PCA GUIDED FAST RESIDUAL VECTOR QUANTIZATION FOR PROCESSING LARGE DATASETS

Salman Aslam

National University of Sciences and Technology, Islamabad, Pakistan
salman@gatech.edu

ABSTRACT

Residual Vector Quantization (RVQ) is a VQ method that has not gained much attention in the literature, first due to the difficulty of producing stable multi-stage designs, and second due to the computational complexity involved. Whereas the first problem has largely been solved, the second problem still limits wide usage of RVQ for processing large datasets. In this work, we introduce a novel method of implementing RVQ. This PCA guided method can be justified mathematically, results in simplifying the design process and greatly reduces the computational complexity required for RVQ design. As a result, large datasets can be efficiently compressed and retrieved in reasonable time.

Index Terms— Residual vector quantization, RVQ, PCA

1. INTRODUCTION

RVQ is a powerful compression method first introduced by Juang et al. [1]. The computational costs and memory requirements of regular Vector Quantization (VQ) are exponential in the data while both these requirements are linear for RVQ [2]. This feature makes RVQ attractive for processing large datasets. In regular VQ, if 64 centroids are used, then intelligently using 64 centroids in an RVQ tree by placing them in 8 stages with 8 centroids per stage can lead to a total of $8^8 = 16,777,216$ terminal centroids! The challenge of course is how to compute those 64 centroids. The design procedure for multi-stage RVQ was developed by Barnes over several years [3]. Since then, RVQ has been used for image coding [4] and more recently for image classification [5, 6], target tracking [7] and for computing classification error bounds [8].

2. QUANTIZATION AND RVQ

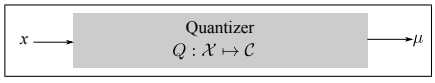


Fig. 1. A quantizer $\mathcal{Q}$ maps symbols from a source alphabet $\mathcal{X}$ to symbols from a reconstruction alphabet $\mathcal{C}$, where in general, the number of elements in $\mathcal{X}$, $N >> K$, the number of elements in $\mathcal{C}$.

Quantization is the process of representing a large, possibly infinite, set of values with a smaller set of values [9]. Figure 1 shows a quantizer $\mathcal{Q}$ that takes values from a source alphabet $\mathcal{X} = \{\mathbf{x} \in \mathbb{R}^D\}$ and maps them to a reconstruction alphabet $\mathcal{C} = \{\mathbf{y}_k \in \mathbb{R}^D \mid k = 1, 2, \ldots K\}$. If the input is scalar, i.e. $D = 1$, the quantizer is called a *scalar quantizer*. For $D > 1$, the quantizer is called a *vector quantizer*.

The reconstruction alphabet $\mathcal{C}$ is known as the *codebook* while the K members $\mathbf{y}_k$ of the reconstruction alphabet $\mathcal{C}$ are called *code-vectors*. The term *centroid* is used interchangeably with code-vector. If we have K code-vectors $\mathbf{y}_k$ in $\mathbb{R}^D$, $\log_2 K$ bits are required to represent each code-vector. The *resolution*, *code rate*, or simply the *rate* r of a quantizer is the number of bits required to represent each sample, i.e., scalar element of $\mathbf{y}_k$. Since there are D samples, the rate $r = \frac{\log_2 K}{D}$.

Figure 2 illustrates the scalar quantization case for an input X with distribution $f_X(x)$ and average distortion e,

$$e = \sum_{k=1}^{K} \int_{b_k}^{b_{k+1}} (x - y_k)^2 f_X(x) dx \tag{1}$$

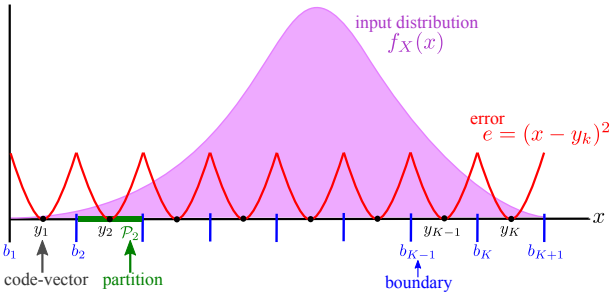


Fig. 2. Given partition $\mathcal{P}_k$, the optimal code-vector for this partition is the centroid of the partition. Note that the reconstruction error increases exponentially as we move away from the centroids.

