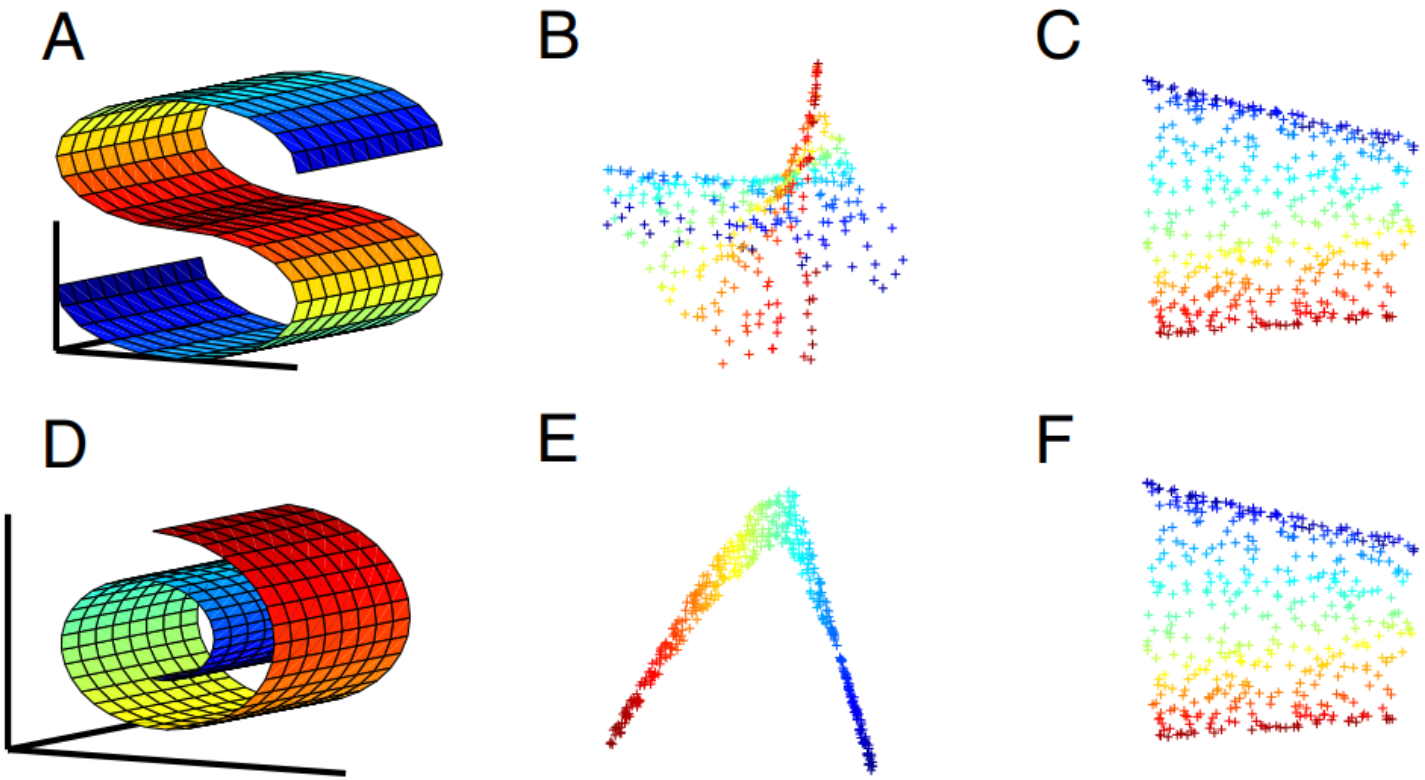
# Manifold Alignment

Shihua Zhang

Fall 2019

# Contents

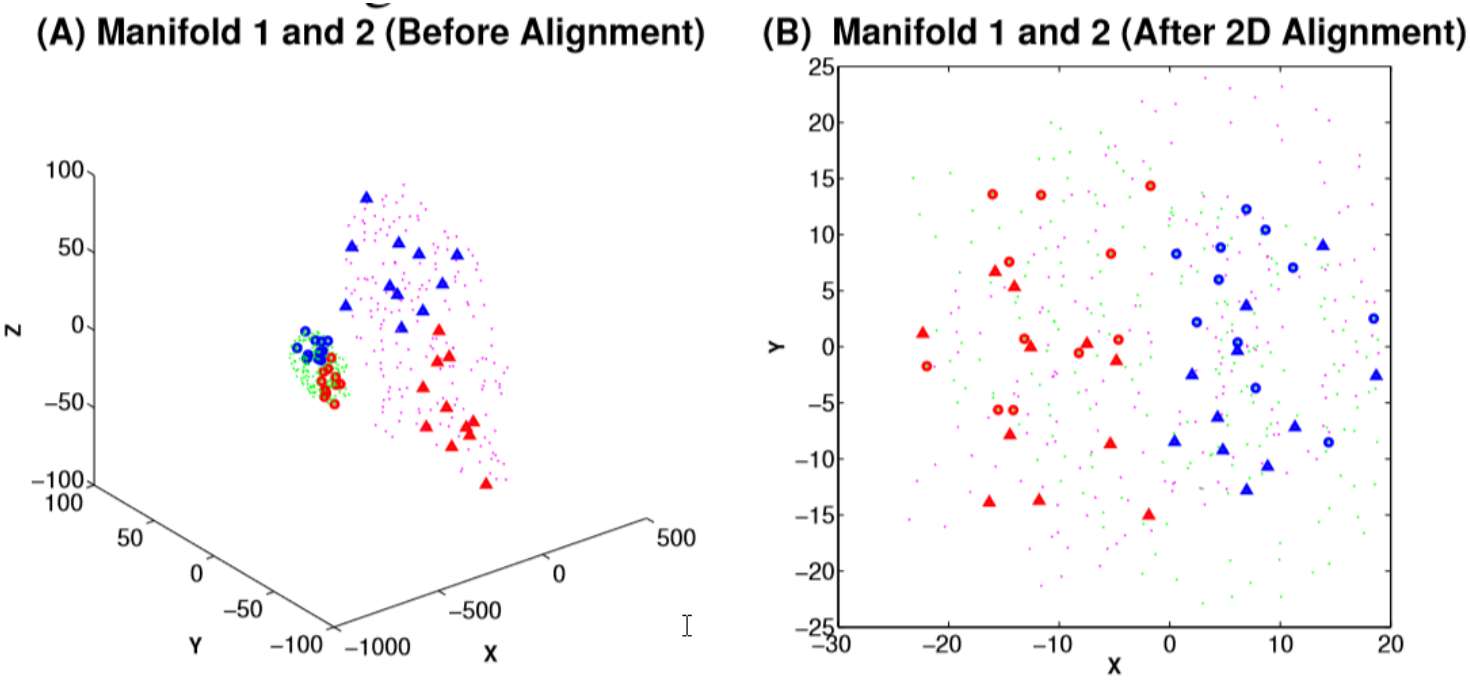
# Learning Correspondences



# Data Matching


Figure: Pose and angle matching

# Data Integration



# Domain Adaptation

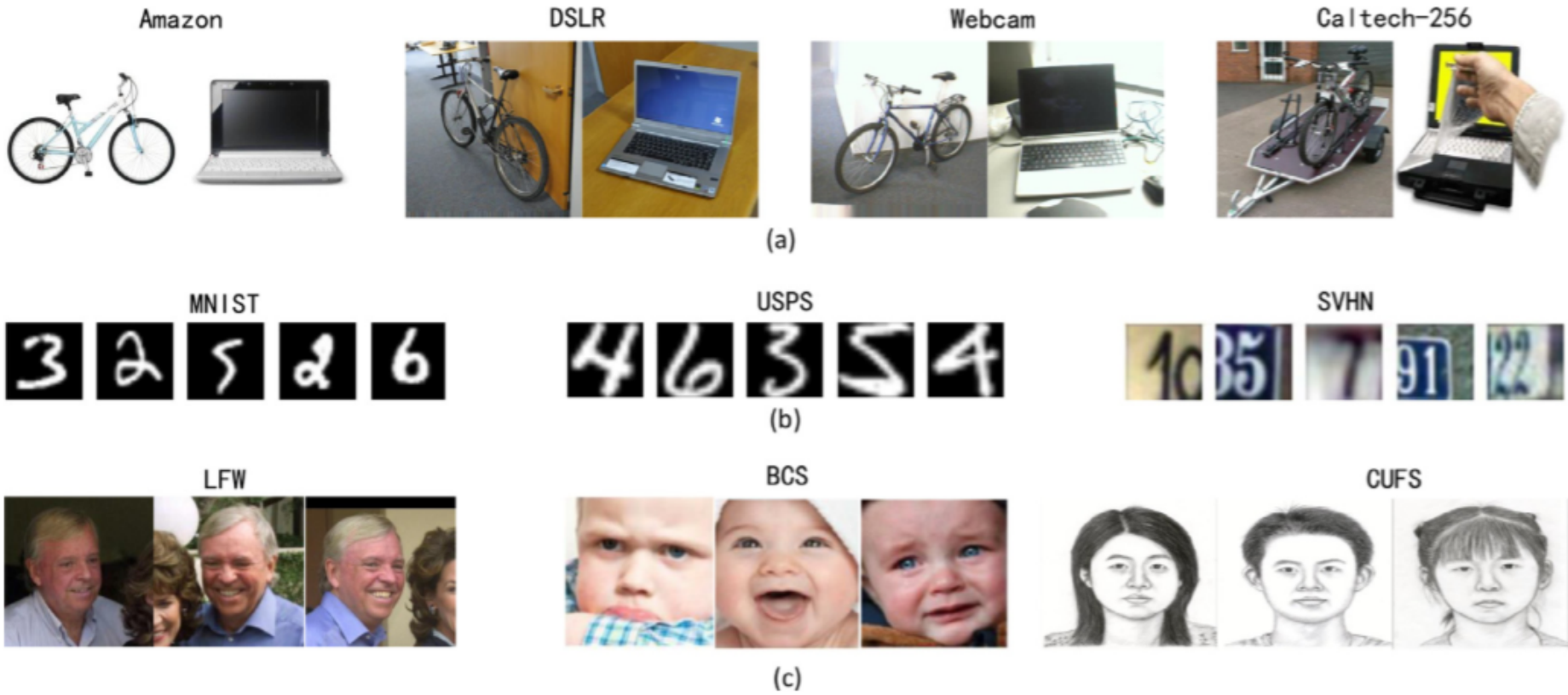



Figure: Distribution shift between domains can degrade the performance of a classifier trained on a source domain.

# Manifold Learning and Manifold Alignment

## Manifold learning (ML)

Aim to recover a low-dimensional manifold embedded in a high-dimensional ambient space.

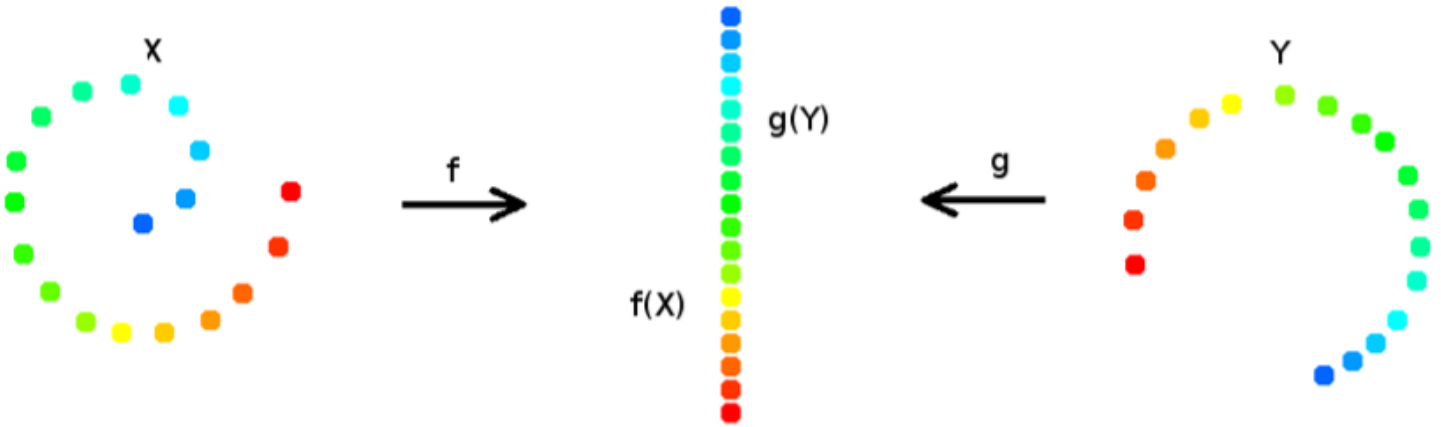
## Manifold alignment (MA)

Aim to project multiple sets of data into a common latent space, given that those original datasets lie on a common manifold.

- Most data sets can be modeled by manifolds.
- MA **aligns their underlying structures** and **transfers knowledge across** data sets.
- MA can be formalized as **dimensionality reduction with constraints** induced by the **correspondences** among data sets.

# Problem Statement

- Given datasets $X \in \mathbb{R}^{n \times p}$ and $Y \in \mathbb{R}^{m \times q}$ , whose instances lie on the same manifold $\mathcal{Z}$.
- MA is to find two functions $f : \mathbb{R}^p \to \mathbb{R}^k$, $g : \mathbb{R}^p \to \mathbb{R}^k$, such that $f(x_i)$ is close to $g(y_j)$ in terms of Euclidean distance if $x_i$ and $y_j$ are close w.r.t. geodesic distance along $\mathcal{Z}$.



# Problem Statement

- $x_i$ and $y_j$ are in exact correspondence $\Leftrightarrow f(x_i) = g(y_i)$.
  - **Prior correspondence** includes any information about the similarity of the instances in $X$ and $Y$, not just exact correspondence information.
- The union of the range of $f$ and $g$ is the joint latent space.
- The concatenation of the new coordinates $[f(X)^T, g(Y)^T]^T$ is the uniform representation of $X$ and $Y$ in the joint space.

# Practical Problems of MA

Many practical problems, ranging from bioinformatics to information retrieval and robotics, involve modeling multiple datasets that contain significant shared underlying structure.

- Data matching
- Protein alignment
- Cross-lingual retrieval
- Data integration
- Transfer learning

# Categories of MA

Instance-level vs. feature-level projections

- Linear (feature-level) alignment
- Non-linear (instance-level) alignment

Inter-data correspondences

- Supervised: with complete correspondence information.
- Semi-supervised: with incomplete correspondence information
- Unsupervised: with no correspondence information, some correspondences must be inferred.

# Two Key Ideas

- Consider local geometry and correspondence information.
- View multiple datasets as being samples on the same manifold.

# Two Key Ideas

- Consider local geometry and correspondence information.
- View multiple datasets as being samples on the same manifold.

$\Longrightarrow$

## Objective 1: Preserving Similarities

Preserve the local similarity and correspondence information.

## Objective 2: Embedding the Joint Laplacian

Form the joint Laplacian. MA is equivalent to Laplacian eigenmaps.

# Two Key Ideas

- Consider local geometry and correspondence information.
- View multiple datasets as being samples on the same manifold.

$\Longrightarrow$

## Objective 1: Preserving Similarities

Preserve the local similarity and correspondence information.

## Objective 2: Embedding the Joint Laplacian

Form the joint Laplacian. MA is equivalent to Laplacian eigenmaps.

The two objectives are equivalent with each other.

# Notations

- $X^{(a)} \in \mathbb{R}^{n_a \times p_a}$: the $a$-th matrix ($n_a$ observations and $p_a$ features).

- $W^{(a)} \in \mathbb{R}^{n_a \times n_a}$: the similarity matrix.
- $D^{(a)} \in \mathbb{R}^{n_a \times n_a}$: a diagonal matrix with $D^{(a)}(i,i) = \sum_j W^{(a)}(i,j)$.
- $L^{(a)} = D^{(a)} - W^{(a)}$: the Laplacian associated with $X^{(a)}$.

- $W^{(a,b)} \in \mathbb{R}^{n_a \times n_b}$: the similarity matrix between $X^{(a)}$ and $X^{(b)}$. $W^{(a,b)}(i,j)$ is the similarity or the strength of correspondence of the two instances $X^{(a)}(i,\cdot)$ and $X^{(b)}(j,\cdot)$.
Typically, $W^{(a,b)}(i,j) = 1$ if the instances $X^{(a)}(i,\cdot)$ and $X^{(b)}(j,\cdot)$ are in correspondence and 0 otherwise.

