Aprendizaje de Similitudes

Emilio Parrado Hernández
MLG

Departamento de Teoría de la Señal y Comunicaciones
Universidad Carlos III de Madrid

1 de febrero de 2013

Referencias

1. **Learning Multimodal Similarity**
   B. McFee, G. Lanckriet
   JMLR 12 (2011)
2. **Metric and Kernel Learning Using a Linear Transformation**
   P. Jain, B. Kulis, J. V. Davis, I. S. Dhillon
   JMLR 13 (2012)
3. **Large Scale Online Learning of Image Similarity Through Ranking**
   G. Chechik, V. Sharma, U. Shalit, S. Bengio
   JMLR 11 (2010)
4. **Learning Discriminative Metrics via Generative Models and Kernel Learning**
   Y. Shi, Y.K. Noh, F. Sha, D. D. Lee
   arXiv:1109.3940v1, 19 Sep. 2011.

Contenidos

Motivación

- Comparar dos objetos es una tarea fundamental en Aprendizaje Máquina (AM).
- Relación inmediata con clasificación por similitudes (vecinos más próximos, métodos núcleo)

Motivación

- Comparar dos objetos es una tarea fundamental en Aprendizaje Máquina (AM).
- Relación inmediata con clasificación por similitudes (vecinos más próximos, métodos núcleo)
- Relación con agrupamiento a través de restricciones sobre parejas de ejemplos que deberían estar (o no) en el mismo grupo.

Motivación

- Comparar dos objetos es una tarea fundamental en Aprendizaje Máquina (AM).
- Relación inmediata con clasificación por similitudes (vecinos más próximos, métodos núcleo)
- Relación con agrupamiento a través de restricciones sobre parejas de ejemplos que deberían estar (o no) en el mismo grupo.
- ¿Similitud o métrica? Métrica implica imponer restricciones de PSD, lo que implica mayor coste computacional

Motivación

- Comparar dos objetos es una tarea fundamental en Aprendizaje Máquina (AM).
- Relación inmediata con clasificación por similitudes (vecinos más próximos, métodos núcleo)
- Relación con agrupamiento a través de restricciones sobre parejas de ejemplos que deberían estar (o no) en el mismo grupo.
- ¿Similitud o métrica? Métrica implica imponer restricciones de PSD, lo que implica mayor coste computacional

Métodos para comparar dos objetos en AM

1. Coger una métrica genérica: coseno, euclídea, etc.
2. Aprender una métrica de Mahalanobis para el problema concreto (equivale a buscar una transformación lineal de los datos).

Métodos para comparar dos objetos en AM

1. Coger una métrica genérica: coseno, euclídea, etc.
2. Aprender una métrica de Mahalanobis para el problema concreto (equivale a buscar una transformación lineal de los datos). Pegas:

Métodos para comparar dos objetos en AM

1. Coger una métrica genérica: coseno, euclídea, etc.
2. Aprender una métrica de Mahalanobis para el problema concreto (equivale a buscar una transformación lineal de los datos). Pegas:
   - Número de parámetros cuadrático con la dimensión

Métodos para comparar dos objetos en AM

1. Coger una métrica genérica: coseno, euclídea, etc.
2. Aprender una métrica de Mahalanobis para el problema concreto (equivale a buscar una transformación lineal de los datos). Pegas:
   - Número de parámetros cuadrático con la dimensión
   - Falla en escenarios no lineales.

Métodos para comparar dos objetos en AM

1. Coger una métrica genérica: coseno, euclídea, etc.
2. Aprender una métrica de Mahalanobis para el problema concreto (equivale a buscar una transformación lineal de los datos). Pegas:
   - Número de parámetros cuadrático con la dimensión
   - Falla en escenarios no lineales. Se puede kernelizar, aunque tiene limitaciones importantes
     - Limitado al escenario transductivo
     - Opciones para escenarios inductivos son muy restrictivas o de difícil optimización: *Multiple kernel learning*, hyperkernels.

Métodos para comparar dos objetos en AM

1. Coger una métrica genérica: coseno, euclídea, etc.
2. Aprender una métrica de Mahalanobis para el problema concreto (equivale a buscar una transformación lineal de los datos). Pegas:
   - Número de parámetros cuadrático con la dimensión
   - Falla en escenarios no lineales. Se puede kernelizar, aunque tiene limitaciones importantes
     - Limitado al escenario transductivo
     - Opciones para escenarios inductivos son muy restrictivas o de difícil optimización: *Multiple kernel learning*, hyperkernels.

Maneras de expresar una similitud implícita

El aprendizaje de la métrica debe estar guiado por restricciones que recogen conocimiento a priori

- **Etiquetas de clases**: Los ejemplos pertenecientes a una misma clase deben estar más cerca entre sí que de los ejemplos de las otras clases
- Supervisión en forma de**restricciones por parejas** de similitud o disimilitud
- **Valor exacto** de similitud entre ejemplos (no suele darse)

La similitud es una supervisión más débil que la clasificación porque no hacen falta etiquetas.

