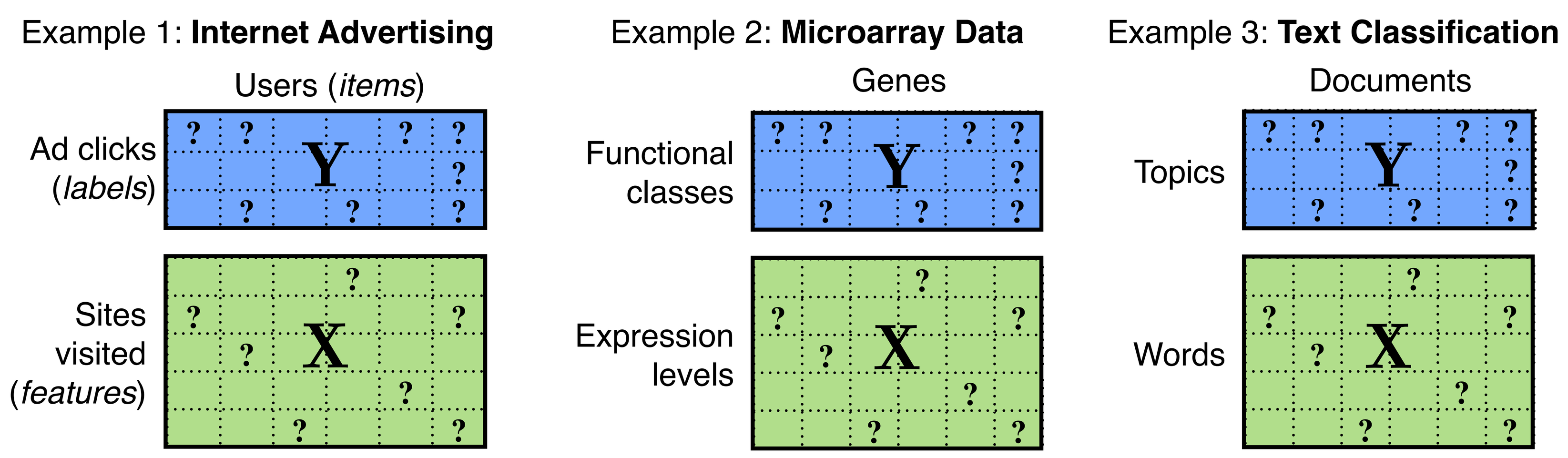
Transduction with Matrix Completion: Three Birds with One Stone

Andrew B. Goldberg[1], Xiaojin Zhu[1], Benjamin Recht[1], Jun-Ming Xu[1], Robert Nowak[2]
Department of [1]Computer Sciences, [2]Electrical and Computer Engineering, University of Wisconsin-Madison

Three Birds: Multi-label + Transduction + Imputation



Problem: (3 birds)
- **Multi-label**: each item has one or more labels based on a set of tasks
- **+ Transduction**: many labels unobserved; want to predict these labels
- **+ Missing features**: many features missing; want to impute them

Formally: $\mathbf{x}_1 \dots \mathbf{x}_n \in \mathbb{R}^d$ $\quad \mathbf{X} = [\mathbf{x}_1 \dots \mathbf{x}_n]$
$\mathbf{y}_1 \dots \mathbf{y}_n \in \{-1, 1\}^t$ $\quad \mathbf{Y} = [\mathbf{y}_1 \dots \mathbf{y}_n]$
Observe only the entries in index sets $\Omega_\mathbf{X}\ \Omega_\mathbf{Y}$

Goals:
- **a.** Predict missing labels y_{ij} for $(i,j) \notin \Omega_\mathbf{Y}$
- **b.** Impute missing features x_{ij} for $(i,j) \notin \Omega_\mathbf{X}$

