ECE 5582 Computer Vision
Lec 16: Subspace Models on Grassmann Manifold

Zhu Li
Dept of CSEE, UMKC
Office: FH560E, Email: lizhu@umkc.edu, Ph: x 2346.
http://l.web.umkc.edu/lizhu
Outline

- Recap:
  - Graph Embedding & Laplacianface
  - Graph Fourier Transform

- Data Partition and Subspace Optimization
  - Query Driven Solution
  - Subspace Indexing on Grassmann Manifold
  - Optimization of Subspace on Grassmann Manifold

- HW-4: Classification
  - Remote sensing data classification

- Summary
Subspace Learning for Face Recognition

- Project face images to a subspace with basis $A$
  - Matlab: $x = \text{faces} \times A(:,1:kd)$

\[
\begin{align*}
\text{eigf}_1 & = 10.9^* \\
\text{eigf}_2 & = 0.4^* \\
\text{eigf}_3 & = 4.7^*
\end{align*}
\]
Graph Embedding Interpretation of PCA/LDA/LPP

- Affinity graph $S$, determines the embedding subspace $W$, via
  \[ XLX^T W = \lambda XDX^T W \]

- PCA and LDA are special cases of Graph Embedding

  - **PCA:**
    \[ S_{i,j} = \frac{1}{n} \]

  - **LDA**
    \[ S_{i,j} = \begin{cases} 
    \frac{1}{n_k}, & \text{if } x_i, x_j \in C_k \\
    0, & \text{else} 
    \end{cases} \]

  - **LPP**
    \[ S_{i,j} = \begin{cases} 
    -\exp\left(\frac{|x_j - x_i|^2}{h}\right), & \text{if } |x_j - x_i| \leq \theta \\
    0, & \text{else} 
    \end{cases} \]
Religious/Philosophical Appreciation of Embedding

Matthew 16:19: Peter's Key

“I will give you the keys of the kingdom of heaven, and whatever you bind on Earth shall be bound in heaven, and whatever you loose on Earth shall be loosed in heaven.”

- Keys: \( y = Ax \)
- Loose/bind affinity: 
  \[
  S_{i,j} = \begin{cases} 
  -\exp\left(\frac{|x_j - x_i|^2}{h}\right), & \text{if } |x_j - x_i| \leq \theta \\
  0, & \text{else}
  \end{cases}
  \]
Laplacian and LDA Embedding

- Laplacian face is powerful. 😊
Applications: facial expression embedding

- Facial expressions embedded in a 2-d space via LPP

- SIFT Compression:

\[ A = \arg \min_W \sum_k \sum_j s_{j,k} |Wx_j - Wx_k|^2 \]

Laplacian embedding and key points topology verification for large scale mobile visual identification

Xin Xin\(^a,\)\(^*\), Zhu Li\(^b\), Aggelos K. Katsaggelos\(^a\)

\(^a\) Department of EECS, Northwestern University, Evanston, IL, USA
\(^b\) Multimedia Standards Research, Samsung Telecom America, Richardson, TX, USA
Signal on Graph

- non-uniformly sampled

- Connected, undirected, weighted graph $\mathcal{G} = (V, E, W)$ where $W_{i,j}$ is the weight of the edge $e = (i, j)$

- Graph signal: a function $f : V \rightarrow \mathbb{R}$ that assigns real values to each vertex of the graph

- Graph description:
  - Weight matrix $W$
  - Degree matrix $\mathcal{D}$: diagonal matrix with sum of weights of incident edges
  - Laplacian matrix $\mathcal{L}$: difference operator
Graph Fourier Transform

- GFT is different from Laplacian Embedding:
  - GFT: nxn transform brings signal to graph spectral domain
  - LLP: dx dx embedding of affinity graph

  - Laplacian is a difference operator \( \mathcal{L} := D - W \)
    
    \[
    (\mathcal{L}f)(i) = \sum_{j \in N_i} W_{i,j} [f(i) - f(j)]
    \]

  - It is a real symmetric matrix
  - It has a complete set of eigenvectors \( \{u_\ell\}_{\ell=0,1,...,N-1} \)
  - The eigenvectors are associated with real, nonnegative eigenvalues \( \{\lambda_\ell\}_{\ell=0,1,...,N-1} \)

    \[
    \mathcal{L}u_\ell = \lambda_\ell u_\ell, \ \forall \ell = 0, 1, \ldots, N - 1
    \]

