

Neighborhood-Based Classification

Exploiting Manifold Information in Statistical Learning

Xuejun Liao, Qiuhua Liu, Chunping Wang, and Lawrence Carin

{xjliao, ql1, cw36, lcarin}@ee.duke.edu

Department of Electrical and Computer Engineering

Duke University

Durham, NC 27708-0291, USA

Outline

- Problem Statement
- Markov random walk on a data manifold
- Neighborhood-Based Classification (NeBC)
 - The *nonparametric* case
 - The *parametric* case
 - The *semi-parametric* case (Dependent Dirichlet distribution)
- Experimental Results
- Summary

Problem Statement

- Given: a discrete data manifold $\mathcal{X} = \{\mathbf{x}_i : i = 1, 2, \dots, N\}$, of which small subset indexed by \mathcal{L} is labeled
- Objective:
 - complete the labels for \mathcal{X} (transduction)
 - predict the labels for data outside of \mathcal{X} (induction)
- Problems
 - Scarce labels, insufficient supervision, over-fitting
 - Data manifold unexploited
 - Lack of prior in choosing classifiers
- Basic approach
 - Modeling \mathcal{X} as a graph, data connected by Markov random walk
 - Labeling a data point based on its neighborhood
 - Using Dirichlet distributions (DD) as nonparametric classifiers

Markov Random Walk

- Graph model of \mathcal{X}
 - $G = (\mathcal{X}, \mathbf{W})$
 - $\mathbf{W} = [w_{ij}]_{N \times N}$ is the affinity matrix, with w_{ij} the strength of immediate connection between \mathbf{x}_i and \mathbf{x}_j

$$w_{ij} = \exp\left(\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2d_i^2}\right) \quad (1)$$

- Markov random walk with one-step transitions

$$T_{ij} = p(\mathbf{x}_j | \mathbf{x}_i) = \begin{cases} \frac{W_{ij}}{\sum_{k \neq i} W_{ik}}, & j \neq i \\ 0, & j = i \end{cases} \quad (2)$$

d_i is roughly the stepsize at \mathbf{x}_i

Nonparametric NeBC (1/3)

- Let $g_{ik} = p(y_i = k | \mathbf{x}_i)$ and $\mathbf{g}_i = [g_{i1}, g_{i2}, \dots, g_{iK}]$, then

$$y_i | \mathbf{g}_i \sim \text{Mult}(y_i; \mathbf{g}_i) = \prod_{k=1}^K (g_{ik})^{\delta(y_i - k)} \quad (3)$$

- Finite-steps random walks:

Let $0 < \gamma < 1$ be a discount factor of the neighbors. Define

$$\mathbf{g}_i^{(n)} = (1 - \gamma)\mathbf{g}_i^* + \gamma \sum_{j=1}^N T_{ij} \mathbf{g}_j^{(n-1)}, \quad i = 1, 2, \dots, N \quad (4)$$

$$\mathbf{G}^{(n)} = (1 - \gamma)\mathbf{G}^* + \gamma \mathbf{T} \mathbf{G}^{(n-1)} \quad (5)$$

where $\mathbf{G} = \begin{bmatrix} \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_N \end{bmatrix}$ and $\mathbf{G}^* = \begin{bmatrix} \mathbf{g}_1^* \\ \vdots \\ \mathbf{g}_N^* \end{bmatrix}$

Nonparametric NeBC (2/3)

Table 1: The Label Iteration Algorithm

Input: \mathbf{T}, γ, n^*

Output: $\mathbf{G}^{(n^*)}$

Initialization: $g_{ik}^{(0)} = \begin{cases} 1, & \mathbf{x}_i \text{ labeled, and } y_i = k \\ 0, & \mathbf{x}_i \text{ labeled, and } y_i \neq k \\ 1/K, & \mathbf{x}_i \text{ unlabeled} \end{cases}$

Label Iteration: Repeat until $n = n^*$,
 $\mathbf{G}^{(n)} = (1 - \gamma)\mathbf{G}^* + \gamma\mathbf{T}\mathbf{G}^{(n-1)}$

