UAM
Universidad Autónoma de Madrid

Advanced Kernel Methods for Multi-Task Learning

Tesis dirigida por José Dorronsoro y Carlos Alaíz

Carlos Ruiz Pastor

14 de enero de 2023

UAM
Universidad Autónoma de Madrid

UAM
Universidad Autónoma de Madrid

Índice

0.

UAM
Universidad Autónoma de Madrid

Índice

1. Introducción

Introducción al Aprendizaje Automático

- El Aprendizaje Automático (AA) 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 $y \in \mathcal{Y}$

Riesgo Esperado

- Dado un espacio de hipótesis $\mathcal{H}$
- Riesgo Esperado:

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

- El objetivo es minimizar el Riesgo Esperado:

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

sin embargo, la distribución $P(x, y)$ es desconocida

Riesgo Empírico

- En su lugar tenemos n muestras de $P(x, y)$:

$$D = \{(x_i, y_i) \sim P(x, y),\ i = 1, \ldots, n\},$$

- Riesgo Empírico

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

- Una estrategia común es minimizar el Riesgo Empírico Regularizado:

$$\arg\min_{h \in \mathcal{H}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, h(x_i)) + \Omega(h) \right\}$$

Aprendizaje Multitarea

1. Introducción

- En aprendizaje multitarea con T tareas tenemos:
  - un espacio de entrada $\mathcal{X}$,
  - un espacio de salida $\mathcal{Y}$,
  - y T distribuciónes $\boldsymbol{P} = (P_1, \ldots, P_T)$ (desconocidas) sobre $\mathcal{X} \times \mathcal{Y}$
- Tenemos que estimar T hipótesis $\boldsymbol{h} = (h_1, \ldots, h_T) \in \mathcal{H}^T$
- El Riesgo Esperado multitarea es:

$$R_{\boldsymbol{P}}(\boldsymbol{h}) = \sum_{r=1}^{T} \int_{\mathcal{X}\times\mathcal{Y}} \ell(y, h_r(x)) dP_r(x, y)$$

Aprendizaje Multitarea

1. Introducción

- Ahora tenemos una muestra multitarea:

$$\boldsymbol{D} = \bigcup_{r=1}^{T} \{(x_i^r, y_i^r) \sim P_r(x, y),\ i = 1, \ldots, m_r\},$$

- El Riesgo Empírico multitarea es:

$$\hat{R}_{\boldsymbol{D}}(\boldsymbol{h}) = \sum_{r=1}^{T} \frac{1}{m_r} \sum_{i=1}^{m_r} \ell(y_i^r, h_r(x_i^r))$$

- Minimizamos el Riesgo Empírico Regularizado multitarea:

$$\arg\min_{\boldsymbol{h} \in \mathcal{H}^T} \left\{ \sum_{r=1}^{T} \frac{1}{m_r} \sum_{i=1}^{m_r} \ell(y_i^r, h_r(x_i^r)) + \Omega(\boldsymbol{h}) \right\}$$

Aprendizaje Multitarea

- Hay tres opciones para minimizar el Riesgo Regularizado multitarea
  - Aprendizaje común (CTL): se usa un modelo común para todas las tareas

$$\arg\min_{h\in\mathcal{H}} \left\{ \sum_{r=1}^{T} \frac{1}{m_r} \sum_{i=1}^{m_r} \ell(y_i^r, h(x_i^r)) + \Omega(h) \right\}$$

  - Aprendizaje independiente (ITL): se usan modelos independientes en cada tarea

$$\arg\min_{\boldsymbol{h}} \left\{ \sum_{r=1}^{T} \sum_{i=1}^{m_r} \ell(h_r(x_i^r), y_i^r) + \sum_{r=1}^{T} \Omega_r(h_r) \right\}$$

  - Aprendizaje multitarea (MTL): se usan modelos específicos que comparten información
    - Basados en características
    - Basados en regularización
    - Basados en combinación

Aprendizaje Multitarea

- Basados en características (más comunes en redes neuronales)

$$\sum_{r=1}^{T} \sum_{i=1}^{m_r} \ell(g_r \circ f(x_i^r), y_i^r) + \Omega(f) + \Omega(g_1, \ldots, g_T)$$

- Basados en regularización (más comunes con modelos lineales)

$$\sum_{r=1}^{T} \sum_{i=1}^{m_r} \ell(h_r(x_i^r), y_i^r) + \Omega(h_1, \ldots, h_T),\ \Omega(h_1, \ldots, h_T) \neq \sum_{r=1}^{T} \Omega_r(h_r)$$

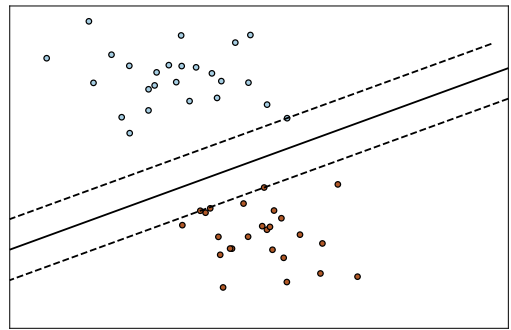
- Basados en combinación (más comunes con métodos de kernel)

$$\sum_{r=1}^{T} \sum_{i=1}^{m_r} \ell(g(x_i^r) + g_r(x_i^r), y_i^r) + \Omega(g) + \Omega(g_1, \ldots, g_T)$$