  - Its spectrum is defined as \( \sigma(\mathcal{L}) := \{\lambda_0, \lambda_1, \ldots, \lambda_{N-1}\} \)
    
    \[
    0 = \lambda_0 < \lambda_1 \leq \lambda_2 \ldots \leq \lambda_{N-1} := \lambda_{\max}
    \]
Graph Spectrum

- Eigenvector vs Zero Crossing counts
  - The graph Laplacian eigenvalues and eigenvectors carry a notion of frequency

![Graph and graph spectrum](image)

Number of zero crossings

$\lambda_\ell$

$u_1$

$u_{50}$
Outline

- Recap:
  - Laplacianface
  - Graph Fourier Transform

- Data Partition and Subspace Optimization
  - Query Driven Solution
  - Subspace Indexing on Grassmann Manifold
  - Optimization of Subspace on Grassmann Manifold

- Summary
Large Scale Recognition/Classification Problems

- **Face Recognition**
  - Identify face from large (e.g., 7 million HK ID) face data set

- **Image Search**
  - Find out the tags associated with the given images

- **Fingerprint Identification**
  - Verification (yes or no) already mature and deployed in HK
  - Identification challenges: stateless feature/indexing efficiency
• **Find a “good” $f()$**
  – Such that after projecting the appearance onto the subspace, the data points belong to different classes are easily separable
  – This kind of “good” function is usually non-linear, hard to obtain from training data.
Graph Embedding Interpretation of PCA/LDA/LPP

- Affinity graph $S_r$ determines the embedding subspace $W$, via

$$XLX^T W = \lambda XDX^T W$$

- PCA and LDA are special cases of Graph Embedding

  - **PCA:**
    
    $$S_{i,j} = 1/n$$

  - **LDA**
    
    $$S_{i,j} = \begin{cases} 
    \frac{1}{n_k}, & \text{if } x_i, x_j \in C_k \\
    0, & \text{else}
    \end{cases}$$

  - **LPP**
    
    $$S_{i,j} = \begin{cases} 
    \frac{-\exp\left(|x_j - x_i|^2\right)}{h}, & \text{if } |x_j - x_i| \leq \theta \\
    0, & \text{else}
    \end{cases}$$
Non-Linearity Issue

- Non-Linear Solutions and challenges:
  - Kernel method: e.g K-PCA, K-LDA, K-LPP, SVM
    » Evaluate inner product $<x_j, x_k>$ with a kernel function $k(x_j, x_k)$, which if satisfy the conditions in Mercer’s Theorem, implicitly maps data via a non-linear function.
    » Typically involves a QP problem with a Hessian of size $n \times n$, when $n$ is large, not solvable.
  - LLE /Graph Laplacian:
    » An algorithm that maps input data $\{x_k\}$ to $\{y_k\}$ that tries to preserve an embedded graph structure among data points.
    » The mapping is data dependent and has difficulty handling new data outside the training set, e.g., a new query point
- How to compromise?
  - Piece-wise Linear Approximation: local data patch linear model fitting
Piece-wise Linear : Query Driven

- **Query-Driven Piece-wise Linear Model**
  - No pre-determined structure on the training data
  - Local neighborhood data patch identified from query point $q$,
  - Local model built with local data, $A(X, q)$

Local Graph Model:

$$Y = A(X, q)X$$

Local data:

$N(X, q)$
DoF Criteria for Local Model

What is a proper size of local data support: \( N(X, q) \) ?

- **The DoF of a linear model:**
  - DoF(A) = \( w \times h \times d \)
  - Example: DoF(A) = 20x24x5 = 2400

- **Discriminant Information to be captured:**
  - LDA: a graph with edges pruned for intra-class points
  - LPP: pruned graph
  - Information function
    \[
    F(N(X, q)) = \|W(\cdot)\|_0
    \]
    as number of edges/relationship among data points

What is a good compromise of data complexity and model power?

\[
\text{DoF} = w \times h \times d \quad F = \|W(\cdot)\|_0
\]
The tradeoffs in local data support size

\[ K = \frac{w \times h \times d}{\|W(:,):\|_0} \]

\[ W(:,:) = [w_{1,1}, w_{1,2}, \ldots w_{1,n}, \ldots, w_{n,1}, w_{n,2}, \ldots, w_{n,n}] \]

Model validation accuracy as functions of DPC

- overfitting
Head Pose Estimation

Given face images as pixels in $\mathbb{R}^{w \times h}$, predict its pose:
Tilt and Pan angles:

Recognition rate is improved:
- DoF: $w=18$, $h=18$, $d=16$, $32$
- Local Data Size: $K=30$

