Prototype Vector Machine for Large Scale Semi-supervised Learning

Kai Zhang\textsuperscript{1} James T. Kwok\textsuperscript{2} Bahram Parvin\textsuperscript{1}

\textsuperscript{1}Life Science Division, Lawrence Berkeley National Lab
\textsuperscript{2}Department of Computer Science and Engineering Hong Kong University of Science and Technology
Outline

1. Semi-supervised Learning
   - Transductive SVM
   - Graph-based Methods
   - Scaling up graph-based SSL

2. Prototype Vector Machine
   - Approximation via Prototypes
   - Low-rank Approximation Prototype
   - Label Reconstruction Prototype
   - Optimization

3. Experiments

4. Conclusion
Outline

1. **Semi-supervised Learning**
   - Transductive SVM
   - Graph-based Methods
   - Scaling up graph-based SSL

2. **Prototype Vector Machine**
   - Approximation via Prototypes
   - Low-rank Approximation Prototype
   - Label Reconstruction Prototype
   - Optimization

3. **Experiments**

4. **Conclusion**
Semi-supervised Learning

Setting:
- limited supervision: \( \{x_i, y_i\}_{i=1}^l \)
- unlabeled data: \( \{x_i\}_{i=l+1}^n \)

Goal:
- prediction using both labeled and unlabeled samples
Transductive SVM

\[
\min_{\{\bar{y}_i\}_{i=1}^u, w, b, \{\xi_i^*\}_{i=1}^u, \{\xi_i\}_{i=1}^l} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \xi_i + C^* \sum_{i=l+1}^n \xi_i^* \\
\text{s.t.} \quad y_i(w^* x_i + b) \leq 1 - \xi_i \\
\quad \quad \quad y_i^*(w^* x_i + b) \leq 1 - \xi_i^*
\]

- transductive SVM (text classification) [Joachims et al. 1999]
- linear SVM [Fung and Mangasarian 2001]
- SDP relaxations [Bie and Cristianini 2004] [Xu et al. 2008]
- CCCP optimization [Collobert et al. 2006]
Graph-based Methods

Graph Regularization (transductive)

$$
\min_{f=\left[f_f \ f_u\right]^T} \underbrace{\text{tr}(f' S f)}_{\text{smoothness}} + \underbrace{C_1 L(f_f, Y_f)}_{\text{loss}} + \underbrace{C_2 \|f_u\|_F^2}_{\text{complexity}}
$$

- $S$: (normalized) Graph Laplacian

Examples:
- local and global consistency [Zhou et al. 2003]
- Gaussian fields and harmonic function [Zhu et al. 2003]
- nonparametric function induction [Delalleau et al. 2005]
Graph-based Methods

Manifold Regularization (inductive)

\[
\min_f \sum_{i=1}^{l} L(f(x_i), y_i) + \gamma_A \|f\|_K + \gamma_I \|f\|_G
\]

\[
\Rightarrow f(x) = \sum_{i=1}^{l+u} \alpha_i K(x, x_i)
\]

- manifold regularization [Belkin 2002]
- Lap-RLS, Lap-SVM
Fast graph-based SSL Methods

Fast algorithms ($O(m^2 n)$)

- **Harmonic mixture** [Zhu et al. 2002]
  - combine generative model with graph-method
- **Nonparametric function induction** [Delalleau et al. 2005]
  - label reconstruction by landmark points
  - ignores important regularization
- **Nyström method** [Gustavo et al. 2007]
  - speed up kernel matrix inverse

Survey

- Semi-supervised learning literature survey [Zhu]
- Large scale semi-supervised learning [Weston]
Outline

1 Semi-supervised Learning
   • Transductive SVM
   • Graph-based Methods
   • Scaling up graph-based SSL

2 Prototype Vector Machine
   • Approximation via Prototypes
   • Low-rank Approximation Prototype
   • Label Reconstruction Prototype
   • Optimization

3 Experiments

4 Conclusion
Observation

**Regularization**: bottleneck of graph-based SSL

- manipulation of $n \times n$ kernel matrix
  - multiplication
  - inverse
- lead to complex model
  - spans over labelled and unlabeled data
    \[ f(x) = \sum_{i=1}^{l+u} \alpha_i K(x, x_i) \]
  - slow training and testing
Approximation via Prototypes