Máquinas de Vectores Soporte

1. Introducción

- Las Máquinas de Vectores Soporte (SVM) son modelos populares del AA
- Se definen usando problemas convexos
- Se puede aplicar el “truco del kernel”
- En clasificación maximizan el margen
- Consideramos tres variantes:
  - L1-SVM
  - L2-SVM
  - LS-SVM



L1-SVM

Problema Primal - L1-SVM

$$\begin{aligned} \min_{w,b,\xi} \quad & C\sum_{i=1}^{n}\xi_i + \frac{1}{2}\|w\|^2 \\ \text{s.t.} \quad & y_i(\langle w, x_i\rangle + b) \geq p_i - \xi_i, \\ & \xi_i \geq 0 \end{aligned}$$

- Se puede ver que esta formulación es equivalente a
  - Máquina de Vectores Soporte para Clasificación (SVC): $p_i = 1$ para $i = 1, \ldots n$
  - Máquina de Vectores Soporte para Regresión (SVR): se duplican los patrones
    - $y_i = 1,\ p_i = t_i - \epsilon$, en la primera mitad
    - $y_i = -1,\ p_i = -t_i - \epsilon$, en la segunda mitad

Problema Dual - L1-SVM

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\boldsymbol{\alpha}^\top Q\boldsymbol{\alpha} - \boldsymbol{\alpha}^\top \boldsymbol{p} \\ \text{s.t.} \quad & \sum_{i=1}^{n} y_i \alpha_i = 0, \\ & 0 \leq \alpha_i \leq C \end{aligned}$$

- Aquí Q es la matriz de kernel
  - SVM lineal: $Q_{ij} = y_i y_j k(x_i, x_j) = y_i y_j \langle x_i, x_j \rangle$
  - SVM no lineal: $Q_{ij} = y_i y_j k(x_i, x_j) = y_i y_j \langle \phi(x_i), \phi(x_j) \rangle$

L2-SVM

1. Introducción

Problema Primal - L2-SVM

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{C}{2}\sum_{i=1}^{m}(\xi_i)^2 + \frac{1}{2}\|w\|^2 \\ \text{s.t.} \quad & y_i(\langle w, \phi(x_i)\rangle + b) \geq 1 - \xi_i \end{aligned}$$

Problema Dual - L2-SVM

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\alpha^\intercal \left(Q + \frac{1}{C}I_n\right)\alpha - \alpha^\intercal p \\ \text{s.t.} \quad & \sum_{i=1}^{n} y_i\alpha_i = 0,\ \alpha_i \geq 0 \end{aligned}$$

Problema Primal - LS-SVM

$$\min_{w,b,\boldsymbol{\xi}} \quad \frac{C}{2}\sum_{i=1}^{n}(\xi_i)^2 + \frac{1}{2}\|w\|^2$$

$$\text{s.t.} \quad y_i(\langle w, x_i\rangle + b) = p_i - \xi_i$$

Problema Dual - LS-SVM

$$\left[\begin{array}{c|c} 0 & \boldsymbol{y}^\intercal \\ \hline \boldsymbol{y} & Q + \frac{1}{C}I_n \end{array}\right] \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{p} \end{bmatrix}$$

UAM
Universidad Autónoma de Madrid

Índice

2. Formulación Convexa para Aprendizaje Multitarea: Métodos de Kernel

Formulación Aditiva con Métodos de Kernel

- Una manera de implementar el MTL es combinar una parte común y otras específicas
- Fue propuesta incialmente para SVM lineales en[1] con los modelos

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

y fue extendida al caso no lineal en[2] con los modelos

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

donde w es el parámetro de la parte común y v_r los de las partes específicas

[1] Theodoros Evgeniou y Massimiliano Pontil. "Regularized multi–task learning". En: *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2004, págs. 109-117.

[2] Feng Cai y Vladimir Cherkassky. "SVM+ regression and multi-task learning". En: *International Joint Conference on Neural Networks*. 2009, págs. 418-424.

Formulación Aditiva para L1-SVM MT

Problema Primal - L1-SVM MT Aditiva

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

- 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)

Formulacion Convexa con Métodos de Kernel

- Proponemos en[3] la siguiente formulación convexa para el aprendizaje multitarea:

$$h_r(\cdot) = \lambda_r \{\langle w, \phi(\cdot)\rangle + b\} + (1-\lambda_r)\{\langle v_r, \phi_r(\cdot)\rangle + d_r\}, \ \lambda_r \in [0,1]$$

- Los hiperparámetros λ_r regulan la influencia de cada parte
- Extendemos esta formulación en[4] y desarrollamos tres variantes de SVM:
  - L1-SVM
  - L2-SVM
  - LS-SVM

[3] Carlos Ruiz, Carlos M. Alaíz y José R. Dorronsoro. "A Convex Formulation of SVM-Based Multi-task Learning". En: *HAIS Proceedings*. Vol. 11734. Springer, 2019, págs. 404-415.

[4] Carlos Ruiz, Carlos M. Alaíz y José R. Dorronsoro. "Convex formulation for multi-task L1-, L2-, and LS-SVMs". En: *Neurocomputing* 456 (2021), págs. 599-608.

