Memetic evolution of training sets with adaptive radial basis kernels for support vector machines

Jakub Nalepa, Wojciech Dudzik and Michal Kawulok

{jakub.nalepa, wojciech.dudzik, michal.kawulok}@polsl.pl
1 Problem statement
2 Proposed method
3 Experiments
4 Conclusions and Outlook
What are the problems

Main problems

- High computational complexity of training $O(t^3)$
- High memory complexity of training $O(t^2)$
- Need to fine-tune the models with hyperparameters
- Growing number of features
- Bigger datasets
- Classification time linearly depends on number of SV ($O(S)$)
Proposed solution
Optimize training set along model optimization by incorporating novel adaptive radial basis function kernel.
Proposed method
Figure: Assigning different $\gamma$'s in the RBF kernel to different $T$ vectors can help better “model” the SVM hyperplane.
Adaptive RBF kernel

Kernel function

For training: \( \mathcal{K}(x_i, x_j) = e^{-\gamma_i \|x_i - x_j\|^2} \),

For inference: \( \mathcal{K}(sv_i, x) = e^{-\gamma_i \|sv_i - x\|^2} \)
Memetic algorithms

Algorithm 1 Memetic evolution of SVM training sets.

1: Select $\vec{\gamma}$
2: for all $\gamma_i$ in $\vec{\gamma}$ do ▶ $\gamma$’s are sorted (ascendingly)
3: $P, P'_{\text{best}} \leftarrow$ Generate population($N, C, \gamma_i, T$)
4: if $\eta(P'_{\text{best}}) > \eta(P_{\text{best}})$ then
5: $P'_{\text{best}} \leftarrow$ RunEvolution()
6: if $\eta(P'_{\text{best}}) > \eta(P_{\text{best}})$ then
7: Add SV($P'_{\text{best}}$) to $S_{\text{best}}$
8: $T \leftarrow$ Shrink($T, P$)
9: $P_{\text{best}} \leftarrow P'_{\text{best}}$
10: end if
11: end if
12: end for
13: return $P_{\text{best}}$
Memetic algorithms

**Figure:** The best solution from initial population.
Figure: The best solution after finishing evolution with first $\gamma$ from $\vec{\gamma}$. 
Memetic algorithms

**Figure:** Shrinked training set that will be used in next iteration with subsequent $\gamma$. Shrinking procedure is based on whole population.
Figure: Solution after second evolution has ended. Added new support vectors marked with red color crosses.
Memetic algorithms

Figure: Adding next $\gamma$ value marked with green vectors provided worse classification performance, these support vectors will be removed.
Memetic algorithms

Figure: Final solution for given dataset containing three different $\gamma$ values.
Experiments
Implementation details

