Graph-Based Semi-Supervised Learning

April 4, 2013

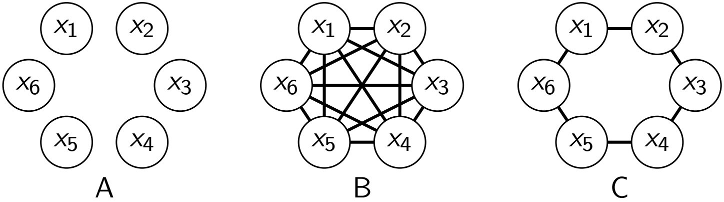
Semi-Supervised Learning

- Semi-supervised learning (SSL) learns from both
  - labeled data (expensive and scarce)
  - unlabeled data (cheap and abundant)
- Given *iid* samples from an unknown distribution $p(\mathbf{x}, y)$ over $\mathbf{x} \in \Omega$ and $y \in \mathbb{Z}$ organized as
  - a labeled set: $\mathcal{X}_l \cup \mathcal{Y}_l = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)\}$
  - an unlabeled set: $\mathcal{X}_u = \{\mathbf{x}_{l+1}, \ldots, \mathbf{x}_{l+u}\}$
- Output missing labels $\hat{\mathcal{Y}}_u = \{\hat{y}_{l+1}, \ldots, \hat{y}_{l+u}\}$ that largely agree with true missing labels $\mathcal{Y}_u = \{y_{l+1}, \ldots, y_{l+u}\}$

Graph Based SSL

- Graph based semi-supervised learning first constructs a graph $\mathcal{G} = (V, E)$ from $\mathcal{X}_l \cup \mathcal{X}_u$ which is usually
  - a sparse graph (using k-nearest neighbors)
  - and a weighted graph (radial basis function weighting)
- Subsequently, $\mathcal{G}$ and $\mathcal{Y}_l$ yield $\hat{\mathcal{Y}}_u$ via a labeling algorithm:
  - Laplacian regularization (Belkin & Niyogi 02)
  - Gaussian fields and harmonic functions (Zhu et al. 03)
  - Local and global consistency (Zhou et al. 04)
  - Laplacian support vector machines (Belkin et al. 06)
  - Transduction via alternating minimization (Wang et al. 08)

Graph Construction



A Given the full dataset $\mathcal{X}_l \cup \mathcal{X}_u$ of $n = l + u$ samples

B Form full weighted graph $\mathcal{G}$ with adjacency matrix $A \in \mathbb{R}^{n \times n}$ using any kernel $k(.,.)$ elementwise as $A_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$

- Kernel choice is application dependent and only locally reliable
- Equivalent to use distances and matrix $D \in \mathbb{R}^{n \times n}$ defined as $D_{ij} = \sqrt{k(\mathbf{x}_i, \mathbf{x}_i) + k(\mathbf{x}_j, \mathbf{x}_j) - 2k(\mathbf{x}_i, \mathbf{x}_j)}$

C Sparsify graph with pruning matrix $P \in \mathbb{B}^{n \times n}$

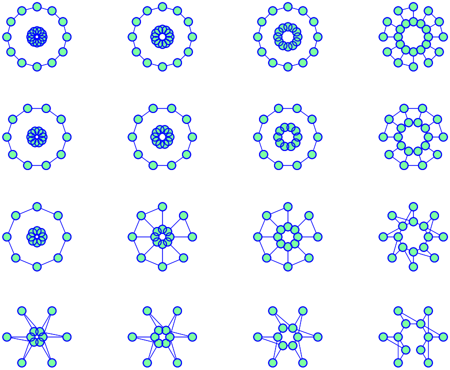
Neighborhood Graphs

- ϵ-NEIGHBORHOOD Set $P \in \mathbb{B}^{n \times n}$ as $P_{ij} = \delta(D_{ij} \leq \epsilon)$
  - The ϵ-neighborhood often forms disconnected graphs
  - Better to make ϵ adaptive using k-nearest neighbors algorithm
- k-NEAREST NEIGHBORS Set $P = \max(\hat{P}, \hat{P}^\top)$ where

$$\hat{P} = \arg \min_{P \in \mathbb{B}^{n \times n}} \sum_{ij} P_{ij} D_{ij} \;\; s.t. \; \sum_j P_{ij} = k, P_{ii} = 0$$