Formulación Convexa para L1-SVM MT

Problema Primal - L1-SVM MT Convexa

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

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

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

- 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 el grado de interdependencia de los modelos

Formulación Convexa para L1-SVM MT

Problema Dual - L1-SVM MT Convexa

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

- Usamos la matriz $\Lambda = \text{diag}(\overbrace{\lambda_1, \ldots, \lambda_1}^{m_1}, \ldots, \overbrace{\lambda_T, \ldots, \lambda_T}^{m_T})$
- 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:

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

Proposicion (Equivalencia entre formulaciones para L1-SVM MT)

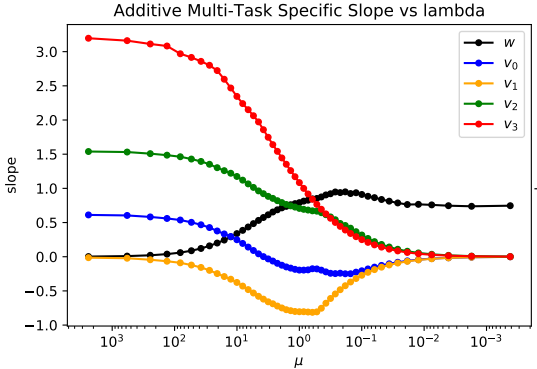
Para valores $\lambda \in (0, 1)$, la formulación aditiva con hiperparámetros C_{add}, μ y la formulación convexa con C_{conv} y un λ común, $\lambda_1, \ldots, \lambda_T = \lambda$, son equivalentes cuando

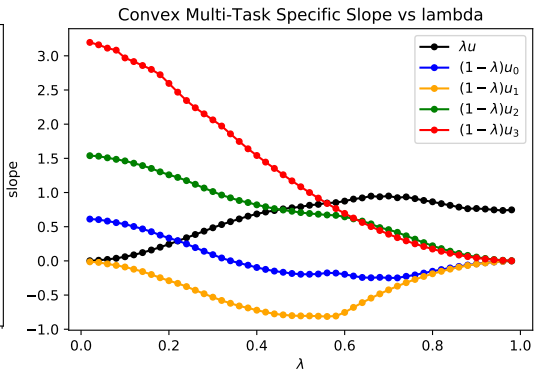
$$C_{add} = (1-\lambda)^2 C_{conv}, \ \mu = (1-\lambda)^2/\lambda^2.$$

- 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. Formulación Convexa para Aprendizaje Multitarea: Métodos de Kernel





Formulación Convexa para L2-SVM MT

Problema Primal - L2-SVM MT Convexa

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

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

Problema Dual - L2-SVM MT Convexa

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\boldsymbol{\alpha}^\intercal\left(\left\{\Lambda Q\Lambda + (I_n-\Lambda)\,K\,(I_n-\Lambda)\right\} + \frac{1}{C}I\right)\boldsymbol{\alpha} - \boldsymbol{p}\boldsymbol{\alpha}$$

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

$$\sum_{i=1}^{m_r}\alpha_i^r y_i^r = 0,\ r=1,\ldots,T$$

Formulación Convexa para LS-SVM MT

Problema Primal - LS-SVM MT Convexa

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

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

Problema Dual - LS-SVM MT Convexa

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

Combinación Convexa de Modelos Preentrenados

2. Formulación Convexa para Aprendizaje Multitarea: Métodos de Kernel

- Consideramos en[5] la combinación convexa de
  - modelo común $g(\cdot)$ ya entrenado
  - modelos específicos $g_r(\cdot)$ ya entrenados

  frente al aprendizaje MT con formulación convexa
- Minimizamos el riesgo eligiendo los hiperparámetros $\lambda_1, \ldots, \lambda_T$ óptimos

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

- Consideramos las pérdidas:
  - Cuadrática y Absoluta (regresión)
  - Hinge y Hinge Cuadrática (clasificación)

[5] Carlos Ruiz, Carlos M. Alaíz y José R. Dorronsoro. "Convex formulation for multi-task L1-, L2-, and LS-SVMs". En: *Neurocomputing* 456 (2021), págs. 599-608.

Formulación Unificada

- En todos los casos lo podemos expresar como

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

- 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

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

Combinación Convexa con Error Cuadrático

- 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 $\lambda^* = \max(\min(\lambda', 1), 0)$

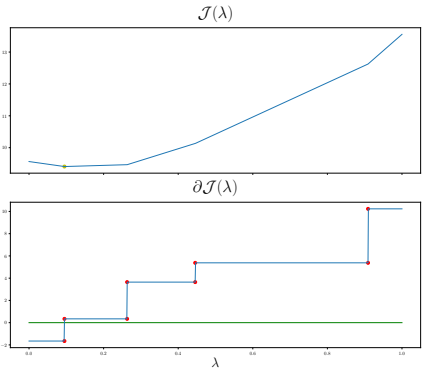
Combinación Convexa con Error Absoluto

- Hay que resolver

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

- El subgradiente de cada sumando es

$$\partial\,|\lambda c_i + d_i| = \begin{cases} -|c_i|\,, & \lambda c_i + d_i < 0, \\ [-|c_i|\,, |c_i|], & \lambda c_i + d_i = 0, \\ |c_i|\,, & \lambda c_i + d_i > 0 \end{cases}$$