Setup:
- Windows 10 machine equipped with i9-7900X CPU and 64 GB of RAM

Settings:
- Objective function AUC with thresholding for optimal accuracy over validation set
- 5-fold cross-validation
- All results shows average from 50 runs (10 times for each fold),
- Folds contains training, validation and test set in 3:1:1 proportion, respectively
Table: The results of all methods. The best results are boldfaced.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>AUC</th>
<th>Acc</th>
<th>F1</th>
<th>Pr</th>
<th>Re</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>GNB</td>
<td>0.839</td>
<td>0.736</td>
<td>0.471</td>
<td>0.454</td>
<td>0.708</td>
<td>0.397</td>
</tr>
<tr>
<td>LR</td>
<td>0.862</td>
<td>0.912</td>
<td>0.599</td>
<td>0.691</td>
<td>0.567</td>
<td>0.545</td>
</tr>
<tr>
<td>$k$-NN(3)</td>
<td>0.857</td>
<td>0.930</td>
<td>0.674</td>
<td>0.729</td>
<td>0.644</td>
<td>0.626</td>
</tr>
<tr>
<td>$k$-NN(5)</td>
<td>0.872</td>
<td>0.930</td>
<td>0.632</td>
<td>0.716</td>
<td>0.591</td>
<td>0.590</td>
</tr>
<tr>
<td>$k$-NN(7)</td>
<td>0.879</td>
<td>0.928</td>
<td>0.591</td>
<td>0.695</td>
<td>0.548</td>
<td>0.554</td>
</tr>
<tr>
<td>SVM(Linear)</td>
<td>0.848</td>
<td>0.912</td>
<td>0.584</td>
<td>0.648</td>
<td>0.560</td>
<td>0.528</td>
</tr>
<tr>
<td>SVM(Poly)</td>
<td>0.869</td>
<td>0.924</td>
<td>0.632</td>
<td>0.717</td>
<td>0.605</td>
<td>0.592</td>
</tr>
<tr>
<td>SVM(RBF)</td>
<td>0.807</td>
<td>0.877</td>
<td>0.201</td>
<td>0.303</td>
<td>0.198</td>
<td>0.162</td>
</tr>
<tr>
<td>MASVM</td>
<td>0.880</td>
<td>0.936</td>
<td>0.651</td>
<td>0.763</td>
<td>0.605</td>
<td>0.622</td>
</tr>
<tr>
<td>MASVM(MG)</td>
<td>0.877</td>
<td>0.936</td>
<td>0.676</td>
<td>0.793</td>
<td>0.617</td>
<td>0.644</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>0.888</strong></td>
<td><strong>0.941</strong></td>
<td><strong>0.696</strong></td>
<td><strong>0.810</strong></td>
<td>0.640</td>
<td><strong>0.667</strong></td>
</tr>
</tbody>
</table>
Table: The ranking test over MCC (together with the statistical importance of the differences between our MA and the corresponding approach), for various IR ranges. The meanings of ns, *, **, ***, and ****: $p > 0.05$, $p \leq 0.05$, $p \leq 0.01$, $p \leq 0.001$, and $p \leq 0.0001$. The best results are boldfaced.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>GNB</td>
<td>7.63**</td>
<td>9.04***</td>
<td>8.58****</td>
<td>6.50****</td>
<td>7.94****</td>
</tr>
<tr>
<td>LR</td>
<td>6.08ns</td>
<td>5.79ns</td>
<td>5.75ns</td>
<td>5.29ns</td>
<td>5.73****</td>
</tr>
<tr>
<td>k-NN(3)</td>
<td>5.33ns</td>
<td>4.88ns</td>
<td>4.67ns</td>
<td>5.50ns</td>
<td>5.09*</td>
</tr>
<tr>
<td>k-NN(5)</td>
<td>5.50ns</td>
<td>5.21ns</td>
<td>5.67ns</td>
<td>5.67ns</td>
<td>5.51***</td>
</tr>
<tr>
<td>k-NN(7)</td>
<td>5.21ns</td>
<td>5.63ns</td>
<td>6.58**</td>
<td>6.25**</td>
<td>5.92****</td>
</tr>
<tr>
<td>SVM(Linear)</td>
<td>6.67ns</td>
<td>5.63ns</td>
<td>6.42**</td>
<td>5.71**</td>
<td>6.10****</td>
</tr>
<tr>
<td>SVM(Poly)</td>
<td>5.92ns</td>
<td>4.88ns</td>
<td>5.38ns</td>
<td>4.75ns</td>
<td>5.23**</td>
</tr>
<tr>
<td>SVM(RBF)</td>
<td>7.92**</td>
<td>10.71****</td>
<td>9.83****</td>
<td>8.13****</td>
<td>9.15****</td>
</tr>
<tr>
<td>MASVM</td>
<td>5.17ns</td>
<td>4.92ns</td>
<td>4.79ns</td>
<td>5.04ns</td>
<td>4.98ns</td>
</tr>
<tr>
<td>MASVM(MG)</td>
<td>4.75ns</td>
<td>4.46ns</td>
<td>3.83ns</td>
<td>4.00ns</td>
<td>4.26ns</td>
</tr>
<tr>
<td>Ours</td>
<td>3.75</td>
<td>4.42</td>
<td>2.92</td>
<td>3.21</td>
<td>3.57</td>
</tr>
</tbody>
</table>
Conclusions and outlook
Conclusions

- Our technique outperforms SVMs optimized using other evolutionary methods and other supervised learners with grid-searched hyperparameters.
- It delivers consistent results across sets of various characteristics.
- Our technique can be easily applied in imbalanced classification, where it outperformed all other methods.
- Assigning different $\gamma$’s to different training vectors is useful in heterogeneous ("difficult") parts of the input space, as visually shown for our synthetic datasets.
Thank you

{jakub.nalepa, wojciech.dudzik, michal.kawulok}@polsl.pl