Basic Idea

Basic idea: approximate regularization via prototypes

1. Low-rank approximation prototypes
   - preserve structures of kernel matrix
   - crucial for manifold regularization
   - less space

2. Label-reconstruction prototypes
   - reduce model complexity
   - fast testing
Low-rank Approximation

Given \( n \times n \) kernel matrix \( K \) (on \( \mathcal{X} \))

- find \( K \approx GG', \ G \in \mathbb{R}^{n \times m} \) (\( m \ll n \))

Nyström Method

1. Choose \( m \ll n \) columns \( E_{n \times m} \)
   - corresponds to landmark set \( \mathcal{Z} \), \( |\mathcal{Z}|=m \)
   - \( W_{m \times m} \): kernel matrix on \( \mathcal{Z} \)

2. Reconstruct by \( K \approx EW^{-1}E' \)
Low-rank Approximation Prototype

\[ z_i' \in \mathcal{Z} \]: low-rank approximation prototypes

- can be chosen as k-means clustering centers for
  - Gaussian
  - linear
  - polynomial

Detailed analysis in [Zhang et. al. 2008]

Nyström low-rank approximation quality depends on the encoding power of landmark points.
A small set of prototypes (with labels estimated) can reconstruct the overall label landscape.

Label reconstruction:  \( g(x) = \sum_{i=1}^{k} f_i K(x, v_i) \) or \( f = Hf_v \)

\( v_i \)'s: label reconstruction prototypes.
Using $g$ to approximate $f$:

$$\min_{\beta_i, v_i} D(\sum_{i=1}^{l+u} \alpha_i K(x, x_i), \sum_{i=1}^{m} \beta_i K(x, v_i))$$

$$\min_{\beta_i, v_i} \left\{ f(x) \bigg| \sum_{i=1}^{l+u} \alpha_i K(x, x_i), \sum_{i=1}^{m} \beta_i K(x, v_i) \right\}$$

- $\alpha_i$’s unknown
- alternative: basis in $f$ should be well-coded by those in $g$.

$$Q = \sum_{i=1}^{l+u} \sum_{j=1}^{k} \min D_{KL} \left[ K(x, x_i) \| K(x, v_j) \right]$$

Gaussian kernel $K \Rightarrow Q = \frac{1}{4h^2} \sum_i \sum_j \min \| x_i - v_j \|^2 \Rightarrow k$-means centers as $v_j$’s.
Rephrasing Optimization with Prototypes

Two types of prototypes

1. low-rank approximation \( K \approx EW^{-1}E' \)
   - \( E \in \mathbb{R}^{n \times m} \), \( W \in \mathbb{R}^{m \times m} \),

2. label reconstruction \( f \approx Hf_v \)
   - \( f \in \mathbb{R}^{n \times 1}; f_v \in \mathbb{R}^{k \times 1}, H \in \mathbb{R}^{n \times k} \)

Regularization can be approximated by

\[
    f^T S f \approx f'_v H' (\tilde{D} - EW^{-1}E^T) H f_v \\
    O(\left(m + k\right)^2 n)
\]
Optimization

$L_2$ Loss Function

- multiclass, $L_2$-loss function
- labels $Y_l \in \mathbb{R}^{l \times C}$,

$$
\min_{f_v \in \mathbb{R}^{m \times k}} \text{tr} \left( (Hf_v)'S(Hf_v) \right) + C_1 \| Hf_v - Y_l \|_F^2 + C_2 \| H_f v \|_F^2
$$

**training**

$$f^*_v = (H'SH + C_1 H_l' H_l + C_2 H_u' H_u)^{-1} E'_l Y_l$$

**testing**

$$f = Hf_v$$

$O(n(m + k)^2)$ time
Hinge Loss Function

- Binary, $Y_l \in \{\pm 1\}^{l \times 1}$, Hinge loss,
- $H_l = [e_1, e_2, ..., e_l]^T$
- $A = H^T S H + C_2 H_u^T H_u \in \mathbb{R}^{k \times k}$
- $Q = H_l A^{-1} H_l^T \odot Y_l Y_l^T \in \mathbb{R}^{l \times l}$