Low Rank Assumption for Semi-Supervised Learning

Problem is ill-posed without further assumptions

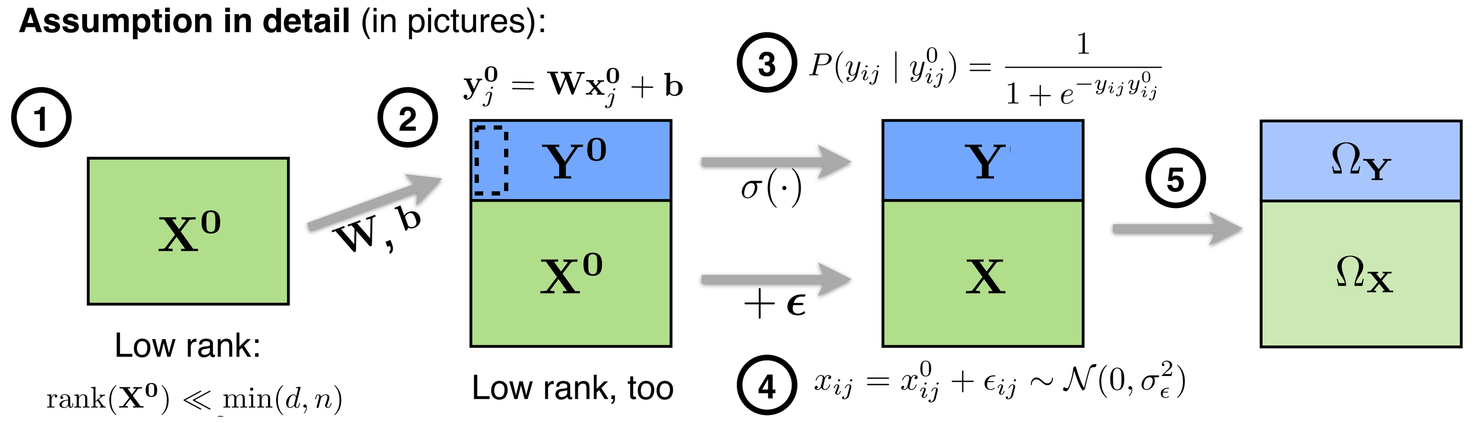
Novel assumption: Feature-by-item matrix $\mathbf{X}$ and label-by-item matrix $\mathbf{Y}$ are **jointly low rank**
- $\mathbf{X}$ and $\mathbf{Y}$ jointly produced by an underlying low-rank matrix, coupling the tasks and the features
- Can implicitly use observed labels for one task to predict unobserved labels for another
- Similarly, observed features can help predict missing ones due to few underlying factors

Assumption in detail (in words):
1. Low rank pre-feature matrix $\mathbf{X}^0_{d \times n}$ $\quad \mathrm{rank}(\mathbf{X}^0) \ll \min(d,n)$
2. Soft labels via affine transformation $\mathbf{Y}^0 = \mathbf{W}\mathbf{X}^0 + \mathbf{b}\mathbf{1}^\top$
3. Noisy discrete labels $\mathbf{Y} = \mathrm{Bernoulli}(\sigma(\mathbf{Y}^0))$
4. Noisy features $\epsilon_{ij} \sim \mathcal{N}(0, \sigma_\epsilon^2)$ $\quad \mathbf{X} = \mathbf{X}^0 + \epsilon$
5. Random masks reveal only: $x_{ij} \iff (i,j) \in \Omega_\mathbf{X}$, $\quad y_{ij} \iff (i,j) \in \Omega_\mathbf{Y}$

Combined (noise-free) matrix is also low rank $\begin{bmatrix}\mathbf{Y}^0 \\ \mathbf{X}^0\end{bmatrix}$

$$\mathrm{rank}([\mathbf{Y}^0; \mathbf{X}^0]) \leq \mathrm{rank}(\mathbf{X}^0) + 1$$

Assumption in detail (in pictures):



One Stone: Matrix Completion (MC)

- **Given:** partially observed features and labels $\mathbf{X}, \mathbf{Y}, \Omega_\mathbf{X}, \Omega_\mathbf{Y}$
- **Do:** recover the intermediate low-rank matrix $[\mathbf{Y}^0; \mathbf{X}^0] = \mathbf{Z}$

Ideally want to solve:

$$\underset{\mathbf{Z} \in \mathbb{R}^{(t+d) \times n}}{\mathrm{argmin}} \ \mathrm{rank}(\mathbf{Z})$$
$$\text{s.t.} \ \mathrm{sign}(z_{ij}) = y_{ij}, \ \forall (i,j) \in \Omega_\mathbf{Y}$$
$$z_{(i+t)j} = x_{ij}, \ \forall (i,j) \in \Omega_\mathbf{X}$$

But rank is non-convex!
Relax with convex nuclear norm: $\|\mathbf{Z}\|_* = \sum_{k=1}^{\min(t+d,n)} \sigma_k(\mathbf{Z})$

But features and labels are noisy! Use loss functions.
Squared loss for features: $c_x(u,v) = \frac{1}{2}(u-v)^2$
Logistic loss for labels: $c_y(u,v) = \log(1 + \exp(-uv))$

How to handle the bias term? Two Formulations

- Nuclear norm MC assumes that rows of labels can be recovered as linear combinations of rows of features ($\mathbf{Y}^0 = \mathbf{W}\mathbf{X}^0$)
- Need special handling to account for the bias vector $\mathbf{b}$ (as in $\mathbf{Y}^0 = \mathbf{W}\mathbf{X}^0 + \mathbf{b}\mathbf{1}^\top$)
- Can model $\mathbf{b}$ explicitly or implicitly

