AdaBoosting Neural Networks for Character Recognition

This presentation is based on the paper:
AdaBoosting Neural Networks: Application to on-line Character Recognition (1997)
by Holger Schwenk and Yoshua Bengio

Introduction

- The main emphasis of this presentation is to:
  - Describe AdaBoosting.
  - Provide an overview of the Auto Associative Network and the Diabolo Classifier.
  - Give a summary of the performance of AdaBoosted Diabolo Classifiers when used for on-line character recognition.
**Boosting**

- *Boosting* is a technique for improving the predictive power of classifier learning systems.
- In general, a succession of weak classifiers is *boosted* to a strong classifier that is at least as accurate, and usually much more accurate than, the best weak classifier.

**AdaBoosting**

- *AdaBoosting* is a popular version of *Boosting*.
  - It constructs a composite classifier by sequentially training classifiers.
  - All the training examples are initially given equal weights that determine their selection for training (i.e., higher weight - higher chance to be used in training).
  - In an iteration $t$, the weighted error $E_t$ of the classifier used is distributed among the incorrectly classified examples. (i.e., correctly classified examples have their weights multiplied by $E_t^{-1}(1 - E_t)$ before all weights are renormalised to 1).
  - If $E_t$ is less than or equals 0.5, the weights are adjusted for the next iteration.
  - If $E_t$ equals 0, no more iterations are done and $t$ is the last iteration.
  - If $E_t$ is greater than 0.5, iteration $t-1$ was the last iteration.
  - The composite classifier is thus obtained by summing the votes of the trial classifiers with each weighted by $\log(1 - E_t) / E_t$. 


**AdaBoosting**

**Example**

- The weights attached to each instance in a given trial are re-adjusted for the next trial (with instances misclassified given greater weights)
- Consider the point (4,7)
- Here the class derived from each trial is weighted with \( \log(1 - E^t / E) \) with \( E^t \) being the sum of the weights of the misclassified instances in that trial.
- The weights for each class is tallied with the above formula and the class with the highest overall weight is the class that the composite (boosted) classifier classifies the instance (4,7) under.
- In this case it is '+'.

---

**AdaBoosting**

- If \( E^t \) is always less than 0.5, the error rate of the composite classifier on the given examples under the initial (uniform) distribution approaches zero exponentially quickly as the number of iterations increases.
- There is however no guarantee on the composite classifier's performance on unseen instance (possibly due to overfitting).
Auto Associative Networks

- Conventionally, neural networks used for classification are trained to map an input vector to an output vector that encodes directly the classes.
- Auto-associative neural networks, also known as autoencoders or Diabolo networks, instead learn to model each class.
- These networks are trained to approximate the original featurespace with less degrees of freedom than available in the original featurespace - done via a bottleneck layer.
- The distance between the input vector and the reconstructed output vector expresses the likelihood that a particular example is part of a corresponding class.

Diabolo Networks

- The architecture for a simple non-linear Diabolo network is as follows:
  - The data is compressed from $n$ to $m$ dimensions before it is reconstructed from $m$ to $n$ dimensions.
  - The learning algorithm typically used is the Backpropagation learning algorithm.
  - This mapping may either be supervised or unsupervised. In the case of unsupervised learning, class information is not used, instead, learning is done by approximating the values of the outputs to approximate the inputs.
  - In the simplest case, each Diabolo network is trained only with examples of a corresponding class.
**Diabolo Classifier**

- The architecture of the *Diabolo Classifier* that is used to classify the *characters* (inputs) is as follows:
  - The input and output vectors are points referring to the sequences pertaining to a character. Following the sequence of points will form the character.
  - The objective function (provides a performance/distance measure) utilises the *tangent-distance*.
  - In order to learn, the network weights are updated so that the reconstruction distance is minimised for the network of the desired class and maximised for the closest incorrect one - similar to *LVQ2*.

  * See handout for more details.

**Experimental Setup**

- In order to accommodate a variety of different writing styles, the architecture is extended by using several *autoencoders* per class, each specialising a particular writing style.
- The *Adaboosting* algorithm was modified slightly where resampled examples from the training set, random local affine transformations were done to the original characters.
**Experimental Results**

- The unboosted classifier error rates are as follows:

<table>
<thead>
<tr>
<th></th>
<th>Diabolo Classifier</th>
<th>Fully connected MLP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No Subclasses</td>
<td>Hand-selected</td>
</tr>
<tr>
<td>Training Error</td>
<td>2.2%</td>
<td>0.6%</td>
</tr>
<tr>
<td>Testing Error</td>
<td>3.3%</td>
<td>1.2%</td>
</tr>
</tbody>
</table>

- The Diabolo classifier performed best when combining 16 classifiers where the error of 1.4% was close the Diabolo classifier that used the hand-selected subclasses.

- Although AdaBoost significantly improves the generalization error of the MLPs (e.g. 22-10-10 went from 9.5% to 2.4% error), these results were not as good as the boosted Diabolo classifier.

**Conclusion**

- AdaBoost was found to bring training error to zero after only a few steps (even with MLP with only 10 hidden neurons).

- The generalization error is also considerably improved and continues to decrease asymptotically after zero training error is reached.

- AdaBoost can significantly improve neural classifiers such as multi-layer networks and Diabolo networks.

- This behaviour confirms previous observations on other learning algorithms.

- In terms of character recognition, after boosting, the Diabolo classifier performed better than the MLP networks.
Questions

- Please feel free to ask any questions regarding this presentation

This presentation is based on the paper:

*Adaboosting Neural Networks: Application to on-line Character Recognition* (1997)

by

Holger Schwenk and Yoshua Bengio