Semisupervised Learning Using Negative Labels

Hou, Ch. *et al.*

IEEE Transactions on Neural Networks
March 2011

Intro & Motivation

- Semi-supervised approach which attempts to incorporate partial information from unlabeled data points

Intro & Motivation

- Semi-supervised approach which attempts to incorporate partial information from unlabeled data points
- Normally for SSL: a small portion of data is labeled, the rest is unlabeled

Intro & Motivation

- Semi-supervised approach which attempts to incorporate partial information from unlabeled data points
- Normally for SSL: a small portion of data is labeled, the rest is unlabeled
- In current approach: some “negative” information about unlabeled data is known, i.e. it is known to which category they don’t belong to

Intro & Motivation

- Examples

Intro & Motivation

- Examples

Human, sport, animal?

Intro & Motivation

- Examples

Human, sport, ani~~x~~al?

Intro & Motivation

- Examples

Human, sport, ani~~m~~al?

Mountain, water, building?

Intro & Motivation

- Examples

Human, sport, animal?

Mountain, water, building?

Intro & Motivation

- Examples

Human, sport, animal?

Mountain, water, building?

Instead of:	?	?	?
We have:	?	0	?

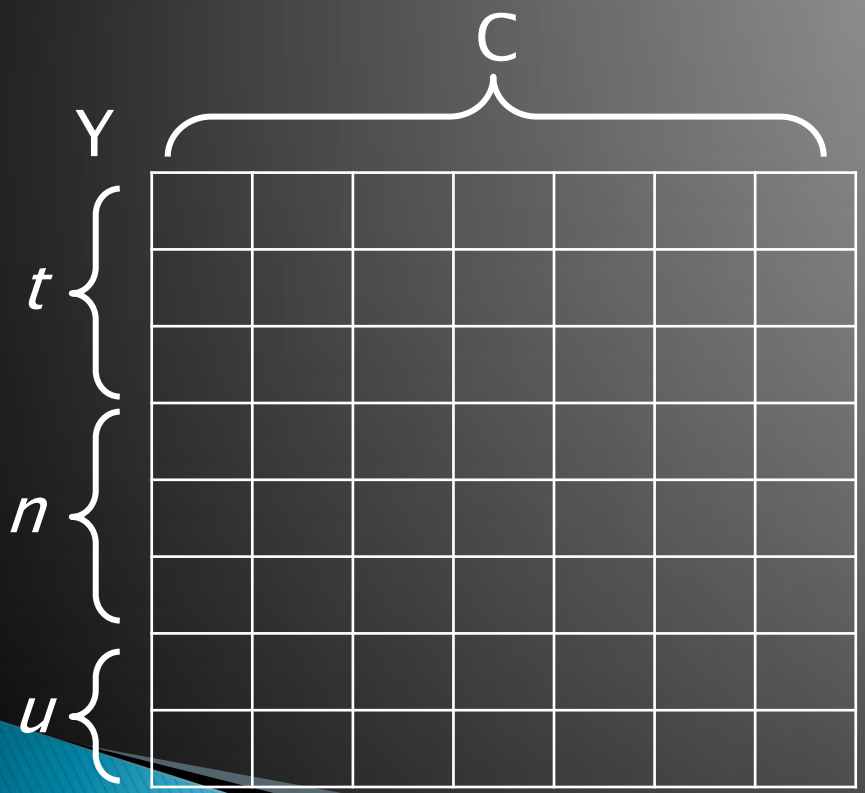
Algorithm NLP

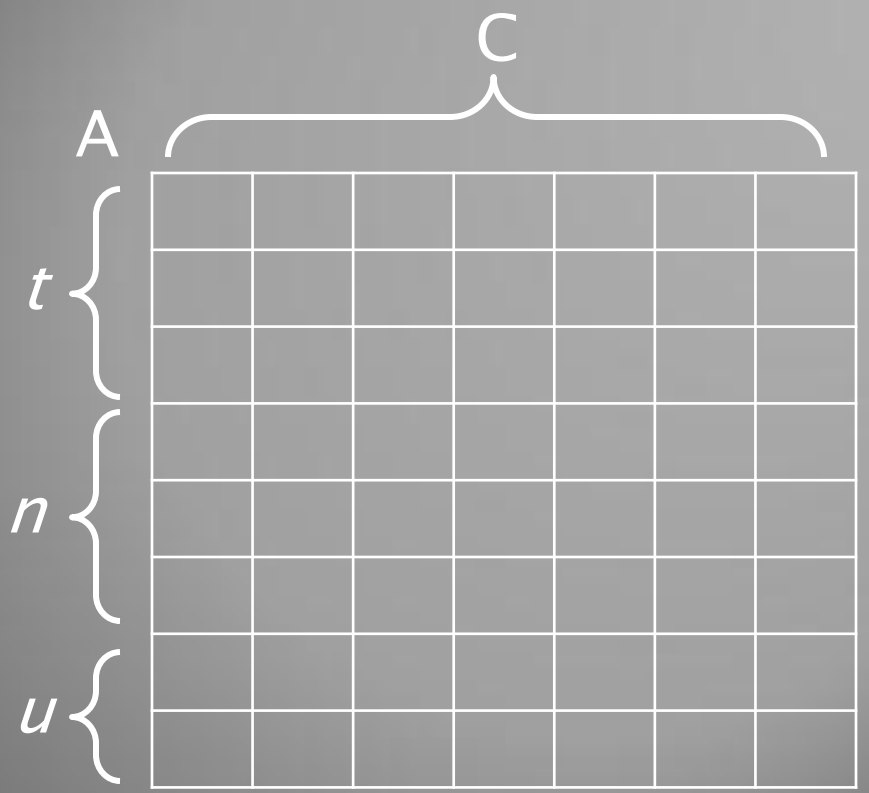
Some notation

- TL – true labels, $|\chi^{(T)}| = t$
- NL –negative labels, $|\chi^{(N)}| = n$
- UL – unlabeled data, $|\chi^{(U)}| = u$
- C – number of classes