Nonparametric NeBC (3/3)

- Convergence of Label Iteration:

$$\begin{aligned}\|\mathbf{G}^{(n+1)} - \mathbf{G}^{(n)}\| &= \|\gamma\mathbf{T}\mathbf{G}^{(n)} - \gamma\mathbf{T}\mathbf{G}^{(n-1)}\| \\ &\leq \gamma\|\mathbf{T}\|\|\mathbf{G}^{(n)} - \mathbf{G}^{(n-1)}\| \\ &\leq \|\mathbf{G}^{(n)} - \mathbf{G}^{(n-1)}\|\end{aligned}\tag{6}$$

since $\gamma < 1$ and $\|\mathbf{T}\| \leq 1$.

- Infinite-steps random walks ($0 < \gamma < 1$):

$$\mathbf{G}^{(\infty)} = (1 - \gamma)(\mathbf{I} - \gamma\mathbf{T})^{-1}\mathbf{G}^*\tag{7}$$

- Let $[b_{ij}]_{N \times N} = \mathbf{B} = (1 - \gamma)(\mathbf{I} - \gamma\mathbf{T})^{-1}$, then $b_{ij} \geq 0$ and $\sum_{j=1}^N b_{ij} = 1$.

Parametric NeBC (1/2)

- Let $\sigma_{ik} = p(y_i = k | \mathbf{x}_i, \boldsymbol{\theta})$ be a classifier parameterized by $\boldsymbol{\theta}$, $\boldsymbol{\sigma}_i = [\sigma_{i1}, \dots, \sigma_{iK}]$, and

$$y_i | \mathbf{g}_i \sim \text{Mult}(y_i; \mathbf{g}_i) = \prod_{k=1}^K (g_{ik})^{\delta(y_i - k)} \quad (8)$$

$$\mathbf{g}_i = \sum_{j=1}^N b_{ij} \boldsymbol{\sigma}_j \quad (9)$$

where $[b_{ij}]_{N \times N} = \mathbf{B} = (1 - \gamma)(\mathbf{I} - \gamma \mathbf{T})^{-1}$.

- In the case the classifier is a multinomial logistic regression,

$$\sigma_{ik} = \frac{\exp(\boldsymbol{\theta}_k^T \mathbf{x}_i)}{\sum_{j=1}^K \exp(\boldsymbol{\theta}_j^T \mathbf{x}_i)}.$$

Parametric NeBC (2/2)

- Estimating θ by maximizing the likelihood of observed labels

$$\begin{aligned} p(\{y_i : i \in \mathcal{L}\} | \mathcal{X}, \theta) &= \prod_{i \in \mathcal{L}} \prod_{k=1}^K \left(\sum_{j=1}^N b_{ij} \sigma_{jk} \right)^{\delta(y_i - k)} \\ &= \prod_{i \in \mathcal{L}} \sum_{j=1}^N b_{ij} p(y_j = y_i | \mathbf{x}_j, \theta) \end{aligned} \quad (10)$$

- The Expectation-Maximization (EM) algorithm:

$$\begin{aligned} \text{E - step : } & \begin{cases} \psi_{ij} = \frac{b_{ij} p(y_j = y_i | \mathbf{x}_j, \theta^n)}{\sum_{j=1}^N b_{ij} p(y_j = y_i | \mathbf{x}_j, \theta^n)} \\ Q(\theta | \theta^n) = \sum_{i \in \mathcal{L}} \sum_{j=1}^N \psi_{ij} \ln p(y_j = y_i | \mathbf{x}_j, \theta) \end{cases} \\ \text{M - step : } & \theta^{n+1} = \theta^n + c [\nabla^2 Q(\theta | \theta^n)]^{-1} \nabla Q(\theta | \theta^n) \end{aligned} \quad (11)$$

Semi-parametric NeBC (1/6)

- Let $g_{ik} = p(y_i = k | \mathbf{x}_i)$, $\mathbf{g}_i = [g_{i1}, g_{i2}, \dots, g_{iK}]$, then

$$y_i | \mathbf{g}_i \sim \text{Mult}(y_i; \mathbf{g}_i) = \prod_{k=1}^K (g_{ik})^{\delta(y_i - k)} \quad (12)$$

