Multi-Task Feature Learning

Andreas Argyriou
Dept. of Computer Science, University College London

Theodoros Evgeniou
Technology Management, INSEAD

Massimiliano Pontil
Dept. of Computer Science, University College London

Learning Multiple Tasks Simultaneously

- Learning multiple related tasks vs. learning independently.
- Few data per task; pooling data across related tasks.
- Examples:
  - user preferences (movies, products etc.)
  - computer vision (recognizing faces, objects etc.)
  - text classification

  etc.

Multi-Task Feature Learning

- Assumption: common underlying representation across tasks.

- A *small* set of *shared* features ([Baxter 1995], [Torralba et al. 2004], [Ando & Zhang 2005] etc.).

Learning Paradigm

- Tasks $t = 1, \ldots, T$.
- m examples per task: $(x_{t1}, y_{t1}), \ldots, (x_{tm}, y_{tm}) \in \mathbb{R}^d \times \mathbb{R}$.
- Estimate $f_t : \mathbb{R}^d \to \mathbb{R}, \quad t = 1, \ldots, T$.
- Consider features

$$h_1(x), \ldots, h_d(x)$$

- Predict using functions

$$f_t(x) = \sum_{i=1}^{d} a_{it} h_i(x)$$

Weighting Features

- *Feature importance vs. tasks* is described by the matrix

$$A = \begin{pmatrix} a_{11} & \dots & a_{1T} \\ \vdots & \ddots & \vdots \\ a_{d1} & \dots & a_{dT} \end{pmatrix} = \begin{pmatrix} - & a^1 & - \\ & \vdots & \\ - & a^d & - \end{pmatrix} = \begin{pmatrix} | & & | \\ a_1 & \dots & a_T \\ | & & | \end{pmatrix}$$

where

$$a^i = (a_{i1}, \dots, a_{iT})$$

$$a_t = \begin{pmatrix} a_{1t} \\ \vdots \\ a_{dt} \end{pmatrix}$$

Sharing Features Across Tasks

- Desiderata:

  1. a *low-dimensional data representation* shared across the tasks

  2. the importance of each feature is *preserved across the tasks*

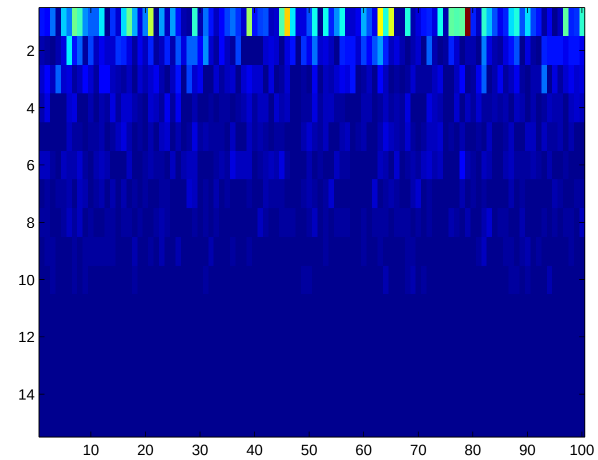
  3. *convex* formulation

Sharing Features Across Tasks

- In terms of matrix A:

  1. *most* a^i *should equal zero*

  2. for each i, the $|a_{it}|$ *should be similar*



$(2,1)$-Norm

- Approximate desiderata $1, 2$ using the norm

$$\|A\|_{2,1} \ := \ \sum_{i=1}^{d} \sqrt{\sum_{t=1}^{T} a_{it}^2}$$

  - First compute the 2-norms of the rows: $\|a^1\|_2, \ldots, \|a^d\|_2$
  - Then compute the 1-norm of the resulting vector: $\sum_{i=1}^{d} \|a^i\|_2$.

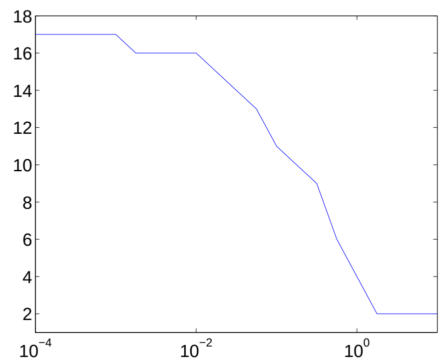
$(2,1)$-Norm

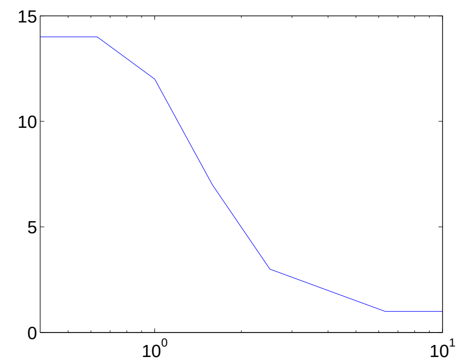
- Want the $(2,1)$-norm to be *small*.
- Small 1-norm favors sparsity and small 2-norm favors uniformity.
- Hence, small $(2,1)$-norm means
  - many rows a^i are ≈ 0
  - for each i, the $|a_{it}|$ are similar.