**Primal**

$$\min_{f_v \in \mathbb{R}^{m \times 1}} \frac{1}{2} f_v^T A f_v + C_1 \sum_{i=1}^{l} \xi_i$$

s.t. $y_i e_i^T f_v \geq 1 - \xi_i$, $\xi_i \geq 0$

**Dual**

$$\max -\frac{1}{2} \beta^T Q \beta + 1_l^T \beta$$

s.t. $0 \leq \beta_i \leq C_1$, $i = 1, 2, ..., l.$
Outline

1 Semi-supervised Learning
   - Transductive SVM
   - Graph-based Methods
   - Scaling up graph-based SSL

2 Prototype Vector Machine
   - Approximation via Prototypes
   - Low-rank Approximation Prototype
   - Label Reconstruction Prototype
   - Optimization

3 Experiments

4 Conclusion
Experimental Setting

- methods compared
  - **LGC**: local and global consistency;
  - **Lap-RLS**: Laplacian-regularized RLS;
  - **NYS-LGC**: Nyström-based LGC;
  - **NFI**: nonparametric function induction;
  - **PVM(1)**: $L_2$ loss;
  - **PVM(2)** Hinge loss

- 15 data sets (semi-supervised learning, libsvm)
- Gaussian kernel ($m = k$).
- $m = 0.1n$ for $n \leq 3000$; $m = 200$ for larger $n$
- 50 labels per class; randomly repeat 30 times
## Benchmark Data

Classification errors of different algorithms.

<table>
<thead>
<tr>
<th>Data(#cls)</th>
<th>LGC</th>
<th>LAP-RLS</th>
<th>NYS-LGC</th>
<th>NFI</th>
<th>PVM(1)</th>
<th>PVM(2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>g241c(2)</td>
<td>21.92</td>
<td>22.02</td>
<td>24.19</td>
<td>28.07</td>
<td>24.50</td>
<td>23.21</td>
</tr>
<tr>
<td>g241d(2)</td>
<td>28.10</td>
<td>22.36</td>
<td>30.98</td>
<td>30.82</td>
<td>25.15</td>
<td>24.85</td>
</tr>
<tr>
<td>digit1(2)</td>
<td>5.74</td>
<td>5.74</td>
<td>6.68</td>
<td>9.83</td>
<td>4.18</td>
<td>3.72</td>
</tr>
<tr>
<td>USPS(2)</td>
<td>4.57</td>
<td>6.11</td>
<td>9.72</td>
<td>5.49</td>
<td>5.29</td>
<td>6.35</td>
</tr>
<tr>
<td>coil(6)</td>
<td>12.38</td>
<td>21.17</td>
<td>18.75</td>
<td>30.93</td>
<td>13.41</td>
<td>–</td>
</tr>
<tr>
<td>BCI(2)</td>
<td>44.43</td>
<td>29.16</td>
<td>45.45</td>
<td>45.67</td>
<td>33.59</td>
<td>31.65</td>
</tr>
<tr>
<td>Text(2)</td>
<td>23.09</td>
<td>23.99</td>
<td>34.40</td>
<td>32.54</td>
<td>30.4</td>
<td>26.29</td>
</tr>
<tr>
<td>usps3589(4)</td>
<td>2.46</td>
<td>4.54</td>
<td>6.89</td>
<td>7.14</td>
<td>3.66</td>
<td>–</td>
</tr>
<tr>
<td>splice(2)</td>
<td>22.85</td>
<td>19.78</td>
<td>30.56</td>
<td>34.56</td>
<td>23.47</td>
<td>25.32</td>
</tr>
<tr>
<td>dna(3)</td>
<td>27.31</td>
<td>17.72</td>
<td>29.53</td>
<td>43.38</td>
<td>15.87</td>
<td>–</td>
</tr>
<tr>
<td>svmgd1a(2)</td>
<td>–</td>
<td>–</td>
<td>6.32</td>
<td>14.21</td>
<td>5.24</td>
<td>6.08</td>
</tr>
<tr>
<td>usps-full(10)</td>
<td>–</td>
<td>–</td>
<td>17.68</td>
<td>14.43</td>
<td>7.35</td>
<td>–</td>
</tr>
<tr>
<td>satimage(6)</td>
<td>–</td>
<td>–</td>
<td>16.36</td>
<td>19.27</td>
<td>14.97</td>
<td>–</td>
</tr>
</tbody>
</table>
## Benchmark Data

Time consumptions (seconds) of different algorithms.

