Metric Learning with Rank and Sparsity Constraints

Bubacarr Bah

Inst. of Computational Engineering & Sciences (ICES), and Dept. of Mathematics
The University of Texas at Austin

ICIAM 2015
Minisymposium: Mathematics of Information & Low Dimensional Models
Beijing, China
August 12

Joint work with
Stephen Becker @ University of Colorado, Boulder
Baran Gözcü, and Volkan Cevher @ EPFL
Thus this talk ... 

1. Metric learning with a twist
2. Formulation of metric into an optimization problem
3. Proximal Gradient method
4. Experimental results
Problem setup and background
Metric learning

Metric learning is about learning a “good” distance metric

- Given points $x_i \in \mathcal{X}$ for $i = 1, \ldots, M$, find $d_Z(\cdot, \cdot)$ in which similar $x_i$ are closer while dissimilar $x_i$ are not

- $\Rightarrow$ the metric rearranges the $x_i$ into a similarity set $\mathcal{S}$ and a dissimilarity set $\mathcal{D}$

**Metric learning contd.**

**Standard problem linear model**

\[
\begin{align*}
\min_{\mathbf{B} \succeq 0} & \quad \sum_{\{\mathbf{x}_i, \mathbf{x}_j\} \in \mathcal{S}} \|\mathbf{x}_i - \mathbf{x}_j\|_B^2 \\
\text{subject to} & \quad \sum_{\{\mathbf{x}_i, \mathbf{x}_j\} \in \mathcal{D}} \|\mathbf{x}_i - \mathbf{x}_j\|_B^2 \geq 1
\end{align*}
\]

- \( \|\mathbf{x}\|_B := \sqrt{\mathbf{x}^T \mathbf{B} \mathbf{x}} \) is a semi-norm
- \( \mathcal{S} \) and \( \mathcal{D} \) are the dissimilarity and dissimilarity sets respectively

Metric learning has many applications

- Data classification and retrieval
- \( k \)-nearest neighbors, clustering, ...
- Signal processing, compressed sensing, ...
Rank and sparsity constraints

In general the solution, $\mathbf{B}$, dense and full rank

- high storage complexity
- high computational complexity

There is a desire to have a solution, which is low-rank and sparse

- low rankness: $\mathbf{B} = \mathbf{A} \mathbf{A}^T$, $\mathbf{A} \in \mathbb{R}^{N \times r}$ for $r \ll N$
- sparsity: $\|\mathbf{A}\|_0 \leq \sigma$ for $\sigma \in \mathbb{N}$ and $\sigma \ll N \times N$

This approach is also useful in machine learning

- matrix factorization
- autoencoding
- neural learning
Optimization formulation of problem
Problem description

This work: a slightly different metric learning

- no similarity set but a more stringent constraint

\[ \sum_{\{x_i, x_j\} \in \mathcal{D}} \|x_i - x_j\|_B^2 \geq |\mathcal{D}| \]

\[ \Rightarrow (1 - \delta) \leq \|x_i - x_j\|_B^2 \leq (1 + \delta) \]

- Instead of imposing a certain distance between dissimilar points, we want to preserve pairwise distances

\[ (1 - \delta)\|x_i - x_j\|_2^2 \leq \|x_i - x_j\|_B^2 \leq (1 + \delta)\|x_i - x_j\|_2^2 \]

- Making a change of variables \( B = AA^T, \ A \in \mathbb{R}^{N \times r} \)

\[ (1 - \delta)\|x_i - x_j\|_2^2 \leq \|A^T(x_i - x_j)\|_2^2 \leq (1 + \delta)\|x_i - x_j\|_2^2 \]

- Reminds us of JL/RIP and measurement matrix design in CS?
A remark on data embedding

Data embedding has a variety of different approaches

Traditionally, **PCA** has been the favorite tool

▷ Computationally **efficient** and **simple**

▷ Mapping may **not** preserve local geometries

Alternatively, **random projections** are also used

▷ Computationally **efficient**, **universal** and **bi-Lipschitz**

▷ **Cannot** exploit any structure in data

A bi-Lipschitz \( f \) satisfies

\[
L_1 |u - v| \leq |f(u) - f(v)| \leq L_2 |u - v|
\]

for constants \( L_1 \) and \( L_2 \) independent of \( u \) and \( v \)
A remark on data embedding contd.

