is proposed for AQ encoding. It utilizes a bottom-up approach, consecutively merging the codebooks to obtain new candidate solutions in a tree structure. Fig. 1 shows the proposed scheme. The first merge results in K^2 candidate solutions on each of $M/2$ tree nodes. The contents of every node are independently evaluated and truncated to the best H solutions, similarly to beam search. The following merges naturally result in H^2 solutions in each new node, from which only the top H are retained. This procedure continues until the root of the tree is reached, meaning that all the codebooks are utilized and a single best solution is taken as a final answer.

The advantage of the pyramid encoding arises from the relative simplicity of error calculation when the solutions are merged. Assume that two additive approximations for the target vector x are available – C_1 and C_2, with known quantization errors $E_1 = \|x - C_1\|^2$ and $E_2 = \|x - C_2\|^2$ respectively. Combining these solutions will result in a new quantization error value: $E' = \|x - (C_1 + C_2)\|^2$. The expression for E' can be expanded as follows:

$$E' = \|x - (C_1 + C_2)\|^2 = \\ = \|x\|^2 + \|C_1 + C_2\|^2 - 2\langle x, C_1\rangle - 2\langle x, C_2\rangle \quad (1)$$

Applying the properties of the norms and dot products yields the following alternative formulation for E':

$$E' = \|x - C_1\|^2 + \|x - C_2\|^2 - \|x\|^2 + 2\langle C_1, C_2\rangle = \\ = E_1 + E_2 - \|x\|^2 + 2\langle C_1, C_2\rangle \quad (2)$$

E_1 and E_2 are known by assumption, while the norm of the target vector can be easily precomputed and reused as necessary.

ISBN 978-0-9928626-7-1 © EURASIP 2017

TABLE I
COMPUTATIONAL COMPLEXITY OF ENCODING A SINGLE POINT

Quantization method	Complexity
PQ	$O(KD)$
OPQ	$O(KD + D^2)$
AQ-B	$O(M^3KH + MKD)$
AQ-P	$O(M^2H^2 + MKD)$

All the terms of expression (2) are thus known, excluding the dot product. The fast computation of the latter can be achieved by constructing a lookup table with all the dot products between the different codebooks. As the contents of this table depend only on the quantizer and can be reused for any target vector x, the corresponding computational cost can be excluded from the total estimate. The computational complexity of pyramid encoding for a single target vector is thus $O(M^2H^2 + MKD)$. It is significantly faster than beam search, especially in scaling with respect to M and K. However, due to its tighter constraints, it can be expected to produce less accurate representations. Due to lower complexity, the value $H = 64$ is deemed acceptable for all scenarios.

As the proposed encoding approach searches a smaller region of the solution space, the initial point becomes more important. Two different approaches to pyramid encoding initialization are considered. *Random initialization* is equivalent to the one used for ICM and beam search. *Orthogonal initialization* runs several iterations of PQ on the data and uses the resulting set of codes and codebooks as an initial solution. In the latter case the codebooks start out as orthogonal sets of vectors by definition of PQ. However, as AQ formulation does not enforce this constraint, orthogonality is gradually relaxed during codebook adaptation.

Table I lists the encoding complexities for all methods discussed in this study. Additive quantization even with pyramid encoding is still significantly slower than PQ or OPQ; therefore, a significant search performance benefit is required to justify the approach. When orthogonal initialization is used, the suitable number of PQ iterations to run was experimentally found to be around 10–15. Thus the extra initialization costs do not affect the asymptotic complexity estimate.

IV. EXPERIMENTAL RESULTS

Two datasets of image descriptors – SIFT1M and GIST1M – are the de-facto standard for evaluating ANN search methods [7]. Both datasets are comprised of a *base set*, where the actual search is performed, a *holdout set*, which is used only to train the quantizers, and a *query set*, for which the ground truth (indices of 100 nearest samples from the base set) is provided. Table II contains the quantitative information about the datasets.

Two measures are used to evaluate performance. *Quantization error* is a natural indicator of compressed representation accuracy and was empirically found to be related to the ANN search performance [7]. We report the quantization error values on the base set (after training). *Recall@N* is a popular evaluation metric of ANN search, which gives an estimate of the probability that the true nearest neighbor will be present amongst the top N points returned by the search [7]. We report the experimental results for N = 1, 10, 100 and 1000 (the latter only on GIST1M).

The following baseline methods were used in the experiments: PQ, OPQ/CKM, AQ. All of them have publicly available implementations. For clarity, we denote the original variant of AQ that uses beam search encoding as AQ-B. The proposed solutions are denoted AQ-P and AQ-P(pq) for random and orthogonal initializations, respectively. Table III lists the parameters of the algorithms used during the experiments.