Algorithm NLP

- Introduce two matrices, Y and A

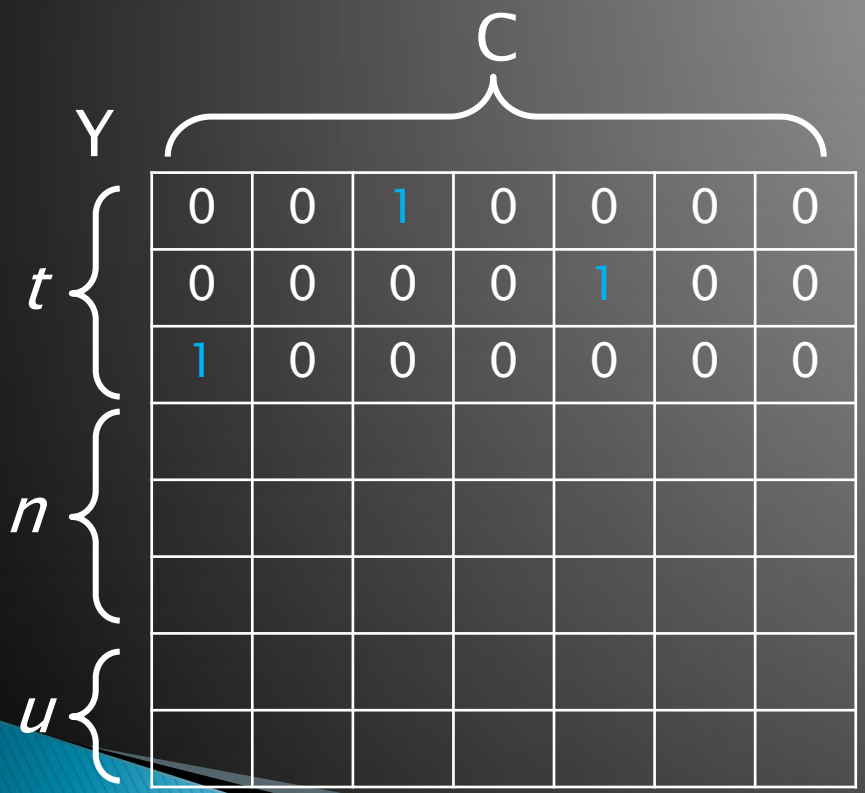


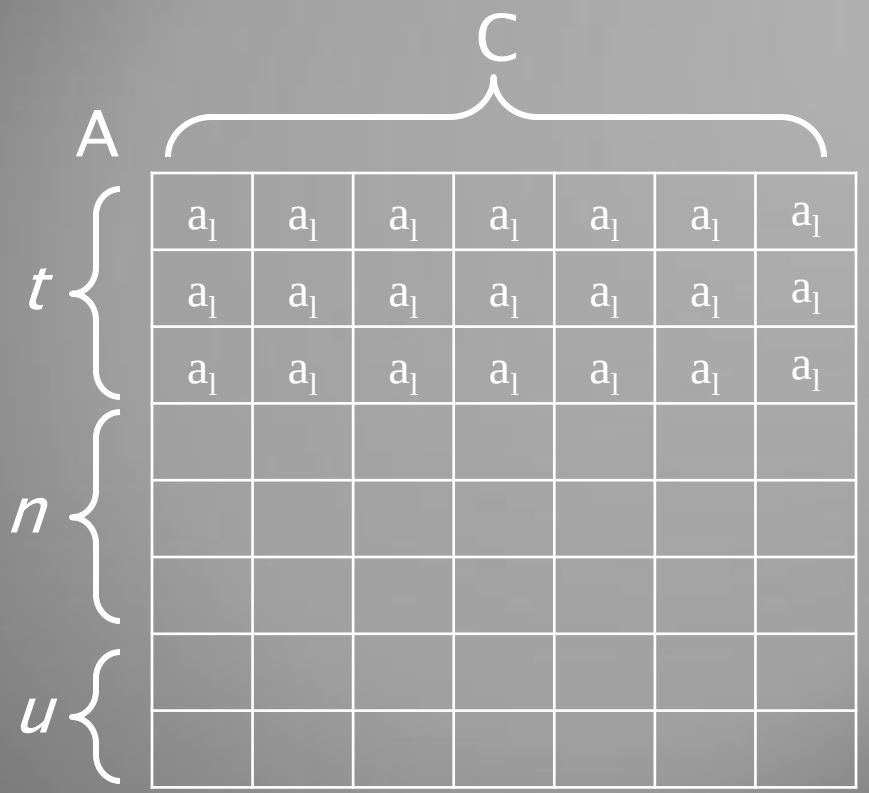


- a_l close to 0
- a_u close to 1

Algorithm NLP

- Introduce two matrices, Y and A

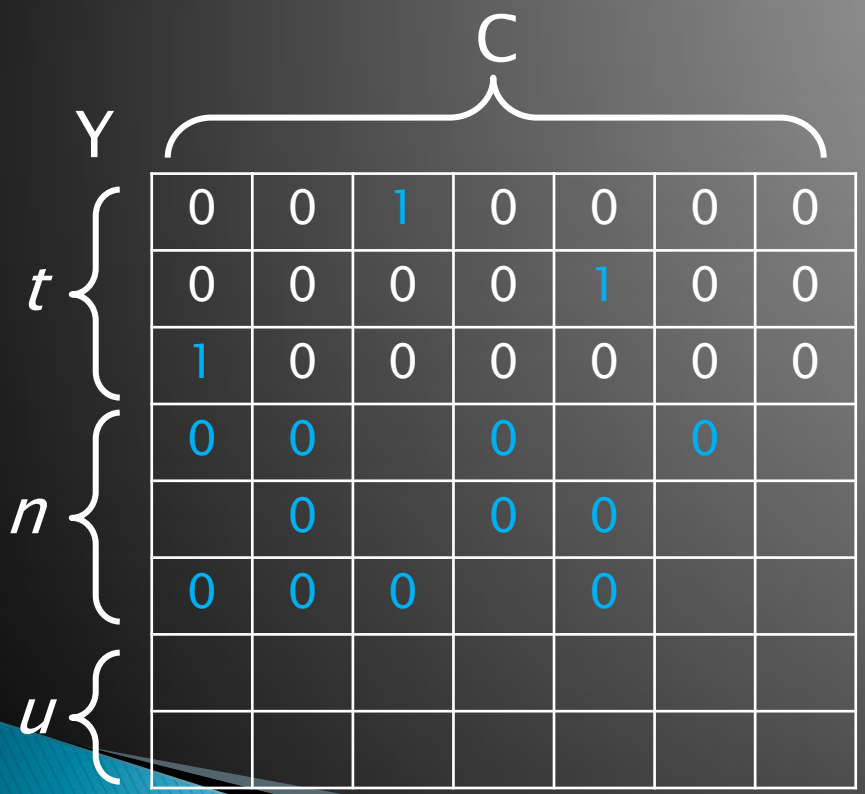


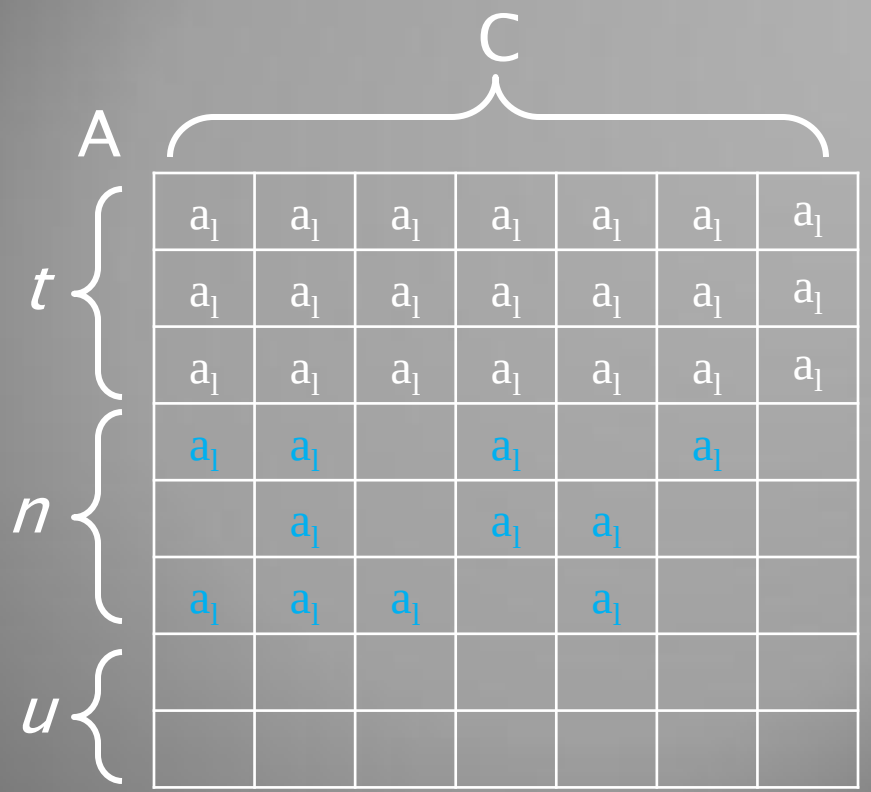


- a_l close to 0
- a_u close to 1

Algorithm NLP

- Introduce two matrices, Y and A





- a_l close to 0
- a_u close to 1

Algorithm NLP

- Introduce two matrices, Y and A

Y

C

0	0	1	0	0	0	0
0	0	0	0	1	0	0
1	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

t

n

u

A

C

a_l	a_l	a_l	a_l	a_l	a_l	a_l
a_l	a_l	a_l	a_l	a_l	a_l	a_l
a_l	a_l	a_l	a_l	a_l	a_l	a_l
a_l	a_l	a_u	a_l	a_u	a_l	a_u
a_u	a_l	a_u	a_l	a_l	a_u	a_u
a_l	a_l	a_l	a_u	a_l	a_u	a_u

t

n

u

- a_l close to 0
- a_u close to 1

Algorithm NLP

- Introduce two matrices, Y and A

Y

C

t	0	0	1	0	0	0	0
t	0	0	0	0	1	0	0
t	1	0	0	0	0	0	0
n	0	0	0	0	0	0	0
n	0	0	0	0	0	0	0
n	0	0	0	0	0	0	0
u	0	0	0	0	0	0	0
u	0	0	0	0	0	0	0

A

C

t	a_l	a_l	a_l	a_l	a_l	a_l	a_l
t	a_l	a_l	a_l	a_l	a_l	a_l	a_l
t	a_l	a_l	a_l	a_l	a_l	a_l	a_l
n	a_l	a_l	a_u	a_l	a_u	a_l	a_u
n	a_u	a_l	a_u	a_l	a_l	a_u	a_u
n	a_l	a_l	a_l	a_u	a_l	a_u	a_u
