Dictionary Learning for Positive Definite Matrices (with Application to Nearest Neighbor Retrieval)

Suvrit Sra\textsuperscript{1} and Anoop Cherian\textsuperscript{2}

\textsuperscript{1} Max Planck Institute for Intelligent Systems, Tübingen
\textsuperscript{2} University of Minnesota, MN
Linear Model

\[ s \approx Dc \]
Introduction

Linear Model

\[ s \approx Dc \]

**Signal** \( s \): Image, audio, video, matrix, tensor

**Dictionary** \( D \): Basis for linear coding

**Code** \( c \): code vector, used in application
Introduction

Linear Model

\[ s \approx Dc \]

- Image compression
- Image denoising
- Texture synthesis
- Audio processing
- Signal classification
- Database search and retrieval
Dictionary learning task

Given examples $s_1, \ldots, s_m$, “learn” dictionary $D$ so that $s_j \approx Dc_j$, for suitable $c_j$

What’s so special about DL?
Dictionary learning task

Given examples $s_1, \ldots, s_m$, “learn” dictionary $D$ so that $s_j \approx Dc_j$, for suitable $c_j$

What’s so special about DL?

- Data dependent encoding
- Versatile applications
Generalized Dictionary Learning

Usually, signals $s_1, \ldots, s_m$ given as vectors in $\mathbb{R}^n$
Usually, signals $s_1, \ldots, s_m$ given as vectors in $\mathbb{R}^n$

We allow $S_1, \ldots, S_m$ to be matrices (or even tensors)
Generalized Dictionary Learning

Usually, signals $s_1, \ldots, s_m$ given as vectors in $\mathbb{R}^n$
We allow $S_1, \ldots, S_m$ to be matrices (or even tensors)

**Generalized Dictionary Learning (GDL)**

$$S \approx DC$$

Matrix $S$: input data
Dictionary $D$: tensor
Code $C$: code matrix
Usually, signals $s_1, \ldots, s_m$ given as vectors in $\mathbb{R}^n$.
We allow $S_1, \ldots, S_m$ to be matrices (or even tensors).

**Generalized Dictionary Learning (GDL)**

$$S \approx DC$$

$$S \approx DCE^T$$

$$\mathcal{D} := (E \otimes D)$$
Example: Suppose \( m \) input images of size \( d \times d \).

Assume dictionary has \( n \) basis elements.
Example: Suppose $m$ input images of size $d \times d$.

Assume dictionary has $n$ basis elements.

<table>
<thead>
<tr>
<th></th>
<th>Vectorized</th>
<th>GDL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage</td>
<td>$(m + n)d^2 + mn$</td>
<td>$md^2 + 2dn + mn$</td>
</tr>
<tr>
<td>Cost / Iter</td>
<td>$O(d^2 n)$</td>
<td>$O(dn)$</td>
</tr>
<tr>
<td>Locality</td>
<td>Lost</td>
<td>Preserved</td>
</tr>
</tbody>
</table>
Example: Suppose $m$ input images of size $d \times d$.

Assume dictionary has $n$ basis elements

<table>
<thead>
<tr>
<th></th>
<th>Vectorized</th>
<th>GDL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage</td>
<td>$(m + n)d^2 + mn$</td>
<td>$md^2 + 2dn + mn$</td>
</tr>
<tr>
<td>Cost / Iter</td>
<td>$O(d^2 n)$</td>
<td>$O(dn)$</td>
</tr>
<tr>
<td>Locality</td>
<td>Lost</td>
<td>Preserved</td>
</tr>
</tbody>
</table>

Let us look at special case of GDL now
GDL - special case

GDL for covariance matrices

**Input:** a set \( \{S_1, \ldots, S_m\} \) of covariance matrices

**GDL:** encode inputs as **sparse** combinations of bases

\[
\text{Find coding matrices } C_1, \ldots, C_m \geq 0, \text{ and dictionary matrix } D
\]

\[
\min_1^2 \sum_{i=1}^m \| S_i - DC_i D^T \|_2^2 + \sum_{i=1}^m \beta_i \text{sp}(C_i)
\]

Note: If \( C \succeq 0 \), then \( DCD^T \succeq 0 \)
GDL - special case

GDL for covariance matrices

Input: a set \( \{ S_1, \ldots, S_m \} \) of covariance matrices

GDL: encode inputs as \textit{sparse} combinations of bases

Find coding matrices \( C_1, \ldots, C_m \geq 0 \), and dictionary matrix \( D \)

\[
\min \frac{1}{2} \sum_{i=1}^{m} \| S_i - D C_i D^T \|_F^2 + \sum_{i=1}^{m} \beta_i \text{sp}(C_i),
\]

Note: If \( C \succeq 0 \), then \( D C D^T \succeq 0 \)
Simplification

\[ S_i \approx \sum_{j=1}^{n} c_j d_j d_j^T \]

\[
\min \frac{1}{2} \sum_{i=1}^{m} \| S_i - DC_i D^T \|_F^2 + \sum_{i=1}^{m} \beta_i \| C_i \|_1
\]

\[ S_i \approx DC_i D^T \]

\( C_i \) is nonnegative, diagonal
GDL - how to optimize

- Number $m$ of input covariances can be large
- Large scale *nonconvex* optimization problem
GDL - how to optimize