# Objective 1: Preserve Similarities

Given $X^{(1)}, \ldots, X^{(c)}$, let $F^{(a)}$ be the embedding of the $a$-th one. The objective 1 has two parts:

- Preserve local similarity within each dataset $X^{(a)}$

$$C_\lambda\left(F^{(a)}\right) = \sum_{i,j} \left\| F^{(a)}(i,\cdot) - F^{(a)}(j,\cdot) \right\|^2 W^{(a)}(i,j)$$

# Objective 1: Preserve Similarities

Given $X^{(1)}, \ldots, X^{(c)}$, let $F^{(a)}$ be the embedding of the $a$-th one. The objective 1 has two parts:

- Preserve local similarity within each dataset $X^{(a)}$

$$C_\lambda\left(F^{(a)}\right) = \sum_{i,j} \left\| F^{(a)}(i,\cdot) - F^{(a)}(j,\cdot) \right\|^2 W^{(a)}(i,j)$$

- Preserve correspondence information across datasets

$$C_\kappa\left(F^{(a)}, F^{(b)}\right) = \sum_{i,j} \left\| F^{(a)}(i,\cdot) - F^{(b)}(j,\cdot) \right\|^2 W^{(a,b)}(i,j)$$

# Objective 1: Preserve Similarities

The complete loss function is

$$
\begin{aligned}
& C_1\left(F^{(1)}, \ldots, F^{(c)}\right) \\
= \; & \nu \sum_a C_\lambda\left(F^{(a)}\right) + \mu \sum_{a \neq b} C_\kappa\left(F^{(a)}, F^{(b)}\right) \\
= \; & \nu \sum_a \sum_{i,j} \left\| F^{(a)}(i,\cdot) - F^{(a)}(j,\cdot) \right\|^2 W^{(a)}(i,j) \\
& + \mu \sum_{a \neq b} \sum_{i,j} \left\| F^{(a)}(i,\cdot) - F^{(b)}(j,\cdot) \right\|^2 W^{(a,b)}(i,j)
\end{aligned}
$$

where the scalers $\nu$ and $\mu$ are used to control how much the alignment should try to respect local similarity versus correspondence information. Typically, $\nu = \mu = 1$.

# Objective 2: Embedding the Joint Laplacian

$\mathbf{X}$ is the joint data matrix of $(\sum_i n_i) \times (\sum_i p_i)$
$\mathbf{W}$ is the joint similarity matrix of $(\sum_i n_i) \times (\sum_i n_i)$
$\mathbf{L} = \mathbf{D} - \mathbf{W}$ is the joint Laplacian

$$\mathbf{X} = \begin{pmatrix} X^{(1)} & \cdots & 0 \\ & \cdots & \\ 0 & \cdots & X^{(c)} \end{pmatrix}$$

$$\mathbf{W} = \begin{pmatrix} \nu W^{(1)} & \mu W^{(1,2)} & \cdots & \mu W^{(1,c)} \\ & & \cdots & \\ \mu W^{(c,1)} & \mu W^{(c,2)} & \cdots & \nu W^{(c)} \end{pmatrix}$$

where $\nu$ and $\mu$ are scalers that balance local similarity versus correspondence information, as defined before.

# Objective 2: Embedding the Joint Laplacian

The loss function for Laplacian eigenmaps:

$$C_2(\mathbf{F}) = \sum_{i,j} \|\mathbf{F}(i,\cdot) - \mathbf{F}(j,\cdot)\|^2 \mathbf{W}(i,j)$$

$$\begin{aligned} C_2(\mathbf{F}) &= \sum_{i,j}\sum_{k} [\mathbf{F}(i,k) - \mathbf{F}(j,k)]^2 \mathbf{W}(i,j) \\ &= \sum_{k}\sum_{i,j} [\mathbf{F}(i,k) - \mathbf{F}(j,k)]^2 \mathbf{W}(i,j) \\ &= \sum_{k} \mathrm{tr}\left(\mathbf{F}(\cdot,k)'\mathbf{L}\mathbf{F}(\cdot,k)\right) \\ &= \mathrm{tr}\left(\mathbf{F}'\mathbf{L}\mathbf{F}\right) \end{aligned}$$

# Equivalence of the Two Objectives

Check the definition of the joint Laplacian $\mathbf{L}$,

$$\mathbf{W}(i,j) = \begin{cases} \nu W^{(a)}(i,j) & \text{if } \mathbf{X}(i,\cdot) \text{ and } \mathbf{X}(j,\cdot) \text{ are both from } X^{(a)} \\ \mu W^{(a,b)}(i,j) & \text{if } \mathbf{X}(i,\cdot) \text{ and } \mathbf{X}(j,\cdot) \text{ are corresponding instances from } X^{(a)} \text{ and } X^{(b)} \\ 0 & \text{otherwise.} \end{cases}$$

The terms of $C_2(\mathbf{F})$ containing instances from the same dataset are exactly the $C_\lambda\left(F^{(a)}\right)$ terms, and the terms of containing instances from different datasets are exactly the $C_\kappa\left(F^{(a)}, F^{(b)}\right)$ terms.

$$C_1\left(F^{(1)}, \ldots, F^{(c)}\right) = C_2(\mathbf{F}) = C(\mathbf{F})$$

Embed the joint Laplacian $\Leftrightarrow$ Preserve local similarity and correspondence information.

# Optimization Problem

- An **additional constraint** is needed mathematically,

$$\mathbf{F}'\mathbf{D}\mathbf{F} = \mathbf{I}$$

  Without this constraint, the trivial solution of mapping all instances to zero would minimize the loss function.

- The final optimization formulation of MA is

$$\underset{\mathbf{F}:\mathbf{F}'\mathbf{D}\mathbf{F}=\mathbf{I}}{\arg\min} C(\mathbf{F}) = \underset{\mathbf{F}:\mathbf{F}'\mathbf{D}\mathbf{F}=\mathbf{I}}{\arg\min} \operatorname{tr}\left(\mathbf{F}'\mathbf{L}\mathbf{F}\right)$$

  - The optimal solution of it can be derived using the method of Lagrange multipliers.
  - Its solution $\Leftrightarrow$ that of Laplacian eigenmaps.

# Algorithm

## Joint Laplacian MA Algorithm

1. Find the adjacency matrices $W^{(1)}, \ldots, W^{(c)}$ of each dataset.
   - possibly only including a weight between two instances if one is in the k-nearest neighbors of the other.
2. Construct the joint Laplacian $\mathbf{L}$.
3. Compute the $d$ smallest nonzero eigenvectors of $\mathbf{L}f = \lambda \mathbf{D}f$.
4. The rows $1 + \sum_{l=0}^{g-1} n_l, 2 + \sum_{l=0}^{g-1} n_l, \ldots, n_g + \sum_{l=0}^{g-1} n_l$ of $\mathbf{F}$ are the new coordinates of $X^{(g)}$.

# Linear Restriction

## Nonlinear MA (as discussed above)

- The eigenvectors of the Laplacian are exactly the new coordinates of the embedded instances.
- No simple closed form for the mapping function.

# Linear Restriction

## Nonlinear MA (as discussed above)

- The eigenvectors of the Laplacian are exactly the new coordinates of the embedded instances.
- No simple closed form for the mapping function.

## Linear MA

- Enforces an explicit, linear embedding function.
- Useful for making out-of-sample estimates.
- Finds relationships between the features of multiple datasets instead of just between instances.
- Identify a common linear subspace of the original datasets.

# Linear MA

## Problem Statement

- Given two datasets $X^{(1)} \in \mathbb{R}^{n \times p}$ and $X^{(2)} \in \mathbb{R}^{m \times q}$, whose instances lie on the same manifold $\mathcal{Z}$.
- Linear alignment is to find two matrices $\mathbf{F}^{(1)} \in \mathbb{R}^{p \times k}$ and $\mathbf{F}^{(2)} \in \mathbb{R}^{q \times k}$, such that $x_i \mathbf{F}^{(1)}$ is close to $x_j \mathbf{F}^{(2)}$ in terms of Euclidean distance if $x_i^{(1)}$ and $x_j^{(2)}$ are close w.r.t. geodesic distance along $\mathcal{Z}$.

## The Loss Function

$$C(\mathbf{F}) = \sum_{i \neq j} \|\mathbf{X}(i,\cdot)\mathbf{F} - \mathbf{X}(j,\cdot)\mathbf{F}\|^2 \mathbf{W}(i,j) = \mathrm{tr}\left(\mathbf{F}'\mathbf{X}'\mathbf{L}\mathbf{X}\mathbf{F}\right)$$

# Linear MA

## The optimization problem

$$\max C(\mathbf{F}) = \text{tr}\left(\mathbf{F}'\mathbf{X}'\mathbf{L}\mathbf{X}\mathbf{F}\right)$$
$$\text{s.t. } \mathbf{F}'\mathbf{X}'\mathbf{D}\mathbf{X}\mathbf{F} = \mathbf{I}$$

- Linear alignment reduces to locality preserving projections on the joint Laplacian of the datasets.
- The solution is the minimum eigenvectors of the generalized eigenvector problem:

$$\mathbf{X}'\mathbf{L}\mathbf{X}f = \lambda\mathbf{X}'\mathbf{D}\mathbf{X}f$$

# Canonical Correlation Analysis (CCA)

- First introduced by [HOTELLING, 1936].
- Inferring information from cross-covariance matrices.

# Canonical Correlation Analysis (CCA)

- First introduced by [HOTELLING, 1936].
- Inferring information from cross-covariance matrices.

- $\mathbf{X}$, $\mathbf{Y}$: two data matrix with $N$ observations (rows) and $p$, $q$ features (columns), respectively.
- $\Sigma_{XX}$, $\Sigma_{YY}$: the within-set covariance matrix of $\mathbf{X}$ and $\mathbf{Y}$.
- $\Sigma_{XY}$: the between-sets covariance matrix (cross-covariance).

CCA is to find the vectors $u \in \mathbb{R}^p$ and $v \in \mathbb{R}^q$ that maximize the correlation between $\mathbf{X}u$ and $\mathbf{Y}v$.

# Optimization Problem of CCA

## Maximize the correlation after linear projection

$$\max_{u,v} \ \text{corr}(\mathbf{X}u, \mathbf{Y}v) = \frac{u^\top \Sigma_{XY} v}{\sqrt{(u^\top \Sigma_{XX} u)(v^\top \Sigma_{YY} v)}}$$

## The equivalent optimization problem

$$\max_{u,v} \ u^\top \Sigma_{XY} v$$

$$\text{subject to} \ \ u^\top \Sigma_{XX} u = 1 \text{ and } v^\top \Sigma_{YY} v = 1$$

# Optimization Problem of CCA

## Maximize the correlation after linear projection

$$\max_{u,v} \ \text{corr}(\mathbf{X}u, \mathbf{Y}v) = \frac{u^\top \Sigma_{XY} v}{\sqrt{(u^\top \Sigma_{XX} u)(v^\top \Sigma_{YY} v)}}$$

## The equivalent optimization problem

$$\max_{u,v} \ u^\top \Sigma_{XY} v$$

$$\text{subject to} \ \ u^\top \Sigma_{XX} u = 1 \text{ and } v^\top \Sigma_{YY} v = 1$$

The Lagrange function is

$$L(u, v) = u^\top \Sigma_{XY} v - \frac{\lambda}{2}(u^\top \Sigma_{XX} u - 1) - \frac{\mu}{2}(v^\top \Sigma_{YY} v - 1)$$

# CCA: The Optimal Solution

Differentiate w.r.t. $u$, $v$, $\lambda$ and $\mu$, and set to zeros:

$$\Sigma_{XY} v = \lambda \Sigma_{XX} u, \quad \Sigma_{XY} u = \lambda \Sigma_{YY} v$$
$$u^\top \Sigma_{XX} u = 1, \quad v^\top \Sigma_{YY} v = 1$$

which will gives: $\lambda = \mu$.
Find the vectors $u$ and $v$ by solving the eigenvalue problems:

$$\Sigma_{XX}^{-1} \Sigma_{XY} \Sigma_{YY}^{-1} \Sigma_{XY} u = \lambda^2 u$$
$$\Sigma_{YY}^{-1} \Sigma_{XY} \Sigma_{XX}^{-1} \Sigma_{XY} v = \lambda^2 v$$

For $d > 1$, the optimal solution is the eigenvectors of the $d$ biggest eigenvalues of $\Sigma_{XX}^{-1} \Sigma_{XY} \Sigma_{YY}^{-1} \Sigma_{XY}$ and $\Sigma_{YY}^{-1} \Sigma_{XY} \Sigma_{XX}^{-1} \Sigma_{XY}$.

# Summary

- Two ideas of manifold alignment: preserving the similarities and embedding the joint Laplacian.
- The consequent loss functions are equivalent.
- The non-linear embeddings generally produces more accurate alignments.
- Linear projections allow any new instances to be easily embedded in the manifold space.

# Correspondence Inference

How to infer the cross-datasets correspondences?

## Unsupervised case

- Using relations (rather than features) to represent local geometry [Chang and Mahadevan, 2009].
- Iterative methods [Cui et al., 2014].

## Semi-supervised case

- Building a bridge with known correspondences [Wang and Mahadevan, 2013].
- Utilize the class label information [Wang, 2010].

# Generalized Unsupervised MA (GUMA)

- Fully **unsupervised** matching without correspondences
- Formulated into an explicit integer optimization problem
- Efficient iterative alignment without computations in all permutation cases.

**Problem Setup:**

- Two datasets (normalized) $\mathbf{X} \in \mathbb{R}^{d_x \times n_x}, \mathbf{Z} \in \mathbb{R}^{d_z \times n_z}$(assume $n_x \leqslant n_z$), residing in two different manifolds $\mathcal{M}_x$ and $\mathcal{M}_z$
- The correspondences matrix $\mathbf{F} \in \{0,1\}^{n_x \times n_z}$

$$\Pi = \{\mathbf{F} | \mathbf{F} \in \{0,1\}^{n_x \times n_z}, \mathbf{F}\mathbf{1}_{n_z} = \mathbf{1}_{n_x}, \mathbf{1}_{n_x}^\top \mathbf{F} \leqslant \mathbf{1}_{n_z}^\top, n_x \leqslant n_z\}$$

- Linear projections $\mathbf{P}_x \in \mathbb{R}^{d \times d_x}$ and $\mathbf{P}_z \in \mathbb{R}^{d \times d_z} (d \leqslant d_x, d \leqslant d_z)$ from the two datasets to a mutual embedding space $\mathcal{M}$

# GUMA Problem

- **Overall objective function**

$$\min_{\mathbf{P}_x, \mathbf{P}_z, \mathbf{F}} E_s + \gamma_f E_f + \gamma_p E_p$$
$$\text{s.t. } \mathbf{F} \in \Pi, \ \mathbf{P}_x, \mathbf{P}_z \in \Theta$$

# GUMA Problem

- **Overall objective function**

$$\min_{\mathbf{P}_x, \mathbf{P}_z, \mathbf{F}} E_s + \gamma_f E_f + \gamma_p E_p$$
$$\text{s.t. } \mathbf{F} \in \Pi, \ \mathbf{P}_x, \mathbf{P}_z \in \Theta$$

- **The geometry matching term**

$$E_s = \|\mathbf{K}_x - \mathbf{F}\mathbf{K}_z\mathbf{F}^\top\|_F^2,$$

where $[\mathbf{K}]_{ij} = d(\mathbf{X}_{\cdot i}, \mathbf{X}_{\cdot j})$ is the full adjacency matrix, $d$ is geodesic distance.

- **The feature matching term**

$$E_f = \|\mathbf{P}_x^\top \mathbf{X} - \mathbf{P}_z^\top \mathbf{Z}\mathbf{F}^\top\|_F^2$$

which ensures the aligned data points have similar intrinsic feature representations in the mutual embedding space $\mathcal{M}$.

- **The feature matching term**

$$E_f = \|\mathbf{P}_x^\top \mathbf{X} - \mathbf{P}_z^\top \mathbf{Z}\mathbf{F}^\top\|_F^2$$

which ensures the aligned data points have similar intrinsic feature representations in the mutual embedding space $\mathcal{M}$.

- **The geometry preserving term**

$$\begin{aligned} E_p &= \sum_{i,j} \|\mathbf{P}_x^\top(\mathbf{x}_i - \mathbf{x}_j)\|^2 w_{ij}^x + \sum_{i,j} \|\mathbf{P}_z^\top(\mathbf{z}_i - \mathbf{z}_j)\|^2 w_{ij}^z \\ &= \operatorname{tr}(\mathbf{P}_x^\top \mathbf{X}\mathbf{L}_x\mathbf{X}^\top\mathbf{P}_x + \mathbf{P}_z^\top \mathbf{Z}\mathbf{L}_z\mathbf{Z}^\top\mathbf{P}_z). \end{aligned}$$

where $w_{ij}^x\,(w_{ij}^z)$ is the weight between the $i^{th}$ point and the $j^{th}$ point, $\mathbf{L}_x = \operatorname{diag}(\sum_j w_{1j}^x, \cdots, \sum_j w_{n_x j}^x) - \mathbf{W}_x$.