Distancia de Mahalanobis en el espacio de características

- Datos $X = \{\mathbf{x}_i\}_{i=1}^n \in \mathbb{R}^d$
- Proyección no lineal a un espacio de características $\phi(\mathbf{x}_i)$ inducida mediante un kernel conocido $\kappa_0(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$
- Distancia de Mahalanobis en el espacio de características

$$d_W\left(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\right) = \phi(\mathbf{x}_i)^T W \phi(\mathbf{x}_j) = \kappa(\mathbf{x}_i, \mathbf{x}_j)$$

- Aprender $d_W(\cdot,\cdot)/\kappa(\cdot,\cdot)$ incorporando restricciones en forma de similitudes entre pares de datos.

Extracción de similitudes a partir de etiquetados humanos

La similitud puede ser un concepto subjetivo.
Multiples etiquetadores pueden no estar de acuerdo.

Es más robusto preguntar por órdenes que por cantidades absolutas

"¿Son i y j más parecidos que k y l?"
"¿el elemento i es más parecido a j o a k?"

Extracción de similitudes a partir de etiquetados humanos

La similitud puede ser un concepto subjetivo.
Multiples etiquetadores pueden no estar de acuerdo.

Es más robusto preguntar por órdenes que por cantidades absolutas

"¿Son i y j más parecidos que k y l?"
"¿el elemento i es más parecido a j o a k?"

Multidimensional Scaling (MDS)

Encontrar un espacio euclídeo de representación de los datos que respete las distancias "percibidas" por los etiquetadores

Extracción de similitudes a partir de etiquetados humanos

La similitud puede ser un concepto subjetivo.
Multiples etiquetadores pueden no estar de acuerdo.

Es más robusto preguntar por órdenes que por cantidades absolutas

"¿Son i y j más parecidos que k y l?"
"¿el elemento i es más parecido a j o a k?"

Multidimensional Scaling (MDS)

Encontrar un espacio euclídeo de representación de los datos que respete las distancias "percibidas" por los etiquetadores

MDS No métrico

Encontrar un espacio euclídeo de representación de los datos que respete una ordenación parcial "percibida" por los etiquetadores

Extracción de similitudes a partir de etiquetados humanos

La similitud puede ser un concepto subjetivo.
Multiples etiquetadores pueden no estar de acuerdo.

Es más robusto preguntar por órdenes que por cantidades absolutas

"¿Son i y j más parecidos que k y l?"
"¿el elemento i es más parecido a j o a k?"

Multidimensional Scaling (MDS)

Encontrar un espacio euclídeo de representación de los datos que respete las distancias "percibidas" por los etiquetadores

MDS No métrico

Encontrar un espacio euclídeo de representación de los datos que respete una ordenación parcial "percibida" por los etiquetadores

Orden Parcial Estricto

Relación binaria R definida en un conjunto Z ($R \subseteq Z^2$) que satisface

- **Irreflexiva**: $(a, a) \notin R$
- **Transitiva**: $(a, b) \in R \wedge (b, c) \in R \Rightarrow (a, c) \in R$
- **Antisimétrica**: $(a, b) \in R \Rightarrow (b, a) \notin R$

Se puede capturar mediante un grafo acíclico dirigido (DAG) C donde cada nodo es un par de muestras y las aristas indican la relación R

- Si hay **ciclos**, no hay una representación que capture la relación
- Si no hay ciclos, cualquier representación que capture una reducción transitiva de C captura al propio C.

Simplificación del grafo

La subjetividad y el desacuerdo entre etiquetadores provocan que el grafo resultante de compilar todas las etiquetas presente ciclos.

Simplificación del grafo

1. Romper ciclos. Algoritmo greedy de Berger and Shor, 1990
2. Eliminar aristas redundantes aplicando transitividad: Si dos nodos están unidos a través de un camino del grafo, se puede eliminar su conexión directa.

Simplificación del grafo

La subjetividad y el desacuerdo entre etiquetadores provocan que el grafo resultante de compilar todas las etiquetas presente ciclos.

Simplificación del grafo

1. Romper ciclos. Algoritmo greedy de Berger and Shor, 1990
2. Eliminar aristas redundantes aplicando transitividad: Si dos nodos están unidos a través de un camino del grafo, se puede eliminar su conexión directa.

