WIKIPEDIA
The Free Encyclopedia

Vector quantization

Vector quantization (**VQ**) is a classical quantization technique from signal processing that allows the modeling of probability density functions by the distribution of prototype vectors. It was originally used for data compression. It works by dividing a large set of points (vectors) into groups having approximately the same number of points closest to them. Each group is represented by its centroid point, as in k-means and some other clustering algorithms.

The density matching property of vector quantization is powerful, especially for identifying the density of large and high-dimensional data. Since data points are represented by the index of their closest centroid, commonly occurring data have low error, and rare data high error. This is why VQ is suitable for lossy data compression. It can also be used for lossy data correction and density estimation.

Vector quantization is based on the competitive learning paradigm, so it is closely related to the self-organizing map model and to sparse coding models used in deep learning algorithms such as autoencoder.

Training

The simplest training algorithm for vector quantization is:[1]

1. Pick a sample point at random
2. Move the nearest quantization vector centroid towards this sample point, by a small fraction of the distance
3. Repeat

A more sophisticated algorithm reduces the bias in the density matching estimation, and ensures that all points are used, by including an extra sensitivity parameter:

1. Increase each centroid's sensitivity s_i by a small amount
2. Pick a sample point P at random
3. For each quantization vector centroid c_i, let $d(P, c_i)$ denote the distance of P and c_i
4. Find the centroid c_i for which $d(P, c_i) - s_i$ is the smallest
5. Move c_i towards P by a small fraction of the distance
6. Set s_i to zero
7. Repeat

It is desirable to use a cooling schedule to produce convergence: see Simulated annealing. Another (simpler) method is LBG which is based on K-Means.

The algorithm can be iteratively updated with 'live' data, rather than by picking random points from a data set, but this will introduce some bias if the data are temporally correlated over many samples.

Applications

Vector quantization is used for lossy data compression, lossy data correction, pattern recognition, density estimation and clustering.

Lossy data correction, or prediction, is used to recover data missing from some dimensions. It is done by finding the nearest group with the data dimensions available, then predicting the result based on the values for the missing dimensions, assuming that they will have the same value as the group's centroid.

For density estimation, the area/volume that is closer to a particular centroid than to any other is inversely proportional to the density (due to the density matching property of the algorithm).

Use in data compression

Vector quantization, also called "block quantization" or "pattern matching quantization" is often used in lossy data compression. It works by encoding values from a multidimensional vector space into a finite set of values from a discrete subspace of lower dimension. A lower-space vector requires less storage space, so the data is compressed. Due to the density matching property of vector quantization, the compressed data has errors that are inversely proportional to density.

The transformation is usually done by projection or by using a codebook. In some cases, a codebook can be also used to entropy code the discrete value in the same step, by generating a prefix coded variable-length encoded value as its output.

The set of discrete amplitude levels is quantized jointly rather than each sample being quantized separately. Consider a k-dimensional vector $[x_1, x_2, \ldots, x_k]$ of amplitude levels. It is compressed by choosing the nearest matching vector from a set of n-dimensional vectors $[y_1, y_2, \ldots, y_n]$, with $n < k$.

All possible combinations of the n-dimensional vector $[y_1, y_2, \ldots, y_n]$ form the vector space to which all the quantized vectors belong.

Only the index of the codeword in the codebook is sent instead of the quantized values. This conserves space and achieves more compression.

Twin vector quantization (VQF) is part of the MPEG-4 standard dealing with time domain weighted interleaved vector quantization.

Video codecs based on vector quantization

- Bink video[2]
- Cinepak
- Daala is transform-based but uses pyramid vector quantization on transformed coefficients[3]
- Digital Video Interactive: Production-Level Video and Real-Time Video
- Indeo
- Microsoft Video 1
- QuickTime: Apple Video (RPZA) and Graphics Codec (SMC)

- Sorenson SVQ1 and SVQ3
- Smacker video
- VQA format, used in many games

The usage of video codecs based on vector quantization has declined significantly in favor of those based on motion compensated prediction combined with transform coding, e.g. those defined in MPEG standards, as the low decoding complexity of vector quantization has become less relevant.

Audio codecs based on vector quantization

- AMR-WB+
- CELP
- Codec 2
- DTS
- G.729
- iLBC
- Ogg Vorbis[4]
- Opus is transform-based but uses pyramid vector quantization on transformed coefficients
- TwinVQ

Use in pattern recognition

VQ was also used in the eighties for speech[5] and speaker recognition.[6] Recently it has also been used for efficient nearest neighbor search [7] and on-line signature recognition.[8] In pattern recognition applications, one codebook is constructed for each class (each class being a user in biometric applications) using acoustic vectors of this user. In the testing phase the quantization distortion of a testing signal is worked out with the whole set of codebooks obtained in the training phase. The codebook that provides the smallest vector quantization distortion indicates the identified user.

The main advantage of VQ in pattern recognition is its low computational burden when compared with other techniques such as dynamic time warping (DTW) and hidden Markov model (HMM). The main drawback when compared to DTW and HMM is that it does not take into account the temporal evolution of the signals (speech, signature, etc.) because all the vectors are mixed up. In order to overcome this problem a multi-section codebook approach has been proposed.[9] The multi-section approach consists of modelling the signal with several sections (for instance, one codebook for the initial part, another one for the center and a last codebook for the ending part).