# GUMA Algorithm

1. Fix $\mathbf{P}_x$ and $\mathbf{P}_z \rightarrow$ solve a non-convex quadratic integer program
2. Fix $\mathbf{F} \rightarrow$ analytic solution for $\mathbf{P}_x$ and $\mathbf{P}_z$
3. Alternately optimize them

## Fixing $\mathbf{P}_x$ and $\mathbf{P}_z$

$$\min_{\mathbf{F} \in \Pi} \Psi_0(\mathbf{F}) = E_s + \gamma_f E_f$$

$$\Psi_0(\mathbf{F}) = \|\mathbf{K}_x\mathbf{F} - \mathbf{F}\mathbf{K}_z\|_F^2 + \mathrm{tr}(\mathbf{F}^\top \mathbf{1}\mathbf{1}^\top \mathbf{F}\mathbf{K}_{zz}) + \mathrm{tr}(\mathbf{F}^\top \mathbf{B})$$

where $\widehat{\mathbf{X}} = \mathbf{P}_x^\top \mathbf{X}$, $\widehat{\mathbf{Z}} = \mathbf{P}_z^\top \mathbf{Z}$, $\mathbf{K}_{zz} = \mathbf{K}_z \odot \mathbf{K}_z$ and $\mathbf{B} = \gamma_f(\mathbf{1}\mathbf{1}^\top(\widehat{\mathbf{Z}} \odot \widehat{\mathbf{Z}}) - 2\widehat{\mathbf{X}}^\top\widehat{\mathbf{Z}}) - \mathbf{1}\mathbf{1}^\top\mathbf{K}_{zz}$

# Subproblem about Learning Alignment

Relax this optimization problem

- Relax $\Pi$ into a compact convex set

$$\Pi' = \{\mathbf{F} | \mathbf{F} \geqslant 0, \mathbf{F}\mathbf{1}_{n_z} = \mathbf{1}_{n_x}, \mathbf{1}_{n_x}^\top \mathbf{F} \leqslant \mathbf{1}_{n_z}^\top, n_x \leqslant n_z\}$$

- Relax $\Psi_0$ into a convex function

$$\Psi(\mathbf{F}) = \|\mathbf{K}_x\mathbf{F} - \mathbf{F}\mathbf{K}_z\|_F^2 + \mathrm{tr}(\mathbf{F}^\top\mathbf{1}\mathbf{1}^\top\mathbf{F}\mathbf{K}_{zz} + \lambda\mathbf{F}^\top\mathbf{F}) + \mathrm{tr}(\mathbf{F}^\top\mathbf{B})$$

where $\lambda = n_x \times \max\{-\min(\mathrm{eig}(\mathbf{K}_{zz})), 0\}$

- $\Psi(\mathbf{F}_k)$ is non-increasing at each iteration and $\{\mathbf{F}_1, \mathbf{F}_2, \cdots\}$ will converge into a fixed point

# Subproblem about Learning Transformations

## Fixing **F**

The function to minimize

$$\begin{aligned} E_s &+ \gamma_p E_p \\ &= \operatorname{tr}(\mathbf{P}_x^\top \mathbf{X}(\gamma_f \mathbf{I} + \gamma_p \mathbf{L}_x)\mathbf{X}^\top \mathbf{P}_x + \mathbf{P}_z^\top \mathbf{Z}(\gamma_f \mathbf{F}^\top \mathbf{F} + \gamma_p \mathbf{L}_z)\mathbf{Z}^\top \mathbf{P}_z \\ &- 2\gamma_f \mathbf{P}_x^\top \mathbf{X}\mathbf{F}\mathbf{Z}^\top \mathbf{P}_z) \end{aligned}$$

Centralize $\mathbf{X}, \mathbf{Z}$ and reformulate the optimization problem

$$\begin{aligned} \max_{\mathbf{P}_x, \mathbf{P}_z} \quad & \operatorname{tr}\left(\mathbf{P}_x^\top \mathbf{X}\mathbf{F}\mathbf{Z}^\top \mathbf{P}_z\right) \\ \text{s.t.} \quad & \mathbf{P}_x^\top \mathbf{X}\left(\gamma_f \mathbf{I} + \gamma_p \mathbf{L}_x\right)\mathbf{X}^\top \mathbf{P}_x = \mathbf{I}, \mathbf{P}_z^\top \mathbf{Z}\left(\gamma_f \mathbf{F}^\top \mathbf{F} + \gamma_p \mathbf{L}_z\right)\mathbf{Z}^\top \mathbf{P}_z = \mathbf{I} \end{aligned}$$

Then use Canonical Correlation Analysis (CCA) to solve it.

# GUMA - Experiments

The balance parameters $\gamma_f, \gamma_p$ are set to 1. The embedding dimension $d$ is set to the minimal rank of two sets minus 1.

## GUMA for Set Matching (Expressions and Poses)

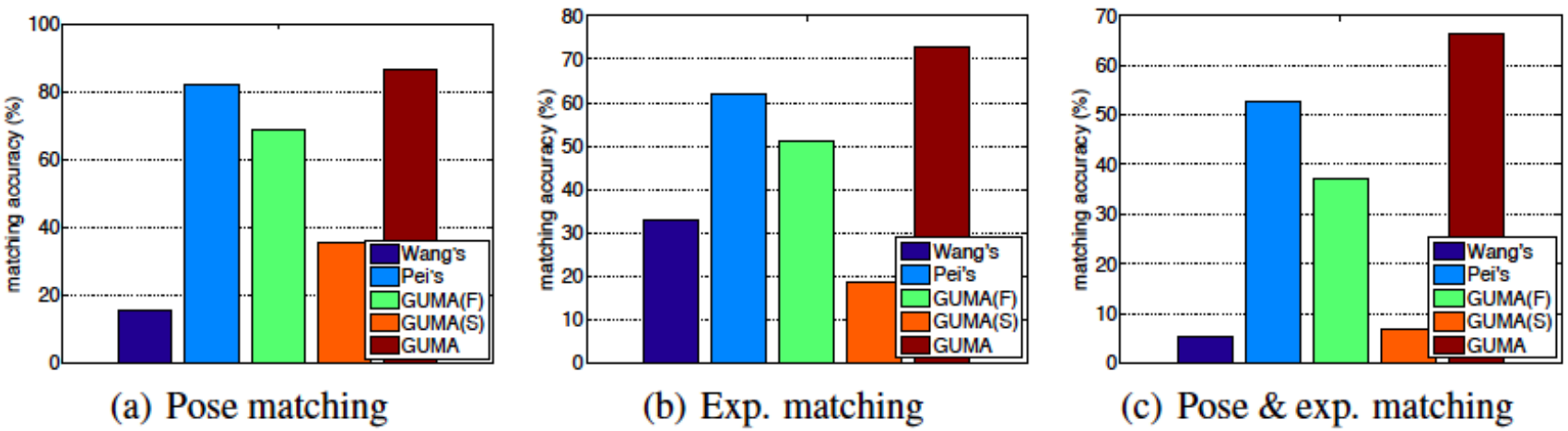
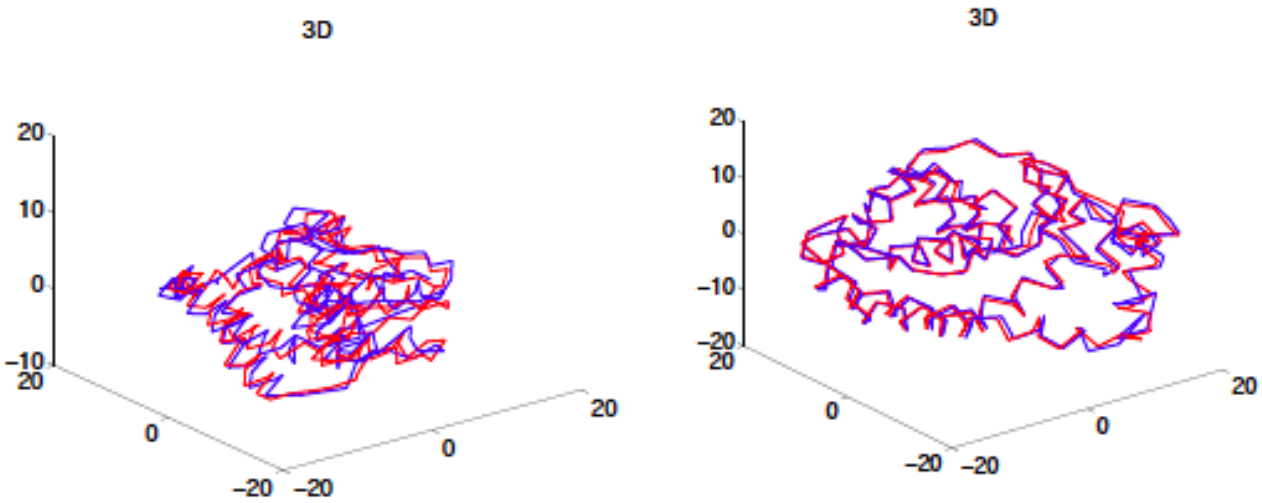



Figure: Alignment accuracy of face image sets from Multi-PIE [Cui et al., 2014] [Chang and Mahadevan, 2009] [Pei et al., 2012].

# GUMA - Experiments

## GUMA for Set Matching (Protein Structures)




(b) Wang's[29] (c) GUMA

Figure: The structure alignment results of two protein structures 1G7O-10 and 1G7O-21.

# GUMA - Experiments

## GUMA for Video-based Face Verification

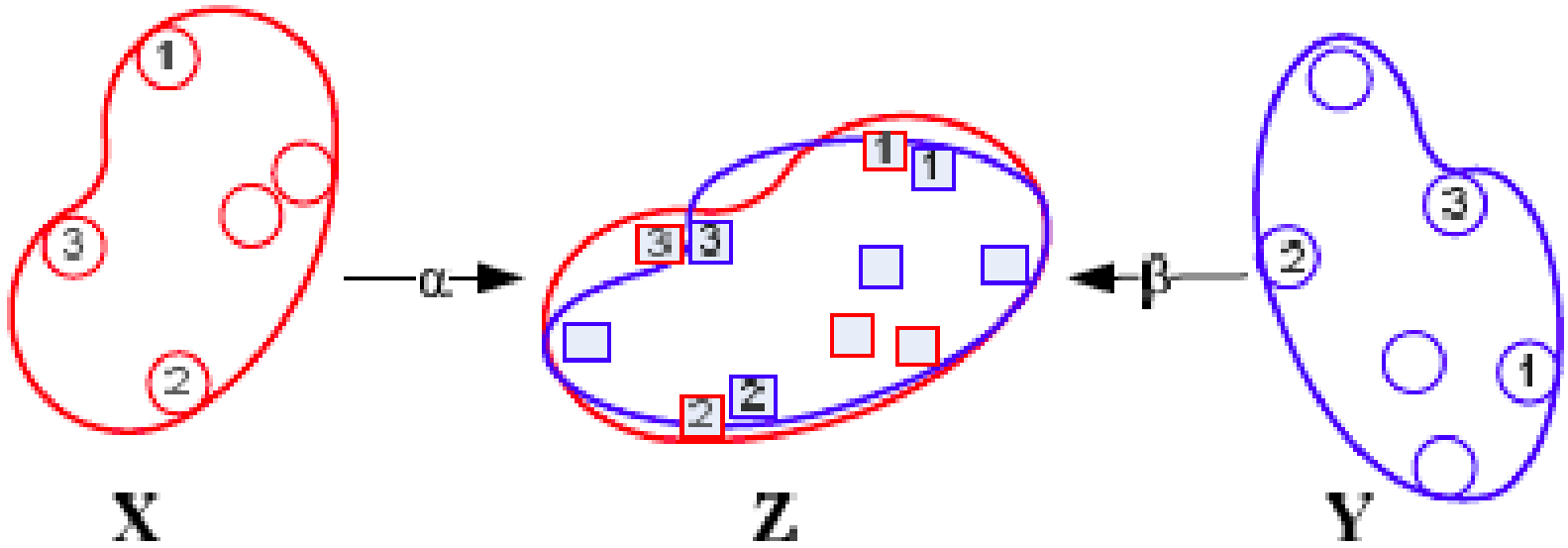
1. Align two videos by GUMA
2. Accumulate Euclidean distances of the counterparts as their dissimilarity

| Method | MSM[34] | MMD[31] | AHISD[4] | CHISD[4] | SANP[17] | DCC[18] | Unaligned | GUMA(un) | GUMA(su) |
|---|---|---|---|---|---|---|---|---|---|
| Mean Accuracy | 62.54 | 64.96 | 66.50 | 66.24 | 63.74 | 70.84 | 61.80 | 65.74 | 75.00 |
| Standard Error | ±1.47 | ±1.00 | ±2.03 | ±1.70 | ±1.69 | ±1.57 | ±2.29 | ±1.81 | ±1.56 |

Figure: The comparisons on YouTube faces dataset (%) [Cui et al., 2014].

# Motivation of Global MA

- The global manifold geometry needs to be respected.
- Directly represent the global geometry of the data.
- Using given correspondences, can deal with many to many correspondences.



# Global MA - Problem

## The Cost Function

$$C(\alpha, \beta, k) = \|\tau(\mathcal{D}) - \tau\left(\mathcal{D}_{X,Y,\alpha,\beta,k}\right)\|_2^2 =$$

$$\left\|\tau(\mathcal{D}) - k\left[\alpha^T X, \beta^T Y\right]^T \left[\alpha^T X, \beta^T Y\right]\right\|_2^2$$

where $\beta$ is a $d \times p$ matrix, $\alpha$ is a $d \times q$ matrix, $k$ is a positive number to rescale mapping functions.

# Global MA - Problem

## The Cost Function

$$C(\alpha, \beta, k) = \|\tau(\mathcal{D}) - \tau\left(\mathcal{D}_{X,Y,\alpha,\beta,k}\right)\|_2^2 =$$

$$\left\|\tau(\mathcal{D}) - k\left[\alpha^T X, \beta^T Y\right]^T \left[\alpha^T X, \beta^T Y\right]\right\|_2^2$$

where $\beta$ is a $d \times p$ matrix, $\alpha$ is a $d \times q$ matrix, $k$ is a positive number to rescale mapping functions.

- $\mathcal{D} = \begin{pmatrix} D_{x,x} & D_{x,y} \\ D_{y,x} & D_{y,y} \end{pmatrix}$ is a $(m+n) \times (m+n)$ matrix and $D_{x,y}(i,j) = \min_{u \in [1,l]} \left(D_{x,x}\left(x_i, x_{a_u}\right) + D_{y,y}\left(y_j, y_{b_u}\right)\right)$

# Global MA - Problem

## The Cost Function

$$C(\alpha, \beta, k) = \|\tau(\mathcal{D}) - \tau\left(\mathcal{D}_{X,Y,\alpha,\beta,k}\right)\|_2^2 =$$

$$\left\|\tau(\mathcal{D}) - k\left[\alpha^T X, \beta^T Y\right]^T \left[\alpha^T X, \beta^T Y\right]\right\|_2^2$$

where $\beta$ is a $d \times p$ matrix, $\alpha$ is a $d \times q$ matrix, $k$ is a positive number to rescale mapping functions.

- $\mathcal{D} = \begin{pmatrix} D_{x,x} & D_{x,y} \\ D_{y,x} & D_{y,y} \end{pmatrix}$ is a $(m+n) \times (m+n)$ matrix and $D_{x,y}(i,j) = \min_{u \in [1,l]} \left(D_{x,x}\left(x_i, x_{a_u}\right) + D_{y,y}\left(y_j, y_{b_u}\right)\right)$

# Global MA - Problem

## The Cost Function

$$C(\alpha, \beta, k) = \|\tau(\mathcal{D}) - \tau\left(\mathcal{D}_{X,Y,\alpha,\beta,k}\right)\|_2^2 =$$

$$\left\|\tau(\mathcal{D}) - k\left[\alpha^T X, \beta^T Y\right]^T \left[\alpha^T X, \beta^T Y\right]\right\|_2^2$$

where $\beta$ is a $d \times p$ matrix, $\alpha$ is a $d \times q$ matrix, $k$ is a positive number to rescale mapping functions.

- $\mathcal{D} = \begin{pmatrix} D_{x,x} & D_{x,y} \\ D_{y,x} & D_{y,y} \end{pmatrix}$ is a $(m+n) \times (m+n)$ matrix and $D_{x,y}(i,j) = \min_{u \in [1,l]} \left(D_{x,x}\left(x_i, x_{a_u}\right) + D_{y,y}\left(y_j, y_{b_u}\right)\right)$

- $\tau(A) = -HSH/2$ is a Gram matrix, $S_{i,j} = A_{i,j}^2$ is the matrix of squared distance, $H_{i,j} = \delta_{i,j} - 1/m$ is the "centering matrix".

# Global MA - Problem

## Theorem 1 [Wang and Mahadevan, 2013]

Let $Z = \begin{pmatrix} X & 0 \\ 0 & Y \end{pmatrix}$, then the eigenvectors corresponding to the $d$ maximum eigenvalues of $Z\tau(\mathcal{D})Z^T\gamma = \lambda ZZ^T\gamma$ provide optimal mappings to minimize $C(\alpha, \beta, k)$.

# Global MA - Algorithm

1. **Rescale data set $Y$:**
   $Y = \eta Y$, where $\eta = \text{tr}\left(D_b^T D_a\right) / \text{tr}\left(D_b^T D_b\right)$.

# Global MA - Algorithm

1. **Rescale data set $Y$:**
   $Y = \eta Y$, where $\eta = \text{tr}\left(D_b^T D_a\right) / \text{tr}\left(D_b^T D_b\right)$.
2. **Construct distance matrix $\mathcal{D}$, modeling the joint graph:**
   $\mathcal{D} = \begin{pmatrix} D_{x,x} & D_{x,y} \\ D_{y,x} & D_{y,y} \end{pmatrix}$, where
   $D_{y,x}(j,i) = D_{x,y}(i,j) = \min_{u \in [1,l]} \left(D_{x,x}\left(x_i, x_{a_u}\right) + D_{y,y}\left(y_j, y_{b_u}\right)\right)$.