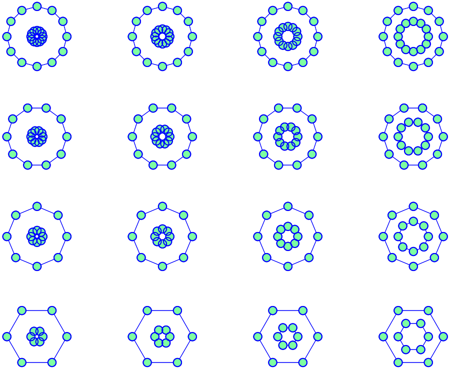
  - Despite its name, this algorithm doesn't give k neighbors
  - Due to symmetrization of $\hat{P}$, $\sum_i P_{ij} \geq k$ neighbors
  - Alternatively, can take $P = \min(\hat{P}, \hat{P}^\top)$, then $\sum_i P_{ij} \leq k$

k-Nearest Neighbor Graphs



- Above is k-nearest neighbors with $k = 2$
- Related to the so-called Kissing Number (see Wikipedia)

b-Matching Graphs



- Above is unipartite b-matching with $b = 2$
- Fixes the so-called Kissing Number issue

b-Matching Graphs

- b-MATCHING is k-nearest neighbors with explicit symmetry

$$P = \arg\min_{P \in \mathbb{B}^{n\times n}} \sum_{ij} P_{ij} D_{ij} \;\; s.t. \sum_j P_{ij} = b, P_{ii} = 0, P_{ij} = P_{ji}$$

- Known as unipartite generalized matching
- Efficient combinatorial solver known (Edmonds 1965)
- Like an LP with exponentially many blossom inequalities
- Fastest solvers now use max product belief propagation
  - Exact for bipartite b-matching in $O(bn^3)$ (Huang & J 2007)
  - Under mild assumptions get $O(n^2)$ (Salez & Shah 2009)
  - Exact for integral unipartite b-matching (Sanghavi et al. 2008)
  - Exact for unipartite perfect graph b-matching (J 2009)

Bipartite 1-Matching

	Motorola	Apple	IBM
"laptop"	0$	2$	2$
"server"	0$	2$	3$
"phone"	2$	3$	0$

$$\rightarrow P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

- Given C, $\max_{P \in \mathbb{B}^{n\times n}} \sum_{ij} C_{ij} P_{ij}$ such that $\sum_i P_{ij} = \sum_j P_{ij} = 1$
- Classical Hungarian marriage problem $O(n^3)$
- Creates a very loopy graphical model
- Max product takes $O(n^3)$ for exact MAP (Bayati et al. 2005)
- Use $C = -D$ to mimic minimization of distances

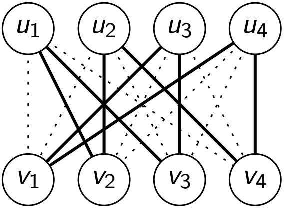
Bipartite b-Matching

	Motorola	Apple	IBM
"laptop"	0\$	2\$	2\$
"server"	0\$	2\$	3\$
"phone"	2\$	3\$	0\$

$$\rightarrow P = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

- Given C, $\max_{P \in \mathbb{B}^{n \times n}} \sum_{ij} C_{ij} P_{ij}$ such that $\sum_i P_{ij} = \sum_j P_{ij} = b$
- Combinatorial b-matching problem $O(bn^3)$, (Google Adwords)
- Creates a very loopy graphical model
- Max product takes $O(bn^3)$ for exact MAP (Huang & J 2007)
- Use $C = -D$ to mimic minimization of distances
- Code also applies to unipartite b-matching problems

