UAM
Universidad Autónoma de Madrid

Advanced Kernel Methods for Multi-Task Learning

Tesis dirigida por José Dorronsoro y Carlos Alaíz

Carlos Ruiz Pastor

January 10, 2023

UAM
Universidad Autónoma de Madrid

Acknowledgements 1

Acknowledgements 2

UAM
Universidad Autónoma de Madrid

Outline
o

UAM
Universidad Autónoma de Madrid

Table of Contents

1 Introducción

Introducción al Aprendizaje Automático

- El Aprendizaje Automático intenta automatizar el proceso de aprendizaje
- En el aprendizaje supervisado tenemos:
  - un espacio de entrada $\mathcal{X}$,
  - un espacio de salida $\mathcal{Y}$,
  - y una distribución $P(x, y)$ (desconocida) sobre $\mathcal{X} \times \mathcal{Y}$
- Dada una función $f : \mathcal{X} \to \mathcal{Y}$, definimos una función de pérdida como

$$\begin{aligned} \ell : \mathcal{Y} \times \mathcal{Y} &\to [0, \infty) \\ (y, f(x)) &\to \ell(y, f(x)), \end{aligned}$$

tal que $\ell(y, y) = 0$ para todo $y \in \mathcal{Y}$

Loss Functions

- In classification, with the class labels $y_i \in \{-1, 1\}$, we can use:

$$\ell(y, f(x)) = [1 - yf(x)]_+ = \begin{cases} 0, & yf(x) \geq 1, \\ 1 - yf(x), & yf(x) < 1. \end{cases}$$

-

Expected Risk

- Given a space of hypothesis $\mathcal{H} = \{h(\cdot, \alpha), \alpha \in A\}$
- Definition: Expected Risk

$$R_P(\alpha) = \int_{\mathcal{X}\times\mathcal{Y}} \ell(y, h(x, \alpha)) dP(x, y)$$

- Our goal is to find

$$\alpha^* = \arg\min_{\alpha\in A} \left\{ R_P(\alpha) = \int_{\mathcal{X}\times\mathcal{Y}} \ell(y, h(x, \alpha)) dP(x, y) \right\},$$

  however the distribution $P(x, y)$ is unknown

Empirical Risk

- Instead, we have a set of n instances sampled from $P(x, y)$:

$$D_n = \{(x_i, y_i),\ i = 1, \ldots, n\},$$

- Definition: Empirical Risk

$$\hat{R}_{D_n}(\alpha) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, h(x_i, \alpha))$$

- Instead of the Expected Risk, we minimize this empirical risk:

$$\arg\min_{\alpha \in A} \left\{ \hat{R}_D(\alpha) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, h(x_i, \alpha)) \right\}$$

UAM
Universidad Autónoma de Madrid

Table of Contents

1 Introducción

Multi-Task Learning

1 Introducción

UAM
Universidad Autónoma de Madrid

Table of Contents

UAM
Universidad Autónoma de Madrid

Table of Contents

2 Una Formulación Convexa para Aprendizaje Multitarea

Formulación Aditiva

- Una manera de implementar el MTL es combinar una parte común y otras específicas
- La formulación aditiva para el aprendizaje multitarea es

$$h_r(\cdot) = g(\cdot) + g_r(\cdot)$$

  donde
  - $g(\cdot)$ es la parte común
  - $g_r(\cdot)$ es la parte específica
- Fue propuesta para SVM lineales con los modelos

$$h_r(\cdot) = \langle w + v_r, \cdot \rangle + b_r$$

Formulación Convexa

- Proponemos la siguiente formulación convexa para el aprendizaje multitarea:

$$h_r(\cdot) = \lambda_r g(\cdot) + (1 - \lambda_r) g_r(\cdot),$$

  con $\lambda_r \in [0, 1]$.
- Los hiperparámetros λ_r regulan la influencia de cada parte:
  - $\lambda_1, \ldots, \lambda_T = 0$: modelos independientes (ITL)
  - $\lambda_1, \ldots, \lambda_T = 1$: modelo común (CTL)
- La interpretación de los hiperparámetros es más sencilla

UAM
Universidad Autónoma de Madrid

Table of Contents

2 Una Formulación Convexa para Aprendizaje Multitarea

Formulacion Convexa con Métodos de Kernel

2 Una Formulación Convexa para Aprendizaje Multitarea

- La formulación aditiva con métodos de kernel puede expresarse con los modelos:

$$h_r(\cdot) = \{\langle w, \phi(\cdot)\rangle + b\} + \{\langle v_r, \phi_r(\cdot)\rangle + d_r\}$$

- Con nuestra formulación convexa los modelos son:

$$h_r(\cdot) = \lambda_r \{\langle w, \phi(\cdot)\rangle + b\} + (1-\lambda_r)\{\langle v_r, \phi_r(\cdot)\rangle + d_r\}$$

- Desarrollamos tres variantes de SVM:
  - L1-SVM
  - L2-SVM
  - LS-SVM

Formulación Aditiva para MTL L1-SVM

Problema Primal - L1-SVM Aditiva

$$\begin{aligned} \underset{\boldsymbol{w},\boldsymbol{v},\boldsymbol{b},\boldsymbol{\xi}}{\arg\min} \quad & J(\boldsymbol{w},\boldsymbol{v},\boldsymbol{b},\boldsymbol{\xi}) = C\sum_{r=1}^{T}\sum_{i=1}^{m_r}\xi_i^r + \frac{1}{2}\sum_{r=1}^{T}\|v_r\|^2 + \frac{\mu}{2}\|w\|^2 \\ \text{s.t.} \quad & y_i^r(\langle w,\phi(x_i^r)\rangle + \langle v_r,\phi_r(x_i^r)\rangle + b_r) \geq p_i^r - \xi_i^r, \\ & \xi_i^r \geq 0;\ i=1,\ldots,m_r,\ r=1,\ldots,T. \end{aligned}$$

- El parámetro μ (junto con C) regula la influencia de cada parte:
  - $\mu \to \infty$: modelos independientes (ITL)
  - $C \to 0,\ \mu \to 0$: modelo común (CTL)

Formulación Convexa para L1-SVM MT

Problema Primal - L1-SVM Convexa

$$\min_{w,\boldsymbol{v},b,\boldsymbol{d},\boldsymbol{\xi}} \quad J(w,\boldsymbol{v},b,\boldsymbol{d},\boldsymbol{\xi}) = C\sum_{r=1}^{T}\sum_{i=1}^{m_r}\xi_i^r + \frac{1}{2}\sum_{r=1}^{T}\|v_r\|^2 + \frac{1}{2}\|w\|^2$$

$$\text{s.t.} \quad y_i^r\left(\lambda_r\left\{\langle w,\phi(x_i^r)\rangle + b\right\} + (1-\lambda_r)\left\{\langle v_r,\phi_r(x_i^r)\rangle + d_r\right\}\right) \geq p_i^r - \xi_i^r,$$

$$\xi_i^r \geq 0,\ i = 1,\ldots,m_r,\ r = 1,\ldots,T.$$

- Los hiperparámetros λ_r regulan la influencia de cada parte:
  - $\lambda_1,\ldots,\lambda_T = 0$: modelos independientes (ITL)
  - $\lambda_1,\ldots,\lambda_T = 1$: modelo común (CTL)
- El hiperparámetro C no interviene en la definición de los modelos

Formulación Convexa para L1-SVM MT

Problema Dual - L1-SVM Convexa

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \Theta(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^\intercal \left(\Lambda Q \Lambda + (I_n - \Lambda)\, K\, (I_n - \Lambda)\right)\boldsymbol{\alpha} - \boldsymbol{p}\boldsymbol{\alpha} \\ \text{s.t.} \quad & 0 \leq \alpha_i^r \leq C;\ i = 1, \ldots, m_r,\ r = 1, \ldots, T, \\ & \sum_{i=1}^{m_r} \alpha_i^r y_i^r = 0;\ r = 1, \ldots, T, \end{aligned}$$

- Usamos la matriz $\Lambda = \mathrm{diag}(\overbrace{\lambda_1, \ldots, \lambda_1}^{m_1}, \ldots, \overbrace{\lambda_T, \ldots, \lambda_T}^{m_T})$
- La matriz Q es común entre todas las tareas usando el kernel k correspondiente a ϕ
- La matriz K es diagonal por bloques, con los kernel k_r correspondientes a ϕ_r
- La función de kernel es:

$$\widehat{k}(x_i^r, x_j^s) = \lambda_r \lambda_s k(x_i^r, x_j^s) + \delta_{rs}(1 - \lambda_r)(1 - \lambda_s) k_r(x_i^r, x_j^s)$$

Formulación Convexa para L1-SVM MT

Problema Dual - L1-SVM Convexa (λ común)