Combinación Convexa con Error Absoluto

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

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

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

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

Combinación Convexa: Tabla

	$\boldsymbol{\lambda^* \in (0,1)}$
Cuadrática	$0 < -\frac{\sum_{i=1}^m d_i c_i}{\sum_{i=1}^m (c_i)^2} < 1$
Absoluta	$-\sum_{i:\ \lambda_{(k)}>\lambda_{(i)}} \lvert c_{(i)}\rvert + \sum_{i:\ \lambda_{(k)}<\lambda_{(i)}} \lvert c_{(i)}\rvert \in \left[-\lvert c_{(k)}\rvert, \lvert c_{(k)}\rvert\right]$
Hinge	$-\sum_{i:\ \lambda_{(k)}>\lambda_{(i)}} \text{máx}\left(0, c_{(i)}\right) - \sum_{i:\ \lambda_{(k)}<\lambda_{(i)}} \text{mín}\left(0, c_{(i)}\right) \in \left[\text{mín}\left(0, c_{(k)}\right), \text{máx}\left(0, c_{(k)}\right)\right]$
Hinge Cuad.	$-\frac{\sum_{i:\ \lambda_{(k+1)}\geq\lambda_{(i)}} \text{máx}\left(0,c_{(i)}\right) d_{(i)} + \sum_{i:\ \lambda_{(k)}\leq\lambda_{(i)}} \text{mín}\left(0,c_{(i)}\right) d_{(i)}}{\sum_{i:\ \lambda_{(k+1)}\geq\lambda_{(i)}} \text{máx}\left(0,c_{(i)}\right)^2 + \sum_{i:\ \lambda_{(k)}\leq\lambda_{(i)}} \text{mín}\left(0,c_{(i)}\right)^2} \in [\lambda_{(k)}, \lambda_{(k+1)}]$
	$\boldsymbol{\lambda^* = 0}$
Cuadrática	$-\frac{\sum_{i=1}^m d_i c_i}{\sum_{i=1}^m (c_i)^2} \leq 0$
Absoluta	$-\sum_{i:\ 0>\lambda_{(i)}} \lvert c_{(i)}\rvert + \sum_{i:\ 0<\lambda_{(i)}} \lvert c_{(i)}\rvert \leq 0$
Hinge	$-\sum_{i:\ 0>\lambda_{(i)}} \text{máx}\left(0, c_{(i)}\right) - \sum_{0<\lambda_{(i)}} \text{mín}\left(0, c_{(i)}\right) \leq 0$
Hinge Cuad.	$-\sum_{i:\ 0>c_{(i)}, 0<\lambda_{(i)}} 2c_i d_i - \sum_{i:\ 0<c_{(i)}, 0>\lambda_{(i)}} 2c_{(i)} d_{(i)} \leq 0$
	$\boldsymbol{\lambda^* = 1}$ **en otro caso**

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

Experimentos: Problemas

2. Formulación Convexa para Aprendizaje Multitarea: Métodos de Kernel

Problema	Tamaño	Dimensión	Nº tareas	Tam. tarea medio	Tam. tarea mín.	Tam. tarea máx.
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 y 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

- Por limitaciones computacionales nos restringimos a la búsqueda de tres hiperparámetros para la búsqueda en rejilla
- Las anchuras de kernel para los modelos MTL se extraen de los modelos CTL e ITL

	Rejilla	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. Formulación Convexa para Aprendizaje Multitarea: Métodos de Kernel

	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)

2. Formulación Convexa para Aprendizaje Multitarea: Métodos de Kernel

	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)

2. Formulación Convexa para Aprendizaje Multitarea: Métodos de Kernel

	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

UAM
Universidad Autónoma de Madrid

Experimentos: Resultados de Clasificación (Accuracy)

2. Formulación Convexa para Aprendizaje Multitarea: Métodos de Kernel

	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

Índice

3. Formulación Convexa para Aprendizaje Multitarea: Redes Neuronales

Redes Neuronales MT

- Los métodos de kernel ofrecen propiedades deseables como
  - convexidad
  - dualidad
  - truco del kernel

  pero tienen una limitación computacional
- Las redes neuronales son mejores alternativas para grandes volúmenes de datos
- Existen arquitecturas neuronales específicas para algunos tipos de datos:
  - imágenes
  - texto
  - sonido
- Proponemos una arquitectura MT para redes basada en la combinación convexa

Redes Neuronales MT: *Hard Sharing*

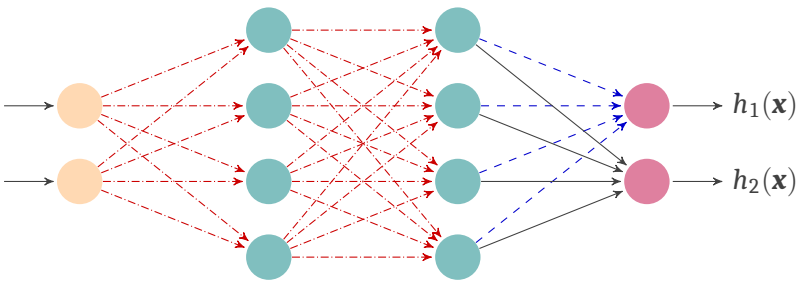
- La manera más común de adaptar las redes neuronales es el *hard sharing*[6]
  - Capas ocultas compartidas por todas las tareas
  - Capas de salida específicas para cada tarea
- El modelo se puede expresar como:

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

  - w_r, d_r son los parámetros de las capas de salida específicas
  - Θ son los parámetros de las capas ocultas compartidas
- Se comparte la misma representación en todas las tareas

[6] Rich Caruana. "Multitask Learning". En: *Mach. Learn.* 28.1 (1997), págs. 41-75.