Bipartite b-Matching

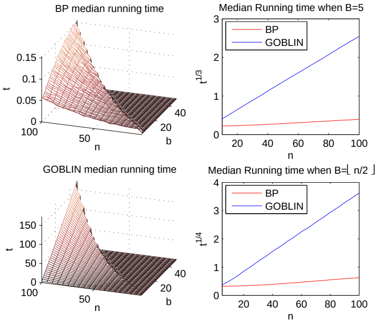


- Graph $G = (U, V, E)$ with $U = \{u_1, \ldots, u_n\}$ and $V = \{v_1, \ldots, v_n\}$ and $M(.)$, a set of neighbors of node u_i or v_j
- Define $x_i \in X$ and $y_i \in Y$ where $x_i = M(u_i)$ and $y_i = M(v_j)$
- Then $p(X, Y) = \frac{1}{Z} \prod_i \prod_j \psi(x_i, y_j) \prod_k \phi(x_k)\phi(y_k)$ where $\phi(y_j) = \exp(\sum_{u_i \in y_j} C_{ij})$ and $\psi(x_i, y_j) = \neg(v_j \in x_i \oplus u_i \in y_j)$

b-Matching

- Code at http://www.cs.columbia.edu/∼jebara/code
- Also applies to unipartite b-matching

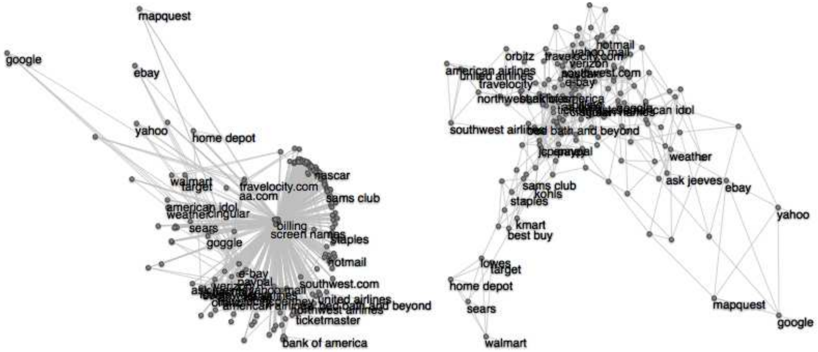
b-Matching



Applications:
clustering (J & S 2006)
classification (H & J 2007)
collaborative filtering (H & J 2008)
visualization (S & J 2009)

Max product is $O(n^2)$, beats other solvers (Salez & Shah 2009)

b-Matching



- Left is k-nearest neighbors, right is unipartite b-matching.

Graph Weighting

Given sparsification matrix P, obtain final adjacency matrix W graph for $\mathcal{G}$ using any of the following weighting schemes

BN BINARY Set $W = P$

GK GAUSSIAN KERNEL Set $W_{ij} = P_{ij} \exp(-d(\mathbf{x}_i, \mathbf{x}_j)/2\sigma^2)$ where $d(.,.)$ is any distance function (ℓ_p distance, chi squared distance, cosine distance, etc.)

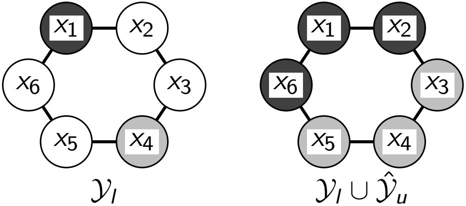
LLR LOCALLY LINEAR RECONSTRUCTION Set W to reconstruct each point with its neighborhood (Roweis & Saul 00)

$$W = \arg\min_{W \in \mathbb{R}^{n \times n}} \sum_i \|\mathbf{x}_i - \sum_j P_{ij} W_{ij} \mathbf{x}_j\|^2 \; s.t. \sum_j W_{ij} = 1, W_{ij} \geq 0$$

Graph Labeling

- Given known labels $\mathcal{Y}_l$ and sparse weighted graph $\mathcal{G}$ with W
- Output $\hat{\mathcal{Y}}_u$ by diffusion or propagation
- Define the following intermediate matrices
  - Degree $\mathcal{D} \in \mathbb{R}^{n \times n}$ where $\mathcal{D}_{ii} = \sum_j W_{ij}$, $\mathcal{D}_{ij} = 0$ for $i \neq j$
  - Laplacian $\Delta = \mathcal{D} - W$
  - Normalized Laplacian $L = \mathcal{D}^{-1/2} \Delta \mathcal{D}^{-1/2}$
  - Classification function $F \in \mathbb{R}^{n \times c}$ where $F = \begin{bmatrix} F_l \\ F_u \end{bmatrix}$
  - Label matrix $Y \in \mathbb{B}^{n \times c}$, $Y_{ij} = \delta(y_i = j)$ and $Y = \begin{bmatrix} Y_l \\ Y_u \end{bmatrix}$
- Consider these algorithms for producing F and Y
  - Gaussian Random Fields (GRF)
  - Local and Global Consistency (LGC)
  - Graph Transduction via Alternating Minimization (GTAM)