Partial Order Embedding

- Encontrar $g : X \longrightarrow \mathbb{R}^d$ tal que

$$\forall (i,j,k,l) \in C : \|g(i) - g(j)\|^2 + 1 \leq \|g(k) - g(l)\|^2$$

se da un margen unitario para estabilidad numérica

- Asumir que los datos ya están en un cierto espacio $\mathbb{R}^D$ y que la función g es una proyección lineal

$$g(\mathbf{x}) = M\mathbf{x}$$

Partial Order Embedding

- Encontrar $g : X \longrightarrow \mathbb{R}^d$ tal que

$$\forall (i,j,k,l) \in C : \|g(i) - g(j)\|^2 + 1 \leq \|g(k) - g(l)\|^2$$

se da un margen unitario para estabilidad numérica

- Asumir que los datos ya están en un cierto espacio $\mathbb{R}^D$ y que la función g es una proyección lineal

$$g(\mathbf{x}) = M\mathbf{x}$$

- La matriz de Gram en el espacio imagen es $K = X^T M^T M X$

Partial Order Embedding

- Encontrar $g : X \longrightarrow \mathbb{R}^d$ tal que

$$\forall (i,j,k,l) \in C : \|g(i) - g(j)\|^2 + 1 \leq \|g(k) - g(l)\|^2$$

se da un margen unitario para estabilidad numérica

- Asumir que los datos ya están en un cierto espacio $\mathbb{R}^D$ y que la función g es una proyección lineal

$$g(\mathbf{x}) = M\mathbf{x}$$

- La matriz de Gram en el espacio imagen es $K = X^T M^T M X$
- Restricciones del grafo:
$(\mathbf{x}_i - \mathbf{x}_j)^T M^T M (\mathbf{x}_i - \mathbf{x}_j) + 1 \leq (\mathbf{x}_k - \mathbf{x}_l)^T M^T M (\mathbf{x}_k - \mathbf{x}_l)$

Partial Order Embedding

- Encontrar $g : X \longrightarrow \mathbb{R}^d$ tal que

$$\forall (i,j,k,l) \in C : \|g(i) - g(j)\|^2 + 1 \leq \|g(k) - g(l)\|^2$$

se da un margen unitario para estabilidad numérica

- Asumir que los datos ya están en un cierto espacio $\mathbb{R}^D$ y que la función g es una proyección lineal

$$g(\mathbf{x}) = M\mathbf{x}$$

- La matriz de Gram en el espacio imagen es $K = X^T M^T M X$
- Restricciones del grafo:
$(\mathbf{x}_i - \mathbf{x}_j)^T M^T M (\mathbf{x}_i - \mathbf{x}_j) + 1 \leq (\mathbf{x}_k - \mathbf{x}_l)^T M^T M (\mathbf{x}_k - \mathbf{x}_l)$

Optimización para encontrar M

1. **slack variables** ξ_{ijkl} para que haya solución al problema.
2. **empirical hinge loss** para las violaciones de las restricciones
3. **Regularización** con $\text{tr}(M^T M)$ para evitar sobreajuste

Optimización

$$\min_{M,\xi_{ijkl}} \text{tr}(M^T M) + \frac{\beta}{|C|} \sum_{C} \xi_{ijkl}$$

sujeto a $(\mathbf{x}_i - \mathbf{x}_j)^T M^T M(\mathbf{x}_i - \mathbf{x}_j) + 1 \leq (\mathbf{x}_k - \mathbf{x}_l)^T M^T M(\mathbf{x}_k - \mathbf{x}_l) + \xi_{ijkl}$ y $\xi_{ijkl} \geq 0$

Resolver para $W = M^T M$ con W PSD, que es convexo

Optimización para encontrar M

1. **slack variables** ξ_{ijkl} para que haya solución al problema.
2. **empirical hinge loss** para las violaciones de las restricciones
3. **Regularización** con $\text{tr}(M^T M)$ para evitar sobreajuste

Optimización

$$\min_{M,\xi_{ijkl}} \text{tr}(M^T M) + \frac{\beta}{|C|} \sum_{C} \xi_{ijkl}$$

sujeto a $(\mathbf{x}_i - \mathbf{x}_j)^T M^T M(\mathbf{x}_i - \mathbf{x}_j) + 1 \leq (\mathbf{x}_k - \mathbf{x}_l)^T M^T M(\mathbf{x}_k - \mathbf{x}_l) + \xi_{ijkl}$ y $\xi_{ijkl} \geq 0$

Resolver para $W = M^T M$ con W PSD, que es convexo

Versión kernelizada

Asumir que la función g es una transformación no lineal con d componentes

$$g(\mathbf{x}) = M(\phi(\mathbf{x})) = [\langle \mathbf{w}_p, \phi(\mathbf{x}) \rangle_{\mathcal{H}}]_{p=1}^{d}$$

donde $\phi(\cdot)$ es una proyección inducida por un kernel $\kappa_0(\cdot, \cdot)$

No resoluble en dim. infinita...