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \Theta(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^\intercal \left(\lambda^2 Q + (1-\lambda)^2 K\right)\boldsymbol{\alpha} - \boldsymbol{p}\boldsymbol{\alpha} \\ \text{s.t.} \quad & 0 \leq \alpha_i^r \leq C;\ i = 1, \ldots, m_r,\ r = 1, \ldots, T, \\ & \sum_{i=1}^{m_r} \alpha_i^r y_i^r = 0;\ r = 1, \ldots, T, \end{aligned}$$

- La función de kernel es:

$$\widehat{k}(x_i^r, x_j^s) = \lambda^2 k(x_i^r, x_j^s) + (1-\lambda)^2 \delta_{rs} k_r(x_i^r, x_j^s)$$

- El hiperparámetro λ regula la influencia de cada parte:
  - $\lambda = 0$: modelos independientes (ITL)
  - $\lambda = 1$: modelo común (CTL)

Proposiciones

Proposicion (Equivalencia entre formulaciones para L1-SVM)

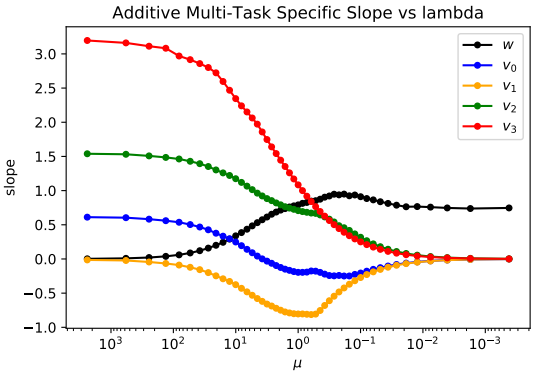
Para valores $\lambda \in (0, 1)$, la formulación aditiva con hiperparámetros C_{add}, μ y la formulación convexa con C_{conv} y un λ común, $\lambda_1, \ldots, \lambda_T = \lambda$, son equivalentes cuando

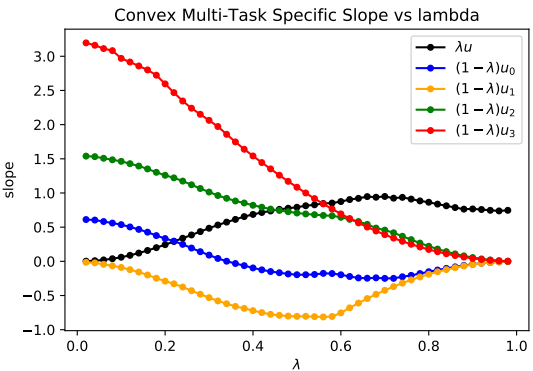
$$C_{add} = (1 - \lambda)^2 C_{conv}, \ \mu = (1 - \lambda)^2 / \lambda^2.$$

- Para $\lambda = 0$, la formulación convexa con un λ común es equivalente a modelos independientes (ITL).
- Para $\lambda = 1$ la formulación convexa con un λ común es equivalente a un modelo común (CTL).

Formulación Aditiva vs Formulación Convexa

2 Una Formulación Convexa para Aprendizaje Multitarea





Formulación Convexa para L2-SVM MT

Problema Primal - MTL L2-SVM Convexa

$$\begin{aligned} \underset{w,\boldsymbol{v},b,\boldsymbol{d},\xi}{\arg\min} \quad & J(w,\boldsymbol{v},b,\boldsymbol{d},\boldsymbol{\xi}) = \frac{C}{2}\sum_{r=1}^{T}\sum_{i=1}^{m_r}(\xi_i^r)^2 + \frac{1}{2}\sum_{r=1}^{T}\|v_r\|^2 + \frac{1}{2}\|w\|^2 \\ \text{s.t.} \quad & y_i^r\left(\lambda_r\left\{\langle w,\phi(x_i^r)\rangle + b\right\} + (1-\lambda_r)\left\{\langle v_r,\phi_r(x_i^r)\rangle + d_r\right\}\right) \geq p_i^r - \xi_i^r, \end{aligned}$$

Problema Dual - MTL L2-SVM Convexa

$$\begin{aligned} \min_{\alpha} \quad & \Theta(\alpha) = \frac{1}{2}\boldsymbol{\alpha}^\intercal\left(\left\{\Lambda Q\Lambda + (I_n-\Lambda)\,K\,(I_n-\Lambda)\right\} + \frac{1}{C}I\right)\boldsymbol{\alpha} - \boldsymbol{p}\boldsymbol{\alpha} \\ \text{s.t.} \quad & 0 \leq \alpha_i^r,\ i=1,\ldots,m_r,\ r=1,\ldots,T, \\ & \sum_{i=1}^{m_r}\alpha_i^r y_i^r = 0,\ r=1,\ldots,T. \end{aligned}$$

Formulación Convexa para LS-SVM MT

Problema Primal - MTL LS-SVM Convexa

$$\underset{w,\boldsymbol{v},b,\boldsymbol{d},\xi}{\arg\min} \quad J(w,\boldsymbol{v},b,\boldsymbol{d},\boldsymbol{\xi}) = \frac{C}{2}\sum_{r=1}^{T}\sum_{i=1}^{m_r}(\xi_i^r)^2 + \frac{1}{2}\sum_{r=1}^{T}\|v_r\|^2 + \frac{1}{2}\|w\|^2$$

$$\text{s.t.} \qquad y_i^r\left(\lambda_r\left\{\langle w,\phi(x_i^r)\rangle + b\right\} + (1-\lambda_r)\left\{\langle v_r,\phi_r(x_i^r)\rangle + d_r\right\}\right) = p_i^r - \xi_i^r,$$

Problema Dual - MTL LS-SVM Convexa

$$\left[\begin{array}{c|c|c} 0 & \mathbf{0}_T^\intercal & \boldsymbol{y}^\intercal\Lambda \\ \hline \mathbf{0}_T & \mathbf{0}_{T\times T} & A^\intercal Y(I_n-\Lambda) \\ \hline \boldsymbol{y} & YA & \widehat{Q}+\frac{1}{C}I_n \end{array}\right]\begin{pmatrix} b \\ d_1 \\ \vdots \\ d_T \\ \boldsymbol{\alpha} \end{pmatrix} = \begin{pmatrix} 0 \\ \mathbf{0}_T \\ \boldsymbol{p} \end{pmatrix},$$

UAM
Universidad Autónoma de Madrid

Table of Contents

2 Una Formulación Convexa para Aprendizaje Multitarea

Combinación Convexa de modelos Preentrenados

- Alternativa a la formulación convexa para aprendizaje MT
- Consideramos la combinación convexa de
  - modelo común $g(\cdot)$ entrenado
  - modelos específicos $g_r(\cdot)$ entrenados
- Minimizamos el riesgo eligiendo los hiperparámetros $\lambda_1, \ldots, \lambda_T$ óptimos

$$\hat{R}_D(\lambda_1, \ldots, \lambda_T) = \sum_{r=1}^{T} \sum_{i=1}^{m_r} \ell(\lambda_r g(x_i^r) + (1 - \lambda_r) g_r(x_i^r), y_i^r),$$

Formulación Unificada Clasificación

2 Una Formulación Convexa para Aprendizaje Multitarea

- Hinge loss (classification):

$$\hat{R}_D(\lambda_1, \ldots, \lambda_T) = \sum_{r=1}^{T} \sum_{i=1}^{m_r} \left[1 - y_i^r \left\{\lambda_r g(x_i^r) + (1 - \lambda_r) g_r(x_i^r)\right\}\right]_+ .$$

- Squared hinge loss (classification):

$$\hat{R}_D(\lambda_1, \ldots, \lambda_T) = \sum_{r=1}^{T} \sum_{i=1}^{m_r} \left[1 - y_i^r \left\{\lambda_r g(x_i^r) + (1 - \lambda_r) g_r(x_i^r)\right\}\right]_+^2 .$$

- Ambas se pueden expresar como:

$$\sum_{r=1}^{T} \sum_{i=1}^{m_r} u(\lambda_r c_i^r + d_i^r), \text{ donde } c_i^r = y_i^r (g_r(x_i^r) - g(x_i^r)), \; d_i^r = 1 - y_i^r g_r(x_i^r)$$

Formulación Unificada Regresión

2 Una Formulación Convexa para Aprendizaje Multitarea

- Absolute loss (regression):

$$\hat{R}_D(\lambda_1, \ldots, \lambda_T) = \sum_{r=1}^{T} \sum_{i=1}^{m_r} \left| y_i^r - \{\lambda_r g(x_i^r) + (1 - \lambda_r) g_r(x_i^r)\} \right| .$$

- Squared loss (regression):

$$\hat{R}_D(\lambda_1, \ldots, \lambda_T) = \sum_{r=1}^{T} \sum_{i=1}^{m_r} \left(y_i^r - \{\lambda_r g(x_i^r) + (1 - \lambda_r) g_r(x_i^r)\} \right)^2 .$$