# Global MA - Algorithm

1. **Rescale data set $Y$:**
   $Y = \eta Y$, where $\eta = \text{tr}\left(D_b^T D_a\right) / \text{tr}\left(D_b^T D_b\right)$.
2. **Construct distance matrix $\mathcal{D}$, modeling the joint graph:**
   $\mathcal{D} = \begin{pmatrix} D_{x,x} & D_{x,y} \\ D_{y,x} & D_{y,y} \end{pmatrix}$, where
   $D_{y,x}(j,i) = D_{x,y}(i,j) = \min_{u \in [1,l]} \left(D_{x,x}\left(x_i, x_{a_u}\right) + D_{y,y}\left(y_j, y_{b_u}\right)\right)$.
3. **Find the correspondence between $X$ and $Y$ :**
   Compute the eigenvectors $(\gamma_1, \gamma_2, \cdots, \gamma_d)$ corresponding to $d$ maximum eigenvalues of
   $$\begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix} \tau(\mathcal{D}) \begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix}^T \gamma = \lambda \begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix} \begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix}^T \gamma.$$

# Global MA - Algorithm

1. **Rescale data set $Y$:**
   $Y = \eta Y$, where $\eta = \text{tr}\left(D_b^T D_a\right) / \text{tr}\left(D_b^T D_b\right)$.
2. **Construct distance matrix $\mathcal{D}$, modeling the joint graph:**
   $\mathcal{D} = \begin{pmatrix} D_{x,x} & D_{x,y} \\ D_{y,x} & D_{y,y} \end{pmatrix}$, where
   $D_{y,x}(j,i) = D_{x,y}(i,j) = \min_{u \in [1,l]} \left(D_{x,x}\left(x_i, x_{a_u}\right) + D_{y,y}\left(y_j, y_{b_u}\right)\right)$.
3. **Find the correspondence between $X$ and $Y$ :**
   Compute the eigenvectors $(\gamma_1, \gamma_2, \cdots, \gamma_d)$ corresponding to $d$ maximum eigenvalues of
   $$\begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix} \tau(\mathcal{D}) \begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix}^T \gamma = \lambda \begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix} \begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix}^T \gamma.$$
4. **Construct $\alpha$ and $\beta$ to map $X$ and $Y$ to the same $d$-dimensional space:**
   The $d$-dimensional representations of $X$ and $Y$ are columns of $\alpha^T X$ and $\beta^T Y$, where $[\alpha \quad \beta]^T = [\gamma_1, \cdots, \gamma_d]$.

# Global MA - Experiments

English Arabic Cross-Lingual Retrieval

- **Two collections**: one in English and one in Arabic (manually translated), each of the two document collections has 2,119 documents.
- **The features**: constructed by the language model.
- **Our testing scheme**: for each given English document, we retrieve its top $K$ most similar Arabic documents. The probability that the true match is among the top $K$ documents is used to show the goodness of the method.

# Global MA - Experiments

## English Arabic Cross-Lingual Retrieval

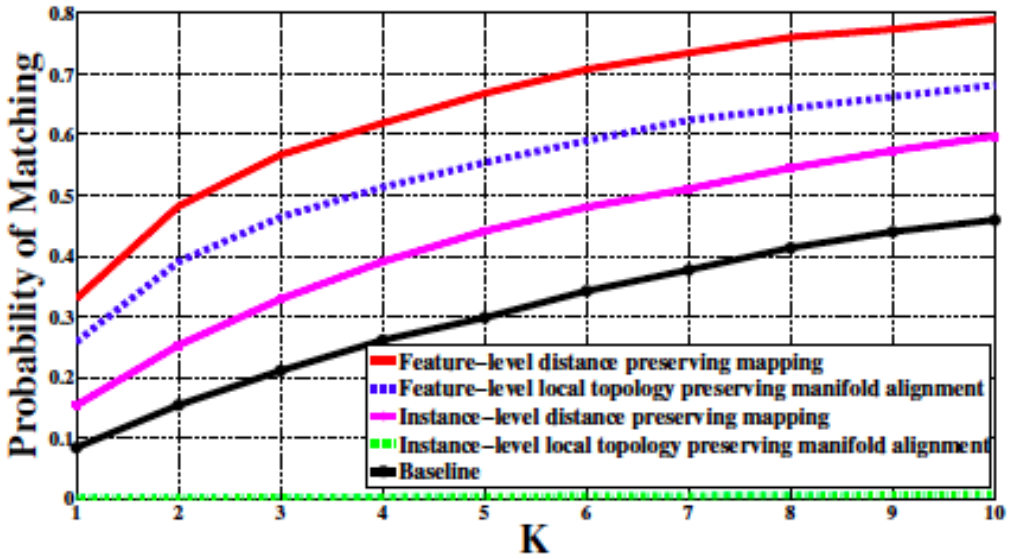



Figure: Test on English Arabic cross-lingual data (25% instances are in the given correspondence) [Wang and Mahadevan, 2013].

# Semi-supervised Manifold Alignment (SSMA)

- Make use of **class labels rather than correspondences** to align the manifolds.
- Handle the situation when the input domains do not share any common features or instances.

# Semi-supervised Manifold Alignment (SSMA)

- Make use of **class labels rather than correspondences** to align the manifolds.
- Handle the situation when the input domains do not share any common features or instances.

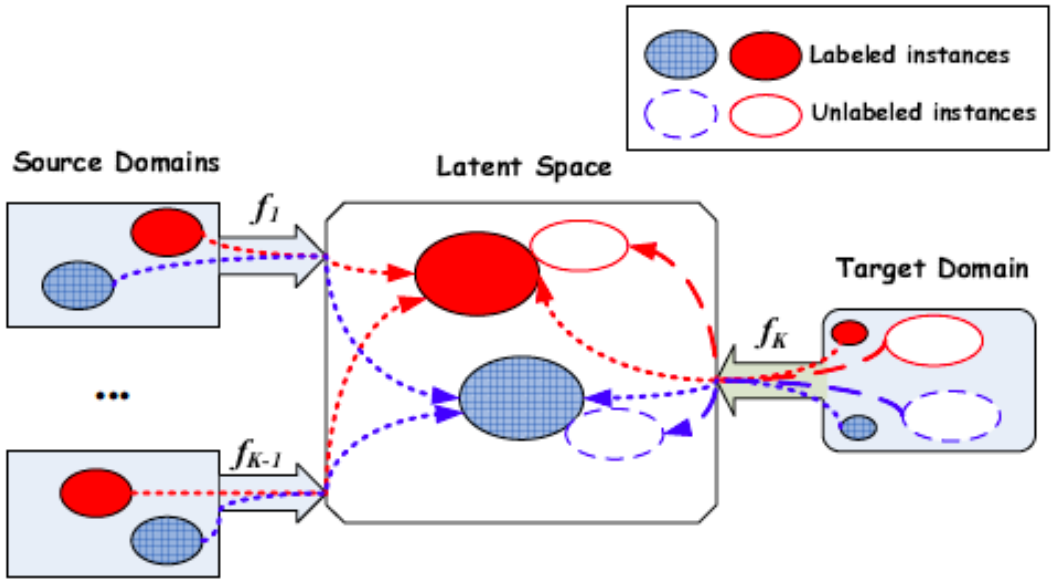



Figure: Illustration of manifold alignment using labels. Different colors represent different classes [Chang and Mahadevan, 2011].

# Idea of SSMA

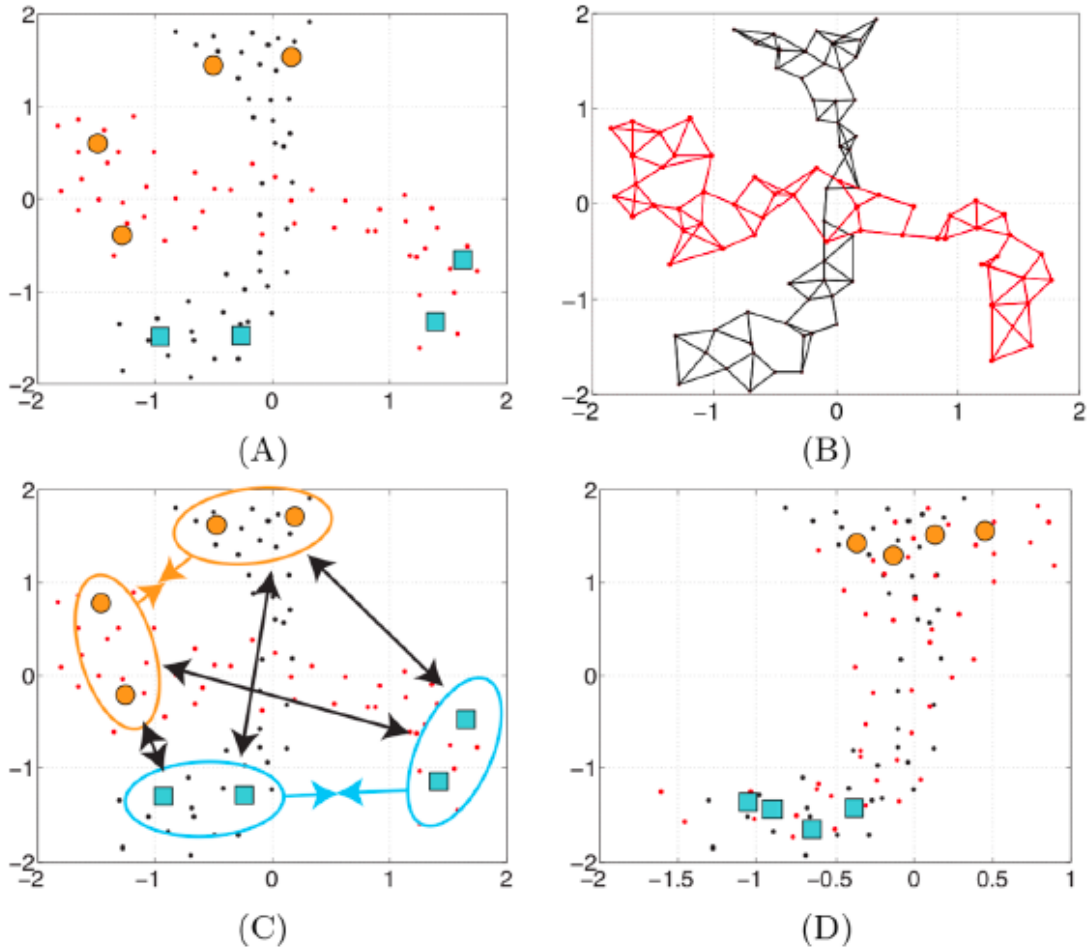
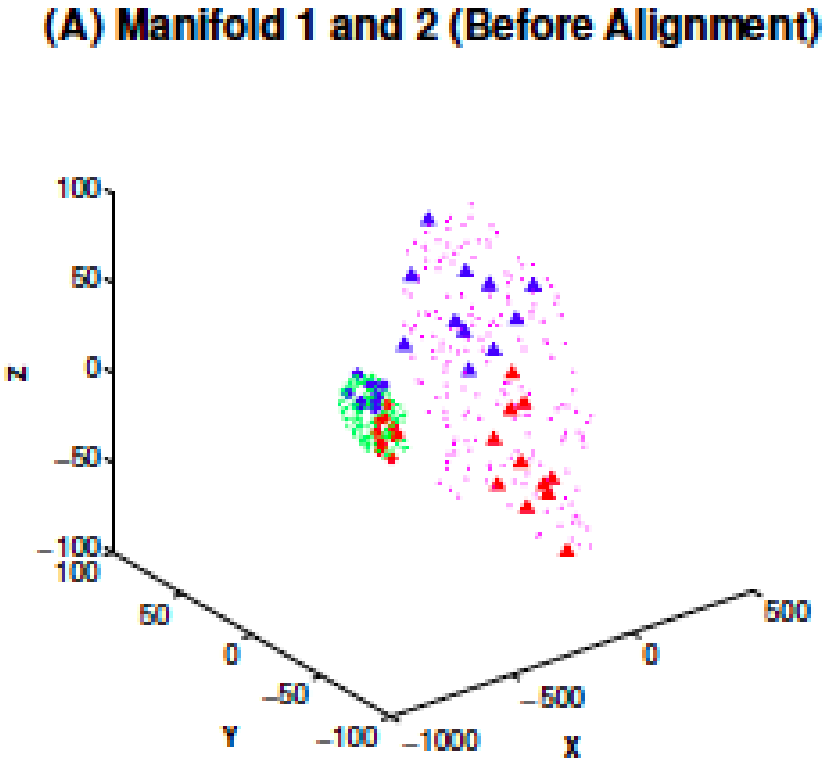
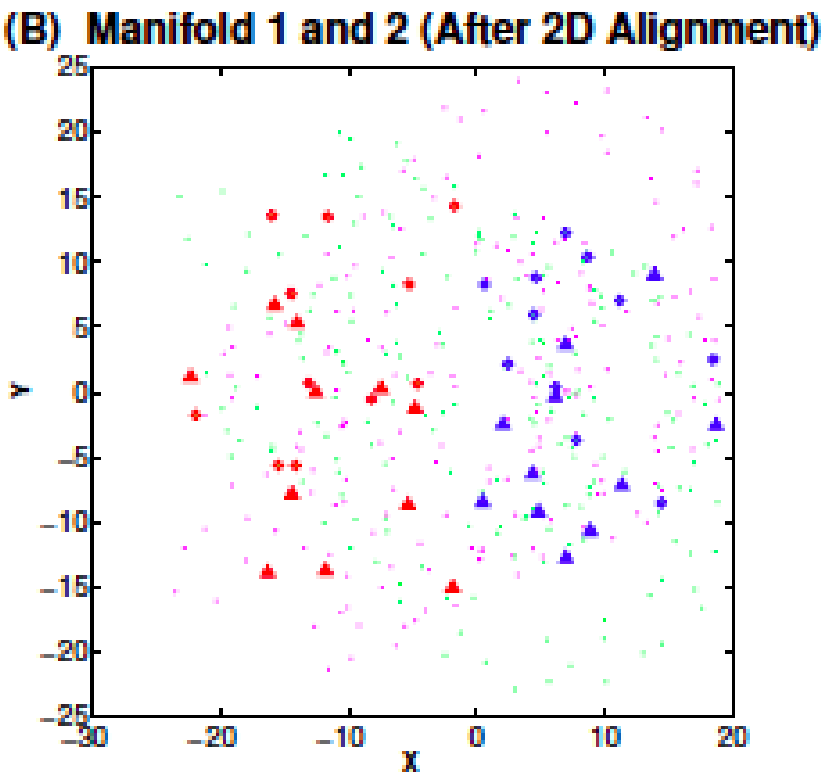



Figure: The idea behind SSMA [Devis et al., 2016].

# Illustration of SSMA



# SSMA - Problem

- Given $K$ input data sets $\mathbf{X}_k = (\mathbf{x}_k^1, \cdots, \mathbf{x}_k^{m_k})$, $k = 1, 2, \cdots, K$ with $p_k$ features each
- The labels for the first $l_k$ instances of $\mathbf{X}_k$ are given as $\mathbf{V}_k = (v_k^1, \cdots, v_k^{l_k})$.
- Similarity matrix $\mathbf{W}_s$ and dissimilarity matrix $\mathbf{W}_d$ are $(m_1 + \cdots + m_K) \times (m_1 + \cdots + m_K)$ matrix

$$\mathbf{W}_s = \begin{pmatrix} \mathbf{W}_s^{1,1} & \cdots & \mathbf{W}_s^{1,K} \\ & \cdots & \\ \mathbf{W}_s^{K,1} & \cdots & \mathbf{W}_s^{K,K} \end{pmatrix}, \mathbf{W}_d = \begin{pmatrix} \mathbf{W}_d^{1,1} & \cdots & \mathbf{W}_d^{1,K} \\ & \cdots & \\ \mathbf{W}_d^{K,1} & \cdots & \mathbf{W}_d^{K,K} \end{pmatrix},$$

where $\mathbf{W}_s^{a,b}(i,j) = 1$ ($\mathbf{W}_d^{a,b}(i,j) = 1$), if $\mathbf{x}_a^i$ and $\mathbf{x}_b^j$ are from the same class (different classes) and 0 otherwise.

# SSMA - Problem

- Let $\mathbf{W}_k(i,j)$ represent the similarity of $\mathbf{x}_k^i$ and $\mathbf{x}_k^j$. This similarity matrix can be computed as $e^{-\left\|\mathbf{x}_k^i - \mathbf{x}_k^j\right\|^2}$.
- The corresponding diagonal row sum matrix is defined as $\mathbf{D}_k(i,i) = \sum_j \mathbf{W}_k(i,j)$, $\mathbf{D}_s(i,i) = \sum_j \mathbf{W}_s(i,j)$ and $\mathbf{D}_d(i,i) = \sum_j \mathbf{W}_d(i,j)$ , and the combinatorial graph Laplacian matrix $\mathbf{L}_k = \mathbf{D}_k - \mathbf{W}_k$, $\mathbf{L}_s = \mathbf{D}_s - \mathbf{W}_s$ and $\mathbf{L}_d = \mathbf{D}_d - \mathbf{W}_d$.
- Matrices $\mathbf{L}$ and $\mathbf{Z}$ are defines as follows:
  - $\mathbf{L} = \text{diag}\{\mathbf{L}_1, \cdots, \mathbf{L}_K\}$ $((m_1 + \cdots + m_K) \times (m_1 + \cdots + m_K))$
  - $\mathbf{Z} = \text{diag}\{\mathbf{X}_1, \cdots, \mathbf{X}_K\}$ $((p_1 + \cdots + p_K) \times (m_1 + \cdots + m_K))$.