Examples of bi-Lipschitz embeddings

* The Johnson-Lindenstrauss (JL) lemma is a bi-Lipschitz embedding of high-dimensional point clouds to a lower dimension

**Johnson-Lindenstrauss lemma**

\[
(1 - \epsilon)\|u - v\|^2_2 \leq \|f(u) - f(v)\|^2_2 \leq (1 + \epsilon)\|u - v\|^2_2
\]

for \(f : \mathbb{R}^N \rightarrow \mathbb{R}^m\), and \(\epsilon \in (0, 1)\), which requires \(m \geq m_0\)

* The restricted isometry property (RIP) allows for isometric embedding of sparse vectors

**RIP**

\[
(1 - \delta)\|x\|^2_2 \leq \|f(x)\|^2_2 \leq (1 + \delta)\|x\|^2_2
\]

for all sparse vectors \(x\)
A compact form of the problem

Data set: \( \mathcal{X} := \{ x_i \}_{i=1}^p \)

Secant set: \( \mathcal{S}(\mathcal{X}) := \left\{ v_{ij} = \frac{x_i - x_j}{\|x_i - x_j\|_2}, \text{ for } i \neq j \right\} \)

▶ Thus we can rewrite

\[
(1 - \delta)\|x_i - x_j\|_2^2 \leq \|A^T (x_i - x_j)\|_2^2 \leq (1 + \delta)\|x_i - x_j\|_2^2
\]

in terms of the secant vectors as follows

\[
|v_{ij}^T Bv_{ij} - 1| \leq \delta \quad \text{for } v_{ij} \in \mathcal{S}(\mathcal{X})
\]
A compact form of the problem contd.

- Re-index $v_{ij}$ to $v_l$ for $l = 1, \ldots, M$ where $M = \binom{p}{2}$.

- We learn $B$ and $\delta$ by solving

$$\minimize |v_l^T B v_l - 1| \text{ for } l = 1, \ldots, M$$

- Form the $N \times M$ matrix $V = [v_1, \ldots, v_M]$.

- Define a linear transform

$$\mathcal{A} : \mathbb{S}^{N \times N}_+ \rightarrow \mathbb{R}^M, \text{ such that } \mathcal{A}(B) := \text{diag}(V^T BV)$$

- Denote $1_M$ as the $M$-vector of ones, then

**Problem becomes ...**

$$\min_B \| \mathcal{A}(B) - 1_M \|_{\infty} \text{ subject to } B \succeq 0$$
A remark on related work

- PCA is one way to learn such a metric
- Another way is to take random projections by a Gaussian matrix
- This is an extension of the approach in [Sadeghian, B. & Cevher 2013]
- Closely related to the NuMax algorithm of [Hegde et al. 2012] which solves the following problem

\[
\begin{align*}
\text{minimize} & \quad \text{rank}(B) \\
\text{subject to} & \quad B \succeq 0, \ B = B^T \\
& \quad 1 - \delta \leq v_l^T B v_l \leq 1 + \delta, \ v_l \in S(\mathcal{X})
\end{align*}
\]

- To solve this the fix a $\delta$ and learn an embedding matrix $B$ and they use SVD and get a solution

\[
\hat{B} = \Sigma_r^{1/2} U_r^T
\]

- A key difference with our method is that we learn $B$ and $\delta$ at the same time and we don’t do the SVD step
Imposing rank and sparsity constraints

- **Low rank** constrained problem

  \[ \min_B \| A(B) - 1_M \|_\infty \quad \text{subject to } B \succeq 0, \ \text{rank}(B) = r \]

- Reformulated as an **unconstrained** optimization:

  \[ \min_{A \in \mathbb{R}^{N \times r}} \| A (AA^T) - 1_M \|_\infty \]

- We use an \( \ell_1 \) regularizer but still enforce sparsity as thus

  \[ \min_{A \in \mathbb{R}^{N \times r}} \| A (AA^T) - 1_M \|_\infty + \lambda \| A \|_1 \quad \text{subject to } \| A \|_0 \leq \sigma \]

- We smooth the \( \ell_\infty \) norm by a smoothing function \( f(\cdot) \)

  \[ \min_{A \in \mathbb{R}^{N \times r}} f \left( A (AA^T) - 1_M \right) + \lambda \| A \|_1 \quad \text{subject to } \| A \|_0 \leq \sigma \]
The smoothing function

- We choose a smoothing function parameterized by \( \mu \)

\[
f(\mathbf{z}) = f_\mu(\mathbf{z}) = \mu \log \left( \sum_{i=1}^{M} e^{z_i/\mu} + e^{z_i/\mu} \right)
\]

- This converges to \( \ell_\infty \) as \( \mu \to 0 \)

\[
\lim_{\mu \to 0} f_\mu(\mathbf{z}) = \| \mathbf{z} \|_\infty
\]

- \( f \) is Lipschitz and so is the gradient with constant \( \mu^{-1} \)

