Weighted Sampling for Large-Scale Boosting

Zdenek Kalal[1] Jiri Matas[2] Krystian Mikolajczyk[1]
[1]University of Surrey
Guildford, UK
{z.kalal,k.mikolajczyk}@surrey.ac.uk
[2]Czech Technical University
Prague, Czech Republic
matas@cmp.felk.cvut.cz

Abstract

This paper addresses the problem of learning from very large databases where batch learning is impractical or even infeasible. Bootstrap is a popular technique applicable in such situations. We show that sampling strategy used for bootstrapping has a significant impact on the resulting classifier performance. We design a new general sampling strategy "quasi-random weighted sampling + trimming" (QWS+) that includes well established strategies as special cases. The QWS+ approach minimizes the variance of hypothesis error estimate and leads to significant improvement in performance compared to standard sampling techniques. The superior performance is demonstrated on several problems including profile and frontal face detection.

1 Introduction

With the information available on the Internet, it is possible to acquire training data sets (pools) of virtually unlimited sizes for many problems in visual recognition. Given a fixed complexity of a classifier, generalization improves with a training set size [18]. Hoverer due to computational limitations and training complexity, it is rarely the case that the entire pool can be processed at once. Methods that are able to process large pools effectively are thus of increasing significance.

One possible solution is to restrict the learner to much smaller *active subset* of the pool and proceed iteratively. In every iteration, the training is performed on the active set and returns a preliminary classifier. The classifier is then used to evaluate the pool and to collect new, more informative active set. This technique is commonly called bootstrapping[1] and proved to be an essential component in many object detection systems based on batch learning [17, 11, 13, 15, 19].

Besides the bootstrapping, on-line learning algorithms exist that are naturally suitable for large datasets [10, 5]. Even though the on-line approach demonstrates some interesting applications, the majority of current object detection algorithms is based on batch learning, where bootstrapping plays an important role.

The bootstrap was first introduced by Sung and Poggio [17]. In every round of training, the active set is *extended* by examples misclassified in previous round, thus emphasizing samples close to the decision boundary. Later Papageorgiou et al. [11] and Dalal

[1]In the statistical literature [2], the term bootstrapping refers to a re-sampling technique used to obtain properties of an estimator such as its variance.

and Triggs [1] applied bootstrap to refine the decision surface of an SVM classifier, Rowley et al. [13] to train the neural network, and Schneiderman and Kanade [15] to estimate the histograms of the part-based detector.

Viola and Jones [19] published the first real-time face detection system based on a cascaded classifier composed of boosted stages. A bootstrapping strategy based on *re-sampling* of the undecided parts of the pool was used. Many authors follow this approach [8, 6, 7, 22]. A number of improvements has been proposed for the classical Viola-Jones cascade. For example, the standard cascade ignores the confidence which is output from each stage thus making the classifier unnecessarily long. This drawback was addressed in [16, 20, 21] where bootstrapping based on *re-sampling* and *re-weighting* of the selected samples according to the confidence was used. Later on, Fleuret and Geman [3] proposed the *weighted sampling*. Instead of weighting examples in the active set, examples were sampled with replacement from the pool with respect to their weights. The approach let to improved classifier performance.

Apart from iterations on the Viola and Jones detector, Friedman et al. [4] suggested *trimming*, a technique that selects only a fraction of examples with the highest weights resulting in improvement of training speed.

We argue and show experimentally that sampling strategy used in bootstrap significantly influences the performance of the final classification system. We build on the weighted sampling proposed by Fleuret and Geman [3] and further improve its performance by quasi-random sampling. We propose a generalized strategy based on *quasi-random weighted sampling + trimming* (QWS+) that optimally combines its components in order to minimize the variance of hypothesis error in each round of training. The QWS+ sampling method is applied to real AdaBoost [14] and leads to significant increase in the classification performance as well as the training efficiency on large datasets compared with standard sampling strategies.

The rest of the paper is organized as follows. Section 2 formalizes the concept of bootstrapping and discusses the well established and proposed sampling strategies. Large scale experiments in the context of face detection are presented in Section 3. The paper is concluded in Section 4.

2 Bootstrap for AdaBoost

Bootstrapping is a general technique that is applicable in conjunction with many learning algorithms. We focus on the real AdaBoost and build on the fact that AdaBoost iteratively calls a weak learner on a weighted training set. We suppose that the training set (pool) is large and calling the weak learner directly is in-practical. Yet, it is possible to maintain the distribution of weights on the pool. In every iteration, the pool is approximated by an active set of a specified size obtained by some *sampling strategy*. The active set is then used as an input to the weak learner that returns a hypothesis, which updates the weights in the pool and the training continues. Figure 1 illustrates the process.