<table>
<thead>
<tr>
<th>Data(n/dim)</th>
<th>LGC</th>
<th>LAP-RLS</th>
<th>NYS-LGC</th>
<th>NFI</th>
<th>PVM(1)</th>
<th>PVM(2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>g241c(1500/241)</td>
<td>140.84</td>
<td>129.86</td>
<td>0.86</td>
<td>0.48</td>
<td>3.30</td>
<td>3.19</td>
</tr>
<tr>
<td>g241d(1500/241)</td>
<td>129.78</td>
<td>142.65</td>
<td>0.84</td>
<td>0.49</td>
<td>3.31</td>
<td>3.16</td>
</tr>
<tr>
<td>digit1(1500/241)</td>
<td>140.51</td>
<td>131.08</td>
<td>0.84</td>
<td>0.48</td>
<td>3.31</td>
<td>3.15</td>
</tr>
<tr>
<td>USPS(1500/241)</td>
<td>139.23</td>
<td>131.59</td>
<td>0.74</td>
<td>0.47</td>
<td>3.28</td>
<td>3.14</td>
</tr>
<tr>
<td>coil2(1500/241)</td>
<td>151.36</td>
<td>120.48</td>
<td>0.87</td>
<td>0.48</td>
<td>3.26</td>
<td>3.47</td>
</tr>
<tr>
<td>coil(1500/241)</td>
<td>146.92</td>
<td>115.22</td>
<td>0.79</td>
<td>0.49</td>
<td>3.35</td>
<td></td>
</tr>
<tr>
<td>BCI(400/117)</td>
<td>3.08</td>
<td>1.94</td>
<td>0.53</td>
<td>0.22</td>
<td>0.71</td>
<td>1.09</td>
</tr>
<tr>
<td>Text(1500/11960)</td>
<td>139.67</td>
<td>216.37</td>
<td>9.14</td>
<td>13.26</td>
<td>30.24</td>
<td>34.24</td>
</tr>
<tr>
<td>2-moon(1000/2)</td>
<td>49.76</td>
<td>16.11</td>
<td>0.026</td>
<td>0.24</td>
<td>0.083</td>
<td>0.21</td>
</tr>
<tr>
<td>usps3589(719/64)</td>
<td>13.94</td>
<td>13.13</td>
<td>0.15</td>
<td>0.086</td>
<td>0.37</td>
<td></td>
</tr>
<tr>
<td>splice(3175/60)</td>
<td>1622.51</td>
<td>1439.51</td>
<td>2.49</td>
<td>0.83</td>
<td>4.87</td>
<td>4.24</td>
</tr>
<tr>
<td>dna(3186/180)</td>
<td>1566.91</td>
<td>1463.75</td>
<td>3.07</td>
<td>1.22</td>
<td>8.92</td>
<td></td>
</tr>
<tr>
<td>svmgd1a(7089/4)</td>
<td>–</td>
<td>–</td>
<td>3.22</td>
<td>1.66</td>
<td>8.06</td>
<td>5.38</td>
</tr>
<tr>
<td>usps-full(7291/256)</td>
<td>–</td>
<td>–</td>
<td>3.96</td>
<td>2.87</td>
<td>22.48</td>
<td></td>
</tr>
<tr>
<td>satimage(6435/36)</td>
<td>–</td>
<td>–</td>
<td>3.34</td>
<td>2.57</td>
<td>11.56</td>
<td></td>
</tr>
</tbody>
</table>
Case Study

Five-class classification

- MNIST digits 3,5,6,8,9
- \( n = 29270 \); \( \text{dim} = 784 \)

Algorithm properties

- scalability
- performance over \# labels
- performance over prototype size
Properties of PVM(1)

From left to right: time v.s. sample size; error v.s. #labels; error v.s.#prototypes.
Outline

1. Semi-supervised Learning
   - Transductive SVM
   - Graph-based Methods
   - Scaling up graph-based SSL

2. Prototype Vector Machine
   - Approximation via Prototypes
   - Low-rank Approximation Prototype
   - Label Reconstruction Prototype
   - Optimization

3. Experiments

4. Conclusion
Conclusions

- Conclusion
  - Computational bottleneck of Graph-based SSL
    - the regularization term
    - alleviated by using prototype approximations

- Future work
  - prototype selection
    - under different kernels
    - using label information
  - different label reconstruction schemes
Thank you!