Methods and Applications for

Distance Based ANN Training

Christoph Lassner, Rainer Lienhart
Multimedia Computing and Computer Vision Lab
Augsburg University
"Make feature learning as simple and fast as possible."

Requirement analysis

"An Analysis of Single-Layer Networks in Unsupervised Feature Learning" (Coates, Lee, Ng; AISTATS 2011)

Distance Based ANN Training

- Formulas
- Preprocessing

Feature learning & compression

Visualization

Outlier Detection
Feature learning rethought

*Use overfitted decision trees as feature selection preprocessing step to filter out noise.
Distance Error: 

\[ E(x_1, x_2) = \frac{1}{2} \left( \theta_f(o,o') - \theta_o(x_1,x_2) \right)^2 \]
Distance Based ANN Training

Weight gradient (last layer):

\[
\frac{\partial E(x_1, x_2)}{\partial w_{ji}} = (\theta_f(o, o') - \theta_o(x_1, x_2)) \cdot (o_j - o_j') \cdot [h'(net_j) \cdot x_{ji} - h'(net_j') \cdot x_{ji}']
\]

Signed difference of node outputs.

→ Difference of weighted deriv. of activ. funcs.

→ Allows for "responsibilities" update "controllable".
Distance Based ANN Training

Weight gradient (last layer):

\[
\frac{\partial E(x_1, x_2)}{\partial w_{ji}} = (\theta_f(o, o') - \theta_o(x_1, x_2)) \cdot (o_j - o'_j) \cdot [h'(net_j) \cdot x_{ji} - h'(net'_j) \cdot x'_{ji}]
\]
Weight gradient (general):

\[
\frac{\partial E(x_1, x_2)}{\partial w_{ji}} = \sum_{r \in \text{outlets}(j)} (\theta_f(o, o') - \theta_o(x_1, x_2)) \cdot (o_r - o'_r) \cdot \sum_{k \in \text{downstream}(j)} w_{kj} \cdot (\delta_{kr} \cdot h'(net_j) \cdot x_{ji} - \delta'_{kr} \cdot h'(net'_j) \cdot x'_{ji})
\]

Difference of weighted deriv. of activ. func. paths.

→ Makes updates "controllable".
Kernel functions

Kernels allow to specify how the objective distance is calculated from data. E.g. $\theta_o(x_1, x_2)$ can be defined as:

- $\kappa_{\text{class}}(x_1, x_2) = \|l(x_1) - l(x_2)\|^2$

- $\kappa_{\text{original}}(x_1, x_2) = \|x_1 - x_2\|^2$

- $\kappa_{\text{orig+class}}(x_1, x_2) = \|x_1 - x_2\|^2 + \|l(x_1) - l(x_2)\|^2$

- $\kappa_{\text{DPCA}}(x_1, x_2) = \|\text{DPCA}^{-1}(\text{DPCA}(x_1 - x_2))\|^2$, reducing the difference onto the "principal differences" between two data points.

$\text{DPCA} = \text{PCA}$ fit on 10000 randomly sampled differences before the DANN training.
**Experiments: Classification**

<table>
<thead>
<tr>
<th>Proben1 Dataset</th>
<th>Dimensions</th>
<th>Classes</th>
<th>Samples (train/val/test)</th>
<th>Balance</th>
<th>SVM</th>
<th>RBF SVM</th>
<th>SVM on DANN features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Proben1 Dataset

<table>
<thead>
<tr>
<th>Dimensions</th>
<th>Classes</th>
<th>Samples (train/val/test)</th>
<th>Balance</th>
<th>SVM</th>
<th>RBF SVM</th>
<th>SVM on DANN features</th>
</tr>
</thead>
<tbody>
<tr>
<td>gene</td>
<td>120</td>
<td>3</td>
<td>1588/794/794</td>
<td>25%/50%/25%</td>
<td>0.906</td>
<td>0.914</td>
</tr>
</tbody>
</table>
### Experiments: Classification

<table>
<thead>
<tr>
<th>Proben1 Dataset</th>
<th>Dimensions</th>
<th>Classes</th>
<th>Samples (train/val/test)</th>
<th>Balance</th>
<th>SVM</th>
<th>RBF SVM</th>
<th>SVM on DANN features</th>
</tr>
</thead>
<tbody>
<tr>
<td>gene</td>
<td>120</td>
<td>3</td>
<td>1588/794/794</td>
<td>25%/50%/25%</td>
<td>0.906</td>
<td>0.914</td>
<td><strong>0.931</strong></td>
</tr>
<tr>
<td>mushroom</td>
<td>125</td>
<td>2</td>
<td>4062/2031/2031</td>
<td>52%/48%</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Experiments: Classification

<table>
<thead>
<tr>
<th>Proben1 Dataset</th>
<th>Dimensions</th>
<th>Classes</th>
<th>Samples (train/val/test)</th>
<th>Balance</th>
<th>SVM</th>
<th>RBF SVM</th>
<th>SVM on DANN features</th>
</tr>
</thead>
<tbody>
<tr>
<td>gene</td>
<td>120</td>
<td>3</td>
<td>1588/794/794</td>
<td>25%/50%/25%</td>
<td>0.906</td>
<td>0.914</td>
<td>0.931</td>
</tr>
<tr>
<td>mushroom</td>
<td>125</td>
<td>2</td>
<td>4062/2031/2031</td>
<td>52%/48%</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>thyroid</td>
<td>21</td>
<td>3</td>
<td>3600/1800/1800</td>
<td>2.5%/95%/2.5%</td>
<td>0.954</td>
<td>0.967</td>
<td>0.987</td>
</tr>
</tbody>
</table>
Experiments: Visualization

Use the $\kappa_{\text{original}}$ kernel to create a visualization in 2D or 3D space.
- Tries to preserve the distances.
- Allows to project new data at a later point in time.
- Learns a nonlinear mapping.
Experiments: Outlier Detection

Use the $\kappa_{\text{original}}$ kernel learn a mapping to a higher-dimensional space.

$\Rightarrow$ Points close together will be mapped to the proper positions first.

- Early stopping allows to isolate these clusters.
- Due to the higher dimensional space, a successively applied weak learner can beat the performance of a tuned RBF kernel One-Class SVM.
Outlook

• Lack of node specialization for very high dimensional data:

\[
\frac{\partial E(x_1, x_2)}{\partial w_{ji}} = (\theta_f(o, o') - \theta_o(x_1, x_2)) \cdot (o_j - o'_j) \cdot [h'(net_j) \cdot x_{ji} - h'(net'_j) \cdot x'_{ji}]
\]

• No exploitation of local correlation in image data: possibly this issue can be addressed by expanding the approach to convolutional ANNs?

Local response normalization could be applied to enforce specialization (Krizhevsky, Sutskever, Hinton; 2012).
Distance based ANN training can be a viable tool for **feature learning** and **compression**:

1. Linear SVMs can **outperform** RBF kernel SVMs on the learned features,
2. this was possible in our experiments using up to only a **tenth of the original feature dimensions**.

It offers unique features for **data visualization**:

1. The learned mapping tries to **preserve distances**,  
2. it allows **mapping of points after training is completed**,  
3. the learned mapping is **nonlinear**.

It might be an interesting tool for **outlier detection**:

1. a **mapping to a higher-dimensional** space can be learned,  
2. exploiting the inductive bias of this learning strategy, in the feature space **clusters of points are naturally closer together** during early stages of training.
Thank you for your interest and attention!