Kernel Learning with a Million Kernels

Ashesh Jain  
IIT Delhi

SVN Vishwanathan  
Purdue University

Manik Varma  
Microsoft Research India
The objective in kernel learning is to jointly learn both SVM and kernel parameters from training data.

Kernel parameterizations
- Linear: \( K = \sum_i d_i K_i \)
- Non-linear: \( K = \prod_i K_i = \prod_i e^{-d_i D_i} \)

Regularizers
- Sparse \( l_1 \)
- Sparse and non-sparse \( l_{p>1} \)
- Log determinant
Kernel Learning for Object Detection

- Vedaldi, Gulshan, Varma and Zisserman ICCV 2009
Kernel Learning for Object Recognition

- Orabona, Jie and Caputo CVPR 2010
Kernel Learning for Feature Selection

- Varma and Babu ICML 2009

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>76.3 ± 0.9</td>
<td>79.5 ± 1.9</td>
<td>71.6 ± 1.4</td>
<td>84.9 ± 1.9</td>
<td>79.5 ± 2.6</td>
<td>81.2 ± 3.2</td>
<td>80.8 ± 0.2</td>
<td>88.7 ± 0.8</td>
</tr>
<tr>
<td>20</td>
<td>-</td>
<td>82.6 ± 0.6</td>
<td>80.5 ± 3.3</td>
<td>87.6 ± 0.5</td>
<td>85.6 ± 0.7</td>
<td>86.5 ± 1.3</td>
<td>83.8 ± 0.7</td>
<td>93.2 ± 0.9</td>
</tr>
<tr>
<td>30</td>
<td>-</td>
<td>83.4 ± 0.3</td>
<td>84.8 ± 0.4</td>
<td>89.3 ± 1.1</td>
<td>88.6 ± 0.2</td>
<td>89.4 ± 2.4</td>
<td>86.3 ± 1.6</td>
<td>95.1 ± 0.5</td>
</tr>
<tr>
<td>50</td>
<td>-</td>
<td>86.9 ± 1.0</td>
<td>88.8 ± 0.4</td>
<td>90.6 ± 0.6</td>
<td>89.5 ± 0.2</td>
<td>91.0 ± 1.3</td>
<td>89.4 ± 0.9</td>
<td>95.5 ± 0.7</td>
</tr>
<tr>
<td>80</td>
<td>-</td>
<td>88.9 ± 0.6</td>
<td>90.4 ± 0.2</td>
<td>-</td>
<td>90.6 ± 1.1</td>
<td>92.4 ± 1.4</td>
<td>90.5 ± 0.2</td>
<td>-</td>
</tr>
<tr>
<td>100</td>
<td>-</td>
<td>89.5 ± 0.2</td>
<td>90.6 ± 0.3</td>
<td>-</td>
<td>90.5 ± 0.2</td>
<td>94.1 ± 1.3</td>
<td>91.3 ± 1.3</td>
<td>-</td>
</tr>
<tr>
<td>150</td>
<td>-</td>
<td>91.3 ± 0.5</td>
<td>90.3 ± 0.8</td>
<td>-</td>
<td>90.7 ± 0.2</td>
<td>94.5 ± 0.7</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>252</td>
<td>-</td>
<td>93.1 ± 0.5</td>
<td>-</td>
<td>-</td>
<td>90.8 ± 0.0</td>
<td>94.3 ± 0.1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>76.3(12.6)</td>
<td>-</td>
<td>91 (221.3)</td>
<td>91 (58.3)</td>
<td>90.8 (252)</td>
<td>-</td>
<td>91.6(146.3)</td>
<td>95.5 (69.6)</td>
</tr>
</tbody>
</table>
The GMKL Primal Formulation

\[ P = \min_{w,b,d} \quad \frac{1}{2} w^t w + C \sum_i L(w^t \phi_d(x_i) + b, y_i) + r(d) \]

\[ \text{s. t.} \quad d \in D \]