Versión kernelizada

Asumir que la función g es una transformación no lineal con d componentes

$$g(\mathbf{x}) = M(\phi(\mathbf{x})) = [\langle \mathbf{w}_p, \phi(\mathbf{x}) \rangle_{\mathcal{H}}]_{p=1}^{d}$$

donde $\phi(\cdot)$ es una proyección inducida por un kernel $\kappa_0(\cdot, \cdot)$
No resoluble en dim. infinita... Regularizar en un Espacio de Hilbert

$$\|M\|_{HS}^2 = \sum_{p=1}^{d} \langle \mathbf{w}_p, \mathbf{w}_p \rangle_{\mathcal{H}}$$

Versión kernelizada

Asumir que la función g es una transformación no lineal con d componentes

$$g(\mathbf{x}) = M(\phi(\mathbf{x})) = [\langle \mathbf{w}_p, \phi(\mathbf{x}) \rangle_{\mathcal{H}}]_{p=1}^{d}$$

donde $\phi(\cdot)$ es una proyección inducida por un kernel $\kappa_0(\cdot, \cdot)$
No resoluble en dim. infinita... Regularizar en un Espacio de Hilbert

$$\|M\|_{HS}^2 = \sum_{p=1}^{d} \langle \mathbf{w}_p, \mathbf{w}_p \rangle_{\mathcal{H}}$$

Representer Theorem: $M = N\Phi^T \rightarrow \|M\|_{HS}^2 = \text{traza}(N^T N K)$

Versión kernelizada

Asumir que la función g es una transformación no lineal con d componentes

$$g(\mathbf{x}) = M(\phi(\mathbf{x})) = [\langle \mathbf{w}_p, \phi(\mathbf{x}) \rangle_{\mathcal{H}}]_{p=1}^{d}$$

donde $\phi(\cdot)$ es una proyección inducida por un kernel $\kappa_0(\cdot, \cdot)$
No resoluble en dim. infinita... Regularizar en un Espacio de Hilbert

$$\|M\|_{HS}^2 = \sum_{p=1}^{d} \langle \mathbf{w}_p, \mathbf{w}_p \rangle_{\mathcal{H}}$$

Representer Theorem: $M = N\Phi^T \rightarrow \|M\|_{HS}^2 = \text{traza}(N^T N K)$

Optimización con kernels

$$g(\mathbf{x}) = M(\phi(\mathbf{x})) = N\Phi^T\phi(\mathbf{x}) = N[\kappa_0(\mathbf{x}_i, \mathbf{x})]_{i=1}^n$$

Las restricciones impuestas por el grafo quedan:
$(K_i - K_j)^T N^T N(K_i - K_j) + 1 \leq (K_k - K_l)^T N^T N(K_k - K_l)$ donde K_x es la columna x de K

Optimización con kernels

$$g(\mathbf{x}) = M(\phi(\mathbf{x})) = N\Phi^T\phi(\mathbf{x}) = N[\kappa_0(\mathbf{x}_i, \mathbf{x})]_{i=1}^n$$

Las restricciones impuestas por el grafo quedan:
$(K_i - K_j)^T N^T N(K_i - K_j) + 1 \leq (K_k - K_l)^T N^T N(K_k - K_l)$ donde K_x es la columna x de K

Optimizar

$$\min_{N,\xi_{ijkl}} \operatorname{tr}(N^T NK) + \frac{\beta}{|C|}\sum_C \xi_{ijkl}$$

sujeto a $(K_i - K_j)^T N^T N(K_i - K_j) + 1 \leq (K_k - K_l)^T N^T N(K_k - K_l) + \xi_{ijkl}$
y $\xi_{ijkl} \geq 0$

Optimización con kernels

$$g(\mathbf{x}) = M(\phi(\mathbf{x})) = N\Phi^T\phi(\mathbf{x}) = N[\kappa_0(\mathbf{x}_i, \mathbf{x})]_{i=1}^n$$

Las restricciones impuestas por el grafo quedan:
$(K_i - K_j)^T N^T N(K_i - K_j) + 1 \leq (K_k - K_l)^T N^T N(K_k - K_l)$ donde K_x es la columna x de K

Optimizar

$$\min_{N,\xi_{ijkl}} \text{tr}(N^T NK) + \frac{\beta}{|C|}\sum_C \xi_{ijkl}$$

sujeto a $(K_i - K_j)^T N^T N(K_i - K_j) + 1 \leq (K_k - K_l)^T N^T N(K_k - K_l) + \xi_{ijkl}$ y $\xi_{ijkl} \geq 0$

Resolver para $W = N^T N$, W PSD que es convexo.

Optimización con kernels

$$g(\mathbf{x}) = M(\phi(\mathbf{x})) = N\Phi^T\phi(\mathbf{x}) = N[\kappa_0(\mathbf{x}_i, \mathbf{x})]_{i=1}^n$$

Las restricciones impuestas por el grafo quedan:
$(K_i - K_j)^T N^T N(K_i - K_j) + 1 \leq (K_k - K_l)^T N^T N(K_k - K_l)$ donde K_x es la columna x de K

Optimizar

$$\min_{N,\xi_{ijkl}} \text{tr}(N^T NK) + \frac{\beta}{|C|}\sum_C \xi_{ijkl}$$

sujeto a $(K_i - K_j)^T N^T N(K_i - K_j) + 1 \leq (K_k - K_l)^T N^T N(K_k - K_l) + \xi_{ijkl}$ y $\xi_{ijkl} \geq 0$

Resolver para $W = N^T N$, W PSD que es convexo.