	MC-b (explicit)	MC-1 (implicit)
Optimization variables	$\mathbf{Z} \in \mathbb{R}^{(t+d) \times n}, \mathbf{b} \in \mathbb{R}^t$	$\mathbf{Z} \in \mathbb{R}^{(t+d+1) \times n}$
$\mathbf{Z}$	$[\mathbf{W}\mathbf{X}^0; \mathbf{X}^0]$	$[\mathbf{Y}^0; \mathbf{X}^0; \mathbf{1}^\top]$
How to predict task-i label of item j	$\mathrm{sign}(z_{ij} + b_i)$	$\mathrm{sign}(z_{ij})$
Optimization method	Fixed Point Continuation (gradient + shrinkage)	FPC (gradient + shrinkage + projection)
Convergence guarantee	Yes, with appropriately chosen step size	No, but converges in practice

MC-b
$$\underset{\mathbf{Z} \in \mathbb{R}^{(t+d) \times n}, \mathbf{b} \in \mathbb{R}^t}{\mathrm{argmin}} \ \mu \|\mathbf{Z}\|_* + \frac{\lambda}{|\Omega_\mathbf{Y}|} \sum_{(i,j) \in \Omega_\mathbf{Y}} c_y(z_{ij} + b_i, y_{ij}) + \frac{1}{|\Omega_\mathbf{X}|} \sum_{(i,j) \in \Omega_\mathbf{X}} c_x(z_{(i+t)j}, x_{ij})$$

MC-1
$$\underset{\mathbf{Z} \in \mathbb{R}^{(t+d+1) \times n}}{\mathrm{argmin}} \ \mu \|\mathbf{Z}\|_* + \frac{\lambda}{|\Omega_\mathbf{Y}|} \sum_{(i,j) \in \Omega_\mathbf{Y}} c_y(z_{ij}, y_{ij}) + \frac{1}{|\Omega_\mathbf{X}|} \sum_{(i,j) \in \Omega_\mathbf{X}} c_x(z_{(i+t)j}, x_{ij})$$
$$\text{s.t.} \quad z_{(t+d+1)\cdot} = \mathbf{1}^\top$$

Optimization Techniques

Can solve both problems using modifications of Fixed Point Continuation (FPC) [Ma *et al*, 2009]

FPC algorithm for MC-b

Input: Initial matrix $\mathbf{Z}_0$, bias $\mathbf{b}_0$, parameters μ, λ, Step sizes $\tau_\mathbf{b}, \tau_\mathbf{Z}$
Determine $\mu_1 > \mu_2 > \cdots > \mu_L = \mu > 0$.
Set $\mathbf{Z} = \mathbf{Z}_0, \mathbf{b} = \mathbf{b}_0$.
foreach $\mu = \mu_1, \mu_2, \dots, \mu_L$ **do**
  while *Not converged* **do**
    Compute $\mathbf{b} = \mathbf{b} - \tau_\mathbf{b} g(\mathbf{b}), \mathbf{A} = \mathbf{Z} - \tau_\mathbf{Z} g(\mathbf{Z})$ — Gradient step
    Compute SVD of $\mathbf{A} = \mathbf{U}\Lambda\mathbf{V}^\top$ — Shrinkage step
    Compute $\mathbf{Z} = \mathbf{U} \max(\Lambda - \tau_\mathbf{Z}\mu, 0)\mathbf{V}^\top$
  end
end
Output: Recovered matrix $\mathbf{Z}$, bias $\mathbf{b}$

Gradient step
Shrinkage step
Projection step

FPC algorithm for MC-1

Input: Initial matrix $\mathbf{Z}_0$, parameters μ, λ, Step size $\tau_\mathbf{Z}$
Determine $\mu_1 > \mu_2 > \cdots > \mu_L = \mu > 0$.
Set $\mathbf{Z} = \mathbf{Z}_0$.
foreach $\mu = \mu_1, \mu_2, \dots, \mu_L$ **do**
  while *Not converged* **do**
    Compute $\mathbf{A} = \mathbf{Z} - \tau_\mathbf{Z} g(\mathbf{Z})$
    Compute SVD of $\mathbf{A} = \mathbf{U}\Lambda\mathbf{V}^\top$
    Compute $\mathbf{Z} = \mathbf{U} \max(\Lambda - \tau_\mathbf{Z}\mu, 0)\mathbf{V}^\top$
    Project $\mathbf{Z}$ to feasible region $z_{(t+d+1)\cdot} = \mathbf{1}^\top$
  end
end
Output: Recovered matrix $\mathbf{Z}$

Experimental Setup

- **Goal:** Evaluate MC as a tool for multi-label transductive classification with missing data
- **Baselines** (two-step approaches combining an imputation and prediction method)**:**
  1. Imputation: FPC, EM with k-component mixture model, Mean imputation, or Zero imputation
  2. Prediction: Set of independent linear SVMs (one per label/task)
- **Procedure:** 10 trials with random selection of observed feature and label entries (and synthetic data)