- \[ K_d(x_i, x_j) = \phi_d^t(x_i) \phi_d(x_j) > 0 \quad \forall d \in D \]
- \[ \nabla_d K \text{ and } \nabla_d r \text{ exist and are continuous} \]
The GMKL Primal Formulation

- The GMKL primal formulation for binary classification.

\[
P = \min_{w, b, d, \xi} \quad \frac{1}{2} w^t w + C \sum_i \xi_i + r(d) \\
\text{s. t.} \quad y_i (w^t \phi_d(x_i) + b) \geq 1 - \xi_i \\
\xi_i \geq 0 \quad \& \quad d \in D
\]
The GMKL Primal Formulation

• The GMKL primal formulation for binary classification.

\[ P = \text{Min}_{w,b,d,\xi} \]
\[ \text{s. t.} \]
\[ \frac{1}{2}w^t w + C \sum_i \xi_i + r(d) \]
\[ y_i (w^t \phi_d(x_i) + b) \geq 1 - \xi_i \]
\[ \xi_i \geq 0 \text{ & } d \in D \]

• Intermediate Dual

\[ D = \text{Min}_d \text{ Max}_\alpha \]
\[ \text{s. t.} \]
\[ 1^t \alpha - \frac{1}{2} \alpha^t YK_d Y \alpha + r(d) \]
\[ 1^t Y \alpha = 0 \]
\[ 0 \leq \alpha \leq C \text{ & } d \in D \]
Projected Gradient Descent
Projected Gradient Descent
Projected Gradient Descent
Projected Gradient Descent
PGD Limitations

- PGD requires many function and gradient evaluations as
  - No step size information is available.
  - The Armijo rule might reject many step size proposals.
  - Inaccurate gradient values can lead to many tiny steps.
PGD Limitations

• PGD requires many function and gradient evaluations as
  • No step size information is available.
  • The Armijo rule might reject many step size proposals.
  • Inaccurate gradient values can lead to many tiny steps.

• Noisy function and gradient values can cause PGD to converge to points far away from the optimum.
PGD Limitations

- PGD requires many function and gradient evaluations as
  - No step size information is available.
  - The Armijo rule might reject many step size proposals.
  - Inaccurate gradient values can lead to many tiny steps.

- Noisy function and gradient values can cause PGD to converge to points far away from the optimum.

- Solving SVMs to high precision to obtain accurate function and gradient values is very expensive.
PGD Limitations

• PGD requires many function and gradient evaluations as
  • No step size information is available.
  • The Armijo rule might reject many step size proposals.
  • Inaccurate gradient values can lead to many tiny steps.

• Noisy function and gradient values can cause PGD to converge to points far away from the optimum.

• Solving SVMs to high precision to obtain accurate function and gradient values is very expensive.

• Repeated projection onto the feasible set might also be expensive.
SPG Solution – Spectral Step Length

- Quadratic approximation: \( \frac{1}{2} \lambda^{-1} x^t x + c^t x + d \)
- Spectral step length: \( \lambda_{SPG} = \frac{\langle x^n - x^{n-1}, x^n - x^{n-1} \rangle}{\langle x^n - x^{n-1}, \nabla f(x^n) - \nabla f(x^{n-1}) \rangle} \)

Original Function

Approximation
• Spectral step length: $\lambda_{SPG} = \frac{\langle x^n - x^{n-1}, x^n - x^{n-1} \rangle}{\langle x^n - x^{n-1}, \nabla f(x^n) - \nabla f(x^{n-1}) \rangle}$
• Accept $P(z^t)$ if it satisfies the Armijo rule
PGD Limitations – Repeated Projections

- Accept $P(z^t)$ if it satisfies the Armijo rule
PGD Limitations – Repeated Projections

- PGD might require many projections before accepting a point.
SPG Solution – Spectral Proj Gradient

- SPG requires a single projection per step
SPG Solution – Non-Monotone Rule

- Handling function and gradient noise.
- Non-monotone rule: $f(x^t - s\nabla f(x^t)) \leq \max_{0 \leq j \leq M} f(x^{t-j}) - \gamma s|\nabla f(x^t)|^2_2$
The Armijo rule might get stuck due to noisy function values.
SPG Solution – SVM Precision Tuning
SPG Advantages

- SPG requires fewer function and gradient evaluations due to
  - The 2\textsuperscript{nd} order spectral step length estimation.
  - The non-monotone line search criterion.