In general, optimal coding of source vectors is not possible unless an exhaustive search over all code-vectors is carried out, as in structurally unconstrained *Exhaustive Search Vector Quantizers* (ESVQs) [2]. For a rate r and dimension D, there are $K = 2^{rD}$ code-vectors. Therefore, the computational cost of ESVQ, C_{ESVQ}, and memory requirements M_{ESVQ} are $\approx 2^{rD}$. A solution to this problem is to impose constraints on the VQ structure. One possible solution is the tree structured vector quantizer (TSVQ) proposed in [10]. A P-level binary TSVQ has run-time search complexity which is only $C_{TSVQ} \approx 2P$ but double storage requirements, $M_{TSVQ} \approx 2M_{ESVQ}$ [3]. So, although TVSQ solves the search complexity problem, it further aggravates the storage problem. A method of reducing both run-time computational and storage complexity is to use a product code VQ [11]. The basic idea in a product code VQ is to break a bigger problem into several smaller problems. Examples include mean-residual VQ, gain-shape VQ, mean-gain-shape VQ and RVQ [3]. See [12] for a comparison of ESVQ, TSVQ and RVQ.

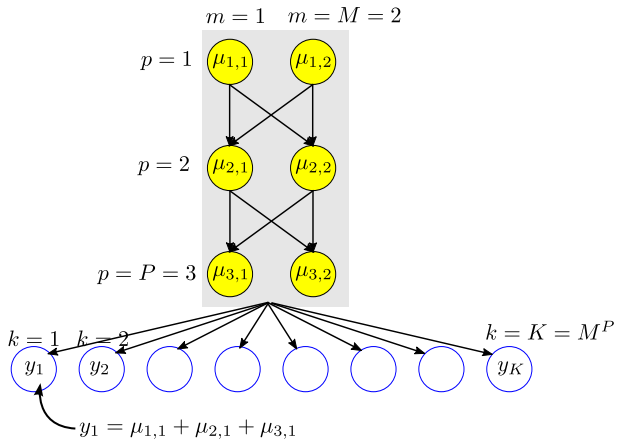


Fig. 3. 3x2 RVQ tree, 3 stages, 2 code-vectors per stage, i.e., P=3, $M = 2$

As with ESVQ and TSVQ, the K-means, or GLA, objective function to be minimized for RVQ for the discrete case can be written as,

$$e = \sum_{k=1}^{K} \sum_{\substack{i=1 \\ x_i \in \mathcal{C}_k}}^{N} (x_i - y_k)^2 \tag{2}$$

For RVQ, the k-th terminal code-vector is a direct sum of P stage code-vectors,

$$y_k = \mu_1^{(k)} + \mu_2^{(k)} + \ldots + \mu_P^{(k)} \tag{3}$$

Substituting this notation in Equation 2 and grouping all stage code-vectors except for the stage code-vector at the ρ-th stage gives us a series of equivalent equations, one equation per stage,

$$\begin{aligned}
e &= \sum_{k=1}^{K} \sum_{\substack{i=1 \\ x_i \in \mathcal{C}_k}}^{N} \left[x_i - \left(\sum_{p=2}^{P} \mu_p^{(k)} + \mu_\rho^{(k)} \right) \right]^2, \quad \rho = 1 \\
&= \sum_{k=1}^{K} \sum_{\substack{i=1 \\ x_i \in \mathcal{C}_k}}^{N} \left[x_i - \left(\sum_{\substack{p=1 \\ p \neq 2}}^{P} \mu_p^{(k)} + \mu_\rho^{(k)} \right) \right]^2, \quad \rho = 2 \\
&\vdots \\
&= \sum_{k=1}^{K} \sum_{\substack{i=1 \\ x_i \in \mathcal{C}_k}}^{N} \left[x_i - \left(\sum_{p=1}^{P-1} \mu_p^{(k)} + \mu_\rho^{(k)} \right) \right]^2, \quad \rho = P
\end{aligned} \tag{4}$$

Equation 4 can be regrouped and written in compact notation as,

$$\begin{aligned}
e &= \sum_{k=1}^{K} \sum_{\substack{i=1 \\ x_i \in \mathcal{C}_k}}^{N} \left[\left(x_i - \sum_{\substack{p=1 \\ p \neq \rho}}^{P} \mu_p^{(k)} \right) - \mu_\rho^{(k)} \right]^2, \quad \rho = \{1, 2, \ldots P\} \\
&= \sum_{k=1}^{K} \sum_{\substack{i=1 \\ g_i \in \mathcal{H}_k}}^{N} (g_i - \mu_\rho^{(k)})^2, \quad \rho = \{1, 2, \ldots P\}
\end{aligned} \tag{5}$$

where g_i is the *graft residual* [13]. As can be seen in Equation 5, the graft residual g_i for a data-point x_i is formed by subtracting from x_i, all stage codevectors that are used to reconstruct x_i except the stage codevector at the ρ-th stage. In this sense, g_i is a causal anti-causal (CAC) residual [13]. The code-vectors at the ρ-th stage are computed using the K-means objective function for that particular stage. The implication of this step is that the RVQ objective function is now a coupled K-means objective function where the design of each stage code-vector depends on stage code-vectors from all other stages, and not just prior stages, hence the name causal anti-causal. A challenge in this coupled K-means setup is that computing the centroids for one stage changes the residual centroids for all other stages.