u	a_u	a_u	a_u	a_u	a_u	a_u	a_u
u	a_u	a_u	a_u	a_u	a_u	a_u	a_u

- a_l close to 0
- a_u close to 1

Algorithm NLP

- For each node x_i calculate edge weight w_{ij} for its k neighbors

$$W_{ij} = W_{ji} = \exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma^2}\right)$$

Algorithm NLP

- For each node x_i calculate edge weight w_{ij} for its k neighbors

$$W_{ij} = W_{ji} = \exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma^2}\right)$$

- Then normalize

$$\hat{W}_{ij} = \frac{W_{ij}}{\sqrt{d_i d_j}}$$

where $d_i = \sum_j W_{ij}$ for $i = 1, 2, \ldots, (t + n + u)$

Algorithm NLP

- Algorithm output is F matrix
- Classification of x_i is $\text{argmax}_j F_{ij}$

Algorithm NLP

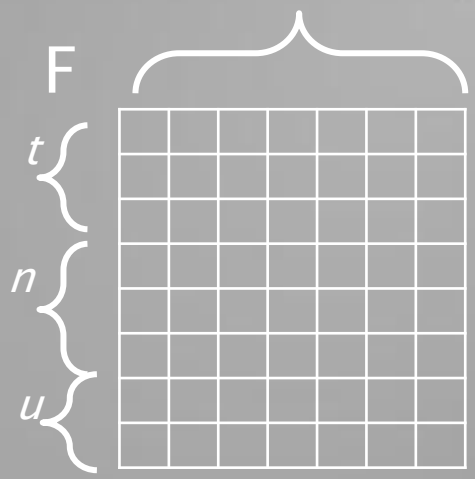
- Algorithm output is F matrix
- Classification of x_i is $\text{argmax}_j F_{ij}$
- Iterate, for step m+1, possibility of x_i belonging to the j th class

$$F_{ij}^{m+1} = A_{ij}\left(\frac{1}{\hat{d}_i}\right)\sum_{l=1}^{t+n+u}\hat{W}_{il}F_{lj}^{m} + (1 - A_{ij})Y_{ij}$$

where $\hat{d}_i = \sum_l \hat{W}_{il}$

Algorithm NLP

$$F_{ij}^{m+1} = A_{ij}\left(\frac{1}{\hat{d}_i}\right)\sum_{l=1}^{t+n+u}\hat{W}_{il}F_{lj}^{m} + (1 - A_{ij})Y_{ij}$$



