Optimized Product Quantization for Approximate Nearest Neighbor Search

Tiezheng Ge, Kaiming He, Qifa Ke, Jian Sun (CVPR 2013)

(+ Cartesian k-means)

Mohammad Norouzi, David J. Fleet (CVPR 2013)

Presented by Relja Arandjelović
Objective

- Memory-efficient approximate nearest neighbour search
- Improve quality of descriptor compression

Specifically:
- Improve Product Quantization (Jegou et. al. 2011)
Related work: hashing

- Hashing for descriptor compression
- Quantize descriptors to binary codes of length $l$ (i.e. $k=2^l$ different codes)
- Find the Euclidean nearest neighbours (NNs) by finding closest binary codes using Hamming distance (very fast)
Locality sensitive hashing (LSH)

- $l$ random projections with random thresholds to form the codes
Locality sensitive hashing (LSH)

- Completely blind, not looking at the data at all
Iterative quantization (ITQ)

- Use PCA for dimensionality reduction (#PCs= $I$)
- Use orthogonal projections:
  - Same as quantizing to the unit hypercube
Iterative quantization (ITQ)

- Pick the best rotation of the data (or of the hypercube) to minimize quantization errors
Instead of restricting to a hypercube, pick best $k (=2^l)$ locations in the space to quantize to – i.e. k-means.
Vector quantization for compression

- **Pros:**
  - Much better fit to the underlying data => lower quantization errors
  - Not restricted by input dimensionality

- **Cons:**
  - Impossible to use for medium to large code lengths \((l)\): \(k=2^l\) so using 64-bits per descriptor would require a \(18 \times 10^{18}\) size codebook:
    - Impossible to learn
    - Impossible to store
    - Too slow to compute a code for a descriptor
  - Can’t use Hamming distance as clusters are arbitrarily assigned binary codes, but this is not important
Product quantization

- Divide a descriptor into $m$ blocks, vector quantize each block independently using $b$ bits ($l=mb$)
- Each block is quantized to $2^b$ centres
- Effectively equivalent to performing vector quantization to $(2^b)^m = 2^{mb} = 2^l$ centres, where the centres are the Cartesian product of the block-wise centres
Product quantization

- Large “product” vocabularies can be easily constructed
- E.g. For 64-bit coding of SIFT descriptors:
  - Divide the 128-D descriptors into m=8 blocks, 16-D each
  - Train a vocabulary of size $2^8=256$ for each 16-D block
  - Each 16-D block is described with a 8-bit code, so 8 blocks * 8 bits = 64 bits
  - Storage requirements for cluster centres: $8 \times (256 \times 16) = 0.1$ MB
- Fast to compute a code for a descriptor: perform VQ on each block independently
- Works very well in practice
Product quantization: problems

- The blocks are assumed to be independent
- The blocks can have unbalanced energies (variance)
Product quantization: problems

- The choice of blocks is important:

<table>
<thead>
<tr>
<th>Order Type</th>
<th>SIFT</th>
<th>GIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Natural order</td>
<td>0.921</td>
<td>0.338</td>
</tr>
<tr>
<td>Random order</td>
<td>0.859</td>
<td>0.285</td>
</tr>
<tr>
<td>Structured order</td>
<td>0.905</td>
<td>0.652</td>
</tr>
</tbody>
</table>

- Current strategy:
  - With lack of expert knowledge (e.g. like for SIFT and GIST) aim at balancing variances
  - Simply using a random rotation balances variances very well
Approach (both papers)

- Use a rotation matrix $R$ to rotate the data
- Perform PQ on rotated vectors
- Find $R$ and PQ clusters such that the quantization loss is minimized
- Note that rotation can:
  - Decorrelate data (e.g. PCA)
  - Balance variances inside blocks (e.g. random rotation)
Approach (both papers)

- Formulation: \( \min \sum_{x} ||R_x - c(i(x))|| \)
  - \( x \): data points (descriptors)
  - \( i(x) \): the ID of the product cluster \( x \) is assigned to
  - \( c \): product clusters
1: Non-parametric solution / ck-means

\[ \min \sum_x \| Rx - c(i(x)) \| \]

- Solve by alternating between optimizing for R and (c, i(x)):
  1. Fixing R, find clusters c and assignments i(x):
     - Same as the original PQ setting – just do k-means
       - In practice: only one step of k-means
  2. Fixing c and assignments i(x), find R:
     \[ \min ||RX - Y||_F, \quad s.t. \quad RTR = I \]
     - \( X = [x_1, x_2, ..], Y = [c(i(x_1)), c(i(x_2)), ..] \)
     - Orthogonal Procrustes problem, closed form solution:
       - \( \text{SVD}(XY^T) \rightarrow XY^T = USV^T \)
       - \( R = VU^T \)
2: Parametric solution

Overview:

- Prove that independence between blocks and balanced variances across blocks are both important
- Propose a simple greedy method to satisfy these goals
Assume data is following a Gaussian distribution \( N(0, \Sigma) \)