training data: 4, 6, 8, 10, 20, 22, 24, 26

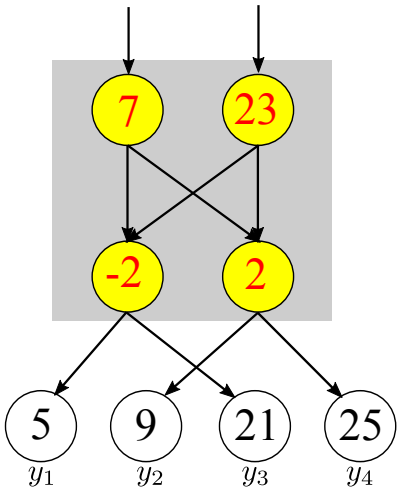


Fig. 4. An example 2x2 RVQ tree. In this case, 2 code-vectors in 2 stages, i.e. a total of 4 stage code-vectors lead to $2^2 = 4$ terminal code-vectors.

An RVQ is different from a traditional VQ in the sense that it partitions the input space $\mathbb{R}^D$ into M cells. The residual space, also in $\mathbb{R}^D$, is then partitioned again into M cells. This process is repeated P times. The advantage of this approach is that in obtaining M^P partitions, we need to run our partitioning algorithm P times and generate M partitions at each stage. In traditional VQ, the partitioning algorithm would run once but have to create M^P partitions. For the binary case (two code-vectors per stage, $M = 2$) and a total of 8 stages (P=8), RVQ only requires 16 searches. In $ESVQ$, this would require 256. The exponential complexity is reduced to linear complexity. In general, structurally constrained quantizers cannot provide performance as good as ESVQ. However, since they are able to more efficiently implement codes, larger and larger vector sizes can be used, and if carefully designed, can achieve better performance that ESVQ for a given computational cost [3]. The general form for an RVQ tree is given in Figure 3 while an example RVQ tree is given in Figure 4. In Figure 5 RVQ reconstruction procedure at run-time is shown while Figure 6 displays the nature of an RVQ tree when used with images.

3. PCA GUIDED RVQ

Having taken care of the preliminaries, we are interested in the relationship between PCA and RVQ. The reason is both academic and practical. From an academic point of view, it is inherently satisfying

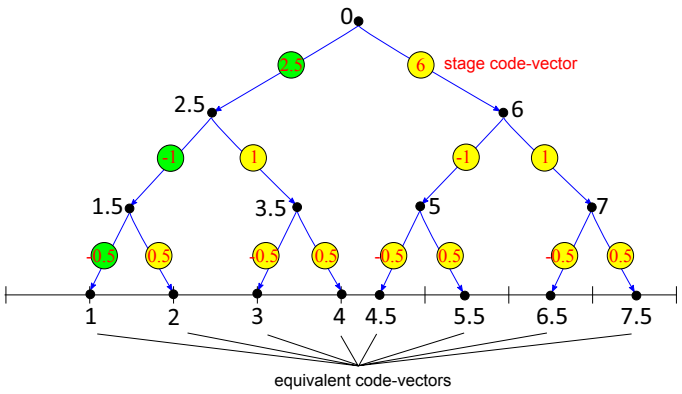
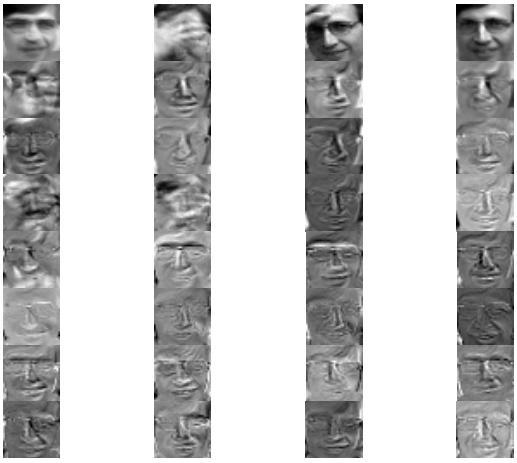


Fig. 5. In this example, the training data is $\{1, 2, ... 7\}$ from which an RVQ tree is generated with 2.5 and 6 as first stage centroids, -1 and 1 as second stage centroids and -0.5 and 0.5 as third stage centroids. At run time, the method shown is used to reconstruct the data points.