C

Y

t	0	0	1	0	0	0	0
	0	0	0	0	1	0	0
	1	0	0	0	0	0	0
n	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
u	0	0	0	0	0	0	0
	0	0	0	0	0	0	0

C

A

t	a_l	a_l	a_l	a_l	a_l	a_l	a_l
	a_l	a_l	a_l	a_l	a_l	a_l	a_l
	a_l	a_l	a_l	a_l	a_l	a_l	a_l
n	a_l	a_l	a_u	a_l	a_u	a_l	a_u
	a_u	a_l	a_u	a_l	a_l	a_u	a_u
	a_l	a_l	a_l	a_u	a_l	a_u	a_u
u	a_u	a_u	a_u	a_u	a_u	a_u	a_u
	a_u	a_u	a_u	a_u	a_u	a_u	a_u

Algorithm NLP

Iterate, for step m+1, possibility of x_i belonging to the j th class

$$F_{ij}^{m+1} = A_{ij}\left(\frac{1}{\hat{d}_i}\right)\sum_{l=1}^{t+n+u}\hat{W}_{il}F_{lj}^{m} + (1 - A_{ij})Y_{ij}$$

where $\hat{d}_i = \sum_l \hat{W}_{il}$

- $I_{(j)}$ is a diagonal matrix with A_{1j} , A_{2j} , . . . , $A_{(t+n+u)j}$ on its diagonal line

Algorithm NLP

Iterate, for step m+1, possibility of x_i belonging to the j th class

$$F_{ij}^{m+1} = A_{ij}\left(\frac{1}{\hat{d}_i}\right)\sum_{l=1}^{t+n+u}\hat{W}_{il}F_{lj}^{m} + (1 - A_{ij})Y_{ij}$$

where $\hat{d}_i = \sum_l \hat{W}_{il}$

- $I_{(j)}$ is a diagonal matrix with A_{1j} , A_{2j} , . . . , $A_{(t+n+u)j}$ on its diagonal line
- $\hat{\Lambda}$ to be a diagonal matrix with $\hat{d}_i$ on diagonal

$$F_{.j}^{m+1} = I_{(j)}PF_{.j}^{m} + (I - I_{(j)})Y_{.j}$$

$$P = \hat{\Lambda}^{-1}\hat{W}$$

Convergence analysis

Theorem 1.

$F_j^0 = Y_j$

Update rule:

$$F_{.j}^{m+1} = I_{(j)} P F_{.j}^{m} + (I - I_{(j)}) Y_{.j}$$

Convergence analysis

Theorem 1.

$F_j^{\,0} = \mathrm{Y_j}$

Update rule:

$$F_{.j}^{m+1} = I_{(j)} P F_{.j}^{m} + (I - I_{(j)}) Y_{.j}$$

$$F_j^1 = I_{(j)} P F_j^0 + \left(I - I_{(j)}\right) Y_j = I_{(j)} P Y_j + \left(I - I_{(j)}\right) Y_j$$

Convergence analysis

Theorem 1.

$F_j^{\,0} = \mathrm{Y_j}$

Update rule:

$$F_{.j}^{m+1} = I_{(j)} P F_{.j}^{m} + (I - I_{(j)}) Y_{.j}$$

$$F_j^1 = I_{(j)} P F_j^0 + \left(I - I_{(j)}\right) Y_j = I_{(j)} P Y_j + \left(I - I_{(j)}\right) Y_j$$

$$F_j^2 = I_{(j)} P F_j^1 + \left(I - I_{(j)}\right) Y_j =$$

Convergence analysis

Theorem 1.

$F_j^{\,0} = \mathrm{Y_j}$

Update rule:

$$F_{.j}^{m+1} = I_{(j)} P F_{.j}^{m} + (I - I_{(j)}) Y_{.j}$$

$$F_j^1 = I_{(j)} P F_j^0 + \left(I - I_{(j)}\right) Y_j = I_{(j)} P Y_j + \left(I - I_{(j)}\right) Y_j$$

$$F_j^2 = I_{(j)} P F_j^1 + \left(I - I_{(j)}\right) Y_j =$$

$$= I_{(j)} P (I_{(j)} P Y_j + \left(I - I_{(j)}\right) Y_j) + \left(I - I_{(j)}\right) Y_j$$

Convergence analysis

Theorem 1.

$F_j^0 = Y_j$

Update rule:

$$F_{.j}^{m+1} = I_{(j)} P F_{.j}^{m} + (I - I_{(j)}) Y_{.j}$$

$$F_j^1 = I_{(j)} P F_j^0 + (I - I_{(j)}) Y_j = I_{(j)} P Y_j + (I - I_{(j)}) Y_j$$

$$F_j^2 = I_{(j)} P F_j^1 + (I - I_{(j)}) Y_j =$$

$$= I_{(j)} P (I_{(j)} P Y_j + (I - I_{(j)}) Y_j) + (I - I_{(j)}) Y_j$$

$$F_{.j}^{m+1} = (I_{(j)} P)^m Y_{.j} + (I - I_{(j)}) \sum_{i=1}^{m} (I_{(j)} P)^i Y_{.j}$$

Convergence analysis

$$F_{.j}^{m+1} = (I_{(j)}P)^m Y_{.j} + (I - I_{(j)}) \sum_{i=1}^{m} (I_{(j)}P)^i Y_{.j}$$

Convergence analysis

$$F_{.j}^{m+1} = (I_{(j)}P)^m Y_{.j} + (I - I_{(j)}) \sum_{i=1}^{m} (I_{(j)}P)^i Y_{.j}$$