$(2,1)$-Norm Regularization

$$\min \left\{ \sum_{t=1}^{T} \sum_{j=1}^{m} L(y_{tj}, \sum_{i=1}^{d} a_{it} h_i(x_{tj})) + \gamma \|A\|_{2,1}^2 \;:\; A \in \mathbb{R}^{d \times T} \right\}$$

- This is a *convex* problem.
- The number of features in the solution decreases with γ





L_1 Regularization

- For *one task*, this is simply L_1 *regularization*:

$$\min\left\{\sum_{j=1}^{m} L(y_j, \sum_{i=1}^{d} a_i h_i(x_j)) + \gamma\, \|a\|_1^2 \;:\; a \in \mathbb{R}^d\right\}$$

- $\|a\|_1$ approximates $\#\{$nonzero entries of $a\}$.

- Many components of the solution will be ≈ 0.

Learning the Features

- How about learning the *features* as well?

- Focus on *linear, orthonormal* features

$$h_i(x) = \langle u_i, x \rangle$$

$$\min \left\{ \sum_{t=1}^{T} \sum_{j=1}^{m} L(y_{tj}, \langle a_t, U^\top x_{tj} \rangle) + \gamma \, \|A\|_{2,1}^2 \;:\; U^\top U = I, \; A \in \mathbb{R}^{d \times T} \right\}$$

- *Non-convex, nonsmooth* problem.

Convex Reformulation

- Variable transformation

$$\underset{d\times T}{W} = \begin{pmatrix} | & & | \\ w_1 & \cdots & w_T \\ | & & | \end{pmatrix} = \underset{d\times d}{U} \quad \underset{d\times T}{A}$$

$$D = U \ \mathrm{Diag}\left(\frac{\|a^i\|_2}{\|A\|_{2,1}}\right) U^\top$$

- Optimal W will be *low-rank*.
- D combines features U and feature weights A.

Convex Reformulation (cont.)

$$\inf\left\{\sum_{t=1}^{T}\sum_{j=1}^{m} L(y_{tj}, \langle w_t, x_{tj}\rangle) + \gamma \sum_{t=1}^{T} \langle w_t, D^{-1} w_t\rangle \right.$$

$$\left. : \; W \in \mathbb{R}^{d\times T},\; D \succ 0,\; \mathrm{trace}(D) \leq 1 \right\}$$

- $\sum_{t=1}^{T} \langle w_t, D^{-1} w_t\rangle$ induces relations between the tasks.

- *Jointly convex* in W and D !

Alternating Algorithm

- Alternate between W (supervised learning) and D (unsupervised "correlating" of tasks).

Initialization: set $D = \frac{I_{d \times d}}{d}$

while convergence condition is not true **do**

for $t = 1, \ldots, T,$ learn w_t *independently*

by minimizing $\sum_{j=1}^{m} L(y_{tj}, \langle w_t, x_{tj} \rangle) + \gamma \langle w_t, D^+ w_t \rangle$

end for

Find the D that best "relates" the tasks:

$$D = \frac{(WW^\top)^{\frac{1}{2}}}{\mathrm{trace}(WW^\top)^{\frac{1}{2}}} \qquad \text{(using SVD)}$$

end while

Experiment 1 (toy data)

- $T = 200$ tasks.

- $h_i(x) = x, \quad i = 1, \ldots, d$.

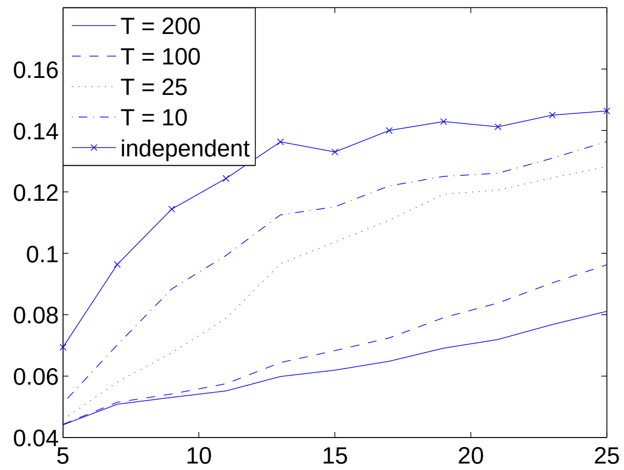
- $a_{it} = \begin{cases} \mathcal{N}(0, \sigma_i) & i = 1, \ldots, 5 \\ 0 & i = 6, \ldots, d \end{cases}$

- 5 training examples per task. Inputs uniformly drawn from $[0, 1]^d$.

- Outputs $y_{tj} = \langle a_t, x_{tj} \rangle +$ noise.