**Define** $A$, $B$ and $C$ with linear mapping

- The topology of each set is preserved ($\mu$ is a weight para.).

$$C = 0.5\mu \sum_{k=1}^{K} \sum_{i=1}^{m_k} \sum_{j=1}^{m_k} \|\mathbf{f}_k^\top \mathbf{x}_k^i - \mathbf{f}_k^\top \mathbf{x}_k^j\|^2 \mathbf{W}_k(i,j),$$

# SSMA - Problem

- The instances from the same class (across the input sets) are mapped to similar locations.

$$A = 0.5 \sum_{a=1}^{K} \sum_{b=1}^{K} \sum_{i=1}^{m_a} \sum_{j=1}^{m_b} ||\mathbf{f}_a^\top \mathbf{x}_a^i - \mathbf{f}_b^\top \mathbf{x}_b^j||^2 \mathbf{W}_s^{a,b}(i,j),$$

- The instances from different classes are well-separated from each other.

$$B = 0.5 \sum_{a=1}^{K} \sum_{b=1}^{K} \sum_{i=1}^{m_a} \sum_{j=1}^{m_b} ||\mathbf{f}_a^\top \mathbf{x}_a^i - \mathbf{f}_b^\top \mathbf{x}_b^j||^2 \mathbf{W}_d^{a,b}(i,j),$$

# SSMA - Problem

SSMA simultaneously minimize $A$, $C$, and maximize $B$. So the overall cost function $\mathcal{C}(\mathbf{f}_1, \cdots, \mathbf{f}_K)$ to be minimized is

$$\mathcal{C}(\mathbf{f}_1, \cdots, \mathbf{f}_K) = (A + C)/B.$$

## Theorem 1 ([Chang and Mahadevan, 2011])

*The embedding that minimizes the cost function* $\mathcal{C}(\mathbf{f}_1, \cdots, \mathbf{f}_K)$ *is given by the eigenvectors corresponding to the smallest non-zero eigenvalues of the generalized eigenvalue decomposition* $\mathbf{Z}(\mu\mathbf{L} + \mathbf{L}_s)\mathbf{Z}^\top\mathbf{x} = \lambda\mathbf{Z}\mathbf{L}_d\mathbf{Z}^\top\mathbf{x}$.

# SSMA - Experiments

## Text Categorization

- One document **only appear in one category**.
- Divide the data set into two subsets **of the same size**.
- Learn LSI (Latent Semantic Analysis) topics from the first subset and LDA (Latent Dirichlet Allocation) topics from the second subset.
- The labels for **all documents** in the first subset and **5%** of the second subset are available.
- Apply domain adaptation algorithms in this **common latent space** to learn the categories for unlabeled documents.

# SSMA - Experiments

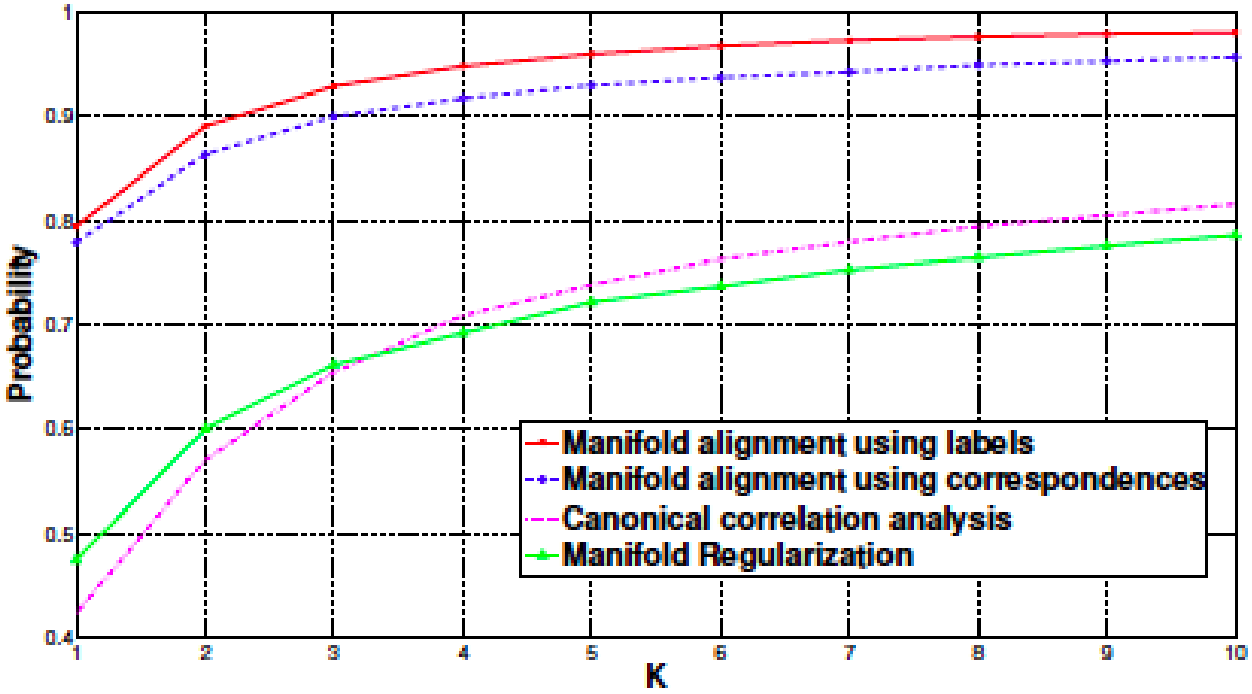



Figure: Text Categorization Test [Chang and Mahadevan, 2011].

# Kernel Manifold Alignment (KEMA)

- KEMA can align manifolds of very **different complexities** or data spaces of **different dimensionality**, and is robust to strong nonlinear feature deformations.

- KEMA generalizes other MA methods. When a linear kernel is used for all the domains, $\mathbf{K}_i = \mathbf{X}_i^\top \mathbf{X}_i$, KEMA reduces to SSMA.

- KEMA is especially well-suited to deal with **high-dimensional** problems (e.g., images and videos), and **under complicated distortions**, twists and warpings of the data manifolds.

# KEMA - Problem

- The original problem: $\mathbf{Z}(\mathbf{L} + \mu\mathbf{L}_s)\mathbf{Z}^\top\mathbf{x} = \lambda\mathbf{Z}\mathbf{L}_d\mathbf{Z}^\top\mathbf{x}$.
- $\phi_i(\cdot) : \mathbf{x} \to \phi(\mathbf{x}) \in \mathcal{H}_i$, $i = 1, 2, \cdots, D$.

# KEMA - Problem

- The original problem: $\mathbf{Z}(\mathbf{L}+\mu\mathbf{L}_s)\mathbf{Z}^\top\mathbf{x} = \lambda\mathbf{Z}\mathbf{L}_d\mathbf{Z}^\top\mathbf{x}$.
- $\phi_i(\cdot) : \mathbf{x} \to \phi(\mathbf{x}) \in \mathcal{H}_i$, $i = 1, 2, \cdots, D$.
- Replacing all the samples with their mapped feature vectors:

$$\mathbf{\Phi}(\mathbf{L}+\mu\mathbf{L}_s)\mathbf{\Phi}^\top\mathbf{U} = \lambda\mathbf{\Phi}\mathbf{L}_d\mathbf{\Phi}^\top\mathbf{U},$$

where $\mathbf{\Phi}$ is a block diagonal matrix containing the data matrices $\mathbf{\Phi}_i = (\phi_i(x_1), \cdots, \phi_i(x_{n_i}))^\top$.

# KEMA - Problem

- The original problem: $\mathbf{Z}(\mathbf{L}+\mu\mathbf{L}_s)\mathbf{Z}^\top\mathbf{x} = \lambda\mathbf{Z}\mathbf{L}_d\mathbf{Z}^\top\mathbf{x}$.
- $\phi_i(\cdot) : \mathbf{x} \to \phi(\mathbf{x}) \in \mathcal{H}_i$, $i = 1, 2, \cdots, D$.
- Replacing all the samples with their mapped feature vectors:

$$\mathbf{\Phi}(\mathbf{L}+\mu\mathbf{L}_s)\mathbf{\Phi}^\top\mathbf{U} = \lambda\mathbf{\Phi}\mathbf{L}_d\mathbf{\Phi}^\top\mathbf{U},$$

  where $\mathbf{\Phi}$ is a block diagonal matrix containing the data matrices $\mathbf{\Phi}_i = (\phi_i(x_1), \cdots, \phi_i(x_{n_i}))^\top$.
- The eigenvectors can be expressed by mapped samples: $\mathbf{u}_i = \mathbf{\Phi}_i\boldsymbol{\alpha}_i$, and in matrix notation $\mathbf{U} = \mathbf{\Phi}\mathbf{\Lambda}$. This leads to the problem:

$$\mathbf{\Phi}(\mathbf{L}+\mu\mathbf{L}_s)\mathbf{\Phi}^\top\mathbf{\Phi}\mathbf{\Lambda} = \lambda\mathbf{\Phi}\mathbf{L}_d\mathbf{\Phi}^\top\mathbf{\Phi}\mathbf{\Lambda},$$

# KEMA - Problem

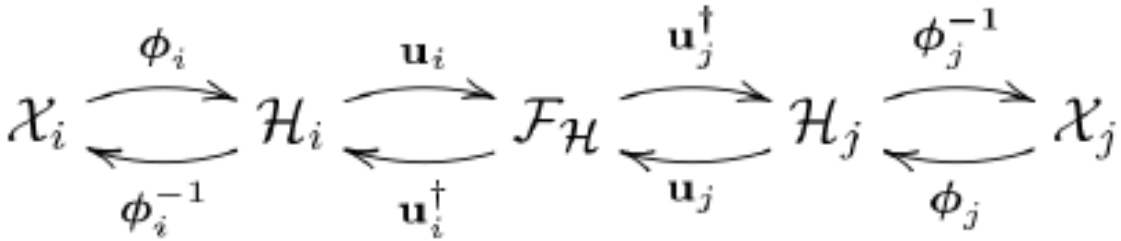
- Let $\mathbf{K}_i = \mathbf{\Phi}^\top \mathbf{\Phi}$, we obtain the final solution:

$$\mathbf{K}(\mathbf{L} + \mu \mathbf{L}_s)\mathbf{K}\mathbf{\Lambda} = \lambda \mathbf{K}\mathbf{L}_d\mathbf{K}\mathbf{\Lambda},$$

  where $\mathbf{K}$ is a block diagonal matrix containing the kernel $\mathbf{K}_i$.
- When a linear kernel is used for all the domains, $\mathbf{K}_i = \mathbf{X}_i^\top \mathbf{X}_i$, KEMA reduces to SSMA;
- Projection to the latent space:

$$P_{\mathcal{F}}(\mathbf{X}_i) = \mathbf{u}_i^\top \mathbf{\Phi}_i = \alpha_i \mathbf{\Phi}_i^\top \mathbf{\Phi}_i = \alpha_i \mathbf{K}_i.$$

- Can be depicted as:



# Learnable Manifold Alignment (LeMA)

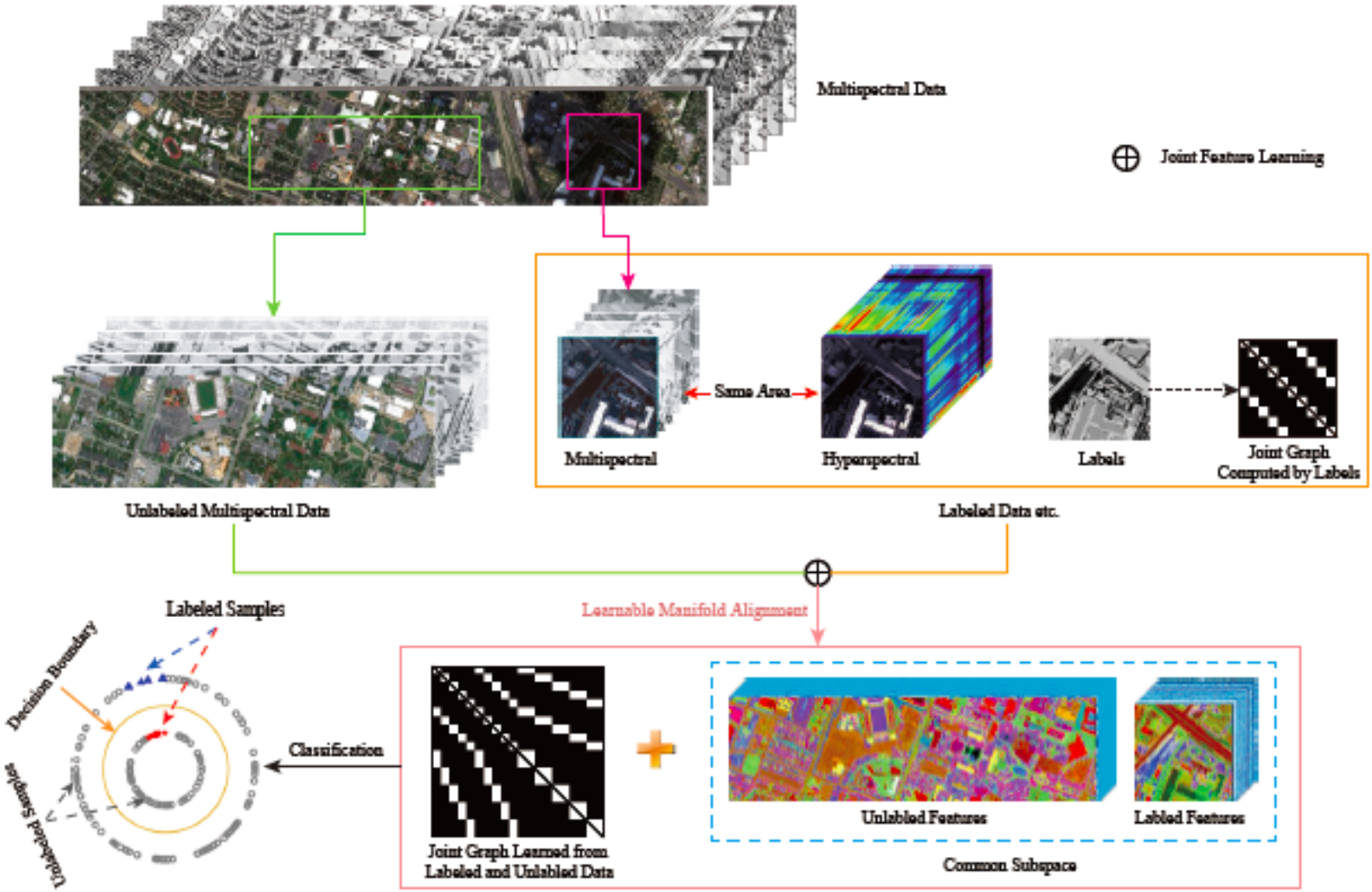
- Multispectral (MS) & hyperspectral (HS) imagery.
- Collect a large amount of labeled and discriminative data.
- Problem: Can a limited amount of HS training data partially overlapping MS data improve the performance of a classification task using a large coverage of MS testing data?

# Learnable Manifold Alignment (LeMA)

- Multispectral (MS) & hyperspectral (HS) imagery.
- Collect a large amount of labeled and discriminative data.
- Problem: Can a limited amount of HS training data partially overlapping MS data improve the performance of a classification task using a large coverage of MS testing data?

- Using a part of the data with labels to learn rest of the data (unlabeled).
- Simultaneously learn the projections, classification and Laplacian matrix.