- Ambas se pueden expresar como:

$$\sum_{r=1}^{T} \sum_{i=1}^{m_r} u(\lambda_r c_i^r + d_i^r), \text{ donde } c_i^r = g(x_i^r) - g_r(x_i^r), \; d_i^r = g_r(x_i^r) - y_i^r$$

Formulación Unificada

2 Una Formulación Convexa para Aprendizaje Multitarea

- En todos los casos tenemos que minimizar

$$\hat{R}_D(\lambda_1, \ldots, \lambda_T) = \sum_{r=1}^{T} \sum_{i=1}^{m_r} u(\lambda_r c_i^r + d_i^r)$$

- Como es separable, tenemos en cada tarea el problema

$$\arg\min_{\lambda_r \in [0,1]} \mathcal{J}(\lambda_r) = \sum_{i=1}^{m_r} u(\lambda_r c_i^r + d_i^r),$$

- Usando el Teorema de Fermat

$$\lambda^* = \arg\min_{0 \leq \lambda \leq 1} \mathcal{J}(\lambda) \iff (0 \in \partial \mathcal{J}(\lambda^*) \text{ and } \lambda^* \in (0,1)) \text{ or } \lambda^* = 0 \text{ or } \lambda^* = 1.$$

Combinación Convexa con Error Cuadrático

2 Una Formulación Convexa para Aprendizaje Multitarea

- La función a minimizar es

$$\arg\min_{\lambda \in [0,1]} \mathcal{J}(\lambda) = \sum_{i=1}^{m} (\lambda c_i + d_i)^2 .$$

- La derivada es

$$\mathcal{J}'(\lambda) = \sum_{i=1}^{m} 2c_i(\lambda c_i + d_i).$$

- Como es derivable, resolviendo $\mathcal{J}'(\lambda) = 0$ obtenemos

$$\lambda' = -\frac{\sum_{i=1}^{m} d_i c_i}{\sum_{i=1}^{m} (c_i)^2}.$$

- La solución es entonces $\lambda^* = \max(\min(\lambda', 1), 0)$

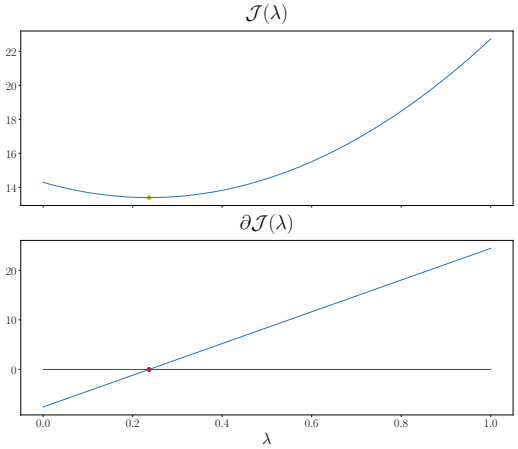


Figure: Error using the squared loss function (top) and its corresponding derivative (bottom). The

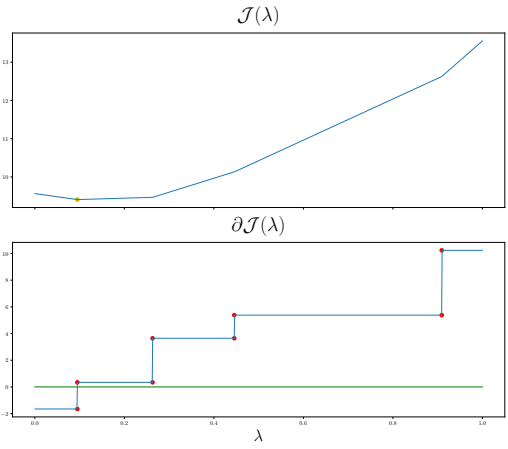
Combinación Convexa con Error Absoluto

Proposicion (λ^* óptimo para el problema con valor absoluto)

- $\lambda^* = 0$ *es óptimo si y solo si:* $-\sum_{i:\, 0>\lambda_{(i)}} |c_{(i)}| + \sum_{i:\, 0<\lambda_{(i)}} |c_{(i)}| \leq 0$
- $\lambda^* \in (0,1)$ *es óptimo si y solo si* $0 < \lambda^* = \lambda_{(k)} < 1$ *para algún* $k = 1, \ldots, m$, *y*

$$-\sum_{i:\, \lambda_{(k)}>\lambda_{(i)}} |c_{(i)}| + \sum_{i:\, \lambda_{(k)}<\lambda_{(i)}} |c_{(i)}| \in \left[-|c_{(k)}|, |c_{(k)}|\right]$$

- $\lambda^* = 1$ *es óptimo en otro caso*


$\mathcal{J}(\lambda)$
13
12
11
10
$\partial\mathcal{J}(\lambda)$
10
8
6
4
2
0
−2
0.0
0.2
0.4
0.6
0.8
1.0
λ

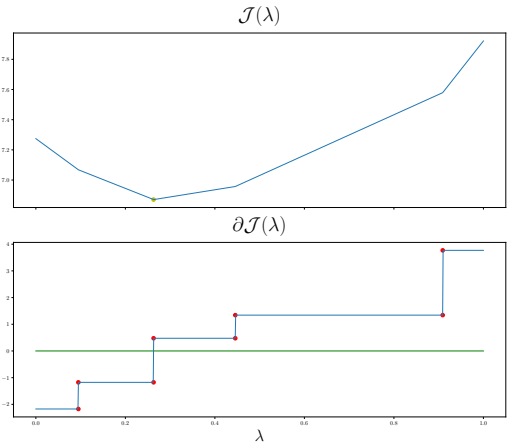
Combinación Convexa con Error Hinge

Proposicion (λ^* óptimo para el problema con error hinge)

- $\lambda^* = 0$ *es óptimo si y solo si:* $-\sum_{i:\, 0>\lambda_{(i)}} \max\left(0, c_{(i)}\right) - \sum_{0<\lambda_{(i)}} \min\left(0, c_{(i)}\right) \leq 0$
- $\lambda^* \in (0,1)$ *es óptimo si y solo si* $0 < \lambda^* = \lambda_{(k)} < 1$ *para algún* $k = 1, \ldots, m$, *y*

$$-\sum_{i:\, \lambda_{(k)}>\lambda_{(i)}} \max\left(0, c_{(i)}\right) - \sum_{i:\, \lambda_{(k)}<\lambda_{(i)}} \min\left(0, c_{(i)}\right) \in \left[\min\left(0, c_{(k)}\right), \max\left(0, c_{(k)}\right)\right]$$

- $\lambda^* = 1$ *es óptimo en otro caso*


$\mathcal{J}(\lambda)$
$\partial\mathcal{J}(\lambda)$
λ

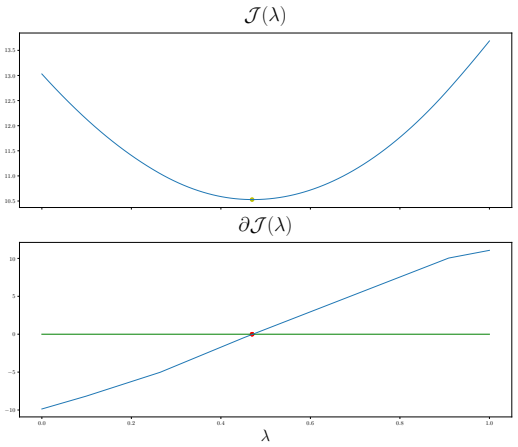
Combinación Convexa con Error Hinge Cuadrático

Proposicion (λ^* óptimo para el problema con error hinge cuadrático)

- $\lambda^* = 0$ *es óptimo si y solo si:* $-\sum_{i:\ 0>c_{(i)},0<\lambda_{(i)}} 2c_i d_i - \sum_{i:\ 0<c_{(i)},0>\lambda_{(i)}} 2c_i d_i \leq 0$
- $\lambda^* \in (0,1)$ *es óptimo si y solo si* $0 < \lambda^* = \widehat{\lambda}_{(k)} < 1$ *para algún* $k = 1, \ldots, m$*, donde*

$$\widehat{\lambda}_{(k)} = -\frac{\sum_{i:\ \lambda_{(k+1)} \geq \lambda_{(i)}} \max\left(0, c_{(i)}\right) d_{(i)} + \sum_{i:\ \lambda_{(k)} \leq \lambda_{(i)}} \min\left(0, c_{(i)}\right) d_{(i)}}{\sum_{i:\ \lambda_{(k+1)} \geq \lambda_{(i)}} \max\left(0, c_{(i)}\right)^2 + \sum_{i:\ \lambda_{(k)} \leq \lambda_{(i)}} \min\left(0, c_{(i)}\right)^2},$$

  y además $\lambda_{(k)} \leq \widehat{\lambda}_k \leq \lambda_{(k+1)}$
- $\lambda^* = 1$ *es óptimo en otro caso*


$\mathcal{J}(\lambda)$
13.5
13.0
12.5
12.0
11.5
11.0
10.5
$\partial\mathcal{J}(\lambda)$
10
5
0
−5
−10
0.0
0.2
0.4
0.6
0.8
1.0
λ

