Amended Cross-Entropy cost: an approach for encouraging diversity in classification ensemble (Brief Announcement)

Ron Shoham and Haim Permuter

CSCML June 2019
Motivation

- Ensemble of models is a fundamental technique
- Diversity between the predictions is necessary
- For Regression - Negative Correlation Learning (NCL)
- For Classification?
Why cross-entropy?

- Sigmoid \( \sigma(x) = \frac{1}{1+e^{-x}} \)
- MSE \( e_i = 0.5(\sigma(x) - y)^2 \)
- Gradient \( \frac{\partial e_i}{\partial x} = (\sigma(x) - y)\sigma(x)(1 - \sigma(x)) \)
- Saturation problem

We wish to get

- \( \frac{\partial e_i}{\partial x} = (\sigma(x) - y) \)
- \( \frac{\partial e_i}{\partial \sigma(x)} = \frac{(\sigma(x) - y)}{\sigma(x)(1 - \sigma(x))} \)
- \( e_i = \int \frac{\partial e_i}{\partial f_i} df_i, \quad f_i = \sigma(x) \)
  \[-y\log(f_i) - (1 - y)\log(1 - f_i) + C \]
  \[= H(y, f_i) + C \]
Negative Correlation Learning
Gavin Brown et al.

\[ f_{ens} = \frac{1}{M} \sum f_i \]

\[ e_i = \frac{1}{2} (f_i - t)^2 + \gamma (f_i - f_{ens}) \sum_{j \neq i} (f_j - f_{ens}). \]

• Gradient analysis

\[ \frac{\partial e_i}{\partial f_i} = (f_i - t) - \gamma \left[ 2 \left( 1 - \frac{1}{M} \right) (f_i - f_{ens}) \right] \]
\[ = (f_i - t) - \lambda (f_i - f_{ens}) \]
\[ = (1 - \lambda)(f_i - t) + \lambda (f_{ens} - t). \]
Amended cross-entropy

\[
\frac{\partial e_i}{\partial z_i} = (1 - \lambda)(f_i - y) + \lambda(f_{ens} - y)
\]
\[
\frac{\partial e_i}{\partial f_i} = \frac{(1 - \lambda)(f_i - y) + \lambda(f_{ens} - y)}{f_i(1 - f_i)}
\]
\[
= \frac{f_i - y}{f_i(1 - f_i)} - \frac{\lambda}{M} \sum_{j \neq i} \frac{f_i - f_j}{f_i(1 - f_i)}
\]
\[
e_i = \int \frac{\partial e_i}{\partial f_i} df_i
\]
\[
= -y \log(f_i) - (1 - y) \log(1 - f_i)
\]
\[
- \frac{\lambda}{M} \sum_{j \neq i} \{-f_j \log(f_i) - (1 - f_j) \log(1 - f_i)\}
\]
\[
= H(y, f_i) - \frac{\lambda}{M} H(f_j, f_i)
\]
Usage – Stacked Diversified Mixture of Classifiers

- Problem: Training an ensemble results in increasing the model.
- Solution: stacking a mixture of classifiers at the final layer of a Deep Neural Network
Database – Cifar 10

Architecture – ResNet 110

Number of parameters:

- Original: 1731002 parameters
- ResNet110 + SDMC (M=10): 1736852 parameters (+0.34%)

<table>
<thead>
<tr>
<th>$M = 1$</th>
<th>$M = 10$</th>
<th>$M = 10$</th>
<th>$M = 10$</th>
<th>$M = 10$</th>
<th>$M = 10$</th>
<th>$M = 10$</th>
<th>$M = 10$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda = 0$</td>
<td>$\lambda = 0.001$</td>
<td>$\lambda = 0.01$</td>
<td>$\lambda = 0.05$</td>
<td>$\lambda = 0.1$</td>
<td>$\lambda = 0.3$</td>
<td>$\lambda = 0.5$</td>
<td></td>
</tr>
<tr>
<td>error(%)</td>
<td>6.43</td>
<td>6.2</td>
<td>6.14</td>
<td>6.12</td>
<td>5.98</td>
<td>6.09</td>
<td>6.13</td>
</tr>
<tr>
<td>CE</td>
<td>0.3056</td>
<td>0.3102</td>
<td>0.3041</td>
<td>0.3048</td>
<td>0.2968</td>
<td>0.2918</td>
<td>0.3137</td>
</tr>
</tbody>
</table>

Optimal lambda reduced error by ~7%