And the cost in computation is rather modest
- Matlab code, online local model $A(X,q)$ learning and NN classification:

<table>
<thead>
<tr>
<th>Method</th>
<th>Pan (d=16)</th>
<th>Tilt (d=16)</th>
<th>Pan (d=32)</th>
<th>Tilt (d=32)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>33.5</td>
<td>44.3</td>
<td>26.9</td>
<td>35.1</td>
</tr>
<tr>
<td>LDA</td>
<td>30.1</td>
<td>33.3</td>
<td>25.8</td>
<td>26.9</td>
</tr>
<tr>
<td>LPP(1)</td>
<td>30.1</td>
<td>31.2</td>
<td>24.7</td>
<td><strong>22.6</strong></td>
</tr>
<tr>
<td>LPP(2)</td>
<td>67.7</td>
<td>76.3</td>
<td>63.4</td>
<td>61.3</td>
</tr>
<tr>
<td>l-PCA</td>
<td>25.2</td>
<td>37.8</td>
<td>24.5</td>
<td>37.6</td>
</tr>
<tr>
<td>l-LPP</td>
<td>33.9</td>
<td>44.5</td>
<td>29.2</td>
<td>40.2</td>
</tr>
<tr>
<td>l-LDA</td>
<td><strong>20.4</strong></td>
<td><strong>30.7</strong></td>
<td><strong>19.1</strong></td>
<td>30.7</td>
</tr>
</tbody>
</table>

**Table 1. Pose estimation error rates**

<table>
<thead>
<tr>
<th>Method</th>
<th>K=30</th>
<th>K=60</th>
<th>K=90</th>
</tr>
</thead>
<tbody>
<tr>
<td>l-LDA, d=16</td>
<td>0.105</td>
<td>0.132</td>
<td>0.121</td>
</tr>
<tr>
<td>l-LDA, d=32</td>
<td>0.145</td>
<td>0.146</td>
<td>0.176</td>
</tr>
<tr>
<td>l-LPP, d=16</td>
<td>0.094</td>
<td>0.122</td>
<td>0.104</td>
</tr>
<tr>
<td>l-LPP, d=32</td>
<td>0.132</td>
<td>0.116</td>
<td>0.144</td>
</tr>
</tbody>
</table>

**Table 2. Computational complexity (sec) per recognition**
Face Recognition

- On ATT Data set: 40 subjects, 400 images:
  - Query point drives 3 local models, $A_1(X, q)$, $A_2(X, q)$, $A_3(X, q)$
  - Local model classification error estimation,
  - Combining the results – weighted voting

Multiple face models with different area and scale:
(a) Upper face model ($18 \times 16$).
(b) Lower face model ($14 \times 18$).
(c) Full face model ($21 \times 28$).

Extra credit: 10pts 😊
- Develop a query driven local Laplacianface model for HW-3
Summary of Query Driven Local Models

- The Good:
  - Allows for local data adaptation, solid gains in classification
  - Problem size is $d \times d$, (vs kernel method, which is $n \times n$, not scaling well)

- The Bad:
  - *Data driven*: need to compute a run-time model that adds computational complexity

- The Ugly:
  - Storage Penalty: need extra storage to store all training data, because the local NN data, patch is generated at run time, as function of the query point
  - Indexing challenge: for very large scale problem, not practical to store all training data, and effective indexing needed to support efficient access of query induced local training data.

- Question:
  - Can we do better?
Closer Examination of DoFs

DoF of the Stiefel manifold:

- All possible $p$ orthonormal basis functions in $d$-dimensional space, $A_{pxd}$, spans Stiefel Manifold, $S(p,d)$ in $R^{d \times p}$, $d > p$.

$$S(p, d) = \{ A \in R^{d \times p}, \text{s.t., } A' A = I_d \}$$

- It requires orthonormal relationship among basis vectors, so the DoF is not $pd$

$$\text{DoF}(S(p,d)) = (d-1) + (d-2) + \ldots + (d-p)$$

$$= pd - (1 + 2 + \ldots + p) = pd - p(p + 1)/2$$

- What is missing?
Grassmann manifold DoF:

- $G(p, d)$ identifies $p$-dimensional subspaces in $d$-dimensional space, it consists of $p$ orthonormal bases in $d$-dimensional space (Stiefel Manifold) and an equivalence constraint on the rotation of the basis functions:

  $$A_1 = A_2, \quad \text{if } \text{span}(A_1) = \text{span}(A_2)$$

- $G(p, d)$ is the quotient space of $S(p, d)/O(p)$, i.e., $A_1 = A_2$,

  $$\text{if exists } R \in \mathbb{R}^{p \times p}, \quad \text{s.t.}, \quad RA_1 = A_2, \quad R \in O_p$$