Experimentos: Modelos

- **Common Task Learning LX-SVM (CTL-LX)**: Un único modelo LX-SVM que es común para todas las tareas
- **Independent Task Learning LX-SVM (ITL-LX)**: Un modelo LX-SVM independiente para cada tarea.
- **Direct Convex Combination of LX-SVMs (CMB-LX)**: Una combinación convexa de los mejores CTL-LX y ITL-LX.
- **Convex Multi-Task Learning LX-SVM (MTL-LX)**: Un modelo multitarea con la formulación convexa basado en la LX-SVM

UAM
Universidad Autónoma de Madrid

Experimentos: Problemas

2 Una Formulación Convexa para Aprendizaje Multitarea

Dataset	Size	No. feat.	No. tasks	Avg. task size	Min. t. s.	Max. t. s.
majorca	15 330	765	14	1095	1095	1095
tenerife	15 330	765	14	1095	1095	1095
california	19 269	9	5	3853	5	8468
boston	506	12	2	253	35	471
abalone	4177	8	3	1392	1307	1527
crime	1195	127	9	132	60	278
binding	32 302	184	47	687	59	3089
landmine	14 820	10	28	511	445	690
adult_(G)	48 842	106	2	24 421	16 192	32 650
adult_(R)	48 842	103	5	9768	406	41 762
adult_(G, R)	48 842	101	10	4884	155	28 735
compas_(G)	3987	11	2	1993	840	3147
compas_(R)	3987	9	4	997	255	1918
compas_(G, R)	3987	7	8	498	50	1525

Experimentos: Procedimiento

- Para majorca u tenerife, usamos los datos de 2013, 2014 and 2015 como conjuntos de entrenamiento, validación y test, respectivamente
- Para el resto, usamos una CV con 3 particiones externas e internas estratificadas por tareas
- Los hiperparámetros se eligen con una búsqueda en rejilla con las particiones de entrenamiento y validación
- Obtenemos 3 scores de test para cada modelo en cada problema

Experimentos: Hiperparámetros

- Due to computational limitations a maximum of three hyperparameters are included in the CV process.
- The kernel widths for the MTL models are selected from the CTL and ITL models.

	Grid	CTL-L1,2	ITL-L1,2	MTL-L1,2	CTL-LS	ITL-LS	MTL-LS
C	$\{4^k : -2 \leq k \leq 6\}$	CV	CV	CV	CV	CV	CV
ϵ	$\left\{\frac{\sigma}{4^k} : 1 \leq k \leq 6\right\}$	CV	CV	CV	-	-	-
γ_c	$\left\{\frac{4^k}{d} : -2 \leq k \leq 3\right\}$	CV	-	CTL-L1,2	CV	-	CTL-LS
γ_s^r	$\left\{\frac{4^k}{d} : -2 \leq k \leq 3\right\}$	-	CV	ITL-L1,2	-	CV	ITL-LS
λ	$\{0.1k : 0 \leq k \leq 10\}$	-	-	CV	-	-	CV

UAM
Universidad Autónoma de Madrid

Experimentos: Resultados de Regresión (MAE)

2 Una Formulación Convexa para Aprendizaje Multitarea

	maj.	ten.	boston	california	abalone	crime
ITL-L1	5.087 (6)	5.743 (3)	2.341±0.229 **(1)**	36883.582±418.435 (2)	1.481±0.051 (3)	0.078±0.001 (2)
CTL-L1	5.175 (7)	5.891 (5)	**2.192±0.244 (1)**	41754.337±270.908 (6)	1.482±0.050 (3)	0.078±0.001 (2)
CMB-L1	**5.047** (5)	**5.340 (1)**	2.239±0.255 **(1)**	36880.238±420.417 **(1)**	1.470±0.052 (2)	0.077±0.002 (2)
MTL-L1	5.050 (5)	5.535 (2)	2.206±0.292 **(1)**	**36711.383±343.333 (1)**	**1.454±0.048 (1)**	**0.074±0.002 (1)**
ITL-L2	4.952 (3)	**5.629** (3)	2.356±0.300 **(1)**	37374.618±433.511 (5)	1.498±0.054 (4)	0.079±0.002 (2)
CTL-L2	5.193 (7)	6.107 (8)	**2.083±0.136 (1)**	42335.612±163.773 (8)	1.503±0.047 (5)	0.080±0.002 (2)
CMB-L2	4.869 (3)	5.963 (6)	2.089±0.128 **(1)**	37374.618±433.511 (4)	1.494±0.050 (4)	0.077±0.003 (2)
MTL-L2	**4.854** (2)	5.784 (4)	2.089±0.134 **(1)**	**37202.603±419.166** (3)	**1.482±0.049** (3)	**0.077±0.002** (2)
ITL-LS	4.937 (3)	5.649 (3)	2.204±0.116 **(1)**	37348.347±441.240 (4)	1.496±0.051 (4)	0.079±0.002 (2)
CTL-LS	5.193 (7)	6.005 (7)	**2.072±0.143 (1)**	42259.492±146.825 (7)	1.502±0.052 (5)	0.079±0.002 (2)
CMB-LS	4.977 (4)	**5.593** (3)	2.081±0.146 **(1)**	37339.179±430.288 (4)	1.486±0.049 (4)	0.079±0.002 (2)
MTL-LS	**4.824 (1)**	5.754 (4)	2.077±0.152 **(1)**	**37231.043±420.992** (4)	**1.478±0.050** (3)	**0.076±0.002** (2)

UAM
Universidad Autónoma de Madrid

Experimentos: Resultados de Regresión (MSE)

	maj.	ten.	boston	california	abalone	crime
ITL-L1	0.845 (6)	0.901 (7)	0.821±0.041 (2)	0.699±0.009 (7)	0.543±0.022 (8)	0.732±0.021 (3)
CTL-L1	0.837 (9)	0.901 (6)	0.854±0.036 **(1)**	0.639±0.006 (10)	0.559±0.014 (6)	0.740±0.027 (3)
CMB-L1	0.844 (6)	0.905 (4)	0.845±0.053 **(1)**	0.699±0.009 (6)	0.555±0.018 (7)	0.741 ±0.029 (3)
MTL-L1	**0.846** (4)	**0.908** (2)	**0.858±0.057 (1)**	**0.703±0.007** (6)	**0.568±0.012** (5)	**0.760±0.024** (2)
ITL-L2	0.846 (5)	0.906 (3)	0.836±0.045 (2)	0.707±0.009 (5)	0.565±0.025 (6)	0.743±0.017 (3)
CTL-L2	0.840 (8)	0.901 (8)	**0.889±0.017 (1)**	0.645±0.005 (9)	0.574±0.013 (4)	0.744±0.028 (3)
CMB-L2	0.850 (3)	0.900 (9)	0.885±0.013 **(1)**	0.707±0.009 (4)	0.571±0.018 (4)	0.755±0.024 (3)
MTL-L2	**0.863** (2)	**0.908 (1)**	0.888±0.015 **(1)**	**0.709±0.008 (1)**	**0.580±0.014** (3)	**0.762±0.028 (1)**
ITL-LS	0.849 (3)	0.907 (3)	0.856±0.008 **(1)**	0.707±0.009 (3)	0.573±0.015 (4)	0.743±0.022 (3)
CTL-LS	0.838 (9)	0.904 (5)	**0.894±0.015 (1)**	0.646±0.005 (8)	0.576±0.016 (4)	0.746±0.032 (3)
CMB-LS	0.843 (7)	0.907 (2)	0.886±0.024 **(1)**	0.707±0.009 (2)	0.581±0.012 (2)	0.746±0.021 (3)
MTL-LS	**0.863 (1)**	**0.910 (1)**	0.890±0.016 **(1)**	**0.709±0.008** (2)	**0.581±0.015 (1)**	**0.763±0.028 (1)**

UAM
Universidad Autónoma de Madrid

Experimentos: Resultados de Clasificación (Score F1)

	comp_(G)	comp_(R)	comp_(G,R)	ad_(G)	ad_(R)	ad_(G,R)	landmine	binding	mean	rank	Wil.
ITL-L1	0.625	0.639	0.630	0.659	0.653	0.657	0.231	0.867	0.620	10	2
CTL-L1	0.623	0.638	0.638	0.657	0.650	0.653	0.255	0.901	0.627	7	2
CMB-L1	0.616	0.638	0.638	0.658	0.650	0.653	0.270	0.901	0.628	6	2
MTL-L1	0.627	0.636	0.640	0.659	0.655	0.659	0.242	0.907	0.628	5	2
ITL-L2	0.636	0.623	0.607	0.668	0.666	0.668	0.256	0.867	0.624	8	2
CTL-L2	0.640	0.647	0.651	0.665	0.661	0.659	0.270	0.903	0.637	2	2
CMB-L2	0.629	0.640	0.645	0.666	0.662	0.661	0.270	0.903	0.634	3	2
MTL-L2	0.634	0.651	0.650	0.668	0.666	0.668	0.263	0.909	0.639	1	1
ITL-LS	0.631	0.622	0.608	0.659	0.659	0.660	0.243	0.867	0.619	12	2
CTL-LS	0.628	0.644	0.649	0.650	0.653	0.647	0.230	0.853	0.619	11	2
CMB-LS	0.630	0.635	0.642	0.657	0.658	0.654	0.238	0.873	0.623	9	2
MTL-LS	0.630	0.641	0.648	0.659	0.659	0.659	0.257	0.906	0.632	4	2