Since $P_{ij} \geq 0$ and $\sum_j P_{ij} = 1$

Convergence analysis

$$F_{.j}^{m+1} = (I_{(j)}P)^m Y_{.j} + (I - I_{(j)}) \sum_{i=1}^{m} (I_{(j)}P)^i Y_{.j}$$

Since $P_{ij} \geq 0$ and $\sum_j P_{ij} = 1$

- spectral radius of P $\leq$ 1

Convergence analysis

$$F_{.j}^{m+1} = (I_{(j)}P)^m Y_{.j} + (I - I_{(j)}) \sum_{i=1}^{m} (I_{(j)}P)^i Y_{.j}$$

Since $P_{ij} \geq 0$ and $\sum_j P_{ij} = 1$

- spectral radius of $P \leq 1$
- $I_{(j)}$ elements $0 < a_l, a_u < 1$

Convergence analysis

$$F_{.j}^{m+1} = (I_{(j)}P)^m Y_{.j} + (I - I_{(j)}) \sum_{i=1}^{m} (I_{(j)}P)^i Y_{.j}$$

Since $P_{ij} \geq 0$ and $\sum_j P_{ij} = 1$

- spectral radius of $P \leq 1$
- $I_{(j)}$ elements $0 < a_l, a_u < 1$
- spectral radius $\rho(I_{(j)}P) < 1$

Convergence analysis

$$F_{.j}^{m+1} = (I_{(j)}P)^m Y_{.j} + (I - I_{(j)}) \sum_{i=1}^{m} (I_{(j)}P)^i Y_{.j}$$

Since $P_{ij} \geq 0$ and $\sum_j P_{ij} = 1$

- spectral radius of P $\leq$ 1
- $I_{(j)}$ elements $0 < a_l, a_u < 1$
- spectral radius $\rho(I_{(j)}P) < 1$

$$\lim_{m \to \infty} (I_{(j)}P)^m = 0$$

Convergence analysis

$$F_{.j}^{m+1} = (I_{(j)}P)^m Y_{.j} + (I - I_{(j)}) \sum_{i=1}^{m} (I_{(j)}P)^i Y_{.j}$$

Since $P_{ij} \geq 0$ and $\sum_j P_{ij} = 1$

- spectral radius of P $\leq$ 1
- $I_{(j)}$ elements $0 < a_l, a_u < 1$
- spectral radius $\rho(I_{(j)}P) < 1$

$$\lim_{m \to \infty} (I_{(j)}P)^m = 0$$

$$\lim_{m \to \infty} \sum_{i=1}^{m} (I_{(j)}P)^i = (I - I_{(j)}P)^{-1}$$

Convergence analysis

$$F_{.j}^{m+1} = (I_{(j)}P)^m Y_{.j} + (I - I_{(j)}) \sum_{i=1}^{m} (I_{(j)}P)^i Y_{.j}$$

Since $P_{ij} \geq 0$ and $\sum_j P_{ij} = 1$

- spectral radius of P ≤ 1
- $I_{(j)}$ elements $0 < a_l, a_u < 1$
- spectral radius $\rho(I_{(j)}P) < 1$

$$\lim_{m \to \infty} (I_{(j)}P)^m = 0$$

$$\lim_{m \to \infty} \sum_{i=1}^{m} (I_{(j)}P)^i = (I - I_{(j)}P)^{-1}$$

$$F_{.j}^{*} = (I - I_{(j)})(I - I_{(j)}P)^{-1} Y_{.j}$$

Regularization Interpretation

Theorem 2.

$$F^*_{\cdot j} = (I - I_{(j)})(I - I_{(j)}P)^{-1}Y_{\cdot j}$$

is the solution to

Regularization Interpretation

Theorem 2.

$$F^*_{\cdot j} = (I - I_{(j)})(I - I_{(j)}P)^{-1}Y_{\cdot j}$$

is the solution to

minimize $\mathcal{L}(F_{\cdot j}) = F_{\cdot j}^T \hat{L} F_{\cdot j} + (F_{\cdot j} - Y_{\cdot j})^T \hat{\Lambda} U_j (F_{\cdot j} - Y_{\cdot j})$

Regularization Interpretation

Theorem 2.

$$F^*_{\cdot j} = (I - I_{(j)})(I - I_{(j)}P)^{-1}Y_{\cdot j}$$

is the solution to

minimize $\mathcal{L}(F_{\cdot j}) = F_{\cdot j}^T \hat{L} F_{\cdot j} + (F_{\cdot j} - Y_{\cdot j})^T \hat{\Lambda} U_j (F_{\cdot j} - Y_{\cdot j})$

$$\mathcal{L}(F_{\cdot j}) = \frac{1}{2} \sum_{i=1}^{t+u+n} \hat{W}_{ji} \| F_{\cdot j} - F_{\cdot i} \|^2 + \sum_{i=1}^{t+u+n} \hat{d}_i \left(\frac{1}{A_{ij}} \right) - 1(F_{ij} - Y_{ij})^2$$

Experiments

Name	# samples	# features	# classes
Coil (image)	1440	32x32	20
Umist (face)	575	23x28	20
USPS (digit)	9298	16x16	10
Isolet5	1559	617	26
Newsgroup	3970	8014	4
Corel (image)	1000	36	10

Experiments

Algorithms:

- SVM – supervised, uses only TL for training

Experiments

Algorithms:

- SVM – supervised, uses only TL for training
- LapRLS – different forms of loss and smoothness functions, treat NL as UL

Experiments

Algorithms:

- SVM – supervised, uses only TL for training
- LapRLS – different forms of loss and smoothness functions, treat NL as UL
- GRF – doesn't use normalized weights, treat NL as UL

Experiments

Algorithms:

- SVM – supervised, uses only TL for training
- LapRLS – different forms of loss and smoothness functions, treat NL as UL
- GRF – doesn't use normalized weights, treat NL as UL
- Consistency – normalized weights, treat NL as UL

Experiments

Algorithms:

- SVM – supervised, uses only TL for training
- LapRLS – different forms of loss and smoothness functions, treat NL as UL
- GRF – doesn't use normalized weights, treat NL as UL
- Consistency – normalized weights, treat NL as UL
- LNP – instead of weights uses local linear approximation coefficients, treat NL as UL