Ejemplo de *Hard Sharing* para dos tareas



Formulación Convexa para Redes Neuronales MT

- Proponemos en[7] la formulación convexa para redes neuronales MT, combinando:
  - Una parte común $g(\cdot; w, b, \Theta)$
  - Una parte específica $g_r(\cdot; w_r, d_r, \Theta_r)$
- Los modelos son:

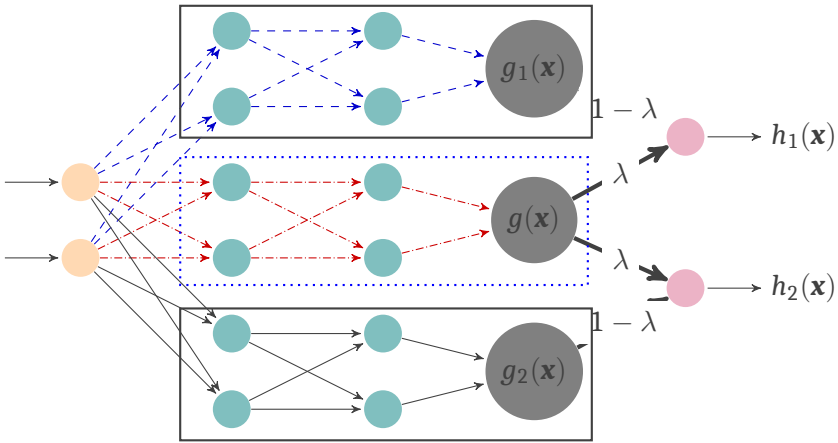
$$\begin{aligned} h_r(\cdot) &= \lambda_r g(\cdot; w, b, \Theta) + (1-\lambda_r) g_r(\cdot; w_r, d_r, \Theta_r) \\ &= \lambda_r \{\langle w, f(\cdot; \Theta)\rangle + b\} + (1-\lambda_r)\{\langle w_r, f_r(\cdot; \Theta_r)\rangle + d_r\} \end{aligned}$$

  - 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)
- No se comparte la representación, se combinan una parte común y partes específicas

[7] Carlos Ruiz, Carlos M. Alaíz y José R. Dorronsoro. "Convex Multi-Task Learning with Neural Networks". En: *HAIS Proceedings*. Vol. 13469. Springer, 2022, págs. 223-235.

Ejemplo de formulación convexa para dos tareas

3. Formulación Convexa para Aprendizaje Multitarea: Redes Neuronales



Formulación Convexa para Redes Neuronales MT

- El riesgo a minimizar en este caso es

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

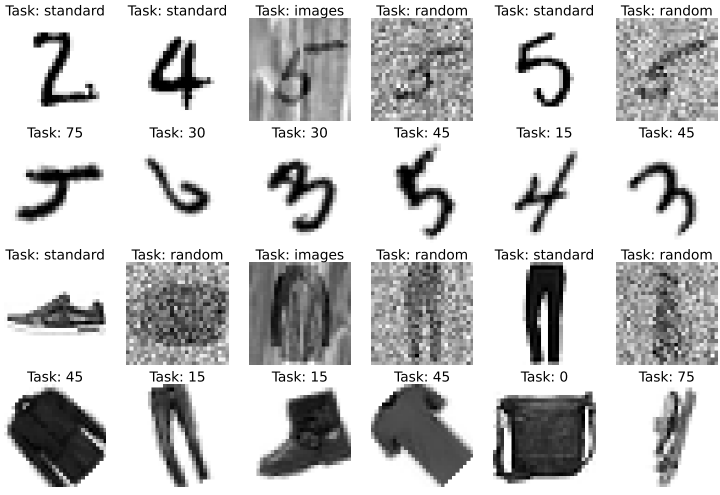
- Se puede aplicar el descenso por gradiente con

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

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

Experimentos: Conjuntos de Datos

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


Task: standard
Task: standard
Task: images
Task: random
Task: standard
Task: random
Task: 75
Task: 30
Task: 30
Task: 45
Task: 15
Task: 45
Task: standard
Task: random
Task: images
Task: random
Task: standard
Task: random
Task: 45
Task: 15
Task: 15
Task: 45
Task: 0
Task: 75

Experimentos: Modelos

- 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*. CV para hiperpar.
  - α (weight decay)
  - λ (en el modelo convexo)

Experimentos: Resultados

3. Formulación Convexa para Aprendizaje Multitarea: Redes Neuronales

	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

Índice

4. Laplaciano Adaptativo para Aprendizaje Multitarea

Aprendizaje Multitarea con Regularización Laplaciana

4. Laplaciano Adaptativo para Aprendizaje Multitarea

- La formulación convexa asume que hay una parte común a todas las tareas
- Sin embargo, pueden existir distintos grados de relación entre las tareas

- Por otra parte, la definición de tareas puede ser arbitraria
- Es posible que definamos tareas distintas que realmente son la misma

- Una alternativa para el aprendizaje MT es usar una regularización laplaciana de grafo

Aprendizaje Multitarea con Regularización Laplaciana

4. Laplaciano Adaptativo para Aprendizaje Multitarea

- 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

4. 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

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

- ¿Cómo definimos la regularización laplaciana en un espacio de kernel?