- \mathbf{g}_i is governed by a dependent Dirichlet distribution (DDD) defined by

$$\mathbf{g}_i = \sum_{j=1}^N b_{ij} \mathbf{g}_j^* \quad (13)$$

$$[b_{ij}]_{N \times N} = \mathbf{B} = (1 - \gamma)(\mathbf{I} - \gamma \mathbf{T})^{-1} \quad (14)$$

$$\mathbf{g}_i^* \sim \text{Dir}(\mathbf{g}_i^*; \alpha \boldsymbol{\sigma}_i) = \frac{\Gamma(\sum_{k=1}^K \alpha \sigma_{ik})}{\prod_{k=1}^K \Gamma(\alpha \sigma_{ik})} \prod_{k=1}^K (g_{ik}^*)^{\alpha \sigma_{ik} - 1} \quad (15)$$

where $\alpha > 0$, $\boldsymbol{\sigma}_i = [\sigma_{i1}, \dots, \sigma_{iK}]$, and $\sigma_{ik} = p(y_i = k | \mathbf{x}_i)$ is the base classifier at location \mathbf{x}_i ; the base classifier can be parameterized.

Semi-parametric NeBC (2/6)

- Let $z_i = j$ indicates $\mathbf{g}_i = \mathbf{g}_j^*$, then

$$y_i | z_i, \mathbf{g}^* \sim \text{Mult}(y_i; \mathbf{g}_{z_i}^*) = \prod_{k=1}^K (g_{z_i k}^*)^{\delta(y_i - k)} \quad (16)$$

- Conditional on z , we can obtain the full conditionals

$$y_n | \{y_i\}_{i \neq n}^N, z \sim \frac{\alpha \sigma_{z_n y_n} + \sum_{i \neq n}^N \delta(z_i - z_n) \delta(y_i - y_n)}{\alpha + \sum_{i \neq n}^N \delta(z_i - z_n)} \quad (17)$$

Semi-parametric NeBC (3/6)

- Marginalizing z , using $p(z_i = j) = b_{ij}$, we get

$$y_n | \{y_i\}_{i \neq n}, \mathbf{b} \sim \sum_{j=1}^N \rho_{nj} \sigma_j y_n + \sum_{l \neq n}^N \lambda_{nl} \delta(y_n - y_l) \quad (18)$$

$$\rho_{nj} = b_{nj} \sum_{m=0}^{N-1} \frac{\alpha p(\sum_{i \neq n}^N \delta(z_i - j) = m)}{\alpha + m} \quad (19)$$

$$\lambda_{nl} = \sum_{j=1}^N b_{nj} b_{lj} \sum_{m=0}^{N-2} \frac{p(\sum_{i \neq n, l}^N \delta(z_i - j) = m)}{\alpha + m + 1} \quad (20)$$

with $\sum_{j=1}^N \rho_{nj} + \sum_{l \neq n}^N \lambda_{nl} = 1$.

- $\lim_{\alpha \rightarrow \infty} \rho_{nj} = b_{nj}$, $\lim_{\alpha \rightarrow \infty} \lambda_{nl} = 0$. However $\lim_{\alpha \rightarrow 0} \rho_{nj} > 0$.

Semi-parametric NeBC (4/6)

- Any parameterized classifier can be used as the base classifier
- For example, the base can be the parametric classifier discussed earlier
- With an inaccurate base classifier, the DDD can give improved prediction of the missing labels by exploiting the data manifold
- Open questions
 - How to learn the base classifier using the exact formulation of the DDD?
 - Why does the base not vanish when α goes to zero?

Semi-parametric NeBC (5/6)

- **Fast Algorithm to compute** $p(\sum_{i \neq n}^N \delta(z_i - j) = m)$

Let us state the problem in a general context. Assume $\mathbf{p} = [p_1, p_2, \dots, p_n]$ and $\mathbf{q} = [q_1, q_2, \dots, q_n]$ are two vectors. We want to decompose $(p_1 + q_1)(p_2 + q_2) \cdots (p_n + q_n)$ into a sum of $n + 1$ terms such that the m -th term involves m and only m p 's, for $m = 0, 1, 2, \dots, n$.