Experiments

NL instances:

x_i	1	0	0	1	0	0	0

Experiments

NL instances:

x_i	1	0	0	1	0	0	0

x_i	1	0	0	1	0	1	0

Experiments

NL instances:

x_i	1	0	0	1	0	0	0

x_i	1	0	0	1	0	1	0

x_i	1	1	0	1	1	1	1

Experiments

NL instances:

x_i	1	0	0	1	0	0	0

x_i	1	0	0	1	0	1	0

TL instances:

x_i	0	0	1	0	0	0	0

Experiments

NL instances:

x_i	1	0	0	1	0	0	0

x_i	1	0	0	1	0	1	0

x_i	1	1	0	1	1	1	1

The more NL/point is given, the more certainty about the classification

Experiment I

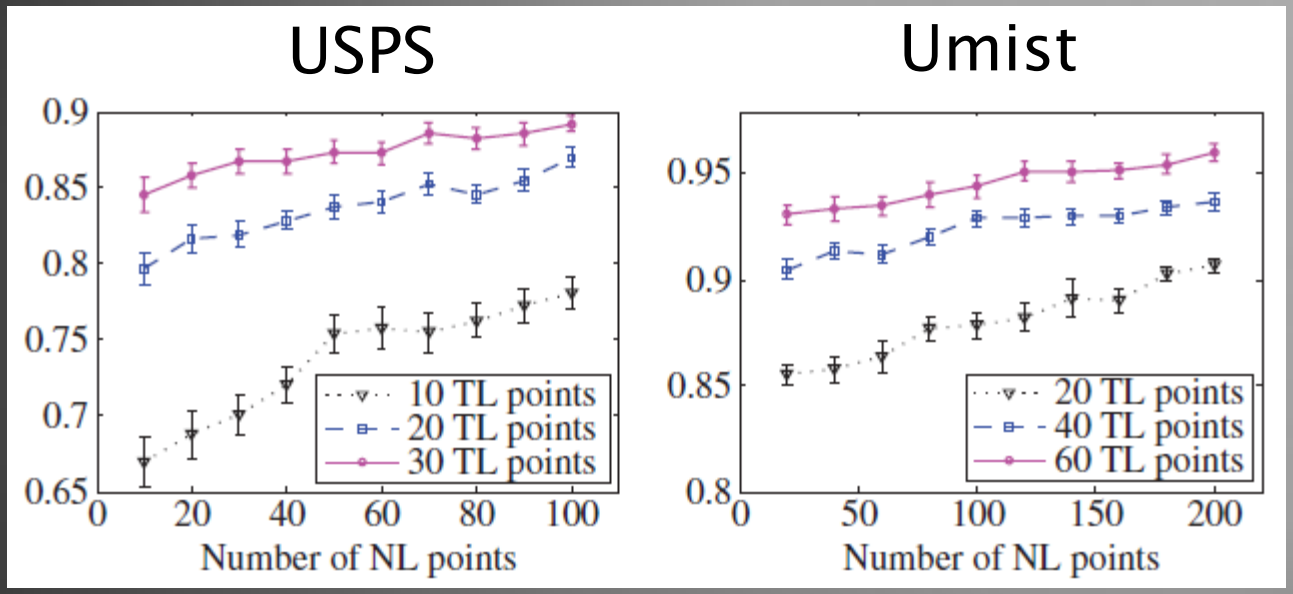
Name	# NL points	# NL/point
Coil (image)	10	5
Umist (face)	10	5
USPS (digit)	10	5
Isolet5	10	5
Newsgroup	50	2

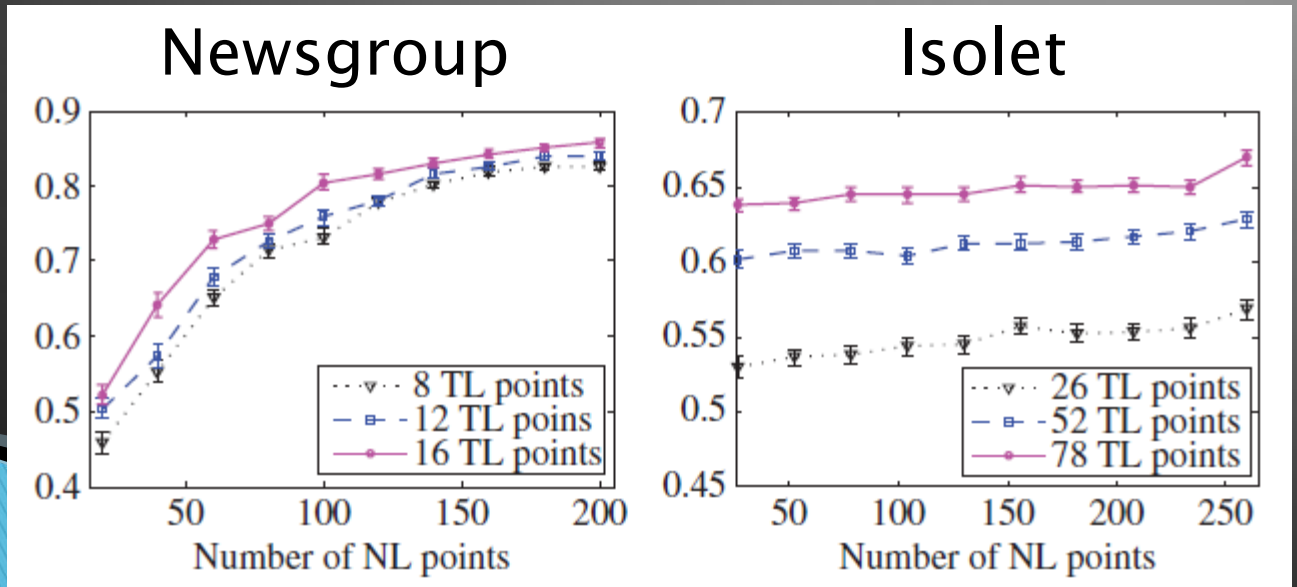
Experiment I

USPS DATASET (MEAN ± STD/%)

t	SVM	LapRLS	GRF	Consist	LNP	NLP
10	41.49±5.72	59.06±3.26	62.14±4.32	74.70±3.47	75.49±3.89	**78.11**±2.44
20	52.39±4.59	67.72±3.62	77.30±3.99	82.98±3.06	83.48±3.91	**85.50**±2.75
30	58.61±4.25	72.69±3.48	84.84±3.97	87.02±2.07	87.52±3.79	**89.32**±2.13
40	60.51±4.20	76.19±2.05	86.62±32.6	88.44±2.46	88.73±2.41	**90.31**±1.48
50	62.74±3.88	77.33±2.53	88.94±2.01	89.00±2.45	89.73±2.65	**91.13**±1.47
60	63.01±4.47	78.87±2.10	89.47±1.36	89.71±1.32	89.77±2.90	**91.91**±0.99