- The DoF of subspaces on Grassmann manifold

  $$\text{DoF}(G(p, d)) = pd - p^2$$
Consider a typical appearance modeling
- Image size 12x10 pixel, appearance space dimension $d=120$, model dimension $p=8$.
- 3D visualization of all $S(8, 120)$ and their covariance eigenvalues”
- Grassmann Manifolds are quotient space $S(8, 120)/O(8)$
The principal angles between two subspaces:

For $A_1$ and $A_2$ in $G(p, d)$, their principal angles are defined as

$$\cos(\theta_k) = \max_{u_k \in \text{span}(A_1), v_k \in \text{span}(A_2)} u_k^T v_k$$

subject to

$$\begin{cases} 
  u_k^T u_k = 1, & v_k^T v_k = 1 \\
  u_k^T u_i = 0, & v_k^T v_i = 0 
\end{cases}$$

where, $\{u_k\}$ and $\{v_k\}$ are called principal dimensions for $\text{span}(A_1)$ and $\text{span}(A_2)$. 
Solving for Principal Angles

Given two subspace models $A_1$ and $A_2$, find the rotations that can max align two:

- Rotating $A_1$, and $A_2$ in $G(p, d)$, such that they are maximally aligned

$$
\max_{R_1, R_2} \text{Trace}(R_1^T A_1^T A_2 R_2), \text{ s.t.}, R_1, R_2 \in O_p
$$

solving by SVD:

$$
[U, S, V] = \text{svd}(A_1^T A_2)
$$

- Where, $U=[u_1, u_2, \ldots, u_p]$, and $V=[v_1, v_2, \ldots, v_p]$ are the principle angles.

- The diagonal of $S$, $[s_1, s_2, \ldots, s_p]$ are the cosine of principal angles, and principal angles are computed as,

$$
\theta_k = \cos^{-1}(s_k)
$$
Subspace Distance on Grassmann Manifold

Grassmann Distance Metrics:

- **Projection Distance**
  
  **Def:**
  
  \[ d_{\text{proj}}(A_1, A_2) = \left( \sum_{i=1}^{p} \sin^2 \theta_i \right)^{1/2} \]
  
  **Computing:**
  
  \[ d_{\text{proj}}^2(A_1, A_2) = p - \sum_{i=1}^{p} \cos^2 \theta_i = m - \|A_1' A_2\|_F^2 \]

- **Binet-Cauchy Distance**
  
  **Def:**
  
  \[ d_{\text{bc}}(A_1, A_2) = (1 - \prod_{i} \cos^2 \theta_i)^{1/2} \]
  
  **Computing:**
  
  \[ d_{\text{bc}}^2(A_1, A_2) = 1 - \prod_{i} \cos^2 \theta_i = 1 - \det^2(A_1' A_2) \]
Arc Distance Metric

- Arc Distance
  Def:

  \[ d_{arc}(A_1, A_2) = \left( \sum_{i} \theta_i^2 \right)^{\frac{1}{2}} \]

  Also known as geodesic distance. It traverse the Grassmannian surface, and two subspace collapse into one, when all principal angles becomes zero.
Linear Combination of Subspaces

How to combine two models?

- Motivation:
  » say if subspace $A_1$ is best for data set $S_1$, and subspace $A_2$ is best for data set $S_2$, can we find a subspace $A_3$ that is good for both?

- When two subspaces are sufficiently close on Grassmannian manifold, we can approximate this by, $A_3 = [t_1, t_2, ....]$

$$t_k = \frac{n_1}{n_1 + n_2} u_k + \frac{n_2}{n_1 + n_2} v_k$$

Where $n_{1,2}$ are the size of data set $S_{1,2}$

- The new sets of basis may not be orthogonal. Can be corrected by Gram-Schmidt orthogonalization.