Consideraciones

- Equivale a aprender una métrica de Mahalanobis con $\Phi W \Phi^T$ en $\mathcal{H}$
- Equivale a aprender un kernel KWK con W PSD
- $K = I$ quiere decir que no hay **conocimiento a priori** sobre la similitud entre puntos. La regularización $\text{tr}(W)$ equivale a minimizar una cota en el rango de W, denotando preferencia por proyecciones de baja dimensión
- Extensión para *Multiple Kernel Learning* útil en problemas con diversas modalidades. Se aprende una combinación ponderada de los distintos kernels.

Función de coste divergencia LogDet

$$D_{l.d}(W, W_0) = \text{tr}(WW_0^{-1}) - \text{logdet}(WW_0^{-1}) - d, \text{ con } W, W_0 \in \mathbb{R}^d$$

Si $W_0 = I$, tenemos maximización de entropía.

Optimización

$$\min_{W \succeq 0} D_{l,d}(W, I) \quad \text{s.t.} \quad d_W(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)) \leq u, \quad (i,j) \in S,$$
$$d_W(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)) \geq l, \quad (i,j) \in D, \qquad (1)$$

- LogDet **invariante a cambios de escala**
- Algoritmo **kernelizable**
- Extensiones a otras matrices PSD W_0
- Extensiones a otro tipo de restricciones lineales

Función de coste divergencia LogDet

$$D_{l.d}(W, W_0) = \text{tr}(WW_0^{-1}) - \text{logdet}(WW_0^{-1}) - d, \text{ con } W, W_0 \in \mathbb{R}^d$$

Si $W_0 = I$, tenemos maximización de entropía.

Optimización

$$\min_{W \succeq 0} D_{l,d}(W, I) \quad \text{s.t.} \quad d_W(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)) \leq u, \quad (i,j) \in S,$$
$$d_W(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)) \geq l, \quad (i,j) \in D, \qquad (1)$$

- LogDet **invariante a cambios de escala**
- Algoritmo **kernelizable**
- Extensiones a otras matrices PSD W_0
- Extensiones a otro tipo de restricciones lineales

Kernelización del aprendizaje de la métrica con LogDet

Dados un conjunto de n puntos con restricciones de (di)similitud que definen una matriz de kernels K_0, $K_0(i,j) = \kappa_0(\mathbf{x}_i, \mathbf{x}_j)$
El problema de encontrar K se puede escribir como:

$$\min_{K \succeq 0} D_{l,d}(K, K_0) \text{ s.t.} \quad \begin{aligned} &\operatorname{tr}\left(K(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T\right) \leq u, \quad (i,j) \in S, \\ &\operatorname{tr}\left(K(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T\right) \geq l, \quad (i,j) \in D \end{aligned} \tag{2}$$

Teorema 1
Sea $K_0 \succ 0$, K^* solución de (2) y W^* solución de (1). Entonces

$$K^* = \Phi^T W^* \Phi, \quad W^* = I + \Phi S \Phi^T$$

$$\text{con } S = K_0^{-1}(K^* - K_0)K_0^{-1}, K_0 = \Phi^T\Phi, \Phi = [\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_n)]$$

Kernelización del aprendizaje de la métrica con LogDet

Dados un conjunto de n puntos con restricciones de (di)similitud que definen una matriz de kernels K_0, $K_0(i,j) = \kappa_0(\mathbf{x}_i, \mathbf{x}_j)$
El problema de encontrar K se puede escribir como:

$$\min_{K \succeq 0} D_{l,d}(K, K_0) \text{ s.t.} \quad \operatorname{tr}\left(K(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T\right) \leq u, \quad (i,j) \in S,$$
$$\operatorname{tr}\left(K(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T\right) \geq l, \quad (i,j) \in D \quad (2)$$

Teorema 1

Sea $K_0 \succ 0$, K^* solución de (2) y W^* solución de (1). Entonces

$$K^* = \Phi^T W^* \Phi, \quad W^* = I + \Phi S \Phi^T$$

$$\text{con } S = K_0^{-1}(K^* - K_0)K_0^{-1}, K_0 = \Phi^T\Phi, \Phi = [\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_n)]$$

Generalización a puntos nuevos

Supóngase que K es la solución al problema (2).

Distancia entre puntos del conjunto de entrenamiento

$$d(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)) = K(i,i) + K(j,j) - 2K(i,j)$$

Distancia entre puntos del conjunto de test

$$d(\phi(\mathbf{z}_1), \phi(\mathbf{z}_2)) = \phi(\mathbf{z}_1)^T W \phi(\mathbf{z}_2) = \phi(\mathbf{z}_1)^T (I + \Phi S \Phi^T \phi(\mathbf{z}_2)$$

$$= \kappa_0(\mathbf{z}_1, \mathbf{z}_2) + \mathbf{k}_1^T S \mathbf{k}_2, \text{con} \mathbf{k}_i = [\kappa_0(\mathbf{x}_1, \mathbf{z}_i), \ldots, \kappa_0(\mathbf{x}_n, \mathbf{z}_i)]^T$$

Generalización a puntos nuevos

Supóngase que K es la solución al problema (2).