- SPG is more robust to noisy function and gradient values due to the non-monotone line search criterion.

- SPG never needs to solve an SVM with high precision due to our precision tuning strategy.

- SPG needs to perform only a single projection per step.
SPG Algorithm

1: $n \leftarrow 0$
2: Initialize $d^0$ randomly
3: repeat
4: $\alpha^* \leftarrow \text{SolveSVM}(K(d^n), \epsilon)$
5: $\lambda \leftarrow \text{SpectralStepLength}$
6: $p^n \leftarrow d^n - P(d^n - \lambda \nabla W(d^n, \alpha^*))$
7: $s^n \leftarrow \text{Non-Monotone}$
8: $\epsilon \leftarrow \text{TuneSVMPrecision}$
9: $d^{n+1} \leftarrow d^n - s^n p^n$
10: until converged
Results on Large Scale Data Sets

- Covertype: Sum of kernels subject to $l_{1.33}$ regularization
  - Number of training points 581,012
  - Number of Kernels 5
  - SPG time taken 64.46 hrs

- SPG took 26 SVM evaluations

- First SVM evaluation took 44 hours

- Only 0.19% of SV were cached
Results on Large Scale Data Sets

- **Sonar**: Sum of kernels subject to $l_{1.33}$ regularization
- Number of training points 208
- Number of Kernels 1 Million
- SPG time taken 105.62 hrs
Results on Large Scale Data Sets

- Sum of kernels subject to $l_{p \geq 1}$ regularization

<table>
<thead>
<tr>
<th>Data Sets</th>
<th># Train</th>
<th># Kernels</th>
<th>$p=1$ PGD (hrs)</th>
<th>$p=1$ SPG (hrs)</th>
<th>$p=1.33$ PGD (hrs)</th>
<th>$p=1.33$ SPG (hrs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adult - 9</td>
<td>32,561</td>
<td>50</td>
<td>35.84</td>
<td>4.55</td>
<td>31.77</td>
<td>4.42</td>
</tr>
<tr>
<td>Cod - RNA</td>
<td>59,535</td>
<td>50</td>
<td>–</td>
<td>25.17</td>
<td>66.48</td>
<td>19.10</td>
</tr>
<tr>
<td>KDDCup04</td>
<td>50,000</td>
<td>50</td>
<td>–</td>
<td>40.10</td>
<td>–</td>
<td>42.20</td>
</tr>
</tbody>
</table>
Results on Small Scale Data Sets

- Sum of kernels subject to $l_1$ regularization

<table>
<thead>
<tr>
<th>Data Sets</th>
<th>SimpleMKL (s)</th>
<th>Shogun (s)</th>
<th>PGD (s)</th>
<th>SPG (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wpbc</td>
<td>$400 \pm 128.4$</td>
<td>$15 \pm 7.7$</td>
<td>$38 \pm 17.6$</td>
<td>$6 \pm 4.2$</td>
</tr>
<tr>
<td>Breast - Cancer</td>
<td>$676 \pm 356.4$</td>
<td>$12 \pm 1.2$</td>
<td>$57 \pm 85.1$</td>
<td>$5 \pm 0.6$</td>
</tr>
<tr>
<td>Australian</td>
<td>$383 \pm 33.5$</td>
<td>$1094 \pm 621.6$</td>
<td>$29 \pm 7.1$</td>
<td>$10 \pm 0.8$</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>$1247 \pm 680.0$</td>
<td>$107 \pm 18.8$</td>
<td>$1392 \pm 824.2$</td>
<td>$39 \pm 6.8$</td>
</tr>
<tr>
<td>Sonar</td>
<td>$1468 \pm 1252.7$</td>
<td>$935 \pm 65.0$</td>
<td>$-$</td>
<td>$273 \pm 64.0$</td>
</tr>
</tbody>
</table>
Results on Large Scale Data Sets