Experimentos: Resultados de Clasificación (Accuracy)

	comp_(G)	comp_(R)	comp_(G,R)	ad_(G)	ad_(R)	ad_(G,R)	landmine	binding	mean	rank	Wil.
ITL-L1	0.750	0.749	0.746	0.852	0.851	0.853	0.941	0.790	0.817	11	3
CTL-L1	0.757	0.759	0.763	0.852	0.847	0.849	0.938	0.850	0.827	6	1
CMB-L1	0.754	0.759	0.763	0.852	0.847	0.849	0.935	0.850	0.826	7	2
MTL-L1	0.753	0.760	0.763	0.853	0.852	0.853	0.933	0.861	0.829	5	1
ITL-L2	0.754	0.762	0.751	0.856	0.855	0.856	0.942	0.791	0.821	8	2
CTL-L2	0.762	0.765	0.767	0.854	0.853	0.851	0.933	0.853	0.830	3	1
CMB-L2	0.757	0.764	0.766	0.854	0.853	0.853	0.934	0.853	0.829	4	1
MTL-L2	0.753	0.766	0.766	0.856	0.855	0.856	0.933	0.864	0.831	1	1
ITL-LS	0.754	0.761	0.750	0.851	0.850	0.851	0.943	0.791	0.819	9	3
CTL-LS	0.757	0.764	0.766	0.845	0.847	0.842	0.914	0.750	0.811	12	3
CMB-LS	0.754	0.764	0.765	0.849	0.850	0.848	0.925	0.793	0.818	10	3
MTL-LS	0.757	0.764	0.767	0.851	0.850	0.851	0.944	0.858	0.830	2	1

UAM
Universidad Autónoma de Madrid

Table of Contents

2 Una Formulación Convexa para Aprendizaje Multitarea

Redes Neuronales MT

2 Una Formulación Convexa para Aprendizaje Multitarea

- La manera más común de adaptar las redes neuronales es el *hard sharing*
  - Capas ocultas compartidas por todas las tareas
  - Capas de salida específicas para cada tarea
- El modelo se puede expresar como:

$$h_r(\cdot) = g_r(\cdot; w_r, \Theta) = \{\langle w_r, f(\cdot; \Theta)\rangle\} + d_r$$

  - w_r, d_r son los parámetros de las capas de salida específicas
  - Θ son los parámetros de las capas ocultas compartidas

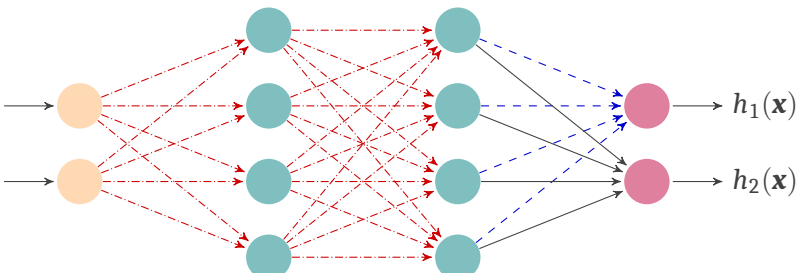


Figure: Ejemplo de *Hard Sharing* para dos tareas .

Formulación Convexa para Redes Neuronales MT

2 Una Formulación Convexa para Aprendizaje Multitarea

- Proponemos la formulación convexa para redes neuronales MT, combinando:
  - Una parte común $g(\cdot; w, \Theta)$
  - Una parte específica $g_r(\cdot; w_r, \Theta_r)$
- Los modelos son:

$$\begin{aligned} h_r(\cdot) &= \lambda_r g(\cdot; w, \Theta) + (1 - \lambda_r) g_r(\cdot; w_r, \Theta_r) \\ &= \lambda_r \{\langle w, f(\cdot; \Theta)\rangle + b\} + (1 - \lambda_r)\{\langle w_r, f_r(\cdot; \Theta_r)\rangle + d_r\}. \end{aligned}$$

  - w, Θ son los parámetros de la red común (capa de salida y ocultas)
  - w_r, Θ_r son los parámetros de las redes específicas (capa de salida y ocultas)

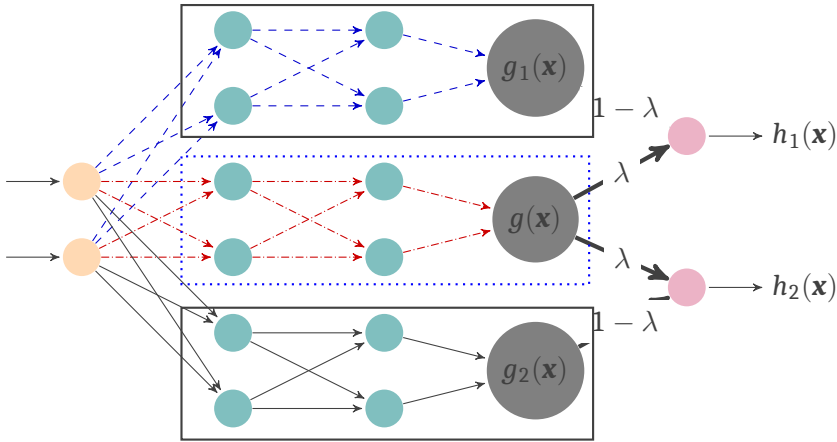


Figure: Ejemplo de formulación convexa con redes neuronales para dos tareas.

Formulación Convexa para Redes Neuronales MT

2 Una Formulación Convexa para Aprendizaje Multitarea

- El riesgo a minimizar en este caso es

$$\hat{R}_D = \sum_{r=1}^{T} \sum_{i=1}^{m_r} \ell(h_r(x_i^r), y_i^r) + \frac{\mu}{2} \left(\|w\|^2 + \sum_{r=1}^{T} \|w_r\|^2 + \Omega(\Theta) + \Omega(\Theta_r) \right).$$

- Se puede aplicar el descenso por gradiente con

$$\begin{aligned}
&\nabla_w h_t(x_i^t) = \lambda_t f(x_i^t, \Theta), && \nabla_\Theta h_t(x_i^t) = \lambda_t \left\langle w, \nabla_\Theta f(x_i^t, \Theta) \right\rangle; \\
&\nabla_{w_t} h_t(x_i^t) = (1-\lambda_t) f_t(x_i^t, \Theta), && \nabla_{\Theta_t} h_t(x_i^t) = (1-\lambda_t) \left\langle w, \nabla_{\Theta_t} f_t(x_i^t, \Theta_t) \right\rangle; \\
&\nabla_{w_r} h_t(x_i^t) = 0, && \nabla_{\Theta_r} h_t(x_i^t) = 0, \text{ for } r \neq t.
\end{aligned}$$

- Los gradientes se escalan adecuadamente con λ_t y $(1-\lambda_t)$

Formulación Convexa para Redes Neuronales MT

2 Una Formulación Convexa para Aprendizaje Multitarea

Algorithm 1: Pase "forward"

Input: $X_{\text{mb}}, t_{\text{mb}}$ `// Minibatch data and task labels`

Output: f `// Forward pass for the minibatch`

Data: λ `// Parameter of convex combination`

Data: $g; g_1, \ldots, g_T$ `// Modules of the common and specific networks`

for $x_i, t_i \in (X_{mb}, t_{mb})$ **do**

  $f_i \leftarrow \lambda g(x_i) + (1-\lambda) g_{t_i}(x_i)$ `// Convex combination`

end

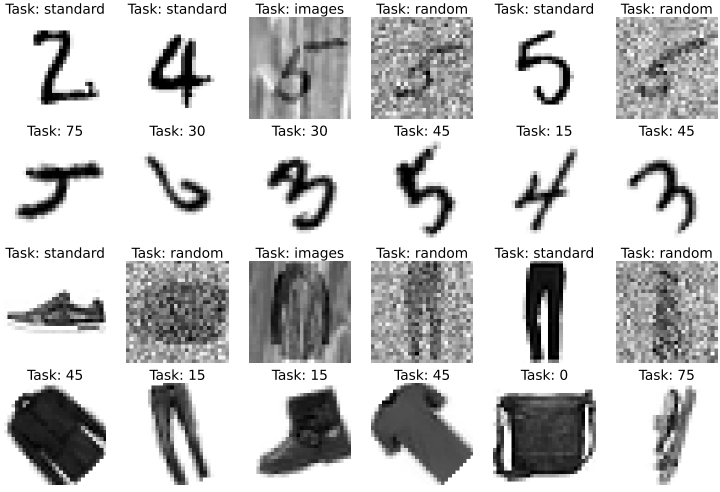
- El pase "backward" se hace con la diferenciación automática de `PyTorch`