# Illustration of LeMA [Hong et al., 2019]



# LeMA - Algorithm

**Algorithm 1:** Learnable Manifold Alignment (LeMA)

**Input:** $\widetilde{\mathbf{Y}}, \widetilde{\mathbf{X}}, \widetilde{\mathbf{X}}', \widetilde{\mathbf{L}}, \alpha, \beta, maxIter.$
**Output:** $\mathbf{P}, \mathbf{\Theta}, \widetilde{\mathbf{L}}$

$t = 1, \zeta = 1e-4;$
**Initializating** $\mathbf{P}$ and $\mathbf{\Theta}$
**while** *not converged* or $t > maxIter$ **do**
  Fix other variables to update $\mathbf{P}$ by Eq. (6)
  Fix other variables to update $\mathbf{\Theta}$ by **Algorithm 2**
  Fix other variables to update $\widetilde{\mathbf{L}}$ by equivalently optimizing $\widetilde{\mathbf{W}}$ in a distributed fashion:
    1. update $\widetilde{\mathbf{W}}_{HU}$ by **Algorithm 3**;
    2. update $\widetilde{\mathbf{W}}_{MU}$ by **Algorithm 3**;
    3. align $\widetilde{\mathbf{W}}_{HU}$ and $\widetilde{\mathbf{W}}_{MU}$ by $\max(\widetilde{\mathbf{W}}_{HU}, \widetilde{\mathbf{W}}_{MU})$;
    4. update $\widetilde{\mathbf{W}}_{UU}$ by **Algorithm 4**
    5. compute $\widetilde{\mathbf{L}} = \widetilde{\mathbf{D}} - \widetilde{\mathbf{W}}, \widetilde{\mathbf{D}}_{ii} = \sum_{i \neq j} \widetilde{\mathbf{W}}_{ij}$
  Compute the objective function value $E^{t+1}$ and check the convergence condition: **if** $|\frac{E^{t+1} - E^t}{E^t}| < \zeta$ **then**
    Stop iteration;
  **else**
    $t \leftarrow t + 1;$
  **end**
**end**

# LeMA - Problem

## The common subspace learning problem (CoSpace)

$$\min_{\mathbf{P},\Theta} \frac{1}{2}\|\widetilde{\mathbf{Y}} - \mathbf{P}\Theta\widetilde{\mathbf{X}}\|_{\mathrm{F}}^2 + \frac{\alpha}{2}\|\mathbf{P}\|_{\mathrm{F}}^2 + \frac{\beta}{2}\,\mathrm{tr}\left(\mathbf{ELE}^{\mathrm{T}}\right)$$

$$\text{s.t. } \mathbf{E} = \Theta\widetilde{\mathbf{X}}, \Theta\Theta^{\mathrm{T}} = \mathbf{I}$$

fidelity term + regularization term + supervised MA

| | |
|---|---|
| $\widetilde{\mathbf{Y}} = [\mathbf{Y}, \mathbf{Y}] \in \mathbb{R}^{d\times 2N}$, $\mathbf{Y} \in \mathbb{R}^{d\times N}$ | the label matrix (one-hot encoding) |
| $\widetilde{\mathbf{X}} = \begin{bmatrix} \mathbf{X}_H & \mathbf{0} \\ \mathbf{0} & \mathbf{X}_M \end{bmatrix} \in \mathbb{R}^{(d_H+d_M)\times 2N}$ | the data matrix |
| $\Theta_H = [\Theta_H, \Theta_M]$ | the common subspace projection |
| $\mathbf{P}$ | the linear projection of the common subspace and label information |
| $\mathbf{L} = \mathbf{D} - \mathbf{W}$ | joint Laplacian matrix |

# LeMA - Problem

$$\min_{\mathbf{P},\Theta,\widetilde{\mathbf{L}}} \frac{1}{2}\|\widetilde{\mathbf{Y}} - \mathbf{P}\Theta\widetilde{\mathbf{X}}\|_{\mathrm{F}}^2 + \frac{\alpha}{2}\|\mathbf{P}\|_{\mathrm{F}}^2 + \frac{\beta}{2}\,\mathrm{tr}\left(\mathbf{H}\widetilde{\mathbf{L}}\mathbf{H}^{\mathrm{T}}\right)$$
$$\text{s.t. } \mathbf{H} = \Theta\widetilde{\mathbf{X}}', \Theta\Theta^{\mathrm{T}} = \mathbf{I}, \widetilde{\mathbf{L}} = \widetilde{\mathbf{L}}^{\mathrm{T}}, \widetilde{\mathbf{L}}_{i,j,i\neq j} \preceq 0, \widetilde{\mathbf{L}}_{i,j,i=j} \succeq 0, \mathrm{tr}(\widetilde{\mathbf{L}}) = s$$

where $\widetilde{\mathbf{X}}' = \begin{bmatrix} \mathbf{X}_H & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{X}_M & \mathbf{X}_U \end{bmatrix} \in \mathbb{R}^{(d_H+d_M)\times(2N+N_U)}$,
$\widetilde{\mathbf{L}} \in \mathbb{R}^{(2N+N_U)\times(2N+N_U)}$, $\mathbf{X}_U \in \mathbb{R}^{d_M\times N_U}$ represents the unlabeled MS samples, and $s > 0$ controls the scale.

# Model Optimization

## Optimization with respect to **P**

a typical least-squares problem

$$\min_{\mathbf{P}} \frac{1}{2}\|\widetilde{\mathbf{Y}} - \mathbf{P}\Theta\widetilde{\mathbf{X}}\|_{\mathrm{F}}^2 + \frac{\alpha}{2}\|\mathbf{P}\|_{\mathrm{F}}^2$$

which has a closed-form solution

$$\mathbf{P} = \left(\widetilde{\mathbf{Y}}\mathbf{E}^{\mathrm{T}}\right)\left(\mathbf{E}\mathbf{E}^{\mathrm{T}} + \alpha\mathbf{I}\right)^{-1},$$

where $\mathbf{E} = \Theta\widetilde{\mathbf{X}}$.

# Model Optimization

## Optimization with respect to $\Theta$

$$\min_{\Theta} \frac{1}{2}\|\widetilde{\mathbf{Y}} - \mathbf{P}\Theta\widetilde{\mathbf{X}}\|_{\mathrm{F}}^2 + \frac{\beta}{2}\operatorname{tr}\left(\mathbf{H}\widetilde{\mathbf{L}}\mathbf{H}^{\mathrm{T}}\right) \text{ s.t. } \mathbf{H} = \Theta\widetilde{\mathbf{X}}', \Theta\Theta^{\mathrm{T}} = \mathbf{I}$$

which has an equivalent form

$$\min_{\Theta,\mathbf{J},\mathbf{G}} \frac{1}{2}\|\widetilde{\mathbf{Y}} - \mathbf{P}\mathbf{J}\|_{\mathrm{F}}^2 + \frac{\beta}{2}\operatorname{tr}\left(\Theta\widetilde{\mathbf{X}}'\widetilde{\mathbf{L}}\left(\Theta\widetilde{\mathbf{X}}'\right)^{\mathrm{T}}\right)$$
$$\text{s.t. } \mathbf{J} = \Theta\widetilde{\mathbf{X}}, \mathbf{G} = \Theta, \mathbf{G}\mathbf{G}^{\mathrm{T}} = \mathbf{I}.$$

This problem can be solved with ADMM.

# Model Optimization

**Algorithm 2:** Solving the subproblem for $\boldsymbol{\Theta}$

**Input:** $\widetilde{\mathbf{Y}}, \mathbf{P}, \mathbf{J}, \widetilde{\mathbf{X}}, \widetilde{\mathbf{X}}', \widetilde{\mathbf{L}}, \beta, maxIter$.

**Output:** $\boldsymbol{\Theta}$.

**Initialization:** $\boldsymbol{\Theta} = \mathbf{0}, \mathbf{G} = \mathbf{0}, \boldsymbol{\Lambda}_1 = \mathbf{0}, \boldsymbol{\Lambda}_2 = \mathbf{0}, \mu = 10^{-3}, \mu_{\max} = 10^6, \rho = 1.5, \varepsilon = 10^{-6}, t = 1$.

**while** *not converged* or $t > maxIter$ **do**

  Fix other variables to update $\mathbf{J}$ by $\mathbf{J} = (\mathbf{P}^{\mathrm{T}}\mathbf{P} + \mu\mathbf{I})^{-1}(\mathbf{P}^{\mathrm{T}}\widetilde{\mathbf{Y}} + \mu\boldsymbol{\Theta}\widetilde{\mathbf{X}} - \boldsymbol{\Lambda}_1)$.

  Fix other variables to update $\boldsymbol{\Theta}$ by

$$\boldsymbol{\Theta} = (\mu\mathbf{J}\widetilde{\mathbf{X}}^{\mathrm{T}} + \boldsymbol{\Lambda}_1\widetilde{\mathbf{X}}^{\mathrm{T}} + \mu\mathbf{G} + \boldsymbol{\Lambda}_2) \times (\mu\widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^{\mathrm{T}} + \mu\mathbf{I} + \beta\widetilde{\mathbf{X}}'\widetilde{\mathbf{L}}\widetilde{\mathbf{X}}'^{\mathrm{T}})^{-1}.$$

  Fix other variables to update $\mathbf{G}$ by

$$[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \mathrm{svd}(\boldsymbol{\Theta} - \boldsymbol{\Lambda}_2/\mu), \quad \mathbf{G} = \mathbf{U}\mathbf{I}_{n\times m}\mathbf{V}.$$

  Update Lagrange multipliers by

$$\boldsymbol{\Lambda}_1 \leftarrow \boldsymbol{\Lambda}_1 + \mu(\mathbf{J} - \boldsymbol{\Theta}\widetilde{\mathbf{X}}), \quad \boldsymbol{\Lambda}_2 \leftarrow \boldsymbol{\Lambda}_2 + \mu(\mathbf{G} - \boldsymbol{\Theta}).$$

  Update penalty parameter by $\mu = \min(\rho\mu, \mu_{\max})$.

  Check the convergence conditions: **if** $\|\mathbf{J} - \boldsymbol{\Theta}\widetilde{\mathbf{X}}\|_{\mathrm{F}} < \varepsilon$ *and* $\|\mathbf{G} - \boldsymbol{\Theta}\|_{\mathrm{F}} < \varepsilon$ **then**

    Stop iteration;

  **else**

    $t \leftarrow t + 1$;

  **end**

**end**

# Model Optimization

## Optimization with respect to $\widetilde{\mathbf{W}}$

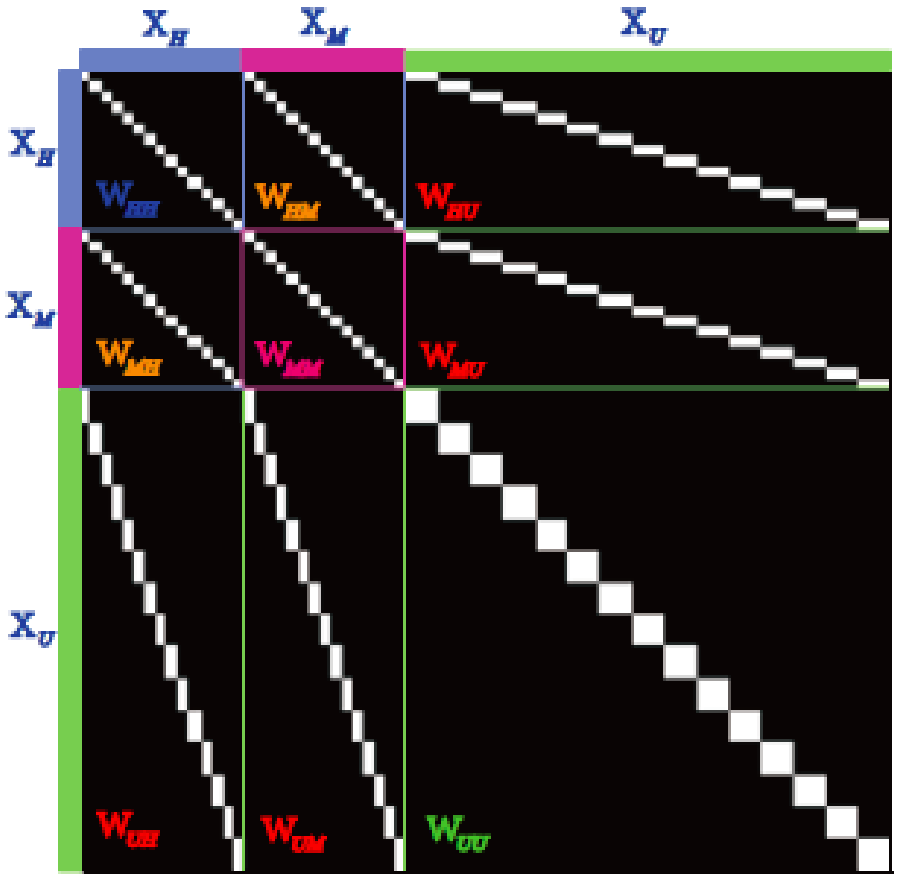



Figure: An example for the joint adjacency matrix $\widetilde{\mathbf{W}}$.

# Model Optimization

## The optimization problems on $\widetilde{\mathbf{W}}_{HU}$ and $\widetilde{\mathbf{W}}_{MU}$

$$\min_{\widetilde{\mathbf{w}}_{HU(MU)}} \frac{\beta}{4}\|\widetilde{\mathbf{W}} \odot \mathbf{Z}\|_{1,1} \text{ s.t. } 1/N_k \succeq \widetilde{\mathbf{W}}_{i,j} \succeq 0, \|\widetilde{\mathbf{W}}\|_{1,1} = s$$

which can be solved by ADMM.

## The optimization problems on $\widetilde{\mathbf{W}}_{UU}$

$$\min_{\widetilde{\mathbf{w}}_{UU}} \frac{\beta}{4}\|\widetilde{\mathbf{W}} \odot \mathbf{Z}\|_{1,1} \text{ s.t. } \widetilde{\mathbf{W}} = \widetilde{\mathbf{W}}^{\mathrm{T}}, 1/N_k \succeq \widetilde{\mathbf{W}}_{i,j} \succeq 0, \|\widetilde{\mathbf{W}}\|_{1,1} = s.$$

# LeMA - Algorithm

**Algorithm 1:** Learnable Manifold Alignment (LeMA)

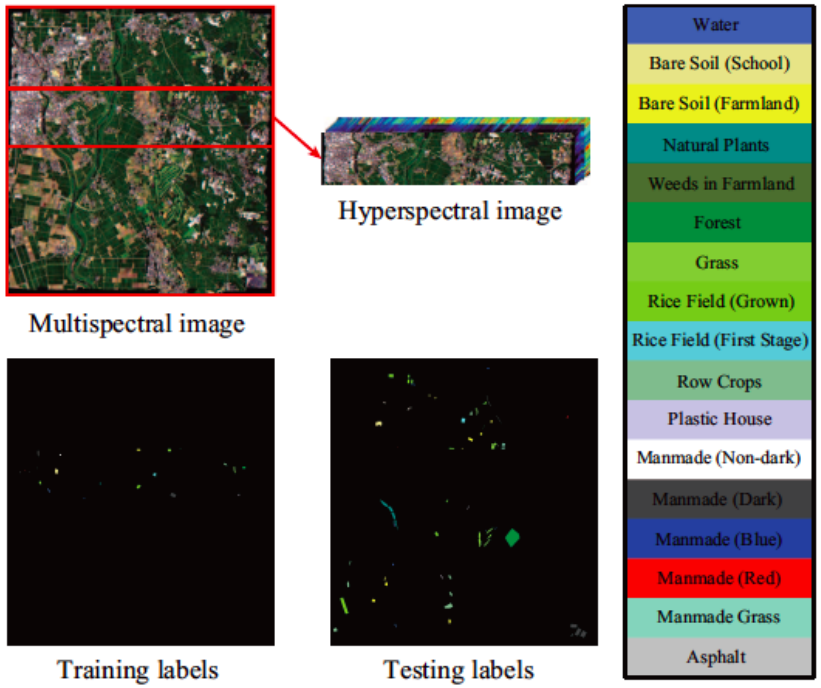
**Input:** $\widetilde{\mathbf{Y}}, \widetilde{\mathbf{X}}, \widetilde{\mathbf{X}}', \widetilde{\mathbf{L}}, \alpha, \beta, maxIter.$
**Output:** $\mathbf{P}, \mathbf{\Theta}, \widetilde{\mathbf{L}}$

$t = 1, \zeta = 1e-4;$
**Initializating** $\mathbf{P}$ and $\mathbf{\Theta}$
**while** *not converged* or $t > maxIter$ **do**
  Fix other variables to update $\mathbf{P}$ by Eq. (6)
  Fix other variables to update $\mathbf{\Theta}$ by **Algorithm 2**
  Fix other variables to update $\widetilde{\mathbf{L}}$ by equivalently optimizing $\widetilde{\mathbf{W}}$ in a distributed fashion:
    1. update $\widetilde{\mathbf{W}}_{HU}$ by **Algorithm 3**;
    2. update $\widetilde{\mathbf{W}}_{MU}$ by **Algorithm 3**;
    3. align $\widetilde{\mathbf{W}}_{HU}$ and $\widetilde{\mathbf{W}}_{MU}$ by $\max(\widetilde{\mathbf{W}}_{HU}, \widetilde{\mathbf{W}}_{MU})$;
    4. update $\widetilde{\mathbf{W}}_{UU}$ by **Algorithm 4**
    5. compute $\widetilde{\mathbf{L}} = \widetilde{\mathbf{D}} - \widetilde{\mathbf{W}}, \widetilde{\mathbf{D}}_{ii} = \sum_{i \neq j} \widetilde{\mathbf{W}}_{ij}$
  Compute the objective function value $E^{t+1}$ and check the convergence condition: **if** $|\frac{E^{t+1} - E^t}{E^t}| < \zeta$ **then**
    Stop iteration;
  **else**
    $t \leftarrow t + 1;$
  **end**
**end**

# LeMA - Experiments

## The Simulated MS-HS Datasets over Chikusei

- MS data: $2517 \times 2335 \times 10$ at a GSD of 2.5 m.
- HS data: 128 bands in the spectral range from 363nm to 1018nm with the 10nm spectral resolution.