Gaussian Random Fields

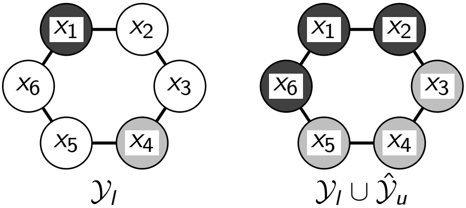


- Gaussian Random Fields smooth classification function over Laplacian while clamping known labels

$$\min_{F \in \mathbb{R}^{n \times c}} \operatorname{tr}(F^\top \Delta F) \ \ s.t.\, \Delta F_u = 0, F_l = Y_l$$

and then obtain Y from F by rounding

Local and Global Consistency

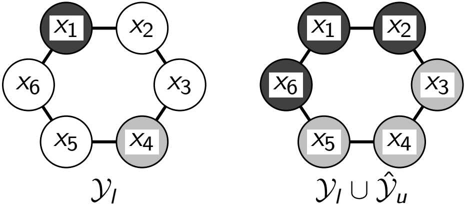


- Local and Global Consistency smooth using normalized Laplacian and softly constrain F_l to Y_l

$$\min_{F \in \mathbb{R}^{n \times c}} \operatorname{tr}\left((F^\top L F) + \mu (F - Y)^\top (F - Y) \right)$$

and then obtain Y from F by rounding

Graph Transduction via Alternating Minimization



- Graph Transduction via Alternating Minimization make the optimization bivariate jointly over F and Y

$$\min_{\substack{F \in \mathbb{R}^{n \times c} \\ Y \in \mathbb{B}^{n \times c}}} \operatorname{tr}\Big(F^\top L F + \mu (F - VY)^\top (F - VY)\Big) \; s.t. \sum_j Y_{ij} = 1$$

  where V is a diagonal matrix containing class proportions
- Given current F, Y is updated greedily one entry at at time

Synthetic Experiments

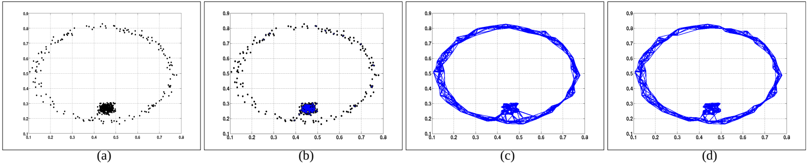


Figure: Synthetic dataset (a) two sampled rings (b) ϵ-neighborhood graph (c) k-nearest graph with $k = 10$ (d) b-matching with $b = 10$.

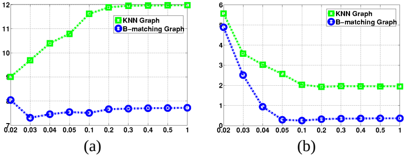
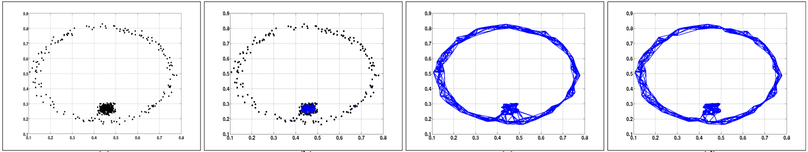


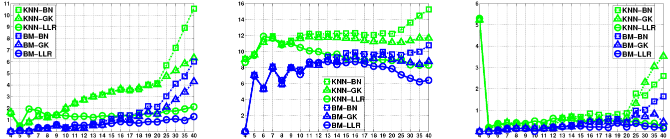
Figure: 50-fold error rate varying σ in Gaussian kernel for (a) LGC and (b) GRF. GTAM (not shown) does best. One point per class labeled.