We call this problem the *non-stationary* triangle problem, since it specializes to the classic Pascal triangle when $\mathbf{p} = \mathbf{q}$ are vectors of all 1's. We give a fast algorithm in Table 2, to solve the *non-stationary* triangle problem.

Semi-parametric NeBC (6/6)

Table 2: The Non-stationary Triangle Algorithm

Input: $\mathbf{p} = [p_1, p_2, \dots, p_n]$
Output: $\mathbf{f} = [f_0, f_2, \dots, f_n]$

$\mathbf{q} = \mathbf{1} - \mathbf{p};$
 $\mathbf{f} = [q_1, p_1];$
for $i = 2$ **to** n
 $\mathbf{f}_{\text{new}} = [q_i \mathbf{f}, 0] + [0, p_i \mathbf{f}];$
 $\mathbf{f} = \mathbf{f}_{\text{new}};$
endfor

Results on Two Helixes

- Algorithm tested: Nonparametric NeBC (Label Iteration)
- movie of label-iteration

Results on Benchmark Data

- Data sets: Pima, Ionosphere, and WDBC
- Training on both labeled and unlabeled data;
- Transduction: testing on unlabeled data seen in training
- Induction: testing on unseen data
- Algorithms being compared:
 - The proposed parametric NeBC
 - The transductive SVM (Joachims, 1999)
 - The algorithm of Szummer & Jaakkola (Szummer & Jaakkola, 2002)
 - GRF (Zhu et. al, 2003)
 - Logistic GRF (Krishnapuram et. al, 2005)