# LeMA - Experiments

## The Simulated MS-HS Datasets over Chikusei

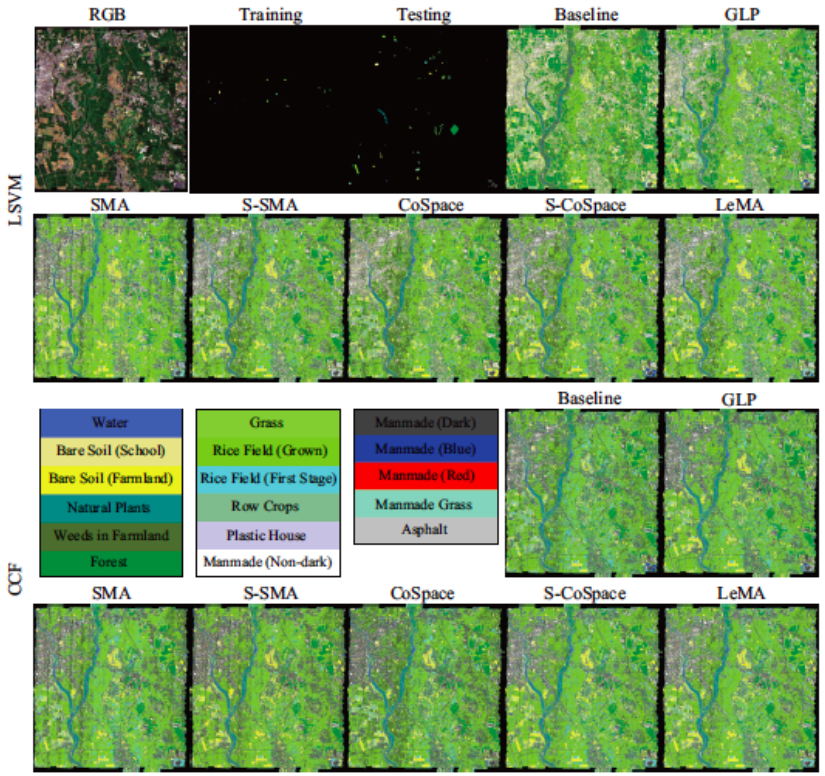



Figure: Classification maps of the different algorithms obtained using two kinds of classifiers on the Chikusei dataset [Hong et al., 2019].

# LeMA - Experiments

## The Simulated MS-HS Datasets over Chikusei

Table 2: Quantitative performance comparison with the different algorithms on the University of Houston data. The best one is shown in bold.

| Methods | Baseline (%) | | GLP (%) | | SMA (%) | | S-SMA (%) | | CoSpace (%) | | S-CoSpace (%) | | LeMA (%) | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Parameter | $d$ | | $(k, \sigma, d)$ | | $d$ | | $(k, \sigma, d)$ | | $(\alpha, \beta, d)$ | | $(\alpha, \beta, d)$ | | $(\alpha, \beta, d)$ | |
| | 10 | | (10, 1, 10) | | 30 | | (10, 0.1, 30) | | (0.01, 0.01, 30) | | (0.1, 0.01, 30) | | (0.01, 0.01, 30) | |
| Classifier | LSVM | CCF | LSVM | CCF | LSVM | CCF | LSVM | CCF | LSVM | CCF | LSVM | CCF | LSVM | CCF |
| OA | 62.12 | 68.21 | 64.71 | 70.01 | 68.01 | 69.59 | 69.29 | 70.10 | 69.38 | 72.17 | 70.41 | 73.75 | 73.42 | **76.35** |
| AA | 65.97 | 70.47 | 68.18 | 72.18 | 70.50 | 71.02 | 72.00 | 72.88 | 71.69 | 73.56 | 73.12 | 75.61 | 74.76 | **77.18** |
| $\kappa$ | 0.5889 | 0.6543 | 0.6164 | 0.6728 | 0.6520 | 0.6695 | 0.6659 | 0.6754 | 0.6672 | 0.6975 | 0.6784 | 0.7146 | 0.7110 | **0.7428** |
| Class1 | 76.39 | 67.95 | 77.83 | 77.97 | 75.25 | 68.53 | 74.25 | 73.53 | 75.54 | 69.96 | **91.85** | 87.98 | 89.56 | 85.84 |
| Class2 | 80.59 | 78.08 | 93.85 | **98.01** | 97.57 | 77.9 | 97.57 | 93.67 | 73.74 | 77.99 | 90.12 | 91.59 | 93.67 | 93.85 |
| Class3 | **100.00** | **100.00** | **100.00** | **100.00** | **100.00** | **100.00** | **100.00** | **100.00** | **100.00** | **100.00** | **100.00** | **100.00** | **100.00** | **100.00** |
| Class4 | 85.51 | 92.27 | 89.66 | 96.62 | 94.78 | 98.74 | 95.85 | 98.55 | 98.74 | 98.26 | 92.75 | 97.29 | 97.49 | **99.61** |
| Class5 | 99.06 | 99.4 | 99.49 | **99.66** | 98.97 | 99.14 | 99.32 | 99.4 | 99.4 | 99.4 | 99.4 | **99.66** | 99.49 | 99.57 |
| Class6 | 86.14 | 86.14 | 96.37 | 99.01 | 86.47 | 70.96 | **99.67** | 99.67 | 85.48 | 85.15 | **99.67** | 96.70 | 86.47 | 86.47 |
| Class7 | 50.62 | 63.76 | 48.63 | 64.01 | 72.32 | 77.14 | 72.15 | 69.66 | 73.98 | 80.05 | 75.06 | 80.96 | 83.21 | **88.03** |
| Class8 | 56.49 | 56.06 | 56.60 | 59.85 | 62.01 | 62.23 | **64.61** | 63.85 | 63.53 | 62.01 | 55.84 | 60.39 | 62.77 | 62.01 |
| Class9 | 56.22 | 70.58 | 69.63 | 69.02 | 49.96 | 61.27 | 50.57 | 45.00 | 59.79 | 64.93 | 65.8 | **71.54** | 64.49 | 61.88 |
| Class10 | 45.36 | 45.25 | 45.46 | 49.89 | 58.12 | 52.32 | 58.33 | 63.61 | **64.14** | 57.70 | 58.97 | 51.79 | 60.97 | 53.59 |
| Class11 | 27.43 | 43.88 | 22.45 | 38.65 | 28.86 | 36.46 | 36.46 | 34.77 | 36.54 | 47.26 | 35.78 | 38.65 | 41.27 | **49.96** |
| Class12 | 31.64 | 56.08 | 31.75 | 37.83 | 35.84 | 62.50 | 34.18 | 55.2 | 46.79 | 62.72 | 34.29 | 58.52 | 45.02 | **76.88** |
| Class13 | 0.00 | 0.67 | 0.00 | 1.11 | 0.00 | 0.00 | 0.00 | 0.45 | 0.00 | 0.45 | 0.00 | 0.89 | 0.00 | **1.78** |
| Class14 | 97.53 | 98.77 | 94.44 | 92.59 | **100.00** | **100.00** | 99.38 | 98.15 | **100.00** | 99.38 | 99.38 | **100.00** | 99.38 | **100.00** |
| Class15 | 96.59 | 98.16 | 96.59 | **98.43** | 97.38 | 98.16 | 97.64 | 97.64 | 97.64 | 98.16 | 97.90 | 98.16 | 97.64 | 98.16 |

# Comparison - Scenarios

- **KEMA (SSMA)**:
  - Making use of class labels rather than correspondences to align the manifolds.
  - Deal with strong distortions, or data from different dimensions.
  - Can adapt to different kernels to apply to various problems.

- **Global MA**:
  - Consider the global geometry instead of only the local one.
  - Based upon given correspondences, not class labels.

# Comparison - Scenarios

- **GUMA**:
  - Fully unsupervised matching without any pre-specified correspondences.
  - Take global geometry structures into consideration by using the geodesic distance.

- **LeMA**:
  - A large amount of low quality data and a small amount of high quality data can be obtained.
  - Linear case.

# Comparison - Correspondence

- **KEMA (SSMA)**:
  - Adjacency Matrix:
    Same data set: Gaussian kernel
    Cross data sets: Class labels
  - Joint Laplacian Matrix:
    Same data set: computed from the adjacency matrix
    Cross data sets: 0
- **LeMA**:
  - Joint Adjacency Matrix:
    Same data set: the LDA-like graph
    $$\widetilde{\mathbf{W}}_{i,j} = \begin{cases} 1/N_k, & \text{if } \mathbf{X}_i \text{ and } \mathbf{X}_j \text{ belong to the } k\text{-th class;} \\ 0, & \text{otherwise.} \end{cases}$$
    Cross data sets: iteratively updated using ADMM
  - Joint Laplacian Matrix:
    Computed from the adjacency matrix

# Comparison - Correspondence

- **GUMA**:
  - Adjacency Matrix:
    geodesic distance

- **Global MA**:
  - Joint Distance Matrix:
    Same data set: Euclidian distance or geodisic distance
    Cross data sets:
    $D_{x,y}(i,j) = \min_{u \in [1,l]} \left( D_{x,x}\left(x_i, x_{a_u}\right) + D_{y,y}\left(y_j, y_{b_u}\right) \right)$
  - Gram Matrix:
    Given the distance matrix $A$, $\tau(A) = -HSH/2$, $S_{i,j} = A^2_{i,j}$ is the matrix of squared distance, $H_{i,j} = \delta_{i,j} - 1/m$ is the "centering matrix"

# Unsupervised Domain Adaptation

## Notations and definitions

- $\mathcal{D}^s = \{\mathcal{X}^s, P(X)^s\}$: source domain and its marginal distributions.
- $\mathcal{T}^s = \{\mathcal{Y}^s, P(Y^s|X^s)\}$: source labels and its conditional distributions.
- $\mathcal{D}^t = \{\mathcal{X}^t, P(X)^t\}$: target domain and its marginal distributions.
- $\mathcal{T}^t = \{\mathcal{Y}^t, P(Y^t|X^t)\}$: target labels and its conditional distributions (to be learned).

## Unsupervised Domain Adaptation Problem

If a model (e.g. a classifier) is learned from a source domain, what is its capacity to correctly label data from the target domain?

# Unsupervised Domain Adaptation

## General Form of The Loss Function

$$\mathcal{L} = \mathcal{L}_{\mathrm{C}}\left(X^{L}, y\right) + \lambda \text{Distance}\left(X^{s}, X^{t}\right)$$

where

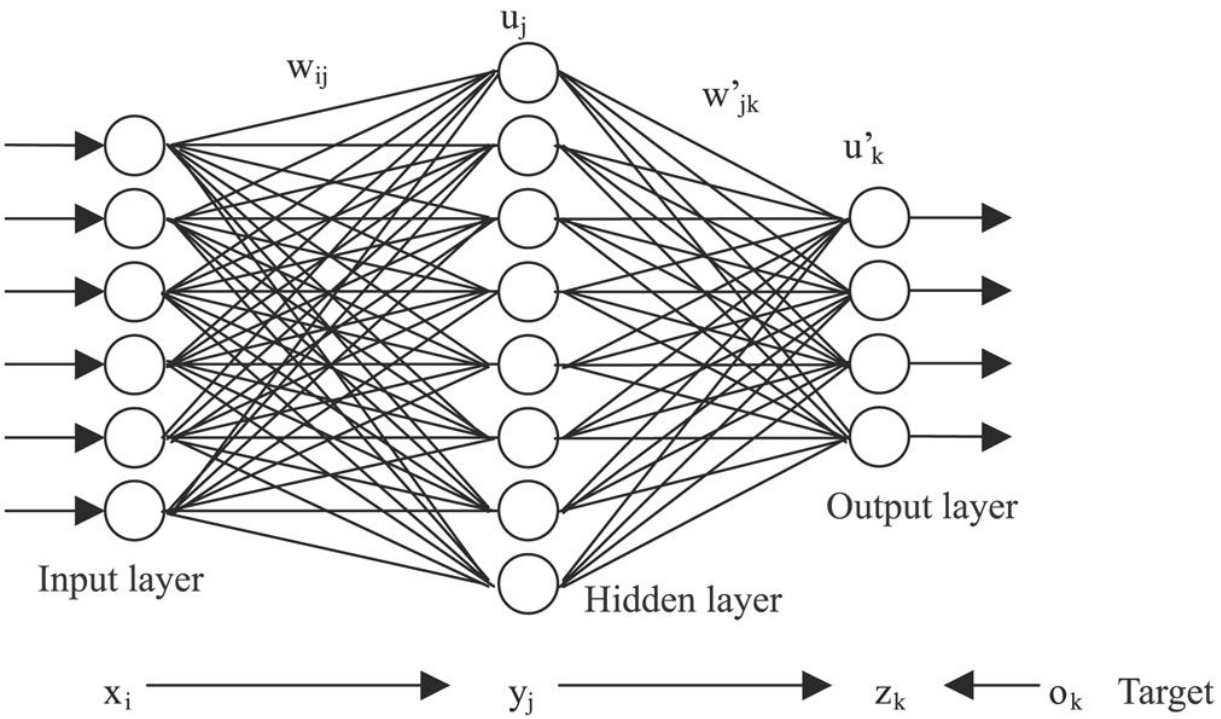
- $\lambda$: a hyperparameter for weighting penalty.
- $\mathcal{L} = \mathcal{L}_{\mathrm{C}}\left(X^{L}, y\right)$: the classification loss on the available labeled data $X^{L}$ and the ground-truth labels $y$.
- Distance $\left(X^{s}, X^{t}\right)$: the distance between the source and target data.

# Generalized MA for Domain Adaptation

## Two Key Ideas

- Using deep neural networks as the projecting function.
- Relax the correspondence-preserving constraint to distribution alignment.
  - Using the adversarial loss from a discriminator for measuring the distance between two distributions.

# Deep Neural Networks




- Strong power of feature representation.
- High-level abstraction.
- End-to-end learning with back-propagation algorithms.

# Align Manifolds or Distributions

## Manifold Alignment

- Accurate alignment between manifolds.
- The alignment results largely depend on the (either given or inferred) **correspondence information** [e.g., the similarities or distances between points, or the given class labels].

# Align Manifolds or Distributions

## Manifold Alignment

- Accurate alignment between manifolds.
- The alignment results largely depend on the (either given or inferred) **correspondence information** [e.g., the similarities or distances between points, or the given class labels].

## Distribution Alignment

- Only needs to minimize the **distance** between two distributions.
- More efficient while less accurate.

# Measure the Distance Between Distributions

How to measure the distance between distributions?

# Measure the Distance Between Distributions

How to measure the distance between distributions?

- KL-divergence.
- Maximum mean discrepancy (MMD).
- Wasserstein Distance.
- Adversarial loss from a discriminator.
- .....

# Adversarial Discriminative Domain Adaptation

- Pre-train a source encoder CNN using labeled source images.
- Perform adversarial adaptation by learning a target encoder CNN.
- During testing, target images are mapped with the target encoder to the shared feature space and classified by the source classifier.

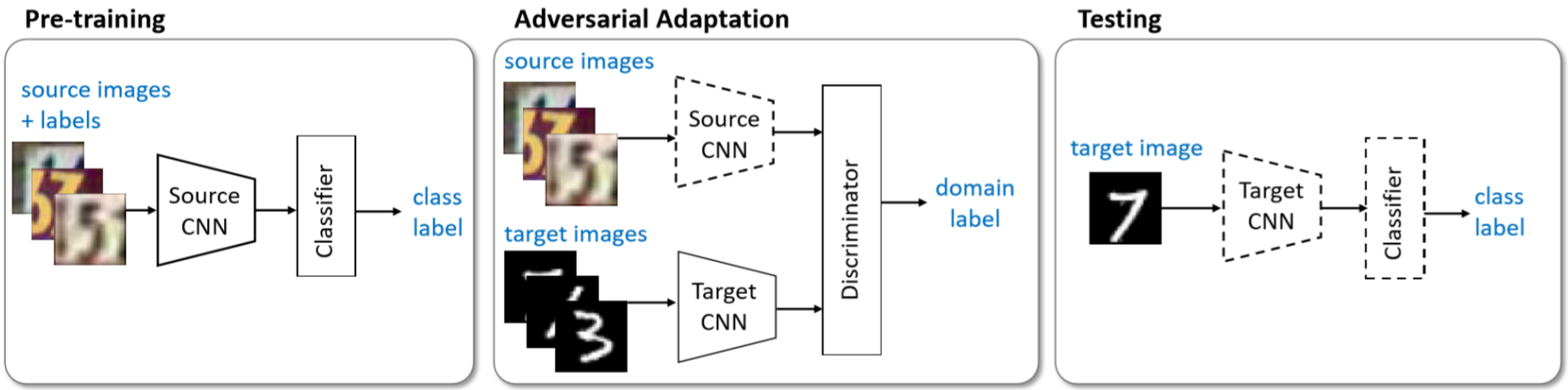



Figure: Overview of the ADDA approach. Dashed lines indicate fixed network parameters.