Laplaciano de Grafo con Métodos de Kernel

4. Laplaciano Adaptativo para Aprendizaje Multitarea

- Inspirados por el trabajo en[8] proponemos lo siguiente:
  - Consideramos el problema de minimización (con E una matriz def. pos.)

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

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

$$R(\boldsymbol{u}) = \sum_{r=1}^{T} \sum_{i=1}^{m_r} \ell(y_i^r, \langle \boldsymbol{u}, e_r \otimes \phi(x_i^r) \rangle) + \mu \left(\boldsymbol{u}^\intercal (E \otimes I) \boldsymbol{u} \right) \tag{2}$$

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

[8] Theodoros Evgeniou, Charles A. Micchelli y Massimiliano Pontil. "Learning Multiple Tasks with Kernel Methods". En: *J. Mach. Learn. Res.* 6 (2005), págs. 615-637.

Lema

Las soluciones $u_1^*, \ldots, u_T^*$ de (1), o equivalentemente la solución $\mathbf{u}^*$ de (2), se pueden obtener minimizando

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

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

4. Laplaciano Adaptativo para Aprendizaje Multitarea

- En[9] proponemos combinar la formulación convexa con la regularización Laplaciana

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

- 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

[9] Carlos Ruiz, Carlos M. Alaíz y José R. Dorronsoro. "Convex Graph Laplacian Multi-Task Learning SVM". En: *ICANN*. Vol. 12397. Springer, 2020, págs. 142-154.

Formulación Convexa para L1-SVM MT con Laplaciano

4. Laplaciano Adaptativo para Aprendizaje Multitarea

Problema Primal - L1-SVM Convexa con Laplaciano

$$\begin{aligned} \min_{\boldsymbol{v},\boldsymbol{b},\boldsymbol{\xi},w} \quad & C\sum_{r=1}^{T}\sum_{i=1}^{m_r}\xi_i^r + \frac{\nu}{2}\sum_{r=1}^{T}\sum_{s=1}^{T}(L)_{rs}\,\langle v_r, v_s\rangle + \frac{1}{2}\sum_{r}\|v_r\|^2 + \frac{1}{2}\|w\|^2 \\ \text{s.t.} \quad & y_i^r(\lambda_r(\langle w, \phi(x_i^r)\rangle) + (1-\lambda_r)(\langle v_r, \psi(x_i^r)\rangle) + b_r) \geq p_i^r - \xi_i^r, \\ & \xi_i^r \geq 0,\ i = 1,\ldots,m_r,\ r = 1,\ldots,T \end{aligned}$$

- 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

4. Laplaciano Adaptativo para Aprendizaje Multitarea

Problema Primal - L2-SVM Convexa con Laplaciano

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

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

Problema Dual - L2-SVM Convexa con Laplaciano

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

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

Formulación Convexa para LS-SVM MT con Laplaciano

4. Laplaciano Adaptativo para Aprendizaje Multitarea

Problema Primal - LS-SVM Convexa con Laplaciano

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

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

Problema Dual - LS-SVM Convexa con Laplaciano

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

Algoritmo Adaptativo para Laplaciano de Grafo

- La selección de la matriz de adyacencia A (y la respectiva L) es determinante
- En[10] proponemos un método para la selección automática de A
- 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: $\boldsymbol{a}^r$: $H(\boldsymbol{a}^r) = \sum_{s=1}^{T}(A)_{rs}\log((A)_{rs})$
- 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, $A = I_T$

[10] Carlos Ruiz, Carlos M. Alaíz y José R. Dorronsoro. "Adaptive Graph Laplacian for Convex Multi-Task Learning SVM". En: *HAIS Proceedings*. Vol. 12886. Springer, 2021, págs. 219-230.

Algoritmo Adaptativo para Laplaciano de Grafo

Problema para Algoritmo Adaptativo

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

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

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

Algoritmo Adaptativo para Laplaciano de Grafo

- Para minimizar este problema alternamos los siguientes pasos:
  - Fijamos A y minimizamos en $w, \boldsymbol{v}, \boldsymbol{b}$:

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

  - 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 1: Algoritmo para laplaciano adaptativo.

Input: $(X, y) = \{(x_i^r, y_i^r), i = 1, \ldots, m_r; r = 1, \ldots, T\}$ // Data

$A = A_0$ // 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

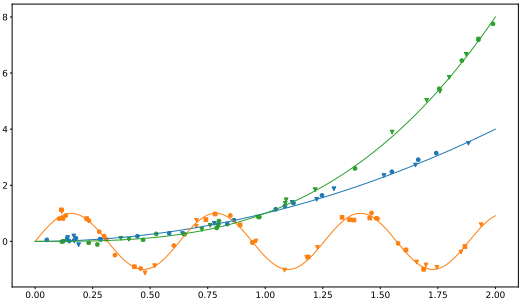
Experimentos Sintéticos: Problemas

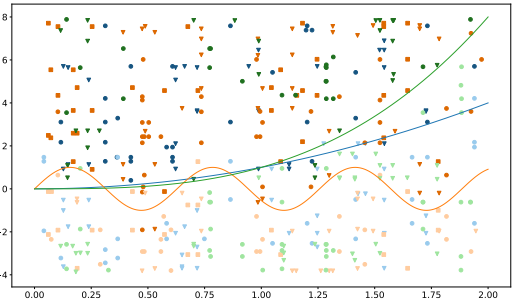
- Definimos problemas donde las tareas están agrupadas en clusters
  - Definimos τ tareas subyacentes (clusters) usando las funciones $f_r, r = 1, \ldots, \tau$
  - La función de regresión la usamos como frontera de clasificación
  - En cada una definimos T_r tareas virtuales, $r = 1, \ldots, \tau$ (las que ven los modelos)