TABLE II
PROPERTIES OF THE DATASETS

	SIFT1M	GIST1M
Dimensionality	128	960
Base set size	1 000 000	1 000 000
Hold-out set size	100 000	500 000
Query set size	10 000	1 000

TABLE III
PARAMETERS OF THE QUANTIZERS

Parameter	Value
Number of codevectors per codebook K	256
Number of codebooks M	4 and 8
Representation codelength	32 bits for $M = 4$, 64 bits for $M = 8$
Search depth H	AQ-B – 16 (training), 64 (encoding) AQ-P – 64
Number of training iterations	30 for SIFT1M 10 for GIST1M
PQ iterations for AQ-P (pq)	15

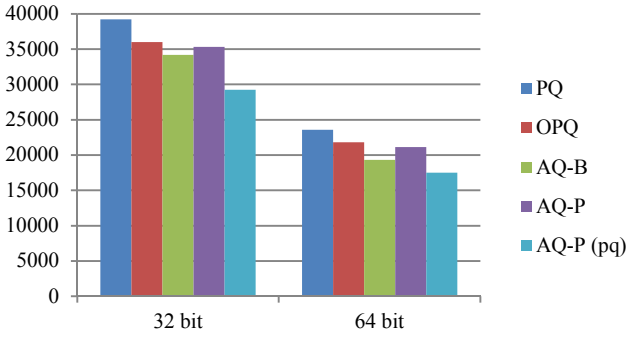

Fig. 2. Quantization error on SIFT1M base set.

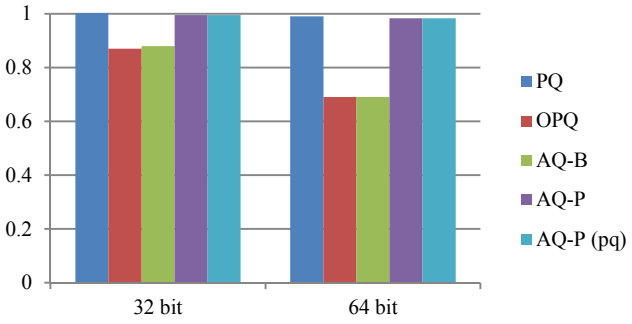

Fig. 3. Quantization error on GIST1M base set.

TABLE IV
RECALL ON SIFT1M DATASET

64 bits ($M = 8$)	R@1	R@10	R@100	32 bits ($M = 4$)	R@1	R@10	R@100
PQ	0,2243	0,599	0,9243	PQ	0,0518	0,2297	0,5945
OPQ	0,2433	0,6384	0,9402	OPQ	0,06756	0,2725	0,6576
AQ-B	0,3114	0,7733	0,983	AQ-B	0,1066	0,415	0,8255
AQ-P	0,2051	0,5808	0,9115	**AQ-P**	0,0560	0,2438	0,6248
AQ-P (pq)	0,2846	0,7125	0,9661	**AQ-P (pq)**	0,1177	0,3983	0,7941

TABLE V
RECALL ON GIST1M DATASET

64 bits (M = 8)	R@1	R@10	R@100	R@1000	32 bits (M = 4)	R@1	R@10	R@100	R@1000
PQ	0,076	0,218	0,504	0,8582	PQ	0,023	0,0675	0,1756	0,5045
OPQ	0,118	0,334	0,715	0,9465	OPQ	0,054	0,1419	0,3964	0,7905
AQ-B	–	–	–	–	AQ-B	0,069	0,189	0,4666	0,809
AQ-P	0,097	0,268	0,563	0,88	**AQ-P**	0,047	0,157	0,382	0,726
AQ-P (pq)	0,097	0,268	0,563	0,88	**AQ-P (pq)**	0,047	0,157	0,382	0,726

Fig. 2 and Fig. 3 show the quantization errors obtained after the base set is encoded with trained quantizers. While randomly initialized pyramid encoding scheme proves inferior to beam search, orthogonal initialization allows for significant improvement upon baseline AQ on SIFT1M with 32-bit codes. The same conclusion holds for longer codes on SIFT1M, although the advantage is not as pronounced. However, the proposed approach is not as justified on GIST1M (regardless of initialization), as it barely surpasses PQ. This is due to the different internal structure of GIST descriptors, which are significantly less suitable for subspace decomposition. Note that SIFT descriptors are composed of 16 histograms with 8 bins each; GIST1M samples, on the other hand, are concatenations of three 320-dimensional subvectors, representing normalized orientation histograms with varying number of bins [7]. Since the number of PQ-initialized codebooks is a power of two, the corresponding subspaces cannot align well with GIST descriptor structure. Adaptive rotation of OPQ and a wider search space of AQ with beam search encoding make the respective quantizers preferable in the latter case, avoiding the rigid dimension allocation. It is worth noting that the baseline AQ provides extremely minor advantage over OPQ on GIST1M.

