Vector compression for similarity search using Multi-layer Sparse Ternary Codes

Sohrab Ferdowski, Slava Voloshynovskiy, Dimche Kostadinov
Department of Computer Science, University of Geneva, Switzerland
(Sohrab.Ferdowski, svolos, Dimche.Kostadinov)@unige.ch
http://sip.unige.ch

Overview

- Applications:
  - Large-scale retrieval systems
  - Learned compression of feature vectors
  - Compressed representation useful for fast similarity search

- Contributions:
  - Rate-distortion (R-D) study of ternary and binary encoding
  - Designing R-D efficient multi-layer Sparse Ternary Codes (STC)

Background

- Ability to search for similarity within a database is crucial for modern retrieval systems.
- A wide-spread solution is binary hashing.
- We proposed ternary hashing [WIFS ’16] as an alternative to binary hashing.
- We showed that ternary encoding has higher coding gain than binary encoding [SIT ’17].
- Here we extend ternary encoding for the task of compression, so that we can have list-reinement.
- Our design challenge: To have good R-D performance within STC limitations.

Problem formulation: ANN search

- Similarity search:
  - Exact Nearest Neighbor (NN) search:
  - Approximate Nearest Neighbor (ANN) search:

- Compressiom:
  - Encoding: $x = \mathbb{Q}(x)$
  - Reconstruction: $f = \mathbb{Q}^{-1}(x)$

- Distortion: $D = \mathbb{E}[d(x, f)]$

- Sparsity per each dimension:

Optimal Re-weighting vector:

$\lambda = \frac{1}{n} \sum_{i=1}^{n} \lambda_i$

Rate: $R = \frac{1}{2} |\lambda| = \frac{1}{2} \sum_{i=1}^{n} \lambda_i = \frac{1}{2} \sum_{i=1}^{n} \log_2 (1 + 2 \lambda_i)$

Sparsity per each dimension: $s_0 = |\lambda| / |\lambda| = |\lambda|$

$
\begin{align*}
\phi_0(t) &= \mathbb{Q}(t) \mathbb{I}_{[0,\lambda]}
\end{align*}$

Encoding:

$f = (A_1 \phi_0(1) + \beta_1) \mathbb{I}_{[0,\lambda]}$

(projection + ternarization + re-weighting)

Reconstruction:

$f = B_k x_k + (A_1 \phi_0(1) + \beta_1) \mathbb{I}_{[0,\lambda]}$

(projection)

Optimizing single-layer STC

- Decompose $B = (A^T A)^{-1} A^T B'$, optimize $B'$

- $B'_d = \arg \min_{B'} \mathbb{E}[d(x, f)]$

- $B' = A^T \mathbb{F}(X)$

- Calculation of distortion:

$D = \mathbb{E}[d(x, f)] = \mathbb{E}[d(x, A \mathbb{F}(X))]$

- Distortion per each dimension ($D_i = \sum_{j=1}^{n} D_j$):

$D_j = \mathbb{E}[|X_j - \beta_j |]$ 

$\Rightarrow D = \mathbb{E} \left[ \sum_{j=1}^{n} \beta_j^2 \mathbb{E} \left[ \left| X_j - \beta_j \right| \right] \right]$

- Optimal Re-weighting vector: $\beta' = \arg \min_{\beta} D$

$\Rightarrow \beta_j = \frac{1}{n} \sum_{i=1}^{n} \beta_i$

- Rate: $R = \frac{1}{2} \mathbb{E} \left[ \sum_{j=1}^{n} \beta_j \right] = \frac{1}{2} \sum_{j=1}^{n} \beta_j$

Search and R-D performance of multi-layer STC on public databases

- Single-layer encoding is insufficient to provide good R-D performance at high rates.
- Residual-based multi-layer encoding can provide reasonable R-D performance.
- Since binary-encoding has rate mismatch, it cannot benefit from multi-layer encoding.
- Ternary encoding with high sparsity has low rate mismatch and can benefit from multi-layer encoding.
- Future work: Joint learning of all layers.
- Python implementation: https://github.com/ssohrab/DSW2018