Transduction with parametric NeBC: Benchmark Data

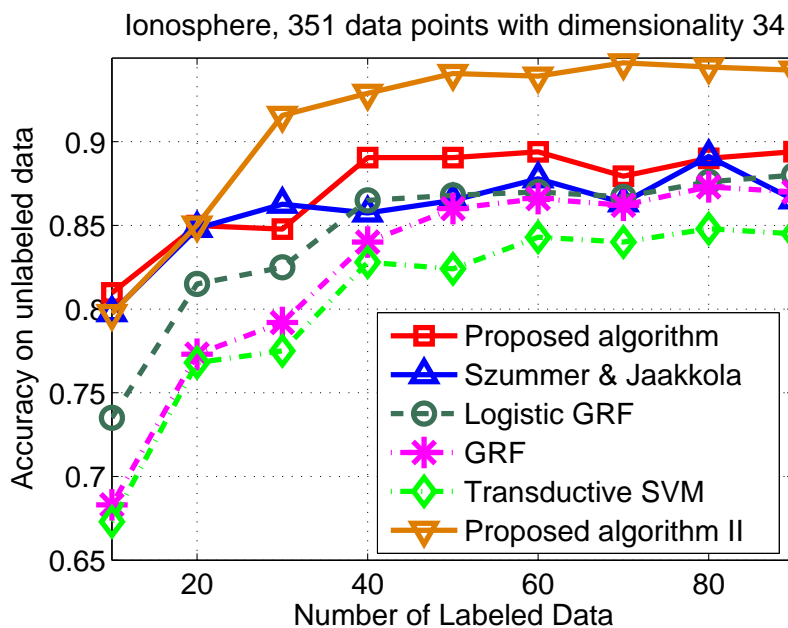
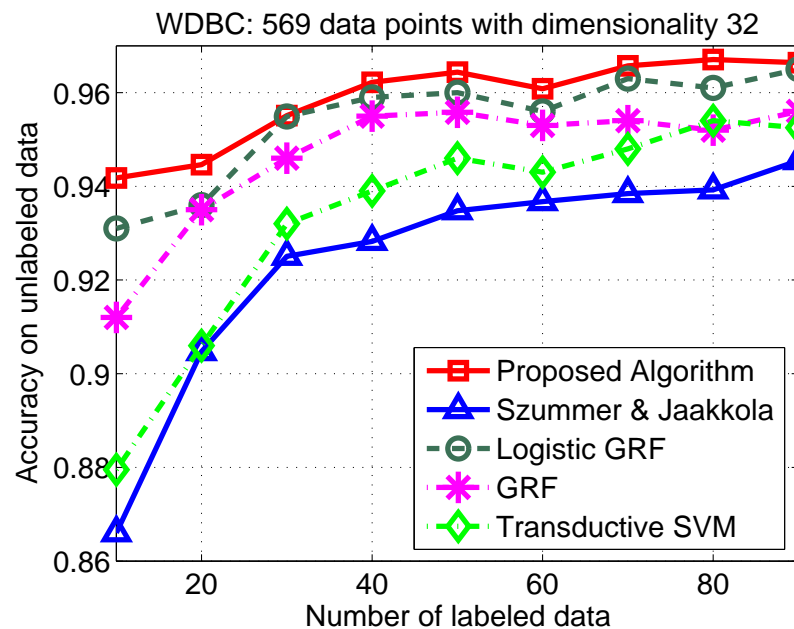
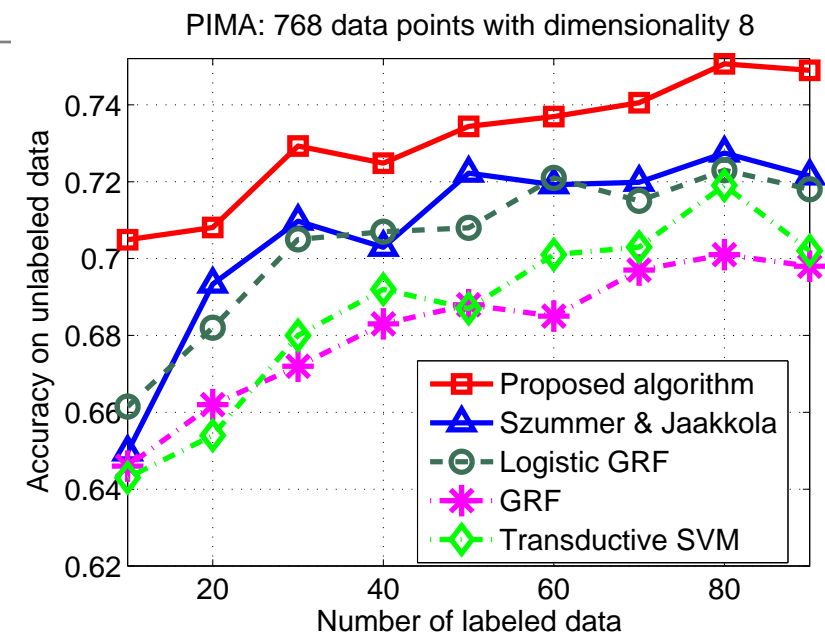


Figure 1: Transductive results. Each curve is an average from 20 independent trials. The horizontal axis is the size of \mathcal{X}_L . The algorithms are tested on \mathcal{X}_U . The algorithm of Szummer & Jaakkola (Szummer & Jaakkola, 2002) and ours use $\sigma_i = \min_j \|\mathbf{x}_i - \mathbf{x}_j\|/3$ and $t = 100$.