From rate distortion theory, for any quantizer with \( k \) codewords \( (k=2^l) \) a distortion \( E \) satisfies:

\[
E \geq k \frac{2}{D} D \left| \frac{1}{\bar{D}} \right|
\]

For PQ with \( M \) blocks:

\[
\Sigma = \begin{pmatrix}
\Sigma_{11} & \ldots & \Sigma_{1M} \\
\vdots & \ddots & \vdots \\
\Sigma_{M1} & \ldots & \Sigma_{MM}
\end{pmatrix}
\]

So the distortion bound for PQ is (note: dimension in each block is \( D/M \)):

\[
E \geq k \frac{2^M D}{M} \frac{1}{D} \sum_{m=1}^{M} |\hat{\Sigma}_{mm}|^{\frac{1}{D}}
\]

Thus, want to minimize:

\[
\sum_{m=1}^{M} |\hat{\Sigma}_{mm}|^{\frac{M}{D}} \quad \text{where} \quad \hat{\Sigma} = \mathbf{R} \Sigma \mathbf{R}^T
\]

Inequality of arithmetic and geometric means (simple from log + Jensen’s inequality)

\[
\sum_{m=1}^{M} |\hat{\Sigma}_{mm}|^{\frac{M}{D}} \geq M \prod_{m=1}^{M} |\hat{\Sigma}_{mm}|^{\frac{1}{D}}
\]

Then from Fisher inequality:

\[
\prod_{m=1}^{M} |\hat{\Sigma}_{mm}| \geq |\bar{\Sigma}| \quad (= |\Sigma|)
\]

Thus, constant lower bound on distortion \( E \):

\[
M |\Sigma|^{\frac{1}{D}}
\]
Assume data is following a Gaussian distribution $N(0, \Sigma)$

From rate distortion theory, for any quantizer with $k$ codewords ($k=2^l$) a distortion $E$ satisfies:

$$E \geq k \frac{2}{D} D \frac{1}{|\Sigma|^{1/D}}$$

For PQ with $M$ blocks:

$$\Sigma = \begin{pmatrix} \Sigma_{11} & \cdots & \Sigma_{1M} \\ \vdots & \ddots & \vdots \\ \Sigma_{M1} & \cdots & \Sigma_{MM} \end{pmatrix}$$

So the distortion bound for PQ is (note: dimension in each block is $D/M$):

$$E \geq k \frac{2M D}{M} \frac{1}{M} \sum_{m=1}^{M} |\hat{\Sigma}_{mm}|^{1/D}$$

Thus, want to minimize:

$$\sum_{m=1}^{M} |\hat{\Sigma}_{mm}|^{1/D} \quad \text{where} \quad \hat{\Sigma} = R\Sigma R^T$$

Inequality of arithmetic and geometric means (simple from log + Jensen’s inequality)

$$\sum_{m=1}^{M} |\hat{\Sigma}_{mm}|^{1/D} \geq \prod_{m=1}^{M} |\hat{\Sigma}_{mm}|^{1/D}$$

Equality when all summed items are equal

Then from Fisher inequality:

$$\prod_{m=1}^{M} |\hat{\Sigma}_{mm}| \geq |\Sigma| \quad (= |\Sigma|)$$

Equality when off-diagonal terms are 0

Thus, constant lower bound on distortion $E$: $\frac{M|\Sigma|^{1/D}}{D}$
2: Parametric solution

- Shown that independence between blocks and balanced variances across blocks are both important

- Simple greedy method – Eigenvalue allocation:
  - To achieve independence: align data by PCA
  - To balance variances: permute the PCA-aligned vectors such that the blocks contain roughly equal variances (products of eigenvalues)
    - Seems this is NP-hard (not sure)
    - Greedy: traverse dimensions in the order of decreasing eigenvalues, assign the current dimension to the (non-full) block with minimal product of allocated eigenvalues (potential problems for eigenvalues<1 ?)

For simplicity, I have replaced eigenvalues with log(eigenvalues) and thus consider the sum of logs instead of the product of eigenvalues; the two are equivalent

<table>
<thead>
<tr>
<th>log(eigs)</th>
<th>log(product)</th>
</tr>
</thead>
<tbody>
<tr>
<td>30 30 30 30 30 30 30 30 30</td>
<td>33</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
</tr>
</tbody>
</table>
Parametric solution

- Simple method
- Can be used to initialize the non-parametric solution (ck-means)

<table>
<thead>
<tr>
<th></th>
<th>theoretical minimum</th>
<th>Eigenvalue Allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIFT</td>
<td>$2.9286 \times 10^3$</td>
<td>$2.9287 \times 10^3$</td>
</tr>
<tr>
<td>GIST</td>
<td>$1.9870 \times 10^{-3}$</td>
<td>$1.9870 \times 10^{-3}$</td>
</tr>
</tbody>
</table>
Results (from Ge et al.)
Results (from Norouzi and Fleet)
Note: this slide was made when I was incorrectly balancing the sum eigenvalues instead of their product. However, using the product gave similar results.