Distancia entre puntos del conjunto de entrenamiento

$$d(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)) = K(i,i) + K(j,j) - 2K(i,j)$$

Distancia entre puntos del conjunto de test

$$d(\phi(\mathbf{z}_1), \phi(\mathbf{z}_2)) = \phi(\mathbf{z}_1)^T W \phi(\mathbf{z}_2) = \phi(\mathbf{z}_1)^T (I + \Phi S \Phi^T \phi(\mathbf{z}_2)$$
$$= \kappa_0(\mathbf{z}_1, \mathbf{z}_2) + \mathbf{k}_1^T S \mathbf{k}_2, \text{con} \mathbf{k}_i = [\kappa_0(\mathbf{x}_1, \mathbf{z}_i), \ldots, \kappa_0(\mathbf{x}_n, \mathbf{z}_i)]^T$$

Extensión para conjuntos grandes de datos

Hemos presentado dos alternativas equivalentes:

- Aprender W, complejidad $O(d^2)$
- Aprender K, complejidad $O(n^2)$

Se puede considerar que

$$W = I + ULU^T$$

con $L = F - I$ de tamaño $k \times k$ definida positiva y U conocida de tamaño $d \times k$ ($k << \text{mín}(d, n)$).

Extensión para conjuntos grandes de datos

Hemos presentado dos alternativas equivalentes:

- Aprender W, complejidad $O(d^2)$
- Aprender K, complejidad $O(n^2)$

Se puede considerar que

$$W = I + ULU^T$$

con $L = F - I$ de tamaño $k \times k$ definida positiva y U conocida de tamaño $d \times k$ ($k << \text{mín}(d, n)$).

Se resuelve para F un problema de tamaño $O(k^2)$

Extensión para conjuntos grandes de datos

Hemos presentado dos alternativas equivalentes:

- Aprender W, complejidad $O(d^2)$
- Aprender K, complejidad $O(n^2)$

Se puede considerar que

$$W = I + ULU^T$$

con $L = F - I$ de tamaño $k \times k$ definida positiva y U conocida de tamaño $d \times k$ ($k << \text{mín}(d, n)$).

Se resuelve para F un problema de tamaño $O(k^2)$

Elecciones para U

- k primeros vectores singulares de Φ
- Centroides de un clustering de las columnas de Φ
- Centroides de un clustering de las medias de cada clase

Extensión para conjuntos grandes de datos

Hemos presentado dos alternativas equivalentes:

- Aprender W, complejidad $O(d^2)$
- Aprender K, complejidad $O(n^2)$

Se puede considerar que

$$W = I + ULU^T$$

con $L = F - I$ de tamaño $k \times k$ definida positiva y U conocida de tamaño $d \times k$ ($k << \text{mín}(d, n)$).

Se resuelve para F un problema de tamaño $O(k^2)$

Elecciones para U

- k primeros vectores singulares de Φ
- Centroides de un clustering de las columnas de Φ
- Centroides de un clustering de las medias de cada clase

Motivación

- Problemas de **escalabilidad**: recursos computacionales y de memoria crecen cuadráticamente con el número de datos
- Etiquetados basados en humanos pueden no ser apropiados en escenarios *big data*

Motivación

- Problemas de **escalabilidad**: recursos computacionales y de memoria crecen cuadráticamente con el número de datos
- Etiquetados basados en humanos pueden no ser apropiados en escenarios *big data*
- La similitud se puede **inferir** a través de parejas de resultados que son devueltas ante la misma *query*

Motivación

- Problemas de **escalabilidad**: recursos computacionales y de memoria crecen cuadráticamente con el número de datos
- Etiquetados basados en humanos pueden no ser apropiados en escenarios *big data*
- La similitud se puede **inferir** a través de parejas de resultados que son devueltas ante la misma *query*
- Inferir similitud de las etiquetas de clases

Motivación

- Problemas de **escalabilidad**: recursos computacionales y de memoria crecen cuadráticamente con el número de datos
- Etiquetados basados en humanos pueden no ser apropiados en escenarios *big data*
- La similitud se puede **inferir** a través de parejas de resultados que son devueltas ante la misma *query*
- Inferir similitud de las etiquetas de clases

OASIS aprende similitud (no garantiza semidefinida positiva) independiente de las etiquetas de las clases

Está basado en los algoritmos Passive-Aggressive para aprendizaje en línea.

Motivación

- Problemas de **escalabilidad**: recursos computacionales y de memoria crecen cuadráticamente con el número de datos
- Etiquetados basados en humanos pueden no ser apropiados en escenarios *big data*
- La similitud se puede **inferir** a través de parejas de resultados que son devueltas ante la misma *query*
- Inferir similitud de las etiquetas de clases

OASIS aprende similitud (no garantiza semidefinida positiva) independiente de las etiquetas de las clases

Está basado en los algoritmos Passive-Aggressive para aprendizaje en línea.

