Recognition Labs – AdaBoost

Jan Šochman, translation Petr Doubek

December 4, 2009

1 Introduction

AdaBoost lineraly combines decisions of several "simple" *weak* classifiers and thus can achieve better results than using the weak classifiers alone.

2 AdaBoost

AdaBoost is a learning algorithm with a training set x, y on the input and binary classifier on output. The resulting classifier $H(x)$ is a linear combination of T weak classifiers $h_t(x)$

$$H(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

In each iteration the linear combination is extended by one weak classifier from the set $\mathcal{H}$. The choice of weak classifier at each iteration is done in a greedy way, so that the upper bound of classifier error is minimized.

The learning is sumarized in Algorithm 1. The training set is composed of pairs (x_i, y_i), where x_i is measurement and y_i is class of the sample $\{-1, 1\}$. AdaBoost weights the importance of the samples by weights D_t, which are initialized uniformly.

The algorithm runs in a loop, doing the following steps at each iteration:

1. find best weak classifier given the current weights D_t of training samples
2. verify that the error is less than 0.5 (i.e. better than random)
3. calculate coefficient α_t of the weak classifier in linear combination $H(x)$,
4. update weights D_t.

Input: $(x_1, y_1), ..., (x_m, y_m)$; $x_i \in \mathcal{X}, y_i \in \{-1, 1\}$
Initialize weights $D_1(i) = 1/m$
For $t = 1, ..., T$:

1. Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j$; $\epsilon_j = \sum_{i=1}^{m} D_t(i) I[y_i \neq h_j(x_i)]$
   to obtain weak classifiers $h_j(x)$, use provided function `findThetaPar()`
2. If $\epsilon_t \geq 1/2$ then stop
3. $\alpha_t = \frac{1}{2} \log(\frac{1-\epsilon_t}{\epsilon_t})$
4. Update
$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$
   kde $Z_t = \sum_{i=1}^{m} D_t(i) \exp(-\alpha_t y_i h_t(x_i))$

Resulting classifier:
$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

Algorithm 1: Discrete AdaBoost.

In step 1, we select a weak classifier whose weighted error on the training set is minimal. Term $I[\textit{statement}]$ is 1 if *statement* is true and 0 if it is false.

The test in step 2 ensures, that the found weak classifier is better than a random choice. It is necessary to guarantee the convergence of the learning algorithm.

The weight update in step 4 increases the weight of misclassified samples and decreases weight of correctly classified samples. Therefore in next iteration, such weak classifier will be prefered, which is better in classification of so far misclassified samples.