Use as clustering algorithm

As VQ is seeking for centroids as density points of nearby lying samples, it can be also directly used as a prototype-based clustering method: each centroid is then associated with one prototype. By aiming to minimize the expected squared quantization error[10] and introducing a decreasing learning gain fulfilling the Robbins-Monro conditions, multiple iterations over the whole data set with a concrete but fixed number of prototypes converges to the solution of k-means clustering algorithm in an incremental manner.

Generative Adversarial Networks (GAN)

VQ has been used to quantize a feature representation layer in the discriminator of Generative adversarial networks. The feature quantization (FQ) technique performs implicit feature matching.[11] It improves the GAN training, and yields an improved performance on a variety of popular GAN models: BigGAN for image generation, StyleGAN for face synthesis, and U-GAT-IT for unsupervised image-to-image translation.

See also

- Speech coding
- Ogg Vorbis
- Voronoi diagram
- Rate-distortion function
- Data clustering
- Learning vector quantization
- Centroidal Voronoi tessellation
- Growing Neural Gas, a neural network-like system for vector quantization
- Image segmentation
- Lloyd's algorithm
- Linde,Buzo,Gray Algorithm (LBG)
- K-means clustering
- Autoencoder
- Deep Learning

Part of this article was originally based on material from the Free On-line Dictionary of Computing and is used with permission under the GFDL.

References

1. Dana H. Ballard (2000). *An Introduction to Natural Computation*. MIT Press. p. 189. ISBN 978-0-262-02420-4.
2. "Bink video" (http://lists.mplayerhq.hu/pipermail/bow/2009-December/000058.html). *Book of Wisdom*. 2009-12-27. Retrieved 2013-03-16.
3. Valin, JM. (October 2012). *Pyramid Vector Quantization for Video Coding* (http://tools.ietf.org/html/draft-valin-videocodec-pvq-00). IETF. I-D draft-valin-videocodec-pvq-00. Retrieved 2013-12-17.
4. "Vorbis I Specification" (http://xiph.org/vorbis/doc/Vorbis_I_spec.html). Xiph.org. 2007-03-09. Retrieved 2007-03-09.
5. Burton, D. K.; Shore, J. E.; Buck, J. T. (1983). "A generalization of isolated word recognition using vector quantization". *IEEE International Conference on Acoustics Speech and Signal Processing ICASSP*. **8**: 1021–1024. doi:10.1109/ICASSP.1983.1171915 (https://doi.org/10.1109%2FICASSP.1983.1171915).
6. Soong, F.; A. Rosenberg; L. Rabiner; B. Juang (1985). "A vector Quantization approach to Speaker Recognition" (https://www.semanticscholar.org/paper/9e1d50d98ae09c15354dbcb126609e337d3dc6fb). *IEEE Proceedings International Conference on Acoustics, Speech and Signal Processing ICASSP*. **1**: 387–390. doi:10.1109/ICASSP.1985.1168412 (https://doi.org/10.1109%2FICASSP.1985.1168412). S2CID 8970593 (https://api.semanticscholar.org/CorpusID:8970593).

7. H. Jegou; M. Douze; C. Schmid (2011). "Product Quantization for Nearest Neighbor Search" (http://hal.archives-ouvertes.fr/docs/00/51/44/62/PDF/paper_hal.pdf) (PDF). *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **33** (1): 117–128. CiteSeerX 10.1.1.470.8573 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.470.8573). doi:10.1109/TPAMI.2010.57 (https://doi.org/10.1109%2FTPAMI.2010.57). PMID 21088323 (https://pubmed.ncbi.nlm.nih.gov/21088323). S2CID 5850884 (https://api.semanticscholar.org/CorpusID:5850884). Archived (https://web.archive.org/web/20111217142048/http://hal.archives-ouvertes.fr/docs/00/51/44/62/PDF/paper_hal.pdf) (PDF) from the original on 2011-12-17.
8. Faundez-Zanuy, Marcos (2007). "offline and On-line signature recognition based on VQ-DTW". *Pattern Recognition*. **40** (3): 981–992. doi:10.1016/j.patcog.2006.06.007 (https://doi.org/10.1016%2Fj.patcog.2006.06.007).
9. Faundez-Zanuy, Marcos; Juan Manuel Pascual-Gaspar (2011). "Efficient On-line signature recognition based on Multi-section VQ" (https://www.semanticscholar.org/paper/acf19e33b76ca5520e85e5c1be54c9920aa590b1). *Pattern Analysis and Applications*. **14** (1): 37–45. doi:10.1007/s10044-010-0176-8 (https://doi.org/10.1007%2Fs10044-010-0176-8). S2CID 24868914 (https://api.semanticscholar.org/CorpusID:24868914).
10. Gray, R.M. (1984). "Vector Quantization". *IEEE ASSP Magazine*. **1** (2): 4–29. doi:10.1109/massp.1984.1162229 (https://doi.org/10.1109%2Fmassp.1984.1162229).
11. Feature Quantization Improves GAN Training https://arxiv.org/abs/2004.02088

External links

- http://www.data-compression.com/vq.html
- QccPack — Quantization, Compression, and Coding Library (open source) (http://qccpack.sourceforge.net)
- VQ Indexes Compression and Information Hiding Using Hybrid Lossless Index Coding (https://dl.acm.org/citation.cfm?id=1535126), Wen-Jan Chen and Wen-Tsung Huang