Algoritmo OASIS

Similitud: $S_W(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T W \mathbf{x}_j$

Restricciones: dada una tripleta de datos $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k$ tales que $r(\mathbf{x}_i, \mathbf{x}_j) > r(\mathbf{x}_i, \mathbf{x}_k)$, se establece la restricción

$$S_W(\mathbf{x}_i, \mathbf{x}_j) > S_W(\mathbf{x}_i, \mathbf{x}_k) + 1$$

Algoritmo OASIS

Similitud: $S_W(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T W \mathbf{x}_j$

Restricciones: dada una tripleta de datos $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k$ tales que $r(\mathbf{x}_i, \mathbf{x}_j) > r(\mathbf{x}_i, \mathbf{x}_k)$, se establece la restricción

$$S_W(\mathbf{x}_i, \mathbf{x}_j) > S_W(\mathbf{x}_i, \mathbf{x}_k) + 1$$

Función de pérdidas tipo *hinge*

$$l_W(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) = \text{máx}\{0, 1 - S_W(\mathbf{x}_i, \mathbf{x}_j) + S_W(\mathbf{x}_i, \mathbf{x}_k)\}$$

Algoritmo OASIS

Similitud: $S_W(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T W \mathbf{x}_j$

Restricciones: dada una tripleta de datos $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k$ tales que $r(\mathbf{x}_i, \mathbf{x}_j) > r(\mathbf{x}_i, \mathbf{x}_k)$, se establece la restricción

$$S_W(\mathbf{x}_i, \mathbf{x}_j) > S_W(\mathbf{x}_i, \mathbf{x}_k) + 1$$

Función de pérdidas tipo *hinge*

$$l_W(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) = \text{máx}\{0, 1 - S_W(\mathbf{x}_i, \mathbf{x}_j) + S_W(\mathbf{x}_i, \mathbf{x}_k)\}$$

OASIS

$$W^i = \arg\min_W \frac{1}{2}\|W - W^{i-1}\|_F^2 + C\xi$$

sujeto a $l_W(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) \leq \xi$ y $\xi \geq 0$

Algoritmo OASIS

Similitud: $S_W(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T W \mathbf{x}_j$

Restricciones: dada una tripleta de datos $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k$ tales que $r(\mathbf{x}_i, \mathbf{x}_j) > r(\mathbf{x}_i, \mathbf{x}_k)$, se establece la restricción

$$S_W(\mathbf{x}_i, \mathbf{x}_j) > S_W(\mathbf{x}_i, \mathbf{x}_k) + 1$$

Función de pérdidas tipo *hinge*

$$l_W(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) = \text{máx}\{0, 1 - S_W(\mathbf{x}_i, \mathbf{x}_j) + S_W(\mathbf{x}_i, \mathbf{x}_k)\}$$

OASIS

$$W^i = \arg\min_W \frac{1}{2}\|W - W^{i-1}\|_F^2 + C\xi$$

sujeto a $l_W(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) \leq \xi$ y $\xi \geq 0$

Aprendizaje de una métrica local discriminativa

Para cada punto de entrenamiento $\mathbf{x}_i$ se obtienen dos conjuntos de datos: objetivos $\mathbf{x}_i^+$, cuya etiqueta coincide con la de $\mathbf{x}_i$ e impostores $\mathbf{x}_i^-$, cuya etiqueta no coincide

Métrica local basada en NN

$$\min_{M,\xi\geq 0} \sum_i \sum_{j\in \mathbf{x}_i^+} d_M^2(\mathbf{x}_i,\mathbf{x}_j) + \gamma \sum_{ijl} \xi_{ijl}$$

sujeto a $1 + d_M^2(\mathbf{x}_i,\mathbf{x}_j) - d_M^2(\mathbf{x}_i,\mathbf{x}_l) \leq \xi_{ijl},\ \forall j \in \mathbf{x}_i^+, l \in \mathbf{x}_i^-$

Métrica Generativa

Cuando el número de datos es finito, el error del 1-NN se desvía del error asimptótico un término de sesgo que es independiente de la métrica.
Si se asume una métrica basada en transformación lineal, el término de sesgo se optimiza con

$$\text{mín}(\text{tr}(M_i^{-1}\Phi_i)^2 \text{ sujeto a } |M_i| = 1, M_i \text{PSD}$$

El óptimo es una matriz PSD cuyos autovectores sean los mismos que los de Φ_i

Aprendizaje discriminativo con múltiples métricas generativas

Métrica generativa local funciona pero hace falta una M_i por cada punto.

Resultados dependen de la pdf usada para el modelado generativo.

Aprendizaje discriminativo con múltiples métricas generativas

Métrica generativa local funciona pero hace falta una M_i por cada punto.