- Product of kernels subject to $l_{p\geq 1}$ regularization

<table>
<thead>
<tr>
<th>Data Sets</th>
<th># Train</th>
<th># Kernels</th>
<th>$p=1$</th>
<th>$p=1.33$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>PGD (hrs)</td>
<td>SPG (hrs)</td>
</tr>
<tr>
<td>Letter</td>
<td>20,000</td>
<td>16</td>
<td>18.66</td>
<td>0.67</td>
</tr>
<tr>
<td>Poker</td>
<td>25,010</td>
<td>10</td>
<td>5.57</td>
<td>0.49</td>
</tr>
<tr>
<td>Adult - 8</td>
<td>22,696</td>
<td>42</td>
<td>–</td>
<td>1.73</td>
</tr>
<tr>
<td>Web - 7</td>
<td>24,692</td>
<td>43</td>
<td>–</td>
<td>0.88</td>
</tr>
<tr>
<td>RCV1</td>
<td>20,242</td>
<td>50</td>
<td>–</td>
<td>18.17</td>
</tr>
</tbody>
</table>
Effect of Individual Components

- Sum of kernels subject to $l_{1.1}$ regularization

<table>
<thead>
<tr>
<th>Data Sets</th>
<th>PGD</th>
<th>PGD + N</th>
<th>PGD + S</th>
<th>PGD + N + S</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time (s)</td>
<td># SVMs</td>
<td>Time (s)</td>
<td># SVMs</td>
</tr>
<tr>
<td>Australian</td>
<td>39.4 ± 6.0</td>
<td>3230</td>
<td>32.7 ± 3.6</td>
<td>116</td>
</tr>
<tr>
<td>Sonar</td>
<td>785.5 ± 471.1</td>
<td>209461</td>
<td>41.6 ± 17.1</td>
<td>3236</td>
</tr>
<tr>
<td>Breast - Cancer</td>
<td>237.3 ± 97.8</td>
<td>109599</td>
<td>42.2 ± 4.1</td>
<td>1187</td>
</tr>
<tr>
<td>Diabetes</td>
<td>73.6 ± 38.8</td>
<td>29347</td>
<td>26.3 ± 9.5</td>
<td>2966</td>
</tr>
<tr>
<td>Wpbc</td>
<td>44.4 ± 11.6</td>
<td>14376</td>
<td>27.9 ± 13.6</td>
<td>9388</td>
</tr>
</tbody>
</table>
### SVM Precision Tuning

- Sum of kernels subject to $l_{1.33}$ regularization

<table>
<thead>
<tr>
<th>Data Sets</th>
<th># Train</th>
<th># Kernels</th>
<th>PGD (hrs)</th>
<th>PGD + N + S (hrs)</th>
<th>SPG (hrs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adult - 9</td>
<td>32,561</td>
<td>50</td>
<td>31.77</td>
<td>8.33</td>
<td>4.43</td>
</tr>
<tr>
<td>Web - 8</td>
<td>49,749</td>
<td>50</td>
<td>4.27</td>
<td>1.73</td>
<td>0.87</td>
</tr>
<tr>
<td>Sonar</td>
<td>208</td>
<td>100,000</td>
<td>53.91</td>
<td>3.35</td>
<td>2.19</td>
</tr>
</tbody>
</table>
SPG Scaling Properties

- Scaling with the number of training points
Conclusions

• Developed a generic and efficient MKL optimizer.

• Experimented with four different MKL formulations and solved both small and large scale problems.

• Combining spectral step length and non-monotone rule gives best performance.

• Quasi Newton methods not suitable for MKL problems due to noisy gradient.

Acknowledgements

• Kamal Gupta (IITD)

• Subhashis Banerjee (IITD)

• The Computer Services Center at IIT Delhi