Induction with parametric NeBC: Benchmark Data

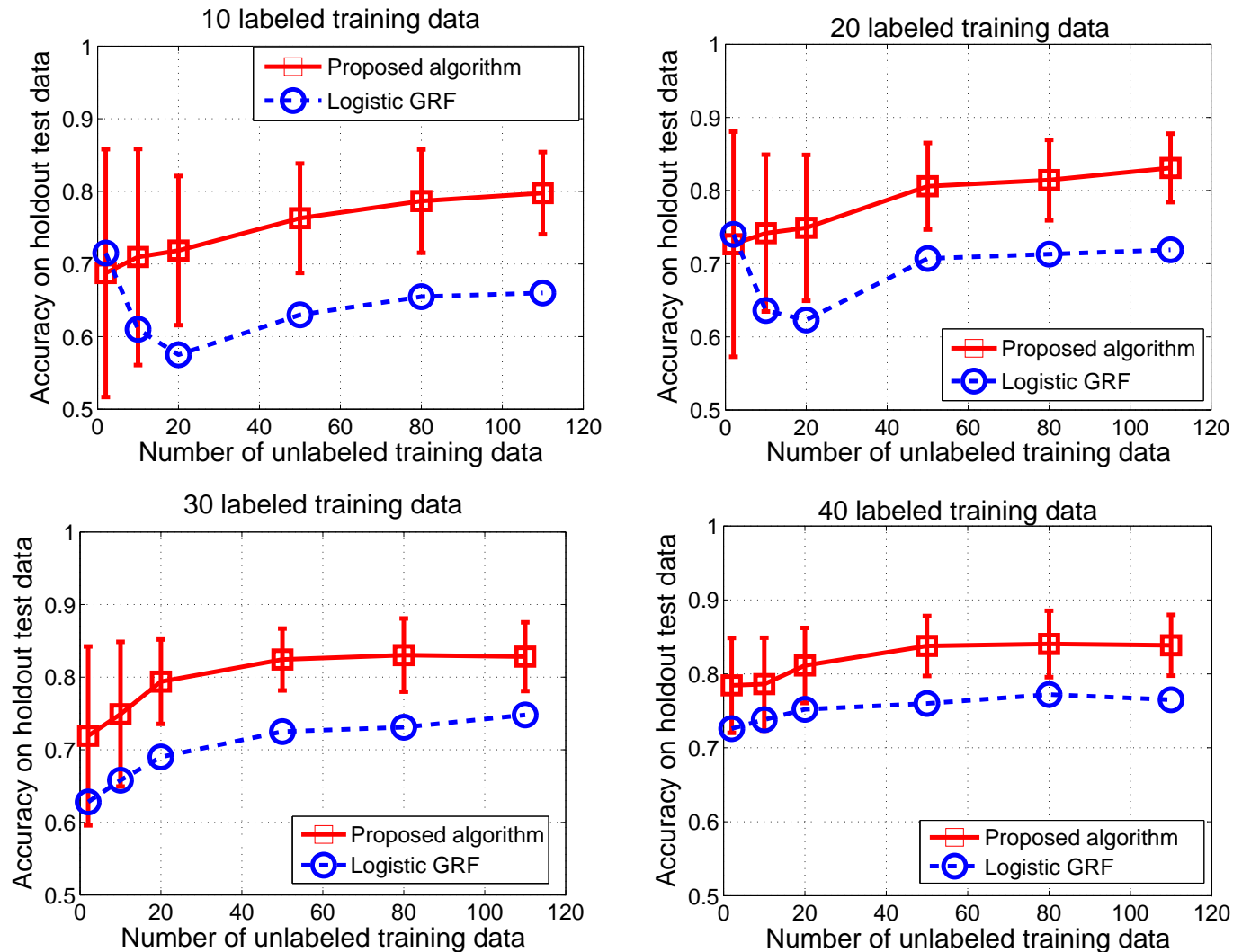


Figure 2: Inductive results. Each curve is an average from 50 independent trials. The horizontal axis is the size of \mathcal{X}_U . From left to right in the sub-figures, the size of \mathcal{X}_L is 10, 20, 30, 40. The algorithms are tested on 200 data randomly sampled from $\mathcal{X} \setminus (\mathcal{X}_L \cup \mathcal{X}_U)$. Error bars are shown for the proposed algorithm, which uses $\sigma_i = \min_j \|\mathbf{x}_i - \mathbf{x}_j\|/3$ and $t = 100$.

Summary

- The nonparametric NeBC works well when the data of the same class are well connected.
- The parametric NeBC facilitate inductive classification and improves generalization when the parametric form is chosen right.
- The semi-parametric NeBC (DDD) is expected to combine the features of the two.
- Undergoing work on how to learn the base classifier with the exact formulation of the DDD.