	τ	f_r	T_r	Total tareas virtuales
regClusters0, clasClusters0	2	$f_1(x) = x^2$	2	7
		$f_2(x) = \sin(10x)$	3	
		$f_3(x) = x^3$	2	
regClusters1, clasClusters1	2	$f_1(x) = x^2$	5	7
		$f_2(x) = \sin(10x)$	2	
regClusters2, clasClusters2	2	$f_1(x) = \sin(10x)$	4	5
		$f_2(x) = x^3$	1	

Experimentos Sintéticos: clusterso

4. Laplaciano Adaptativo para Aprendizaje Multitarea





- Gráficas de regClusterso y clasClusterso

UAM
Universidad Autónoma de Madrid

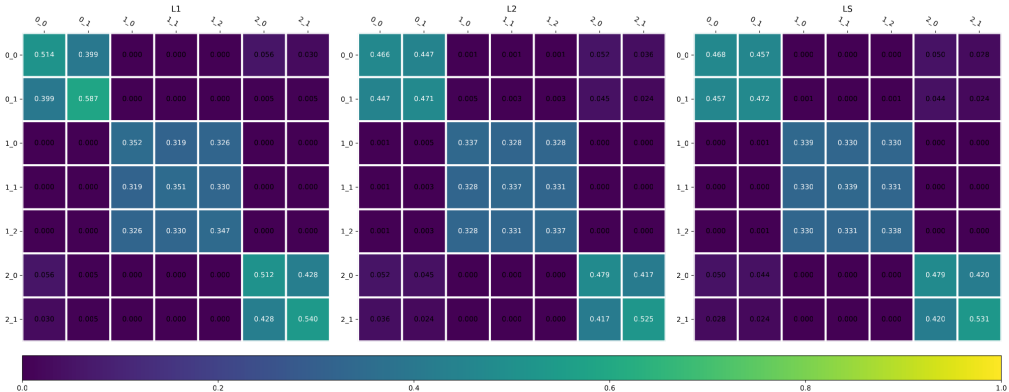
Experimentos Sintéticos: Resultados

4. 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: Matrices de Adyacencia

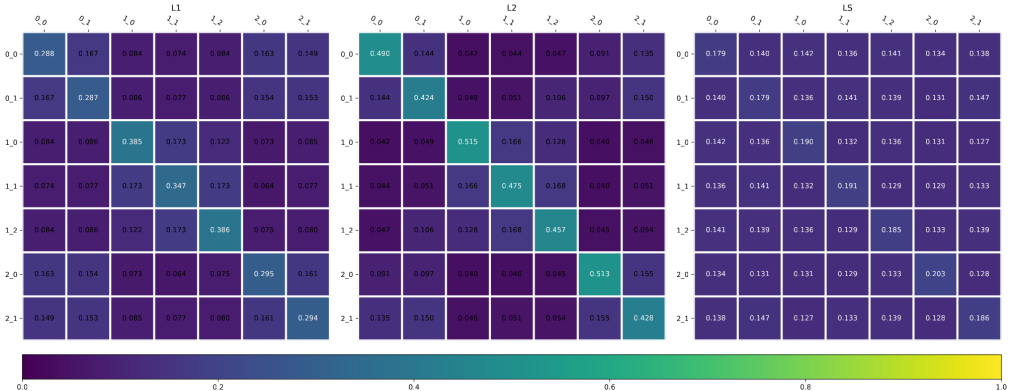
4. Laplaciano Adaptativo para Aprendizaje Multitarea



- Matrices de regClusterso para L1, L2 y LS-SVM MT de laplaciano adaptativo

Experimentos Sintéticos: Matrices de Adyacencia

4. Laplaciano Adaptativo para Aprendizaje Multitarea



- Matrices de clasClusterso para L1, L2 y LS-SVM MT de laplaciano adaptativo

Experimentos Reales: Procedimiento

4. Laplaciano Adaptativo para Aprendizaje Multitarea

	Grid	CTL	ITL	MTL	GLMTL	cvxGLMTL
C	$\left\{4^k : -2 \leq k \leq 2\right\}$	CV	CV	CV	CV	CV
ϵ	$\left\{\frac{\sigma}{4^k} : 1 \leq k \leq 6\right\}$	CV	CV	CV	CV	CV
γ	$\left\{\frac{4^k}{d} : -2 \leq k \leq 3\right\}$	CV	-	CTL	-	CTL
γ_s	$\left\{\frac{4^k}{d} : -2 \leq k \leq 3\right\}$	-	CV	ITL	CTL	CTL
λ	$\{0{,}2k : 0 \leq k \leq 5\}$	-	-	CV	-	CV
μ	$\left\{4^k : -1 \leq k \leq 3\right\}$	-	-	-	CV	GLMTL

- Usamos 5 particiones externas y obtenemos 5 scores de test
- Usamos la Validación Cruzada (CV) con 5 particiones internas para la búsqueda en rejilla de los hiperparámetros

