Gated Classifiers: Boosting under High Intra-Class Variation
Oscar Danielsson, Babak Rasolzadeh and Stefan Carlsson
School of Computer Science and Communication, Royal Inst. of Technology, Stockholm, Sweden
{osda02,babak2,stefanc}@csc.kth.se

1. Motivation
Combinations of weak classifiers that never occur together on any example of the target class may generate false positives.

2. Definition

\[ h_{\text{NAND}}(x) = \neg \left( h_1(x) \wedge \ldots \wedge h_j(x) \right) \]

\[ h_{\text{XOR}}(x) = \left( h_1(x) \wedge \ldots \wedge h_j(x) \right) \oplus \left( h_1(x) \wedge \ldots \wedge h_j(x) \right) \]

Where \( j, \ldots, j \in \{1, \ldots, m\} \) and \( m \) is the current round of boosting and \( n \) is the maximum size of the NAND (XOR) classifier.

3. Learning
Using Gated Classifiers with AdaBoost requires only very small changes to existing boosting code.

**Algorithm 2: AdaBoost with Gated Classifiers**

- **Require:** \( \{(x_i, y_i) \in \mathcal{X} \times \{0, 1\}\} \)
- **Initialize:** \( D_0(i) = 1/|\mathcal{X}| \)
- For \( t = 1 \) to \( T \) do:
  - **Train** \( h_t: \mathcal{X} \rightarrow \{0, 1\} \) using distribution \( D_t \)
  - **Train** \( h_{\text{NAND}} = \neg \bigwedge_{j \in J} h_j(x) \rightarrow y \)
  - **Train** \( h_{\text{XOR}} = \bigoplus_{j \in J} h_j(x) \rightarrow y \)
  - \( h'_t = \text{the best of } h_t, h_{\text{NAND}}, h_{\text{XOR}} \)
  - Choose \( \alpha_t \in \mathbb{R} \)
  - \( D_{t+1} = \text{update weight distribution} \)
- **Return** \( \{\alpha_1, \ldots, \alpha_T\}, \{h'_1, \ldots, h'_T\} \)

A NAND (or XOR) classifier is learnt by:
1. Enumerating all small NAND (XOR) classifiers, typically pairs of previously learnt weak classifiers, and selecting the best.
2. Attempting to extend the NAND (XOR) classifier by greedily appending more weak classifiers

This greedy algorithm has complexity \( O(T^2) \), where \( T \) is the current round of boosting and \( N \) is the maximum size of the NAND (XOR) classifier.

If gated classifiers are constructed from at least one other gated classifier, gate networks result:

4. Experiments and Results

**Synthetic data:** Increasingly difficult datasets generated by adding more components to Gaussian Mixture generating positive examples. A pool of weak classifiers split the \( x \) and \( y \) axes into a set of intervals. By adding first tautology and falsum classifiers and then NAND and XOR classifiers we can better represent complex distributions.

**Pedestrians:** Evaluating representational power on NICTA pedestrian dataset (training set A). Hard negative examples extracted by training a few stages of a cascade detector and using false positive detections.

Using gated classifiers makes the weak learner more flexible and thus increases the risk of over-fitting. We can compute the VC-dimension of the weak learner with gated classifiers (where \( d_g \) is the VC-dimension of the basic weak classifier):

\[ d_g + 5.5 \]

An alternative method for generically handling large intra-class variation would be to use stronger weak classifiers, e.g. deep decision trees, templates. However, these approaches often over-fit.

Gated classifiers lend themselves readily for hardware implementation.

5. Discussion

6. Conclusion

- Using gated classifiers reduces the number of weak classifiers necessary to represent a complex decision boundary.
- Gated classifiers are easy to implement both in software and hardware (FPGA) and can be used to empower any weak learner.
- The VC-dim. when using gated classifiers has a tight bound in terms of the VC-dim. of the basic weak learner.