Resultados dependen de la pdf usada para el modelado generativo.

Mejora combinando métricas locales $K_i(\mathbf{x}_m, \mathbf{x}_n) = \mathbf{x}_m^T M_i \mathbf{x}_n$

Aprendizaje discriminativo con múltiples métricas generativas

Métrica generativa local funciona pero hace falta una M_i por cada punto.

Resultados dependen de la pdf usada para el modelado generativo.

Mejora combinando métricas locales $K_i(\mathbf{x}_m, \mathbf{x}_n) = \mathbf{x}_m^T M_i \mathbf{x}_n$

Cada M_i se aprende en la vecindad de $\mathbf{x}_i$ y es una estimación sesgada de un kernel global.

Aprendizaje discriminativo con múltiples métricas generativas

Métrica generativa local funciona pero hace falta una M_i por cada punto.

Resultados dependen de la pdf usada para el modelado generativo.

Mejora combinando métricas locales $K_i(\mathbf{x}_m, \mathbf{x}_n) = \mathbf{x}_m^T M_i \mathbf{x}_n$

Cada M_i se aprende en la vecindad de $\mathbf{x}_i$ y es una estimación sesgada de un kernel global.

Promediar los kernels locales resulta en un estimador insesgado

$$\kappa(\mathbf{x}_m, \mathbf{x}_n) = \sum_i \alpha_i \kappa_i(\mathbf{x}_m, \mathbf{x}_n) = \mathbf{x}_m^T \left(\sum_i \alpha_i M_i \right) \mathbf{x}_n = \mathbf{x}_m^T M \mathbf{x}_n$$

Aprendizaje discriminativo con múltiples métricas generativas

Métrica generativa local funciona pero hace falta una M_i por cada punto.

Resultados dependen de la pdf usada para el modelado generativo.

Mejora combinando métricas locales $K_i(\mathbf{x}_m, \mathbf{x}_n) = \mathbf{x}_m^T M_i \mathbf{x}_n$

Cada M_i se aprende en la vecindad de $\mathbf{x}_i$ y es una estimación sesgada de un kernel global.

Promediar los kernels locales resulta en un estimador insesgado

$$\kappa(\mathbf{x}_m, \mathbf{x}_n) = \sum_i \alpha_i \kappa_i(\mathbf{x}_m, \mathbf{x}_n) = \mathbf{x}_m^T \left(\sum_i \alpha_i M_i \right) \mathbf{x}_n = \mathbf{x}_m^T M \mathbf{x}_n$$

Consideraciones

- Caso sencillo $\alpha_i = 1$, entonces $M^{\text{UNI}} = \frac{1}{N} \sum_i M_i$
- Extensión no lineal. Kernel local

$$K_{il}(\mathbf{x}_m, \mathbf{x}_n) = \exp\{-(\mathbf{x}_m - \mathbf{x}_n)^T M_i (\mathbf{x}_m - \mathbf{x}_n)/\sigma_l^2\}$$

Consideraciones

- Caso sencillo $\alpha_i = 1$, entonces $M^{\text{UNI}} = \frac{1}{N}\sum_i M_i$
- Extensión no lineal. Kernel local

$$K_{il}(\mathbf{x}_m, \mathbf{x}_n) = \exp\{-(\mathbf{x}_m - \mathbf{x}_n)^T M_i (\mathbf{x}_m - \mathbf{x}_n)/\sigma_l^2\}$$

- Kernel como combinación convexa de kernels locales

$$K(\mathbf{x}_m, \mathbf{x}_n) = \sum_{i,l} \alpha_{il} K_{il}(\mathbf{x}_m, \mathbf{x}_n)$$

y los α_{il} cumplen las restricciones de ser convexos.

Consideraciones

- Caso sencillo $\alpha_i = 1$, entonces $M^{\text{UNI}} = \frac{1}{N}\sum_i M_i$
- Extensión no lineal. Kernel local

$$K_{il}(\mathbf{x}_m, \mathbf{x}_n) = \exp\{-(\mathbf{x}_m - \mathbf{x}_n)^T M_i (\mathbf{x}_m - \mathbf{x}_n)/\sigma_l^2\}$$

- Kernel como combinación convexa de kernels locales

$$K(\mathbf{x}_m, \mathbf{x}_n) = \sum_{i,l} \alpha_{il} K_{il}(\mathbf{x}_m, \mathbf{x}_n)$$

y los α_{il} cumplen las restricciones de ser convexos.

Resumen

- Breve exposición de cuatro artículos sobre aprendizaje de métricas
- Incorporar restricciones sobre parejas, tríos o cuádruplas de elementos que contienen información de dominio del problema
- Poda de grafo para limpiar la supervisión basada en etiquetados hechos por humanos
- Aprendizaje de métrica es equivalente a un aprendizaje de kernel, lo que permite extensión a conjuntos de test
- Aprendizaje de similitud OASIS
- Aprendizaje de métricas globales a partir de métricas locales