Moving along Geodesics?
Training Data Set Partition

• The Plan:
  – Partition the (unlabeled) large training data set into local data patches
  – Compute local models for each data patch with labels, and then optimize a subset for the recognition

• Data Space Partition
  – Partition the training data set by kd-tree, for the kd-tree height of h, we have $2^h$ local data patch as leaf node
  – For each leaf node data patch $k$, build a local LDA/LPP model $A_k$:
Subspace Indexing

• **Subspace Clustering by Grassmann Metric:**
  – It is a VQ like process.
  – Start with a data partition kd-tree, their leaf nodes and associated subspaces \( \{A_k\} \), \( k = 1..2^h \)
  – Repeat
    » Find \( A_i \) and \( A_j \), if \( d_{arc}(A_i, A_j) \) is the smallest among all, and the associated data patch are *adjacent* in the data space.
    » Delete \( A_i \) and \( A_j \), replace with merged new subspace, and update associated data patch leaf nodes set.
    » Compute the empirical identification accuracy for the merged subspace
    » Add parent pointer to the merged new subspace for \( A_i \) and \( A_j \).
    » Stop if only 1 subspace left.
  – Benefit:
    » avoid forced merging of subspace models at data patches that are very different, though adjacent.
MHT Based Identification

- **MHT operation**
  - Organize the leaf nodes models into a new hierarchy, with new models and associated accuracy (error rate) estimation
  - When a probe point comes, first identify its leaf nodes from the data partition tree.
  - Then traverse the MHT from leaf nodes up, until it hits the root, which is the global model, and choose the best model along the path for identification
Simulation

- **The data set**
  - MSRA Multimedia data set, 65k images with class and relevance labels:

  - **Background**
  - **Baby**
  - **Beach**

  ‘Very relevant’ samples from three classes: *background, baby* and *beach*

  - **Background**
  - **Baby**
  - **Beach**

  ‘Relevant’ samples from the three classes

  - **Background**
  - **Baby**
  - **Beach**

  ‘Irrelevant’ samples from the three classes
Simulation

- **Data selection and features**
  - Selected 12 classes with 11k images and use the original combined 889d features from color, shape and texture
  - Performance compared with PCA, LDA and LPP modeling
Simulation

- **Face data set**
  - Mixed data set of 242 individuals, and 4840 face images
  - Performance compared with PCA, LDA and LPP modeling
Grassmann Indexing Summary

- Contributions
  - Address the DoF issues of the BIGDATA recognition problem.
  - Piece-wise linear approach is effective in modeling non-linearity in visual manifolds for a variety of recognition problem.
  - Subspace indexing on Grassmann manifold offers a systemic approach in optimizing the linear models for the localized problem.
  - Solid performance gains over the state of art global linear models and their kernelized non-linear models.
• **Future work**
  - Grassmann Hashing – Penalize projection selection with Grassmannian metric, offers performance gains over LSH and spectral hashing.
  - Grassmann Local Descriptor Aggregation (MPEG CDVS) – significantly improved the indexing efficiency when work with Fisher Vector.
  - SIFT compression with a bank of transforms.

• **Publications:**
  - X. Wang, Z. Li, L. Zhang, and J. Yuan, "Grassmann Hashing for Approx Nearest Neighbour Search in High Dimensional Space", *Proc. of IEEE Int'l Conf on Multimedia & Expo* (ICME), Barcelona, Spain, 2011.
Newtonian Method in Optimization

Recall that in optimizing a functional over vector variables $f(X)$, $X$ in $\mathbb{R}^n$,

1. Start with an initial guess $x_0$ and a tolerance $\epsilon$.

2. Repeat:

   (a) Compute the Newton step $\Delta x = -(\text{Hess}(f)^{-1} \cdot \nabla f)|_{x_n}$.

   (b) Use line search to find the step size $t$.

   (c) Let $x_n = x_{n-1} + t \cdot \Delta x$.

   until $\nabla f(x_n)^T \Delta x < \epsilon$.

Credit: Kerstin Johnsson, Lund Univ
2.5.3. The Gradient of a Function (Grassmann). We must compute the gradient of a function $F(Y)$ defined on the Grassmann manifold. Similarly to §2.4.4, the gradient of $F$ at $[Y]$ is defined to be the tangent vector $\nabla F$ such that

$$\text{tr} \, F_Y^T \Delta = g_c(\nabla F, \Delta) \equiv \text{tr}(\nabla F)^T \Delta$$

for all tangent vectors $\Delta$ at $Y$, where $F_Y$ is defined by Eq. (2.52). Solving Eq. (2.69) for $\nabla F$ such that $Y^T(\nabla F) = 0$ yields

$$\nabla F = F_Y - YY^T F_Y.$$
Hessian on Grassmann Manifold

■ Hessian:

2.5.4. The Hessian of a Function (Grassmann). Applying the definition for the Hessian of $F(Y)$ given by Eq. (2.54) in the context of the Grassmann manifold yields the formula

$$
\text{Hess } F(\Delta_1, \Delta_2) = F_{YY}(\Delta_1, \Delta_2) - \text{tr}(\Delta_1^T \Delta_2 Y^T F_Y),
$$

where $F_Y$ and $F_{YY}$ are defined in §2.4.5. For Newton’s method, we must determine $\Delta = -\text{Hess}^{-1} G$ satisfying Eq. (2.58), which for the Grassmann manifold is expressed as the linear problem

$$
F_{YY}(\Delta) - \Delta(Y^T F_Y) = -G,
$$

$Y^T \Delta = 0$, where $F_{YY}(\Delta)$ denotes the unique tangent vector satisfying Eq. (2.60) for the Grassmann manifold’s canonical metric.

- $F_Y = \text{nxp} \ 1^{\text{st}} \text{ order differentiation}$
- $F_{YY} = \text{2}^{\text{nd}} \text{ order differentiation along } Y$
Newton’s Method on Grassmann Manifold

Overall framework

Newton’s Method for Minimizing $F(Y)$ on the Grassmann Manifold

- Given $Y$ such that $Y^TY = I_p$,
  - Compute $G = F_Y - YY^TF_Y$.
  - Compute $\Delta = -\text{Hess}^{-1}G$ such that $Y^T\Delta = 0$ and
    \[ F_{YY}(\Delta) - \Delta(Y^TF_Y) = -G. \]
- Move from $Y$ in direction $\Delta$ to $Y(1)$ using the geodesic formula
  \[ Y(t) = YV \cos(\Sigma t)V^T + U \sin(\Sigma t)V^T \]
  where $U\Sigma V^T$ is the compact singular value decomposition of $\Delta$ (meaning $U$ is $n$-by-$p$ and both $\Sigma$ and $V$ are $p$-by-$p$).
- Repeat.
Matlab Implementation

Prof. A. Edelman’s matlab package:
- https://umkc.box.com/s/g2oyqvsb2lx2v9wzf0ju60wnspts4t9g

Potential project for the class:
- Graph Fourier Transform optimization: different graph construction strategy gives different signal energy compaction performance, can we use compression efficiency (as entropy sum) as objective functional, and optimize the design of graph affinity matrix (symmetric, p.s.d)?
Outline

- Recap:
  - Graph Embedding & Laplacianface
  - Graph Fourier Transform

- Data Partition and Subspace Optimization
  - Query Driven Solution
  - Subspace Indexing on Grassmann Manifold
  - Optimization of Subspace on Grassmann Manifold

- HW-4: Classification
  - Remote sensing data classification

- Summary
HW-4: Remote Sensing Data Set Classification

The problem:
- classify aerial images into 45 different categories:

- data set: https://umkc.box.com/s/fxvzh5qq2tiob6eklfxfwn89kg3e1io1
Baseline: using pre-trained AlexNet to compute a FC feature from image, and then use PCA+LDA to classify (like in HW-3)

- AlexNet: PyTorch, MatconvNet all have model ready for use.
- Take the final FC as 4096 dimensional input features, use Eigenface/Fisherface methods to generate a base line
HW-4: SoftMax Network

- Re-train with AlexNet, show performance gains (hopefully) over the pretrained AlexNet FC + PCA+LDA

- Alternatively, use VGG16 to retrain a SoftMax network to see where the performance is
Project Option: Triplet Loss Solution

- Use pretrained AlexNet FC feature, PCA+LDA embedding, and then build a kd-tree to partition the input feature space into $2^L$ leaf nodes, for a kd-tree depth of $L$ (=8, 9, 10, e.g.), then do label split with leaf node id, e.g., airport_1101, bridge_776

- Form a triplet loss network on the expanded label space

\[
\sum_{i}^{N} \left[ \| f(x_i^a) - f(x_i^p) \|_2^2 - \| f(x_i^a) - f(x_i^n) \|_2^2 + \alpha \right]^+ 
\]
Summary

Graph Laplacian Embedding is an unifying theory for feature space dimension reduction

- PCA is a special case of graph embedding
  - Fully connected affinity map, equal importance
- LDA is a special case of graph embedding
  - Fully connected intra class
  - Zero affinity inter class
- LPP: preserves pair wise affinity.
- GFT: Eigen vectors of graph Laplacian, has Fourier Transform like characteristics.

Many applications in

- Face recognition
- Pose estimation
- Facial expression modeling
- Compression of Graph signals.