Experiment 1 (toy data)

Test error



Input dimensionality

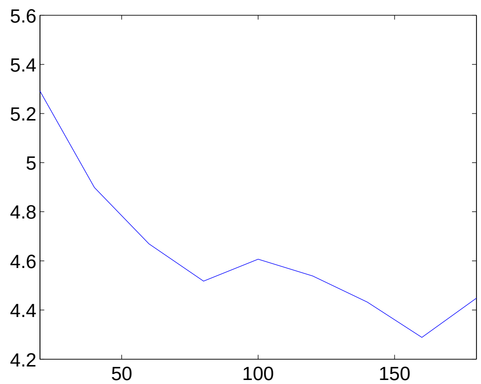
- Learning multiple tasks together improves performance.
- *Improvement is large*, even when most features are irrelevant.
- More tasks lead to better estimates of the features.

Experiment 2 (real data)

- Consumers' ratings of products [Lenk et al. 1996].
- 180 persons (tasks).
- 8 PC models (training examples); 4 PC models (test examples).
- 13 binary input attributes (RAM, CPU, price etc.).
- Integer output in $\{0, \ldots, 10\}$ (likelihood of purchase).

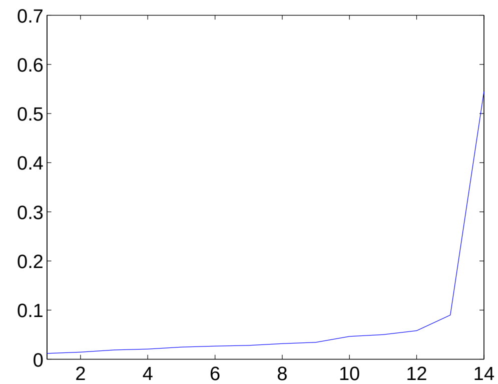
Experiment 2 (real data)

Test error



#tasks

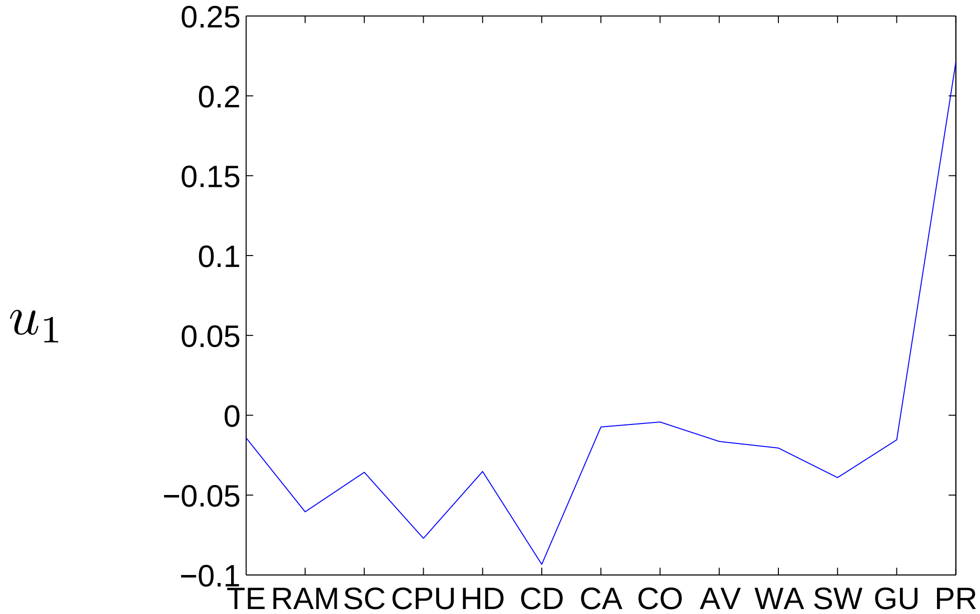
Eig(D)



- Performance improves with more tasks (for independent, error $= 16.53$).

- A single most important feature shared by all persons.

Experiment 2 (real data)



- The most important feature weighs *technical characteristics* (RAM, CPU, CD-ROM) vs. *price*.

Summary

- Multi-task feature learning
  - *low-dimensional data representation* shared by a pool of tasks
  - feature importance *preserved across tasks*.
- *Convex problem*. Converges to global solution.
- Alternating algorithm.
- Solution is *low-rank*. Algorithm *selects the salient features*. Additional tasks enhance prediction.

Future Work

- More general, nonlinear features.
- Handle > 1 clusters of tasks. Hierarchical models of tasks/features.
- Connection to Bayesian methods.

Regularization with the Trace Norm

- Minimizing over D yields

$$\sum_{t=1}^{T}\sum_{i=1}^{m} L(y_{ti}, \langle w_t, x_{ti}\rangle) + \gamma \, \|W\|_{\Sigma}^{2}$$

- Involves the *trace norm* of W (compare to [Srebro et al.]).

- Favors low-rank matrices (also apparent from $W = UA$).

- Convex but nonsmooth problem.