UAM
Universidad Autónoma de Madrid

Experimentos Reales: Resultados

4. Laplaciano Adaptativo para Aprendizaje Multitarea

	computer	parkinson	landmine
CTL-L1	1,985±0,039	4,306±0,209	0,202±0,047
ITL-L1	1,623±0,028	1,949±0,021	0,243±0,014
MTL-L1	**1.526 ±0.052**	**1.942 ±0.019**	0,266±0,022
cvxGLMTL-L1	**1.526 ±0.052**	1,945±0,024	0,249±0,039
AdapGLMTL-L1	1,545±0,083	1,945±0,024	**0.272±0.033**
CTL-L2	2,001±0,028	4,211±0,093	0,235±0,045
ITL-L2	2,105±0,070	1,936±0,026	**0.267±0.019**
MTL-L2	1,506±0,043	1,928±0,037	0,260±0,041
cvxGLMTL-L2	1,507±0,045	**1.927 ±0.037**	0,251±0,048
AdapGLMTL-L2	**1.501 ±0.047**	1,928±0,038	0,249±0,055
CTL-LS	2,002±0,029	4,217±0,108	0,186±0,040
ITL-LS	1,609±0,015	1,928±0,017	0,182±0,007
MTL-LS	1,507±0,042	1,929±0,026	0,186±0,041
cvxGLMTL-LS	**1.502 ±0.047**	**1.917 ±0.022**	**0.197 ±0.033**
AdapGLMTL-LS	1,506±0,046	1,924±0,033	0,196±0,031

UAM
Universidad Autónoma de Madrid

Índice

5. Conclusiones y Trabajo Futuro

Conclusiones

- El Aprendizaje Multitarea ofrece una serie de ventajas, pero es necesario desarrollar técnicas para crear un acoplamiento de los modelos de cada tarea
- Proponemos una formulación que considera la combinación convexa de una parte común a todas las tareas y otra específica
- Formulación convexa con métodos de kernel:
  - Definimos las L1, L2 y LS-SVM MT con esta formulación convexa
  - Proponemos la combinación convexa de modelos preentrenados
  - Mostramos buenos resultados de las SVM MT convexas con varios problemas
- Formulación convexa con redes neuronales:
  - Aplicamos esta formulación convexa también a redes neuronales
  - Mostramos que nuestra propuesta obtiene mejores resultados que el *Hard Sharing* en cuatro conjuntos de imágenes

Conclusiones

5. Conclusiones y Trabajo Futuro

- La formulación convexa asume que todas las tareas comparten una parte común
- Las tareas a veces están repetidas o no todas comparten la misma relación entre ellas
- Con la regularización laplaciana se pueden modelar distintas relaciones entre tareas definiendo una matriz de adyacencia adecuada
- Regularización laplaciana de grafo:
  - Extendemos la regularización laplaciana a los espacios de kernel
  - Definimos modelos que juntan la formulación convexa con la regularización laplaciana, y la aplicamos a la L1, L2 y LS-SVMs
  - Proponemos un algoritmo adaptativo para aprender la matriz de adyacencia
  - Obtenemos buenos resultados en problemas sintéticos y reales

Trabajo Futuro

- Investigar métodos para la selección de hiperparámetros en los modelos MT
- Aprender de forma automática los hiperparámetros λ_r de la combinación convexa de redes usando el descenso por gradiente
- Aplicar la regularización laplaciana a modelos neuronales

UAM
Universidad Autónoma de Madrid

Advanced Kernel Methods for Multi-Task Learning

Gracias por su atención.

UAM
Universidad Autónoma de Madrid

Índice

5. Conclusiones y Trabajo Futuro

Referencias

[1] Feng Cai y Vladimir Cherkassky. "SVM+ regression and multi-task learning". En: *International Joint Conference on Neural Networks*. 2009, págs. 418-424.

[2] Rich Caruana. "Multitask Learning". En: *Mach. Learn.* 28.1 (1997), págs. 41-75.

[3] Theodoros Evgeniou, Charles A. Micchelli y Massimiliano Pontil. "Learning Multiple Tasks with Kernel Methods". En: *J. Mach. Learn. Res.* 6 (2005), págs. 615-637.

[4] Theodoros Evgeniou y Massimiliano Pontil. "Regularized multi–task learning". En: *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2004, págs. 109-117.

[5] Carlos Ruiz, Carlos M. Alaíz y José R. Dorronsoro. "A Convex Formulation of SVM-Based Multi-task Learning". En: *HAIS Proceedings*. Vol. 11734. Springer, 2019, págs. 404-415.

Referencias

[6] Carlos Ruiz, Carlos M. Alaíz y José R. Dorronsoro. "Adaptive Graph Laplacian for Convex Multi-Task Learning SVM". En: *HAIS Proceedings*. Vol. 12886. Springer, 2021, págs. 219-230.

[7] Carlos Ruiz, Carlos M. Alaíz y José R. Dorronsoro. "Convex formulation for multi-task L1-, L2-, and LS-SVMs". En: *Neurocomputing* 456 (2021), págs. 599-608.

[8] Carlos Ruiz, Carlos M. Alaíz y José R. Dorronsoro. "Convex Graph Laplacian Multi-Task Learning SVM". En: *ICANN*. Vol. 12397. Springer, 2020, págs. 142-154.

[9] Carlos Ruiz, Carlos M. Alaíz y José R. Dorronsoro. "Convex Multi-Task Learning with Neural Networks". En: *HAIS Proceedings*. Vol. 13469. Springer, 2022, págs. 223-235.