(a) Snapshots from image sequence taken from [14].



(b) An 8x4 RVQ tree created from the dataset given above.

Fig. 6. In this example, we show what an RVQ tree looks like for 2D data.

to be able to relate a less known algorithm with a widely known one. Additionally, many existing results may become relevant for the new algorithm. In this context, the work by Ding and He [15] of relating PCA to the K-means algorithm is particularly relevant since we have already shown that RVQ is based on coupled K-means. From a practical standpoint, relating RVQ to PCA greatly simplifies the the complex and heuristic design process of RVQ and leads to great reductions in computational complexity.

In this context, we are interested in the following two questions:

1. PCA generates the centroid cluster subspace S while RVQ generates stage subspaces, $S_1, S_2, \ldots, S_P$, one stage subspace for each of the P stages. What is the relationship between the subspace generated by PCA and the subspaces generated by RVQ?
2. PCA generates eigenvectors of the data while RVQ generates stage centroids. What is the relationship between the eigenvectors and the stage centroids?

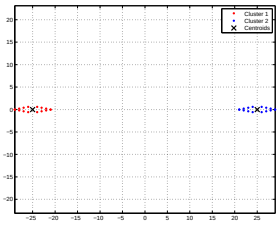
In order to answer the first question, we note that according to Ding and He [15], if we ran the K-means algorithm on the N input data points $x_i \in R^D, K < D, i = 1, 2, \ldots N$, the resulting K equivalent centroids, y_k, would lie in cluster centroid subspace S with dimensionality $K - 1$. Now, if we ran RVQ on this input data, the stage centroids $\mu_{m,p}$ would add up in $K = M^P$ different combinations to make the equivalent centroids y_k. Quite clearly, the equivalent centroids y_k are linear combinations of the stage centroids. Therefore, the stage centroids must also lie in S. Furthermore, the M stage centroids at a particular stage lie in a subspace whose dimensionality is at most $M - 1$. Therefore, stage centroids in the first stage, i.e, $p = 1$ lie in $M - 1$ dimensional stage subspace S_1, stage centroids in the second stage, i.e, $p = 2$ also lie in $M - 1$ dimensional stage subspace S_2, and so on till P stages. Therefore, $S_1, S_2, \ldots, S_P \subset S$.

In order to answer the second question, we again turn to Ding and He in that PCA dimension reduction automatically performs data clustering according to the K-means objective function [15]. For PCA, data reconstruction error is minimized if the data is represented in the $K - 1$ lower dimensional subspace as projections onto the principal directions $\mathbf{u}_k$ where $YY^T\mathbf{u}_k = \lambda_k\mathbf{u}_k$. y_i is the zero-mean version of the input data x_i, Y is the zero-centered input matrix formed from the data y_i and λ's are the eigenvalues. If K-means clustering is to be carried out in the PCA subspace as suggested by [15] , then RVQ stage centroids placed in this subspace must be placed so that they account for maximum variance in the data. This will happen if they are placed on the eigenvectors generated by PCA. A vector sum of these stage centroids will then account for maximum variance in the data. If the data residual from one stage still has more variance than the data's variance in an orthogonal direction, then repeated application of this rule will place subsequent stage centroids along the same eigenvectors until the residual data's variance falls below the variance in an orthogonal dimension. In summary, stage centroids from one stage to another will either lie along the same eigenvector, or along an orthogonal eigenvector. This is illustrated using a simple example in Figure 7.

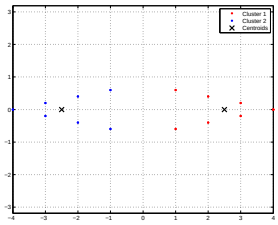
In this example, the input data consists of two elliptically shaped clusters whose variance is greatest along the x-axis. Optimal fist stage centroids are placed on the prinicipal direction, i.e., the x-axis. The variance of the residual data, i.e, input data from which the corresponding centroids have been subtraced also has maximum variance along the x-axis. Second stage optimal centroids are also therefore placed along this dimension. This process repeats for the third stage as well and once again, optimal third stage centroids lie along the x-axis. Finally, in the fourth stage, the variance along the orthogonal direction, the second PCA eigenvector, i.e, y-axis is now greater than along the x-axis and therefore optimal fourth stage centroids now lie along the y-axis.