Suppose we have a pool of size M composed of pairs $X = \{x_i, y_i\}$, where $x_i \in \mathscr{X}$ is an observation and $y_i \in \{-1, 1\}$ is a label. Next we have a set of weights assigned to each example $D = \{d_i\}$ satisfying $\sum_{i=1}^{M} d_i = 1$. For the sake of simplified notation, we consider the pool as triples $S = \{x_i, y_i, d_i\}$. Standard AdaBoost processes this pool in T rounds. In

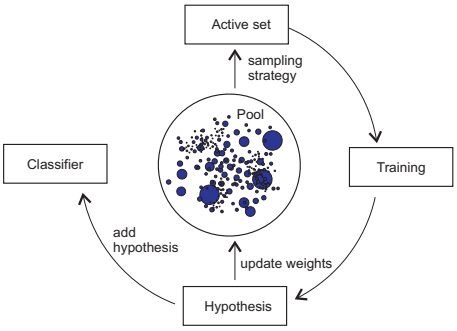

Figure 1: AdaBoost with bootstrapping based on re-sampling of the active set.

each round, a hypothesis h^* minimizing the upper bound of error Z on pool S is found:

$$h^* = \arg\min_h Z(S,h) = \arg\min_h \sum_{i=1}^{M} d_i e^{-y_i h(x_i)}. \tag{1}$$

As stated before, this process can be very time consuming.

If bootstrap is employed, the pool S is first subsampled to form an active set $\hat{S} = \{x_j, y_j, \hat{d}_j\}$ of size N with weights satisfying $\sum_{j=1}^{N} \hat{d}_j = 1$. The AdaBoost learner then finds an approximate hypothesis $\hat{h}$ minimizing the upper bound of error on $\hat{S}$:

$$\hat{h} = \arg\min_h Z(\hat{S},h) = \arg\min_h \sum_{j=1}^{N} \hat{d}_j e^{-y_j h(x_j)}. \tag{2}$$

This process is usually much faster since the parameter N is chosen according to the computational resources available, e.g. the size of available memory.

In the above equations, we use subscript i for indexing samples in the pool. The sampling strategy assigns each sample the estimated weight $\hat{d}_i$. Samples with the estimated weight bigger than zero are added to the active set. The estimated weight of sample i is a random variable with expectation $\mathrm{E}[\hat{d}_i]$ and variance $\mathrm{Var}[\hat{d}_i]$. Note that the randomness is here considered over realizations of the sampling process. If $\mathrm{E}[\hat{d}_i] = d_i$ we consider the estimated weights unbiased. Suppose we have a hypothesis h with some error upper bound on the pool $Z(S,h)$. In training, we do not know this value but we are estimating it on the active set $\hat{S}$ according to the following equation:

$$Z(\hat{S},h) = \sum_{i=1}^{M} \hat{d}_i e^{-y_i h(x_i)} \tag{3}$$

where the estimated weights are linearly combined by the constant coefficients $e^{-y_i h(x_i)}$. It follows that if the estimated weights $\hat{d}_i$ are unbiased so is the estimated error upper bound. If $\mathrm{E}[Z(\hat{S},h)] = Z(S,h)$ we consider the entire strategy unbiased. $\mathrm{Var}[Z(\hat{S},h)]$ will be referred to as a *variance of error estimate*.

Problem formulation. Our objective is to find the sampling strategy that selects N unique samples out of the pool of size M, is unbiased and with minimal variance of error

estimate. Such strategy would guarantee that the approximate hypothesis $\hat{h}$ is as close as possible to the optimal one h^*, which would be trained on the entire pool.

2.1 Commonly used strategies

Trimming (T). This technique selects N samples from the pool with the highest weights. The weights of the selected samples are then normalized so that $\sum_i \hat{d}_i = 1$. This strategy was introduced in [4], where N was set so that a predefined fraction $(90 - 99\%)$ of the total weight mass was used for training. If $N < M$ then the weights are biased $\mathrm{E}[\hat{d}_i] \neq d_i$ with zero variance.