- Number $m$ of input covariances can be large
- Large scale *nonconvex* optimization problem
- Stochastic gradient approach
  1. Initialize dictionary
  2. Receive *mini-batch* of input
  3. Compute coding matrices for mini batch
  4. Update dictionary
  5. Repeat
Number $m$ of input covariances can be large

Large scale *nonconvex* optimization problem

Stochastic gradient approach

1. Initialize dictionary
2. Receive *mini-batch* of input
3. Compute coding matrices for mini batch
4. Update dictionary
5. Repeat

| Coding: Solve large-scale NNLS |
| Dict Update: Projected stoch. gradient step |
GDL – Application to Nearest Neighbors

Coding covariance matrices

- Dict: $n$ elements; say $r$ sparse code a covariance
- $\binom{n}{r}$ possible sparse codes – huge space of codes
- High chance that codes provide unique hash keys
- Hashing facilitates fast nearest neighbor retrieval
Subspace Combination Tuple

- Covariance $S$, and dictionary $D \in \mathbb{R}^{d \times n}$
- Let each element $d_i \in D$ have unique integer identifier $u_i$
- Suppose $d_i, d_j, \cdots, d_k$ used when sparse coding $S$
- *Subspace Combination Tuple* (SCT): Sort($u_i, u_j, \cdots, u_k$)
Hashing

- Encode S into its SCT
- Use SCT to index into a hash table
- Collisions resolved using search
- Allows fast NN search for given input covariance
Experiments

Three main experiments

1. Performance on real-world noisy data (LabelMe images)
2. Larger covariances (40 × 40 from Face image dataset)
3. Larger dataset (40,000 covariances of textures)
Experiments - LabelMe object dataset

- Took 10,000 annotated images dataset.
- For each annotated blob in an image: extracted color RGB values; first and second order gradients.
- Used these to compute the covariance matrix
- Obtained 25,000 such covariances (7 × 7)
Experiments - FERET face pose dataset

- Dataset has facial appearances, with varying poses.
- Aim: recognize person, irrespective of pose.
- We took approx. 10,000 images
- Computed Gabor transform based $40 \times 40$ covariances (as per state-of-the-art)
Texture key in satellite imaging, inspection systems, etc.

- **Brodatz** (111 classes) and **Curret** texture (60 classes)
- Covariances from random patches of each class.
- Pixel coordinates, intensity, first-order gradients to create covariances (5 x 5)
Dictionary Learning - Setup

- Cross-validation to estimate dictionary size
- Subset chosen as database; another fraction as query set
- GDL in MATLAB; while L2LSH, VEC in C
- Riemannian metric used to measure “nearness”
Evaluation

- Exact NN difficult; Approximate NN (ANN) more tractable.
- Let $X_{algo}$ be point found by algo; $X_{ls}$ the exact NN

Accuracy := \# correct matches / \# queries.
Evaluation

- Exact NN difficult; Approximate NN (ANN) more tractable.
- Let $X_{algo}$ be point found by algo; $X_{ls}$ the exact NN
- For query $Q$, define $X_{algo}$ to be an ANN if:

$$\frac{\delta(Q, X_{ls})}{\delta(Q, X_{algo})} > \epsilon$$

We chose $\epsilon = 0.75$.

Using this ANN setup, we define Accuracy as:

$$\text{Accuracy} = \frac{\text{# correct matches}}{\text{# queries}}.$$
Evaluation

- Exact NN difficult; Approximate NN (ANN) more tractable.
- Let $X_{algo}$ be point found by algo; $X_{ls}$ the exact NN
- For query $Q$, define $X_{algo}$ to be an ANN if:
  \[
  \frac{\delta(Q, X_{ls})}{\delta(Q, X_{algo})} > \epsilon
  \]
- We chose $\epsilon = 0.75$
- Using this ANN setup, we define *Accuracy* as:
  \[
  \text{Accuracy} := \frac{\# \text{correct matches}}{\# \text{queries}}.
  \]
Competing Methods

1. Log-Euclidean Embedding
   - Embed into euclidean space using matrix logarithm
   - Locality sensitive hashing (L2LSH, Hamming) based NN

2. Vectorization (VEC)
   Vectorize without embedding, and use LSH

3. Kernelized LSH
   Use pseudo-kernel function based on geodesic distance.
Results

NN retrieval accuracy: faces dataset

Accuracy (%)

L2LSH  HAM  VEC  KLSH  GDL

NN retrieval accuracy: faces dataset
Results

NN retrieval accuracy: texture dataset

Accuracy (%)

L2LSH  |  HAM  |  VEC  |  KLSH  |  GDL

NN retrieval accuracy: texture dataset
Results

NN retrieval accuracy: Objects dataset

Accuracy (%)

L2LSH  HAM  VEC  KLSH  GDL

NN retrieval accuracy: Objects dataset
Summary

- Dictionary learning for matrices and tensors
- Scalable online optimization algorithm
- Application to NN retrieval
- Many more applications to be discovered!
Summary

- Dictionary learning for matrices and tensors
- Scalable online optimization algorithm
- Application to NN retrieval
- Many more applications to be discovered!