Experiment II





Experiment III

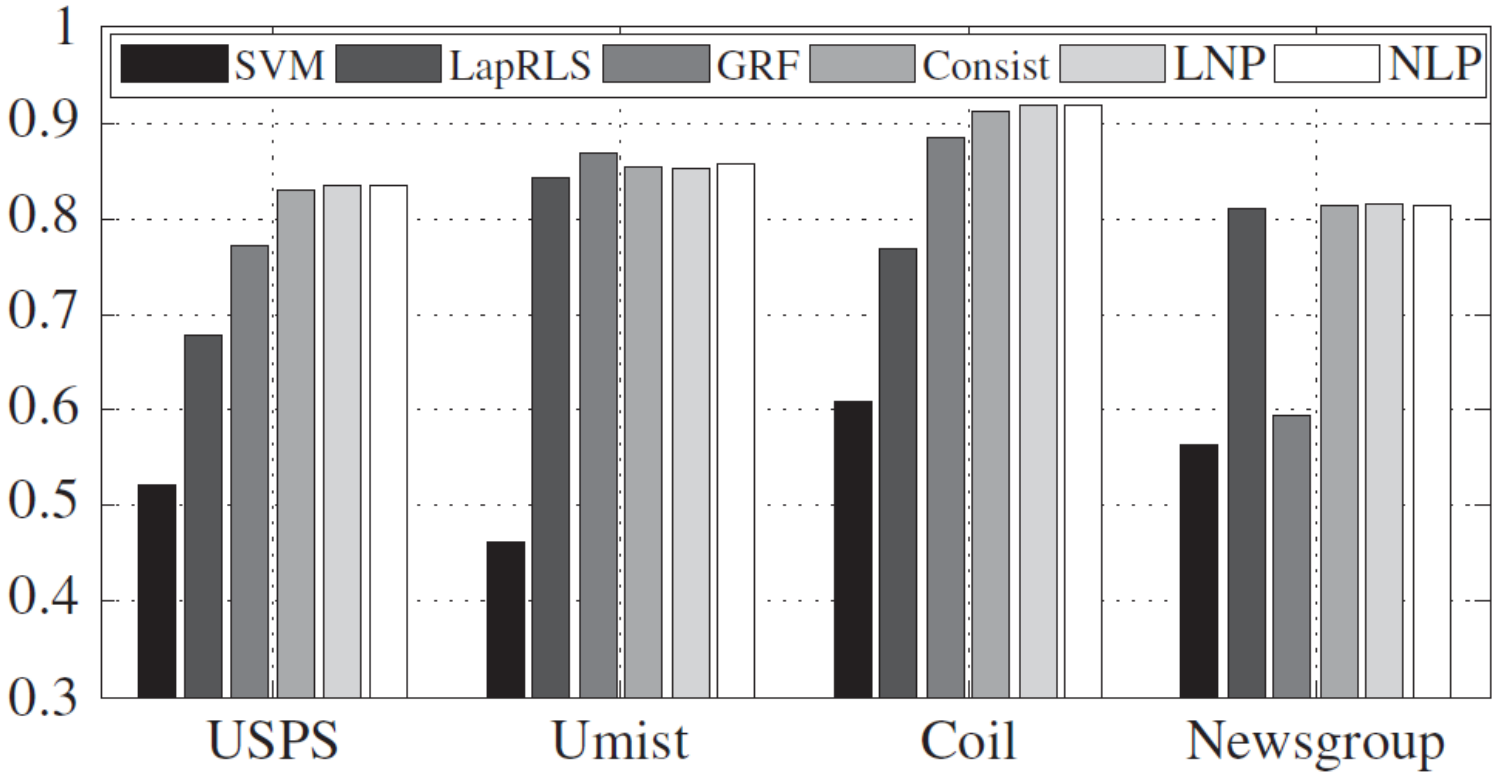


Fig. 3. Classification accuracies of different methods when $t = 20$ and $n = 0$.

Experiment IV

USPS Data. Totally 480 NLs.

Numbers of NL Points Are 0, 60, 80, 120, and 240, Each Point Has 0, 8, 6, 4, and 2 NLs, Respectively.

t	$n = 0$	$n = 60$	$n = 80$	$n = 120$	$n = 240$
10	62.92±5.39	80.10±4.71	78.76±4.33	77.57±4.61	75.86±4.74
20	77.63±3.63	87.88±3.27	87.26±3.94	86.13±3.40	85.47±3.50
40	84.86±3.92	88.92±3.16	89.00±2.77	88.94±2.41	88.78±2.58

Parameter Determination

Four inputs: σ, k, a_l, a_u

Parameter Determination

Four inputs: σ, k, a_l, a_u

$$\exp\left(-\frac{(\bar{d})^2}{2\sigma^2}\right) = \frac{\tau}{k} \Rightarrow \sigma = \frac{1}{2}\sqrt{-\frac{(\bar{d})^2}{\ln(\tau/k)}}.$$

- $\tau \in (0, 1)$; $\bar{d}$ the average of all distances

Parameter Determination

Four inputs: σ, k, a_l, a_u

$$\exp\left(-\frac{(\bar{d})^2}{2\sigma^2}\right) = \frac{\tau}{k} \Rightarrow \sigma = \frac{1}{2}\sqrt{-\frac{(\bar{d})^2}{\ln(\tau/k)}}.$$

- $\tau \in (0, 1)$; $\bar{d}$ the average of all distances

- a_l and a_u by searching the grid {0.01, 0.02, . . . , 0.09} and {0.90, 0.91, . . . , 0.99}