Datasets

- Usamos cuatro 28×28 datasets de imágenes en escala de grises:
  - var-MNIST
  - rot-MNIST
  - var-FMNIST
  - rot-FMNIST
- Cada uno con 70k ejemplos y 10 clases
- Los datasets *variations* tienen 3 tareas: *standard*, *random*, *images*
- Los datasets *rotated* tienen 6 tareas: $0, 15, 30, 45, 60, 75$

Datasets

2 Una Formulación Convexa para Aprendizaje Multitarea



Models

- Comparamos cuatro modelos:
  - ctlNN_conv
  - itlNN_conv
  - cvxmtlNN_conv
  - hsmtlNN_conv
- Todos están basados en una red convolucional de `Pytorch` con
  - Conv. Layer (10 output channels)
  - Conv. Layer (20 output channels)
  - Dropout ($p = 0.5$) and Max. Pooling
  - Fully Connected Layer (320 neurons)
  - Fully Connected Layer (50 neurons)
- Todos los modelos se entrenan con el algoritmo *AdamW*

Resultados

2 Una Formulación Convexa para Aprendizaje Multitarea

	var-MNIST	rot-MNIST	var-FMNIST	rot-FMNIST
		accuracy		
ctlNN	0.964	0.973	0.784	0.834
itlNN	0.968	0.981	0.795	0.873
hsmtlNN	0.971	0.980	0.770	0.852
cvxmtlNN	**0.974** ($\lambda^* = 0.6$)	**0.984** ($\lambda^* = 0.8$)	**0.812** ($\lambda^* = 0.6$)	**0.880** ($\lambda^* = 0.6$)
		categorical cross-entropy		
ctlNN	1.274 ± 0.143	1.145 ± 0.039	2.369 ± 0.183	1.757 ± 0.075
itlNN	1.072 ± 0.029	0.873 ± 0.058	2.356 ± 0.130	1.598 ± 0.042
hsmtlNN	1.087 ± 0.253	0.898 ± 0.073	3.067 ± 0.888	1.888 ± 0.075
cvxmtlNN	$\mathbf{0.924 \pm 0.024}$ ($\lambda^* = 0.6$)	$\mathbf{0.831 \pm 0.029}$ ($\lambda^* = 0.8$)	$\mathbf{2.147 \pm 0.090}$ ($\lambda^* = 0.6$)	$\mathbf{1.482 \pm 0.063}$ ($\lambda^* = 0.6$)

UAM
Universidad Autónoma de Madrid

Table of Contents

3 Laplaciano Adaptativo para Aprendizaje Multitarea

Aprendizaje Multitarea con Regularización Laplaciana

3 Laplaciano Adaptativo para Aprendizaje Multitarea

- Otra manera de acoplar distintas tareas es usar una regularización Laplaciana
- Consideramos un grafo donde
  - Los nodos representan tareas
  - Las aristas y sus pesos representan las relaciones entre las tareas
- La matriz de adyacencia A tiene los pesos de las aristas
- La matriz de grados D es una matriz diagonal donde

$$(D)_{rr} = \sum_{s=1}^{T} (A)_{rs}$$

- La matriz Laplaciana se define como $L = D - A$

Aprendizaje Multitarea con Regularización Laplaciana

3 Laplaciano Adaptativo para Aprendizaje Multitarea

- Dados los modelos para cada tarea definidos como

$$h_r(\cdot) = \langle w_r, \cdot \rangle + b_r$$

- Definimos la regularización

$$\sum_{r=1}^{T} \sum_{s=1}^{T} (A)_{rs} \left\| w_r - w_s \right\|^2 ,$$

- Esta regularización se puede expresar como

$$\sum_{r=1}^{T} \sum_{s=1}^{T} (A)_{rs} \left\| w_r - w_s \right\|^2 = \sum_{r=1}^{T} \sum_{s=1}^{T} (L)_{rs} \langle w_r, w_s \rangle ,$$

UAM
Universidad Autónoma de Madrid

Table of Contents

3 Laplaciano Adaptativo para Aprendizaje Multitarea

Laplaciano de Grafo con Métodos de Kernel

3 Laplaciano Adaptativo para Aprendizaje Multitarea

- Consideramos el problema de minimización

$$R(u_1, \ldots, u_T) = \sum_{r=1}^{T} \sum_{i=1}^{m_r} \ell(y_i^r, \langle u_r, \phi(x_i^r) \rangle) + \mu \sum_r \sum_s (E)_{rs} \langle u_r, u_s \rangle \qquad (0)$$

- Si usamos el vector $\boldsymbol{u}^\intercal = (u_1^\intercal, \ldots, u_T^\intercal)$ lo expresamos como

$$R(\boldsymbol{u}) = \sum_{r=1}^{T} \sum_{i=1}^{m_r} \ell(y_i^r, \langle \boldsymbol{u}, e_r \otimes \phi(x_i^r) \rangle) + \mu \left(\boldsymbol{u}^\intercal (E \otimes I) \boldsymbol{u} \right) \qquad (1)$$

donde $\otimes$ indica el producto tensorial y $e_1, \ldots e_T$ es la base canónica de $\mathbb{R}^T$

Lemma

Las soluciones $u_1^*, \ldots, u_T^*$ de (0), o equivalentemente la solución $\boldsymbol{u}^*$ de (1), se pueden obtener minimizando

$$S(\boldsymbol{w}) = \sum_{r=1}^{T} \sum_{i=1}^{m_r} \ell(y_i^r, \langle \boldsymbol{w}, (B_r \otimes \phi(x_i^r)) \rangle) + \mu \boldsymbol{w}^\top \boldsymbol{w}, \tag{2}$$

donde $\boldsymbol{w} \in \mathbb{R}^p \otimes \mathcal{H}$ con $p \geq T$ y B_r son las columnas de $B \in \mathbb{R}^{p \times T}$, una matriz de rango máximo tal que $E^{-1} = B^\top B$.

El kernel reproductor correspondiente es:

$$\left\langle B_r \otimes \phi(x_i^r), B_s \otimes \phi(x_j^s) \right\rangle = \left(E^{-1}\right)_{rs} k(x_i^r, x_j^s)$$

Laplaciano de Grafo con Métodos de Kernel y Formulación Convexa

3 Laplaciano Adaptativo para Aprendizaje Multitarea

- Propuesta: combinar la formulación convexa con la regularización Laplaciana
- El problema de minimización es

$$\sum_{r=1}^{T}\sum_{i=1}^{m_r}\ell(y_i^r, \lambda_r \langle w, \phi(x_i^r)\rangle + (1-\lambda_r)\langle v_r, \phi(x_i^r)\rangle)$$
$$+ \mu \sum_r \sum_s (L)_{rs}\langle v_r, v_s\rangle + \sum_{r=1}^{T}\langle v_r, v_r\rangle + \langle w, w\rangle$$

- Usando esta formulación y el lema anterior proponemos:
  - L1-SVM MT convexa con regularización laplaciana
  - L2-SVM MT convexa con regularización laplaciana
  - LS-SVM MT convexa con regularización laplaciana

Formulación Convexa para L1-SVM MT con Laplaciano

Problema Primal - L1-SVM Convexa con Laplaciano

$$\min_{\boldsymbol{v},\boldsymbol{b},\boldsymbol{\xi},w} \quad C\sum_{r=1}^{T}\sum_{i=1}^{m_r}\xi_i^r + \frac{\nu}{2}\sum_{r=1}^{T}\sum_{s=1}^{T}(L)_{rs}\langle v_r, v_s\rangle + \frac{1}{2}\sum_r \|v_r\|^2 + \frac{1}{2}\|w\|^2$$

$$\text{s.t.} \quad y_i^r(\lambda_r(\langle w, \phi(x_i^r)\rangle) + (1-\lambda_r)(\langle v_r, \psi(x_i^r)\rangle) + b_r) \geq p_i^r - \xi_i^r,$$

$$\xi_i^r \geq 0,\ i = 1,\ldots,m_r,\ r = 1,\ldots,T.$$

- Los hiperparámetros λ_r regulan la influencia de cada parte:
  - $\lambda_1,\ldots,\lambda_T = 0$: modelos independientes (ITL)
  - $\lambda_1,\ldots,\lambda_T = 1$: modelo común (CTL)
- La matriz laplaciana L establece relaciones entre las partes específicas v_r