Synthetic Experiments



(a) (b) (c) (d)

Figure: Synthetic dataset (a) two sampled rings (b) ϵ-neighborhood graph (c) k-nearest graph with $k = 10$ (d) b-matching with $b = 10$.



(a) (b) (c)

Figure: 50-fold error rate under varying b or k and weighting schemes for (a) LGC, (b) GRF and (c) GTAM. One point per class labeled.

Real Experiment Error Rates

Data set	USPS	COIL	BCI	TEXT
QC + CMN	13.61	59.63	50.36	40.79
LDS	25.2	67.5	49.15	31.21
Laplacian	17.57	61.9	49.27	27.15
Laplacian RLS	18.99	54.54	48.97	33.68
CHM (normed)	20.53	-	46.9	-
GRF-KNN-BN	19.11	64.45	48.77	47.65
GRF-KNN-GK	12.94	61.31	48.98	47.65
GRF-KNN-LLR	19.20	61.19	48.46	47.14
GRF-BM-BN	18.98	60.63	48.44	43.16
GRF-BM-GR	12.82	60.87	48.77	42.88
GRF-BM-LLR	18.95	60.84	48.25	42.94

Data set	USPS	COIL	BCI	TEXT
LGC-KNN-BN	14.7	59.18	48.94	48.79
LGC-KNN-GK	12.42	57.3	48.42	48.09
LGC-KNN-LLR	15.8	56.75	48.65	40.28
LGC-BM-BN	14.4	59.31	48.34	40.44
LGC-BM-GR	11.89	58.17	48.17	37.39
LGC-BM-LLR	14.44	58.69	48.08	39.83
GTAM-KNN-BN	6.42	29.70	47.56	49.36
GTAM-KNN-GK	4.77	16.69	47.29	49.13
GTAM-KNN-LLR	6.69	15.35	45.54	41.48
GTAM-BM-BN	5.2	25.83	47.92	17.81
GTAM-BM-GR	**4.31**	13.65	47.48	28.74
GTAM-BM-LLR	5.45	**12.57**	**43.73**	**16.35**

Real Experiment Error Rates with More Labeling

Data set	USPS		TEXT	
# of labels	10	100	10	100
QC + CMN	13.61	6.36	40.79	25.71
TSVM	25.2	9.77	31.21	24.52
LDS	17.57	4.96	27.15	23.15
Laplacian RLS	18.99	4.68	33.68	23.57
CHM (normed)	20.53	-	7.65	-
GRF-KNN-BN	19.11	9.07	47.65	41.56
GRF-KNN-GK	13.01	5.58	48.2	41.57
GRF-KNN-LLR	19.20	11.17	47.14	35.17
GRF-BM-BN	18.98	9.06	43.16	25.27
GRF-BM-GK	12.92	5.34	41.24	22.28
GRF-BM-LLR	18.95	10.08	42.95	24.54

Data set	USPS		TEXT	
# of labels	10	100	10	100
LGC-KNN-BN	14.99	12.34	48.63	43.44
LGC-KNN-GK	12.34	5.49	49.06	41.51
LGC-KNN-LLR	15.88	13.63	44.86	37.53
LGC-BM-BN	14.62	11.71	40.88	26.19
LGC-BM-GK	11.92	5.21	38.14	23.17
LGC-BM-LLR	14.69	12.19	40.29	24.91
GTAM-KNN-BN	6.59	5.98	49.36	46.67
GTAM-KNN-GK	4.86	**2.56**	49.07	46.06
GTAM-KNN-LLR	6.77	6.19	41.46	39.59
GTAM-BM-BN	6.00	5.08	17.44	16.78
GTAM-BM-GR	**4.62**	3.08	16.85	15.91
GTAM-BM-LLR	5.59	5.16	**16.01**	**14.88**

Conclusions

- Graph-based SSL has top performance
- Investigated 3 sparsifications $\times$ 3 weightings $\times$ 3 algorithms
- GTAM method has better accuracy than other algorithms
- On real data, k-nearest neighbors creates irregular graphs
- Regularity from b-matching ensures balanced manifolds
- b-matching consistently improves k-nearest neighbors
- Fast and exact b-matching code available using max-product
- The runtime of b-matching is not a bottleneck for SSL
- Theoretical guarantees forthcoming