Synthetic Data Results

$\mathbf{X}^0 = \mathbf{L}\mathbf{R}^\top$, $\mathbf{L} \in \mathbb{R}^{d \times r}$, $\mathbf{R} \in \mathbb{R}^{n \times r}$

Meta-averages over 24 synthetic data sets created by fixing # tasks t=10, # features d=20 and varying r (the rank of $\mathbf{X}^0$), # items n, noise level, and observed rate

	MC-b	MC-1	FPC+SVM	EM1+SVM	Mean+SVM	Zero+SVM
Transductive Label Error (% of missing labels predicted incorrectly)	25.6	**21.4**	22.6	24.1	28.6	28.0
Relative Feature Imputation Error $\left(\sum_{ij \notin \Omega_\mathbf{X}} (x_{ij} - \hat{x}_{ij})^2\right) / \sum_{ij \notin \Omega_\mathbf{X}} x_{ij}^2$	**0.66**	**0.66**	0.68	0.78	1.02	1.00

Obs. 1: MC-b and MC-1 best at imputation and better than FPC+SVM, suggesting $\mathbf{Y}$ helps to impute $\mathbf{X}$.
Obs. 2: MC-1 is best for label transduction. Surprisingly, MC-b's imputation does not translate to classification.
Obs. 3: Other results (in paper) show that MC-b and MC-1 improve more as the number of tasks increases.

Real Data Results

Music emotions: predict emotions evoked by songs (n=593, t=6, d=72)

Obs.=40%	60%	80%	Algorithm	Obs.=40%	60%	80%
28.0(1.2)	25.2(1.0)	22.2(1.6)	MC-b	0.69(0.05)	0.54(0.10)	0.41(0.02)
27.4(0.8)	**23.7(1.6)**	**19.8(2.4)**	MC-1	0.60(0.05)	0.46(0.12)	0.25(0.03)
26.9(0.7)	25.2(1.6)	24.4(2.0)	FPC+SVM	0.64(0.01)	0.46(0.02)	0.31(0.03)
26.0(1.1)	**23.6(1.1)**	21.2(2.3)	EM1+SVM	0.46(0.09)	0.23(0.04)	**0.13(0.01)**
26.2(0.9)	**23.1(1.2)**	21.6(1.6)	EM4+SVM	0.49(0.10)	0.27(0.04)	0.15(0.02)
26.3(0.8)	24.2(1.0)	22.6(1.3)	Mean+SVM	**0.18(0.00)**	**0.19(0.00)**	0.20(0.01)
30.3(0.6)	28.9(1.1)	25.7(1.4)	Zero+SVM	1.00(0.00)	1.00(0.00)	1.00(0.00)

transductive label error — relative feature imputation error

Observation: MC-1 among best label-error performers for 60%, 80% observed, despite poor feature imputation.

Yeast microarray: predict gene functional classes (n=2417, t=14, d=103)

Obs. =40%	60%	80%	Algorithm	Obs. =40%	60%	80%
16.1(0.3)	**12.2(0.3)**	**8.7(0.4)**	MC-b	0.83(0.02)	0.76(0.00)	0.73(0.02)
16.7(0.3)	13.0(0.2)	**8.5(0.4)**	MC-1	0.86(0.00)	0.92(0.00)	0.74(0.00)
21.5(0.3)	20.8(0.3)	20.3(0.3)	FPC+SVM	**0.81(0.00)**	**0.76(0.00)**	**0.72(0.00)**
22.0(0.2)	21.2(0.2)	20.4(0.2)	EM1+SVM	1.15(0.02)	1.04(0.02)	0.77(0.01)
21.7(0.2)	21.1(0.2)	20.5(0.4)	Mean+SVM	1.00(0.00)	1.00(0.00)	1.00(0.00)
21.6(0.2)	21.1(0.2)	20.5(0.4)	Zero+SVM	1.00(0.00)	1.00(0.00)	1.00(0.00)

transductive label error — relative feature imputation error

Observation: MC-b and MC-1 significantly outperform baselines in label error, benefiting from simultaneous prediction of missing labels and features.

Summary and Conclusions

- First work to simultaneously perform: 1) multi-label prediction, 2) transduction, and 3) feature imputation
- Novel low-rank SSL assumption leads to formulation as a matrix completion problem
- Introduced two algorithms (MC-b and MC-1) that outperform baselines on synthetic and real data
- **Future work**: Go beyond linear classification by explicit kernelization (e.g., using a polynomial kernel)

Neural Information Processing Systems (NIPS) 2010, Vancouver, Canada
goldberg@cs.wisc.edu, jerryzhu@cs.wisc.edu