Tables IV and V show the recall values for the aforementioned quantization-based ANN algorithms. It is apparent that the proposed approach is very close to the baseline AQ on SIFT1M, especially for shorter code lengths. GIST1M poses difficulties yet again, as simply providing a significant improvement upon OPQ is a challenge. The proposed approach demonstrates subpar performance here, converging to the same solution regardless of initialization. The optimization process ultimately proves to be too local to improve upon the poor starting point given by PQ.

V. CONCLUSION

The proposed pyramid encoding for additive quantization provides a significant complexity reduction compared to the standard beam search, resulting in a faster data processing throughout the ANN search pipeline. The effects of pyramid encoding on the search accuracy depend heavily on the internal structure of the data. If the PQ decomposition is aligned with semantic subspaces of the data, orthogonal initialization ensures highly competitive performance. Otherwise, the pyramid encoding fails to improve upon the initial guess, resulting in a poor solution. Adaptive rotation can be incorporated on different levels of the encoding pyramid to address this limitation, which is the subject of ongoing work.

ACKNOWLEDGMENT

The authors would like to acknowledge the financial support of the Academy of Finland, project no. 289364.

Alexandros Iosifidis was supported from the Academy of Finland Postdoctoral Research Fellowship (No. 295854). He joined Aarhus University on August 2017.

REFERENCES

[1] M. Chen, S. Mao, and Y. Liu, "Big data: A survey," *Mob. Networks Appl.*, vol. 19, no. 2, pp. 171–209, 2014.

[2] R. Weber, H. J. Schek, and S. Blott, "A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces," *Proc. 24th VLDB Conf.*, vol. New York C, pp. 194–205, 1998.

[3] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proceedings of the twentieth annual symposium on Computational geometry*, 2004, pp. 253–262.

[4] Y. Jing and S. Baluja, "Visualrank: Applying pagerank to large-scale image search," *Pattern Anal. Mach. Intell. IEEE Trans.*, vol. 30, no. 11, pp. 1877–1890, 2008.

[5] J. Wang, H. T. Shen, J. Song, and J. Ji, "Hashing for Similarity Search: A Survey," *arXiv:1408.2927*, pp. 1–29, 2014.

[6] E. C. Ozan, S. Kiranyaz, and M. Gabbouj, "M-PCA Binary Embedding for Approximate Nearest Neighbor Search," in *2015 IEEE Trustcom/BigDataSE/ISPA*, 2015, vol. 2, pp. 1–5.

[7] M. Douze, C. Schmid, 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.

[8] Y. Chen, T. Guan, and C. Wang, "Approximate nearest neighbor search by residual vector quantization," *Sensors*, vol. 10, no. 12, pp. 11259–11273, 2010.

[9] L. Ai, J. Yu, Z. Wu, Y. He, and T. Guan, "Optimized residual vector quantization for efficient approximate nearest neighbor search," *Multimed. Syst.*, pp. 1–13, 2015.

[10] R. M. Gray, "Vector quantization," *ASSP Mag. IEEE*, vol. 1, no. 2, pp. 4–29, 1984.

[11] T. Ge, K. He, Q. Ke, and J. Sun, "Optimized product quantization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 4, pp. 744–755, 2014.

[12] M. Norouzi and D. J. Fleet, "Cartesian K-Means," *2013 IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 3017–3024, 2013.

[13] Y. Kalantidis and Y. Avrithis, "Locally optimized product quantization for approximate nearest neighbor search," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2321–2328.

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

[15] A. Babenko and V. Lempitsky, "Additive quantization for extreme vector compression," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 931–938, 2014.

[16] A. Babenko and V. Lempitsky, "Tree Quantization for Large-Scale Similarity Search and Classification," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[17] T. Zhang, J. Wang, and J. M. Com, "Composite Quantization for Approximate Nearest Neighbor Search," *Proc. 31st Int. Conf. Mach. Learn.*, vol. 32, pp. 838–846, 2014.

[18] T. Zhang, G. J. Qi, J. Tang, and J. Wang, "Sparse composite quantization," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 07–12–June, pp. 4548–4556.

[19] G. F. Cooper, "The computational complexity of probabilistic inference using Bayesian belief networks," *Artif. Intell.*, vol. 42, no. 2, pp. 393–405, 1990.

ISBN 978-0-9928626-7-1 © EURASIP 2017