Parameter Determination

Four inputs: σ, k, a_l, a_u

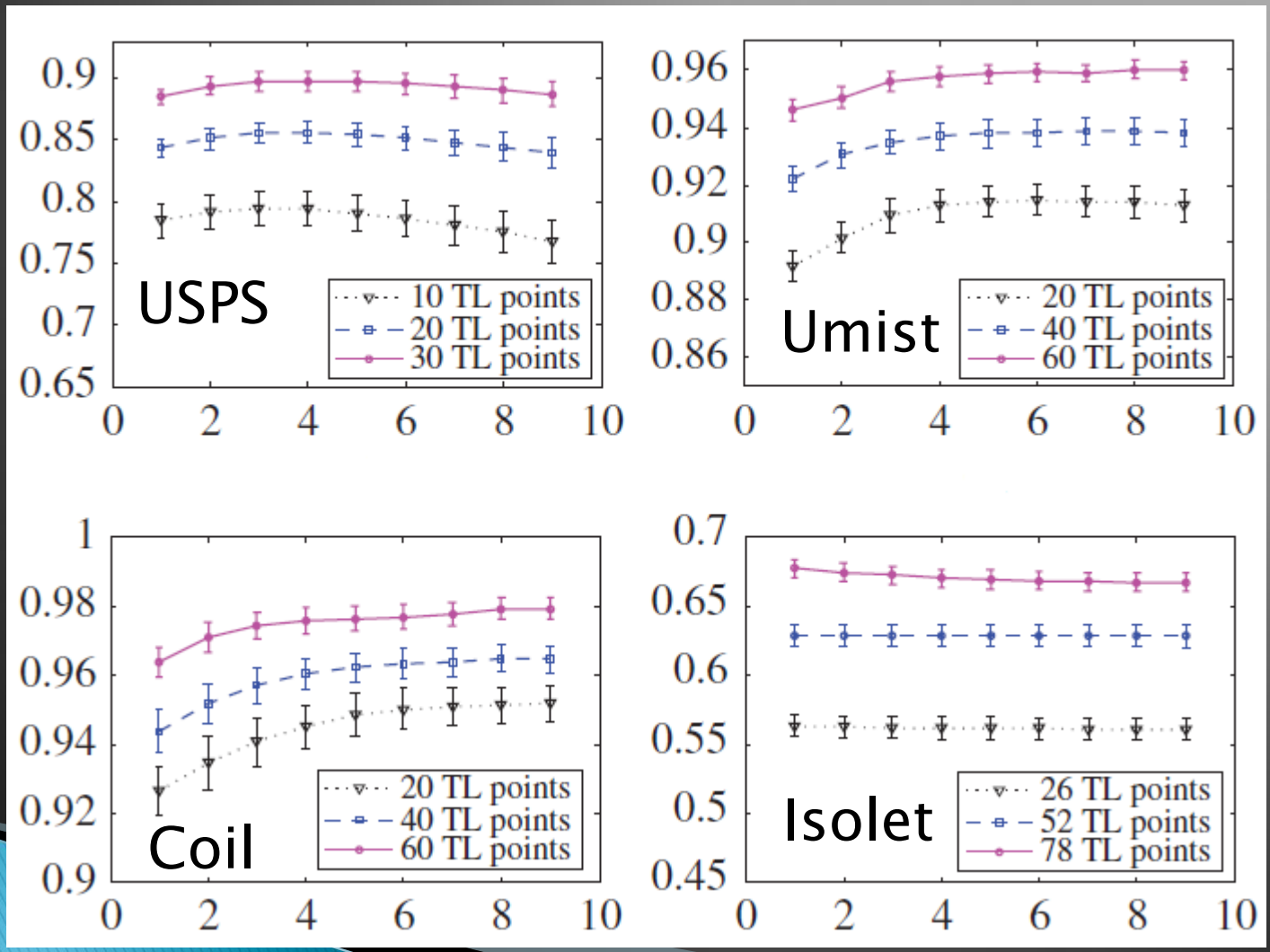
$$\exp\left(-\frac{(\bar{d})^2}{2\sigma^2}\right) = \frac{\tau}{k} \Rightarrow \sigma = \frac{1}{2}\sqrt{-\frac{(\bar{d})^2}{\ln(\tau/k)}}.$$

- $\tau \in (0, 1)$; $\bar{d}$ the average of all distances

- a_l and a_u by searching the grid {0.01, 0.02, . . . , 0.09} and {0.90, 0.91, . . . , 0.99}
- k – ?

Parameter Determination

Ex: i for Isolet, $\tau = 10^{(i-1)}/10$ for $i = 1, 2, \ldots, 9$



Out-Of-Sample Extension

- z is a new point (unlabeled)

$$F_z = \sum_{i:x_i \in \mathcal{N}(z)} \hat{W}(z, x_i) F^*_{i.}$$

- z's label is defined by weighted labels of its neighbors

Computational Complexity

- Time complexity is $O((t + n + u)^2 D)$, where D is the dimensionality of the samples
- Space complexity is $O((t + n + u)^2)$

Strength & Weaknesses

- Variable information incorporation of different type labels (NL vs. UL)

Strength & Weaknesses

- Variable information incorporation of different type labels (NL vs. UL)

- Too much of parameter tuning, i.e. σ, k, a_l, a_u

Strength & Weaknesses

\+ Variable information incorporation of different type labels (NL vs. UL)

— Too much of parameter tuning, i.e. σ, k, a_l, a_u

— Negative labeling defeats the purpose of SSL, i.e. annotating negative labels can be even more time consuming than annotating "positive" labels

References

- Introduction to Semi-Supervised Learning. Xiaojin Zhu, Andrew B. Goldberg. 2007
- Semisupervised Learning Using Negative Labels. Chenping Hou, Feiping Nie, Fei Wang, Changshui Zhang, Yi Wu. IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 22, NO. 3, MARCH 2011.
- Humans Perform Semi-Supervised Classification Too. Xiaojin Zhu, Timothy Rogers, Ruichen Qian, Chuck Kalish. Twenty-Second AAAI Conference on Artificial Intelligence (AAAI-07). 2007.
- Unlabeled data: Now it helps, now it doesn't. Aarti Singh, Robert D. Nowak, Xiaojin Zhu. Advances in Neural Information Processing Systems (NIPS) 22. 2008.
- Manifold Regularization: A Geometric Framework for Learning from Examples. Mikhail Belkin, Partha Niyogi, Vikas Sindhwani. 2004