# Adversarial Discriminative Domain Adaptation

## Notations

- $M_s$: a source mapping (representation), projecting source data into a latent space.
- $M_t$: a target mapping (representation), projecting target data into a latent space.
- $C$: a common classifier for both source and target classification tasks.
- $D$: a domain discriminator for classifying whether a data point is drawn from the source or the target domain.

# Loss Function 1: Source Classifier

The source classification model is then trained using the standard supervised loss below:

$$\min_{M_s, C} \mathcal{L}_{\text{cls}}\left(\mathbf{X}_s,\, Y_s\right) = -\mathbb{E}_{(\mathbf{x}_s, y_s)\sim(\mathbf{X}_s, Y_s)} \sum_{k=1}^{K} \mathbf{1}_{[k=y_s]} \log C\left(M_s\left(\mathbf{x}_s\right)\right)$$

# Loss Function 2: Domain Discriminator

The domain discriminator $D$ is optimized according to a standard supervised loss:

$$\begin{aligned} \min_{D} \quad & \mathcal{L}_{\text{adv}_D}\left(\mathbf{X}_s, \mathbf{X}_t, M_s, M_t\right) \\ = \quad & -\mathbb{E}_{\mathbf{x}_s \sim \mathbf{X}_s}\left[\log D\left(M_s\left(\mathbf{x}_s\right)\right)\right] - \mathbb{E}_{\mathbf{x}_t \sim \mathbf{x}_t}\left[\log\left(1 - D\left(M_t\left(\mathbf{x}_t\right)\right)\right)\right] \end{aligned}$$

where the labels indicate the origin domain (1 for the source domain and 0 for the target domain).

# Loss Function 3: Adversarial Loss

## The GAN loss function for optimizing this mapping

Here the generator aims to mimic another unchanging distribution. The standard choice can be:

$$\min_{M_t} \mathcal{L}_{\text{adv}_M}\left(\mathbf{X}_s, \mathbf{X}_t, D\right) = -\mathbb{E}_{\mathbf{x}_t \sim \mathbf{X}_t}\left[\log D\left(M_t\left(\mathbf{x}_t\right)\right)\right]$$

This objective has the same fixed-point properties as the minimax loss, but provides stronger gradients to the target mapping.

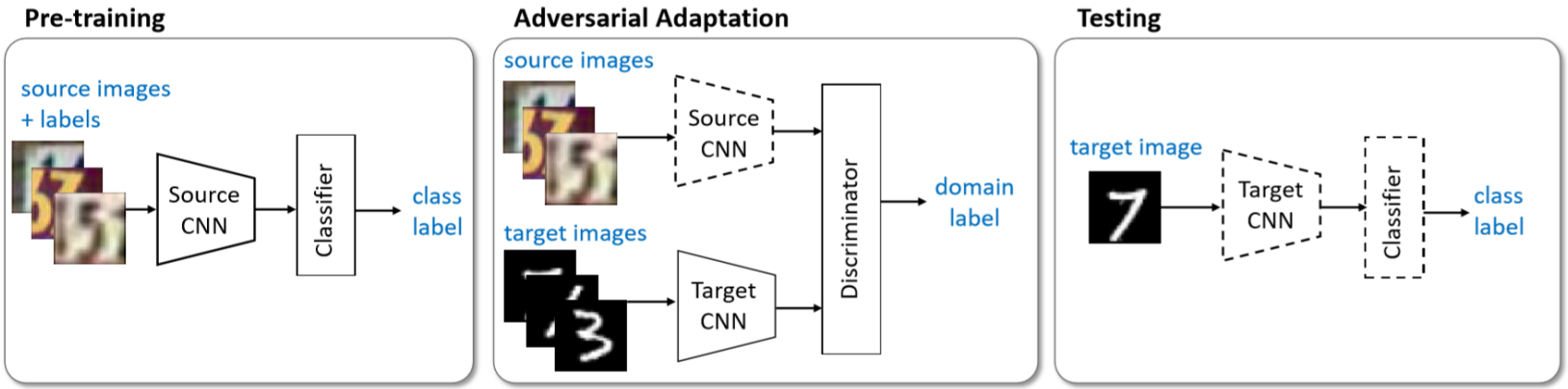
# Final Unconstrained Optimization

$$\min_{M_s, C} \quad \mathcal{L}_{\text{cls}}\left(\mathbf{X}_s, Y_s\right) = -\mathbb{E}_{(\mathbf{x}_s, y_s)\sim(\mathbf{X}_s, Y_s)} \sum_{k=1}^{K} \mathbf{1}_{[k=y_s]} \log C\left(M_s\left(\mathbf{x}_s\right)\right)$$

$$\min_{D} \quad \mathcal{L}_{\text{adv}_D}\left(\mathbf{X}_s, \mathbf{X}_t, M_s, M_t\right)$$

$$= \quad -\mathbb{E}_{\mathbf{x}_s\sim\mathbf{X}_s}\left[\log D\left(M_s\left(\mathbf{x}_s\right)\right)\right] - \mathbb{E}_{\mathbf{x}_t\sim\mathbf{X}_t}\left[\log\left(1 - D\left(M_t\left(\mathbf{x}_t\right)\right)\right)\right]$$

$$\min_{M_t} \quad \mathcal{L}_{\text{adv}_M}\left(\mathbf{X}_s, \mathbf{X}_t, D\right) = -\mathbb{E}_{\mathbf{x}_t\sim\mathbf{X}_t}\left[\log D\left(M_t\left(\mathbf{x}_t\right)\right)\right]$$



# Extract both Common and Specific Information

- Reconstruction-based approach [Bousmalis et al., 2016].
- Using an additional discriminator for common/specific classification [Tsai and Chien, 2017].

# Domain Separation Networks (DSN)

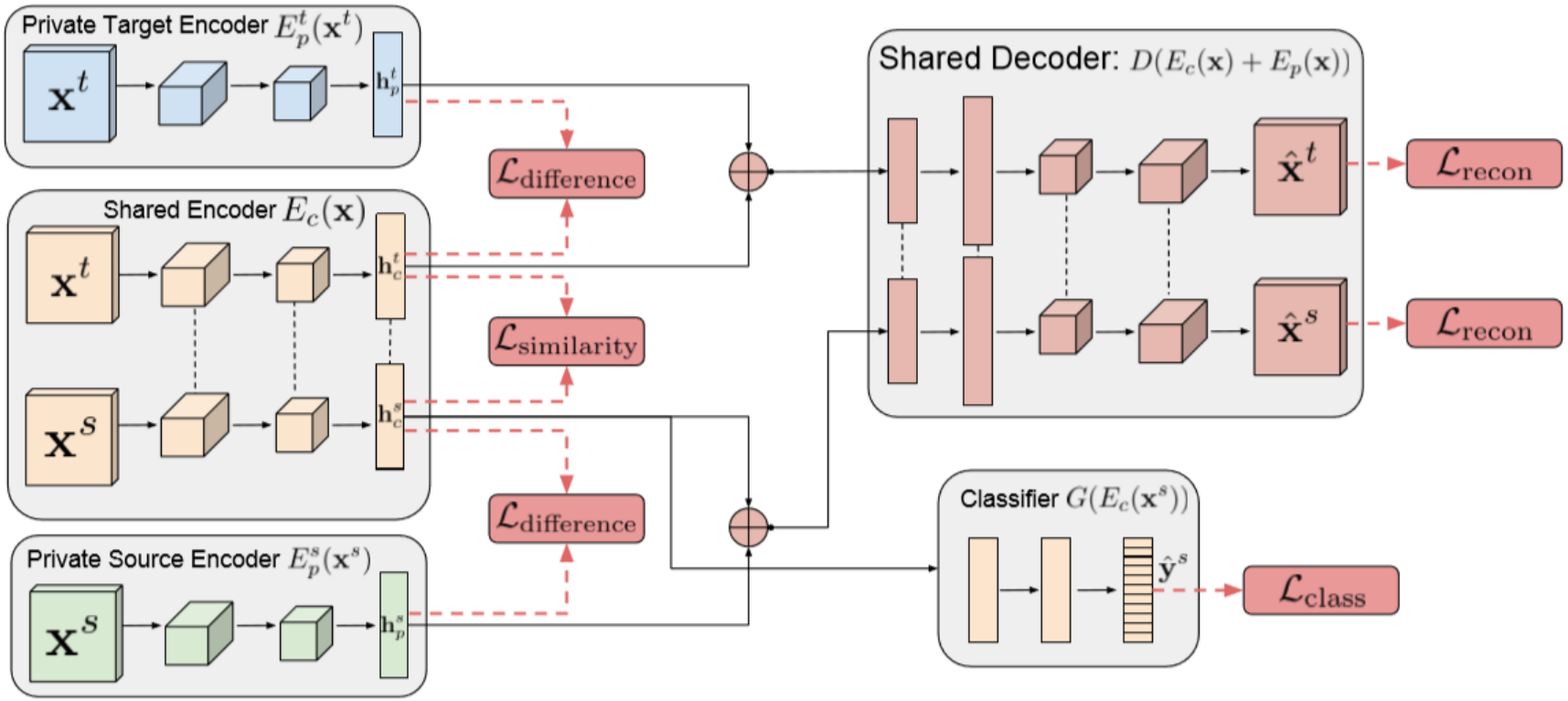



Figure: Training of Domain Separation Networks [Bousmalis et al., 2016].

# Performance Illustration

The reconstructed common and specific features by DSN [Bousmalis et al., 2016].

(a) MNIST (source) (b) MNIST-M (target) (c) Synth Objects (source) (d) LINEMOD (target)

Figure 2: Reconstructions for the representations of the two domains for "MNIST to MNIST-M" and for "Synth Objects to LINEMOD". In each block from left to right: the original image $\mathbf{x}_t$; reconstructed image $D(E_c(\mathbf{x}^t) + E_p(\mathbf{x}^t))$; shared only reconstruction $D(E_c(\mathbf{x}^t))$; private only reconstruction $D(E_p(\mathbf{x}^t))$.

# Adversarial Domain Separation and Adaptation

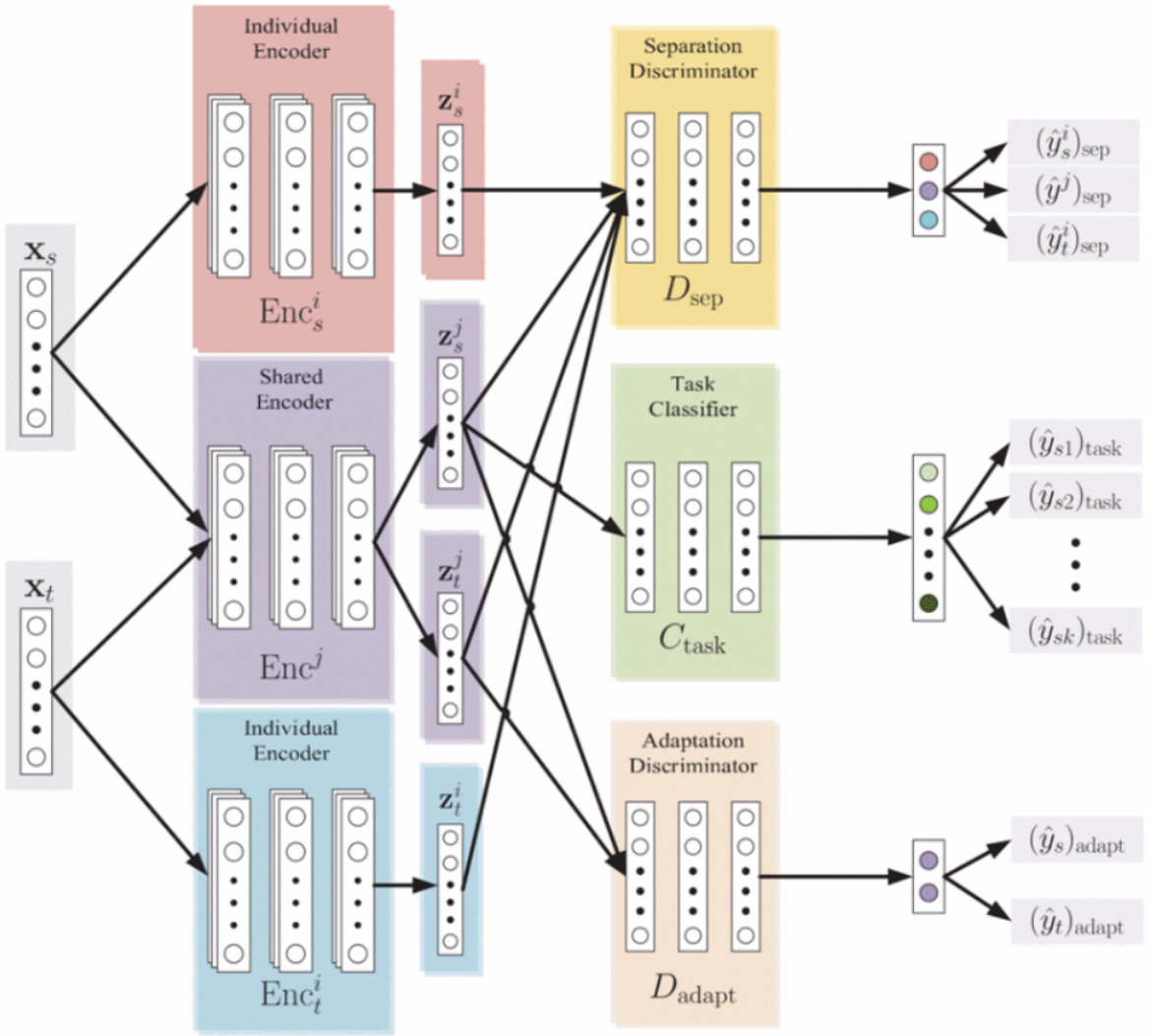



Figure: Overview of the ADSA structure [Tsai and Chien, 2017]

# Performance Comparison

ADSA jointly performed domain separation and adaptation to identify informative features for an improved classification performance [Tsai and Chien, 2017].

| | Baseline | VFAE | DAN | DSN | ADSA |
|---|---|---|---|---|---|
| B → D | 77.2 | 77.6 | 78.4 | 78.2 | **80.2** |
| B → E | 70.3 | 69.0 | 73.3 | 71.8 | **78.3** |
| B → K | 72.8 | 70.3 | 77.9 | 76.6 | **81.3** |
| D → B | 74.2 | 71.4 | 72.3 | 73.6 | **76.4** |
| D → E | 71.8 | 71.2 | 75.4 | 74.8 | **77.7** |
| D → K | 75.6 | 73.5 | 78.3 | 79.7 | **80.1** |
| E → B | 70.9 | 68.8 | 71.1 | 69.7 | **72.6** |
| E → D | 67.9 | 69.7 | 73.8 | 74.1 | **74.5** |
| E → K | 73.0 | 81.5 | 85.4 | **86.4** | 85.5 |
| K → B | 66.9 | 66.8 | 70.9 | 69.4 | **72.5** |
| K → D | 68.0 | 68.4 | 74.0 | 71.3 | **75.8** |
| K → E | 82.5 | 82.4 | **84.3** | 83.0 | 83.6 |
| Average | *73.4* | *72.5* | *76.3* | *76.1* | *78.2* |

Figure: Sentiment classification accuracies (%) for adaptation among different domains

# Overall Summary

- The basic manifold alignment problem
- Feature-level alignment (linear)
- Semi- or unsupervised MA: correspondence inference
- Manifold alignment with deep learning

# References I


Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., and Erhan, D. (2016).
Domain separation networks.
arXiv:1608.06019 [cs.CV].

Chang, W. and Mahadevan, S. (2009).
Manifold alignment without correspondence.
In International Jont Conference on Artifical Intelligence.

Chang, W. and Mahadevan, S. (2011).
Heterogeneous domain adaptation using manifold alignment.
In International Joint Conference on Ijcai.

Cui, Z., Chang, H., Shan, S., and Chen, X. (2014).
Generalized unsupervised manifold alignment.
Advances in Neural Information Processing Systems, 3:2429–2437.

Devis, T., Gustau, C.-V., and Zhaohong, D. (2016).
Kernel manifold alignment for domain adaptation.
Plos One, 11(2):e0148655.

Hong, D., Yokoya, N., Ge, N., and Chanussot, J.and Zhu, X. X. (2019).
Learnable manifold alignment (lema): a semi-supervised cross-modality learning framework for land cover and land use classification.
ISPRS Journal of Photogrammetry and Remote Sensing, 147:193–205.

HOTELLING, H. (1936).
RELATIONS BETWEEN TWO SETS OF VARIATES*.
Biometrika, 28(3-4):321–377.

# References II


Pei, Y., Huang, F., Shi, F., and Zha, H. (2012).

Unsupervised image matching based on manifold alignment.

IEEE Transactions on Pattern Analysis & Machine Intelligence, 34(8):1658.

Tsai, J. and Chien, J. (2017).

Adversarial domain separation and adaptation.

In 2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP), pages 1–6.

Wang, C. (2010).

Heterogeneous domain adaptation using manifold alignment.

Ijcai, pages 1541–1546.

Wang, C. and Mahadevan, S. (2013).

Manifold alignment preserving global geometry.

IJCAI International Joint Conference on Artificial Intelligence, pages 1743–1749.