Formulación Convexa para L1-SVM MT con Laplaciano

Problema Dual - L1-SVM Convexa con Laplaciano

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \Theta(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^t \left(\Lambda Q \Lambda + (I_n - \Lambda)\, \widetilde{Q}\, (I_n - \Lambda) \right) \boldsymbol{\alpha} - \boldsymbol{p}\boldsymbol{\alpha} \\ \text{s.t.} \quad & 0 \leq \alpha_i^r \leq C,\ i = 1, \ldots, m_r, r = 1, \ldots, T, \\ & \sum_{i=1}^{n_r} \alpha_i^r y_i^r = 0,\ r = 1, \ldots, T. \end{aligned}$$

- Usamos la matriz $\Lambda = \mathrm{diag}(\overbrace{\lambda_1, \ldots, \lambda_1}^{m_1}, \ldots, \overbrace{\lambda_T, \ldots, \lambda_T}^{m_T})$
- La matriz Q es común entre todas las tareas usando el kernel k_ϕ correspondiente a ϕ
- La matriz $\tilde{Q}$ se define usando el kernel: $\widetilde{k}_\psi(x_i^r, x_j^s) = \left((\nu L + I_T)^{-1} \right)_{rs} k_\psi(x_i^r, x_j^s)$
- La función de kernel es: $\widehat{k}(x_i^r, x_j^s) = \lambda_r \lambda_s k_\phi(x_i^r, x_j^s) + (1 - \lambda_r)(1 - \lambda_s)\widetilde{k}_\psi(x_i^r, x_j^s)$

Formulación Convexa para L2-SVM MT con Laplaciano

Problema Primal - L2-SVM Convexa con Laplaciano

$$\min_{\substack{v_1,\dots,v_T;\\ b_1,\dots,b_T;\\ \boldsymbol{\xi},w;}} \quad C\sum_{r=1}^{T}\sum_{i=1}^{m_r}(\xi_i^r)^2+\frac{\nu}{2}\sum_{r=1}^{T}\sum_{s=1}^{T}(A)_{rs}\|v_r-v_s\|^2+\frac{1}{2}\sum_r\|v_r\|^2+\frac{1}{2}\|w\|^2$$

$$\text{s.t.} \quad y_i^r(\lambda_r(\langle w,\phi(x_i^r)\rangle)+(1-\lambda_r)(\langle v_r,\psi(x_i^r\rangle))+b_r)\geq p_i^r-\xi_i^r;$$

Problema Dual - L2-SVM Convexa con Laplaciano

$$\min_{\boldsymbol{\alpha}} \quad \Theta(\boldsymbol{\alpha})=\frac{1}{2}\boldsymbol{\alpha}^t\left\{\left(\Lambda Q\Lambda+(I_n-\Lambda)\,\widetilde{Q}\,(I_n-\Lambda)\right)+\frac{1}{C}I_n\right\}\boldsymbol{\alpha}-\boldsymbol{p}\boldsymbol{\alpha}$$

$$\text{s.t.} \quad 0\leq\alpha_i^r,\ i=1,\dots,m_r,\ r=1,\dots,T,$$

$$\sum_{i=1}^{n_r}\alpha_i^r y_i^r=0,\ r=1,\dots,T.$$

Formulación Convexa para LS-SVM MT con Laplaciano

Problema Primal - LS-SVM Convexa con Laplaciano

$$\min_{\substack{v_1,\dots,v_T;\\ b_1,\dots,b_T;\\ \boldsymbol{\xi},w;}} \quad C\sum_{r=1}^{T}\sum_{i=1}^{m_r}(\xi_i^r)^2 + \frac{\nu}{2}\sum_{r=1}^{T}\sum_{s=1}^{T}(A)_{rs}\|v_r - v_s\|^2 + \frac{1}{2}\sum_r \|v_r\|^2 + \frac{1}{2}\|w\|^2$$

$$\text{s.t.} \qquad y_i^r(\lambda_r(\langle w, \phi(x_i^r)\rangle) + (1-\lambda_r)(\langle v_r, \psi(x_i^r\rangle)) + b_r) = p_i^r - \xi_i^r;$$

Problema Dual - LS-SVM Convexa con Laplaciano

$$\left[\begin{array}{c|c} 0_{T\times T} & A^\intercal Y \\ \hline YA & \left(\Lambda Q\Lambda + (I_n - \Lambda)\,\widetilde{Q}\,(I_n - \Lambda)\right) + \frac{1}{C}I_n \end{array}\right] \begin{bmatrix} b_1 \\ \vdots \\ b_T \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} \mathbf{0}_T \\ \boldsymbol{p} \end{bmatrix}. \tag{3}$$

UAM
Universidad Autónoma de Madrid

Table of Contents

3 Laplaciano Adaptativo para Aprendizaje Multitarea

Algoritmo Adaptativo para Laplaciano de Grafo

- La selección de la matriz de adyacencia A (y la respectiva L) determina la relación que se fomenta entre las tareas
- Tiene que tener las siguientes restricciones:
  - A es simétrica
  - $(A)_{rs} \geq 0, r, s = 1, \ldots, T.$
  - $\sum_{s=1}(A)_{rs} = 1$
- La entropía de cada fila es: $\boldsymbol{a}^r$: $H(\boldsymbol{a}^r) = \sum_{s=1}^{T}(A)_{rs}\log((A)_{rs})$
- Definimos la entropía de A como: $H(A) = \sum_{r=1}^{T} H(\boldsymbol{a}^r)$
- Interpretación:
  - $H(A)$ es máxima si A es constante, $A = \frac{1}{T}\mathbf{1}_T\mathbf{1}_T^\intercal$,
  - $H(A)$ es mínima si A es la identidad, $A = I_T$

Algoritmo Adaptativo para Laplaciano de Grafo

Problema para Algoritmo Adaptativo

$$\min_{\substack{\boldsymbol{w},\boldsymbol{v},\boldsymbol{b};\\ A\in(\mathbb{R}_{\geq 0})^{T\times T},\\ A\mathbf{1}_T=\mathbf{1}_T}} C\sum_{r=1}^{T}\sum_{i=1}^{m_r}\ell(\lambda_r\langle w,\phi(x_i^r)\rangle+(1-\lambda_r)\langle v_r,\psi(x_i^r)\rangle+b_r,y_i^r)$$

$$+\frac{\nu}{2}\sum_{r=1}^{T}\sum_{s=1}^{T}(A)_{rs}\|v_r-v_s\|^2+\frac{1}{2}\sum_{r=1}^{T}\|v_r\|^2+\frac{1}{2}\|w\|^2$$

$$-\mu\sum_{r=1}^{T}H(\boldsymbol{a}^r),$$

Algoritmo Adaptativo para Laplaciano de Grafo

- Para minimizar este problema alternamos los siguientes pasos:
  - Fijamos A y minimizamos en $w, \boldsymbol{v}, \boldsymbol{b}$: resolvemos el problema dual (y obtenemos $\boldsymbol{\alpha}^*$) correspondiente a

$$\min_{w,\boldsymbol{v},\boldsymbol{b}} \quad C \sum_{r=1}^{T} \sum_{i=1}^{m_r} \ell(\lambda_r \langle w, \phi(x_i^r) \rangle + (1-\lambda_r) \langle v_r, \psi(x_i^r) \rangle + b_r, y_i^r) + \frac{\nu}{2} \sum_{r=1}^{T} \sum_{s=1}^{T} (A)_{rs} \|v_r - v_s\|^2 + \frac{1}{2} \sum_{r=1}^{T} \|v_r\|^2 + \frac{1}{2} \|w\|^2$$

  - Fjamos $w, \boldsymbol{v}, \boldsymbol{b}$ y minimizamos en A:

$$\min_{\substack{A \in (\mathbb{R}_{\geq 0})^{T \times T}, \\ A\mathbf{1}_T = \mathbf{1}_T}} J(A) = \frac{\nu}{2} \sum_{r=1}^{T} \sum_{s=1}^{T} (A)_{rs} \|v_r - v_s\|^2 - \mu \sum_{r=1}^{T} H(\boldsymbol{a}^r).$$

Algoritmo Adaptativo para Laplaciano de Grafo

3 Laplaciano Adaptativo para Aprendizaje Multitarea

- Si sabemos las distancias $\|v_r - v_s\|^2$, la solución es

$$(A)_{rs} = \frac{\exp -\frac{\nu}{\mu} \|v_r - v_s\|^2}{\sum_t \exp -\frac{\nu}{\mu} \|v_r - v_t\|^2}.$$

- ¿Cómo calculamos estas distancias?
  - Con la matriz $\widetilde{Q^{rs}}$ correspondiente a la función

$$\widetilde{k^{rs}}(x_i^t, x_j^\tau) = (I_T + \nu L)^{-1}_{rt} (I_T + \nu L)^{-1}_{s\tau} k_\psi(x_i^t, x_j^\tau).$$

  - Los productos interiores son

$$\langle v_r, v_s \rangle = \boldsymbol{\alpha}^\intercal (I_n - \Lambda) \widetilde{Q^{rs}} (I_n - \Lambda) \boldsymbol{\alpha},$$