4. CONCLUSIONS

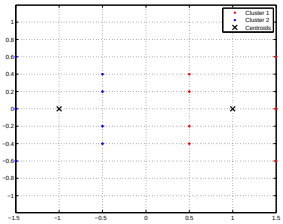
RVQ holds tremendous potential for representing large datasets in compact notation. For instance, an 8x8 RVQ with 8 stages and 8 stage centroids per stage for a total of 64 centroids actually has $8^8 = 16,777,216$ terminal centroids. Despite the large compression ratios possible, RVQ has not found widespread usage. The reason is the difficulty in its design. In this work, we have attempted



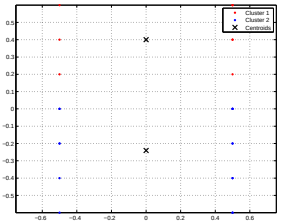
(a) Stage 1. The two first stage centroids lie on the principal eigenvector, x-axis in this case, accounting for maximum variance.



(b) Stage 2. The data points here are the residuals from stage 1. Notice that the variance along the principal eigenvector is still more than the variance in the orthogonal direction. Therefore, the stage centroids of the second stage also lie along the principal eigenvector.



(c) Stage 3. As in stage 2, the variance along the principal direction is still greater than the variance in the orthogonal direction, and therefore once again, the stage centroids lie on the principal eigenvector.



(d) Stage 4. Finally, the variance in the principal direction is less than in its orthogonal direction, and therefore the stage centroids at this stage are aligned along the orthogonal direction.

Fig. 7. PCA guided RVQ. A simple example in $\mathbb{R}^2$. Each data point given in the top left figure is reconstructed using a sum of stage centroids, one from each stage. Notice the decreasing variance of the residual data as each stage progresses (evident from less spread along the x-axis).

to extend the relationship between PCA and K-means to PCA and RVQ resulting in considerably lower design complexity. We have also provided a theoretical justification for this design process. Currently, we are evaluating this design methodology on larger datasets.

5. REFERENCES

[1] Juang Biing-Hwang and Jr. Gray, A., "Multiple stage vector quantization for speech coding," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '82.*, 1982.

[2] C.F. Barnes and R. Frost, "Residual vector quantizers with jointly optimized code books," *Advances in Electronics and Electron Physics*, vol. 84, pp. 1–59, 1992.

[3] C. F. Barnes, S. A. Rizvi, and N. M. Nasrabadi, "Advances in residual vector quantization: a review," *Image Processing, IEEE Transactions on*, vol. 5, no. 2, pp. 226–262, 1996.

[4] F. Kossentini, M.J.T. Smith, and C.F. Barnes, "Image coding with variable rate rvq," in *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., IEEE International Conference on*, 1992.

[5] C.F. Barnes, H. Fritz, and Jeseon Yoo, "Hurricane disaster assessments with image-driven data mining in high-resolution satellite imagery," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 45, no. 6, pp. 1631 –1640, june 2007.

[6] C. F. Barnes, "Image-driven data mining for image content segmentation, classification, and attribution," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, pp. 2964–2978, 2007.

[7] Salman Aslam, Christopher Barnes, and Aaron Bobick, "Target tracking using residual vector quantization," in *Digital Image Computing Techniques and Applications (DICTA), 2012 International Conference on*. IEEE, 2012, pp. 1–8.

[8] Irteza Ali Khan, *Classification Using Residual Vector Quantization*, Ph.D. thesis, Georgia Institute of Technology, 2013.

[9] Khalid Sayood, *Introduction to Data Compression*, Morgan Kaufmann, 3 edition, 2005.

[10] A. Buzo, Jr. Gray, A., R. Gray, and J. Markel, "Speech coding based upon vector quantization," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 28, no. 5, pp. 562 – 574, oct 1980.

[11] Allen Gersho and Robert Gray, *Vector Quantization and Signal Compression (The Springer International Series in Engineering and Computer Science)*, Springer, 1991.

[12] Salman Aslam, *Target Tracking Using RVQ*, Ph.D. thesis, Georgia Institute of Technology, 2011.

[13] C. F. Barnes and R. L. Frost, "Vector quantizers with direct sum codebooks," *IEEE Transactions on Information Theory*, vol. 39, no. Copyright 1993, IEE, pp. 565–80, 1993.

[14] David Ross, Jongwoo Lim, Ruei-Sung Lin, and Ming-Hsuan Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, vol. 77, pp. 125–141, 2008.

[15] Chris Ding and Xiaofeng He, "K-means clustering via principal component analysis," in *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, New York, NY, USA, 2004, p. 29, ACM.