- But not in \( A \)
Solution via Nesterov acceleration
Solving the problem: Proximal gradient method

\[
\min_{A \in \mathbb{R}^{N \times r}} f(A(AA^T) - 1_M) + \lambda \|A\|_1
\]
such that \(\|A\|_0 \leq \sigma\)

\[
\Rightarrow \min_A F(A) + \phi(A)
\]

Proximal gradient algorithm

\[
A_{k+1} = \arg\min_A \nabla F(A_k)^T (A - A_k) + \frac{L}{2} \|A - A_k\|_F^2 + \phi(A) \tag{1}
\]

reduces to gradient descent if \(\phi \equiv 0\). Parameter \(L\) is inverse stepsize

- We actually use Nesterov accelerated variant
- If we drop \(\phi\) term, can use L-BFGS which is very fast

To work well, we need

- To be able to compute \(\nabla F(A_k)\) efficiently
- Compute minimizer in (1) efficiently (this depends on \(\phi\))
Computing the gradient

\[ F(A) = f(A(AA^T) - 1_M) \quad A \in \mathbb{R}^{N \times r}, \quad A : S_+^{N \times N} \rightarrow \mathbb{R}^M \]

Recall \( r \ll N \ll M \), where \( x_i \in \mathbb{R}^N \)

- \( f \) is log-sum-exp, completely separable, so \( f \) & \( \nabla f \) have cost \( O(M) \)
- \( A(B) = \text{diag}(V^T BV) \)
- \( A^*(b) = V \text{diag}(b) V^T \)

Applying chain rule

\[
\nabla F(A) = 2V \text{diag}(\nabla f(A(AA^T) - 1_M))V^T A
\]

If we are careful about the linear algebra,

- \( A(AA^T) \) can be done in \( O(rNM) \) (vs. \( O(N^2(M + r)) \) naively)
- \( 2V \text{diag}(b)V^T A \) can be done in \( O(rNM) \) (vs. \( O(M^2(N + r)) \))
Computing the proximity operator

Need to solve

\[ \tilde{A} = \arg\min_A \phi(A) + \frac{1}{2} \|A - A_0\|_F^2 \]

for arbitrary \( A_0 \), where

\[ \phi(A) = \begin{cases} 
\lambda \|A\|_1 & \text{if } \|A\|_0 \leq \sigma \\
+\infty & \text{otherwise}
\end{cases} \]

- If \( \lambda = 0 \), then just sort \( A_0 \) and keep \( \sigma \) largest entries in magnitude
- If \( \sigma \geq rN \), then just soft-thresholding operation
- Generally, possible by soft-thresholding & sorting a secondary qty.
- **Cost:** sorting \( A \) so \( \mathcal{O}(rN \log(rN)) \) worst-case (theoretically, could avoid sort and do \( \mathcal{O}(rN) \))
- **Overall cost per iteration:** \( \mathcal{O}(rNM) \) compared to \( \mathcal{O}(MN^2) \)

*minimum* for convex methods
Experimental results
Experimental results

- We call our algorithm **FAML**: Fast Adaptive Metric Learning
- We compare with **NuMax**, PCA of data and random projections
- **Data**: Images of white squares in a black background we refer to as “manifold” images, and motorcycle images ($p = 75$, $M = 2775$)

*manifold image sample*  

*samples of motorcycle images*
Experiment-1

- FAML vs. PCA and random projections (using Gaussian matrices)
  We initialized with PCA in the dense and sparse cases
- FAML vs. NuMax, PCA of data and random projections
  We fix a rank and run FAML and $\delta$ is used as input in NuMax

**input data - motorcycle images**

- FAML performs **better**, especially at low ranks
Running times

- FAML vs. others

**input data - manifold images**

- Both dense and sparse versions are faster than NuMax
Experiment-II

- High dimensional case: full resolution motorbike images with
  \[ N = 163 \times 261 = 42543 \]
  - \( p = 50 \) points are selected and \( M = 1225 \) secants
  - We fix a rank and run FAML and \( \delta \) is used as input in NuMax

**input data - motorcyce images**

- FAML performs PCA and Gaussian
- NuMax doesn't fit into memory
Conclusion
Conclusion

Summary

- An optimization formulation for linear metric learning
- Learns sparse metrics which are good for computation
- Learns low rank metrics suitable for certain applications
- Fast convergence due to Nesterov acceleration
- Outperforms NuMax at low rank and high dimensions

Possible extensions

- LDPC codes design
- Full ML problem with a vector of zeros and ones
- NuMax’s column generation to handle more secants
THANK YOU