  - Las distancias son entonces

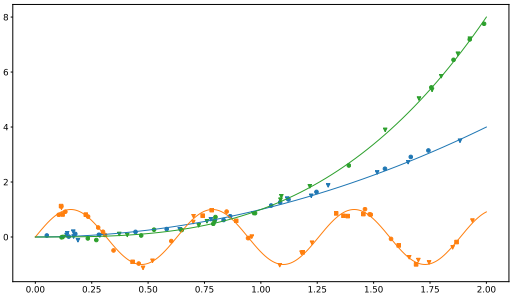
$$\|v_r - v_s\|^2 = \boldsymbol{\alpha}^\intercal (I_n - \Lambda) (\widetilde{Q^{rr}} + \widetilde{Q^{ss}} - 2\widetilde{Q^{rs}}) (I_n - \Lambda) \boldsymbol{\alpha}.$$

Algoritmo Adaptativo para Laplaciano de Grafo

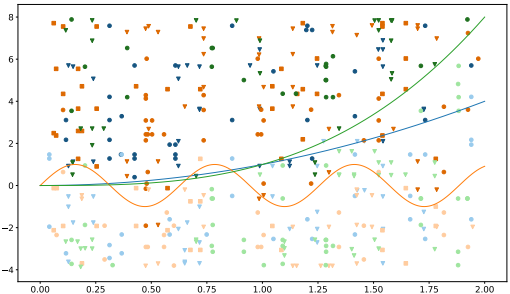
$A = A_0$ // Constant matrix
while *True* **do**
  $L_{\text{inv}} \leftarrow$ getInvLaplacian(A) // Step 0
  $\alpha_{\text{opt}} \leftarrow$ solveDualProblem((X, y), L_{inv}, params) // Step 1
  $o \leftarrow$ computeObjectiveValue((X, y), L_{inv}, α_{opt}) // Objective function value
  if $o^{old} - o \leq \delta_{tol}$ **then**
    break // Exit condition
  end
  $D \leftarrow$ computeDistances((X, y), L_{inv}, α_{opt}) // Step 2
  $A \leftarrow$ updateAdjMatrix(D, params) // Step 3
end
return α_{opt}, A

Experimentos: Problemas Sintéticos

3 Laplaciano Adaptativo para Aprendizaje Multitarea



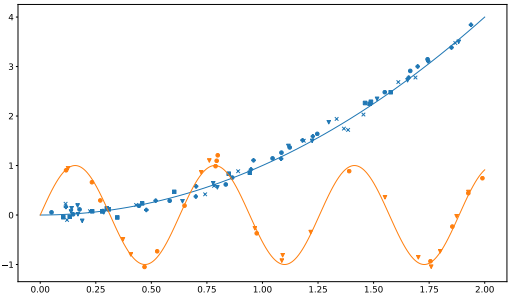
(a) regClusterso.



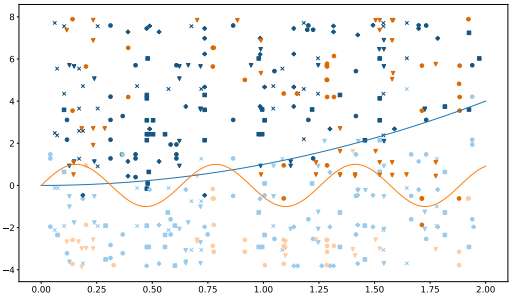
(b) clasClusterso.

Experimentos Sintéticos: Problemas

3 Laplaciano Adaptativo para Aprendizaje Multitarea



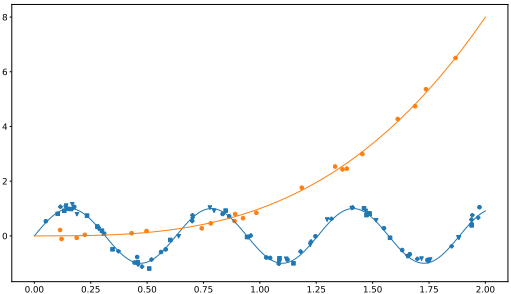
(a) regClusters1.



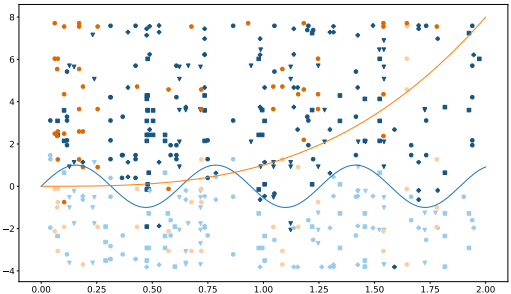
(b) clasClusters1.

Experimentos Sintéticos: Problemas Sintéticos

3 Laplaciano Adaptativo para Aprendizaje Multitarea



(a) regClusters2.



(b) clasClusters2.

Experimentos Sintéticos: Resultados

3 Laplaciano Adaptativo para Aprendizaje Multitarea

	regClusters0	regClusters1	regClusters2	clasClusters0	clasClusters1	clasClusters2
		MAE			F1	
CTL-L1	0.989	0.512	0.541	0.901	0.912	0.904
ITL-L1	0.221	0.212	0.159	0.922	0.923	0.910
MTL-L1	0.213	0.176	0.135	**0.924**	0.925	0.914
cvxGLMTL-L1	0.212	0.173	0.138	0.920	0.926	0.912
AdapGLMTL-L1	**0.152**	**0.116**	**0.107**	**0.924**	**0.929**	**0.916**
CTL-L2	0.990	0.642	0.768	0.904	0.912	0.906
ITL-L2	0.213	0.201	0.154	**0.928**	0.928	0.910
MTL-L2	0.209	0.168	0.131	0.925	0.927	0.913
cvxGLMTL-L2	0.204	0.169	0.131	0.921	0.923	**0.915**
AdapGLMTL-L2	**0.141**	**0.115**	**0.103**	0.924	**0.929**	**0.915**
CTL-LS	0.989	0.642	0.766	0.895	0.908	0.894
ITL-LS	0.212	0.209	0.149	0.914	0.915	0.904
MTL-LS	0.206	0.167	0.131	0.917	0.917	**0.905**
cvxGLMTL-LS	0.207	0.169	0.132	0.919	**0.921**	0.897
AdapGLMTL-LS	**0.136**	**0.115**	**0.106**	**0.920**	**0.921**	0.901

Experimentos Sintéticos: Resultados

3 Laplaciano Adaptativo para Aprendizaje Multitarea

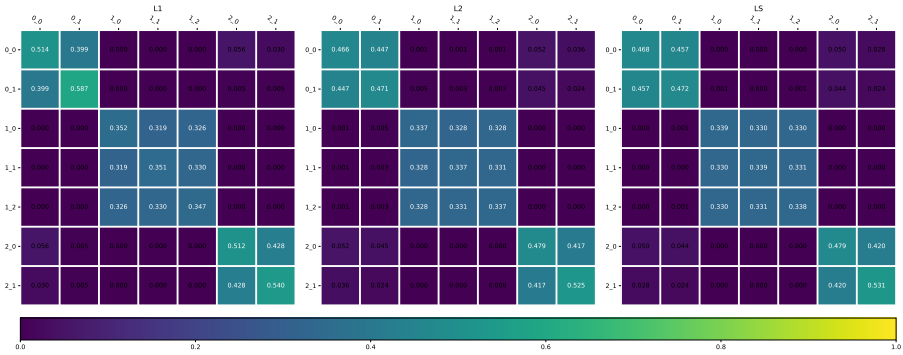


Figure: Prueba

Experimentos Sintéticos: Resultados

3 Laplaciano Adaptativo para Aprendizaje Multitarea

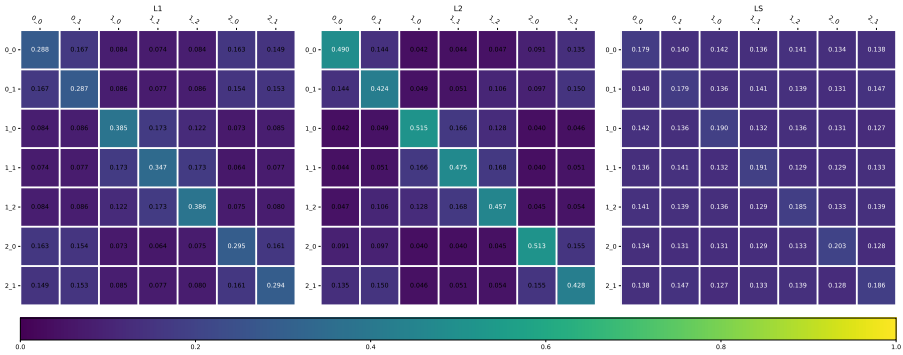


Figure: Prueba

UAM
Universidad Autónoma de Madrid

Table of Contents

4 Summary

Good Luck!

4 Summary

- Enough for an introduction! You should know enough by now

UAM
Universidad Autónoma de Madrid

Advanced Kernel Methods for Multi-Task Learning

Thank you for listening!

Any questions?