Unique uniform sampling (UUS). This strategy selects N unique samples from the pool with a probability of selecting sample i equal to $P(i) = \frac{1}{M}$. The weights of the selected samples are then normalized so that $\sum_i \hat{d}_i = 1$. The estimated weights are in general biased (due to normalization) and have high variance since it is likely to disregard examples that carry significant mass of the distribution. Despite this it is often used in practice [16, 20, 21] for its simplicity.

Weighted sampling (WS). Selects N samples with replacement from the pool. Sample i is selected with probability $P(i) = d_i$ and assigned weight $\hat{d}_i = \frac{1}{N}$. WS is unbiased, since $\mathrm{E}[\hat{d}_i] = \frac{N d_i}{N} = d_i$. Possible implementation: the example weights are represented as intervals arranged on a unit line segment. Next we generate N random points on this line segments which positions determine the index of the selected sample. Since the example weight are approximated by repeated selection of the sample, this strategy does not guarantee N unique samples in the active set. This strategy was proposed in [3].

2.2 Proposed strategies

In this section, we propose new sampling strategies based on WS. These strategies are designed to reduce the variance of error estimate by quasi-random sampling and guarantee selection of N unique samples to the active set.

Quasi-random weighted sampling (QWS). QWS reduces variance of the error estimate by pseudo-random sampling [12]. The weights are represented as intervals and arranged to a unit line segment. The line segment is split into N equal intervals. Within each interval, one random number is generated which position determines the index of the selected sample. This process maintains the weights unbiased but significantly reduces their variance. Intuitively, each sample can be selected at most n-times, where n is number of intervals the sample is sitting on. For standard WS, each sample can be selected at most N-times. Since $n \ll N$, QWS significantly reduces the variance of the estimated weights. QWS selects a unknown number of unique samples, which leads to inefficient usage of available capacity of the active set.

Quasi-random weighted sampling + trimming (QWS+). The QWS+ is a generalization of trimming and QWS, parameterized by K, the number of samples with largest weights in the pool that are trimmed. For $K = N$, QWS+ coincides with trimming. For

$K = 0$, QWS+ becomes a QWS. For any $0 \leq K < N$, the QWS+ strategy is unbiased, since weighted sampling is unbiased and calculations of error upper bound on the trimmed set is exact. Parameter K is set to minimize the variance of error estimate.

Intuitively, QWS+ selects K most dominant samples that would be selected by QWS with probability at lest 50% and assigns them their own weights. The remaining weight mass is distributed amongst the remaining $L = N - K$ samples selected by QWS. Trimming of the most dominant samples reduces the variance of their estimated weight and hence the entire error estimate.

In the following theorem, we show the optimal setting of the parameter K. We use the following notation. Index in parenthesis denotes (i)-th sample with largest weight. $S_K = S \setminus \{x_{(i)}, y_{(i)}, d_{(i)}\}_1^K$ is the pool minus K samples with largest weights, $D_K = 1 - \sum_{i=1}^K d_{(i)}$ is the weight mass of S_K.

Theorem: The optimal size of trimming is the largest $K = N - L$ that satisfies the condition $\frac{1}{L} \leq \frac{d_{(K)}}{D_K}(2 + \frac{d_{(K)}}{D_K})$ if variance of error upper bound of h changes "slowly" (defined below).

Proof: First, we express the variance of QWS+ estimator as a function of K. In the QWS+, the error upper bound for a given hypothesis h on the pool S is estimated by

$$\hat{Z}_{QWS+}(S,h) = \sum_{i=1}^{K} d_{(i)} e^{-y_{(i)} h(x_{(i)})} + \hat{Z}_{QWS}(S_K, h) \tag{4}$$

$$\approx \sum_{i=1}^{K} d_{(i)} e^{-y_{(i)} h(x_{(i)})} + \hat{Z}_{WS}(S_K, h), \tag{5}$$

where $\hat{Z}_{QWS}$ is a random variable representing error upper bound estimated by QWS strategy on S_K. For the purpose of the proof, we assume that $\hat{Z}_{QWS}(S_K,h) \approx \hat{Z}_{WS}(S_K,h)$ which results in simplified analysis. The difference between sampling with/without replacement is in this case very small since all K dominant samples were already removed by trimming.

WS selects $L = N - K$ samples with replacement from S_K. Sample i is selected with probability $P(i) = \frac{d_i}{D_K}$ and assigned weight $\hat{d}_i = \frac{D_K}{L}$. The random variable $\hat{Z}_{WS}$ is thus a re-scaled sum of L i.i.d. random variables corresponding to each sample:

$$\hat{Z}_{WS} = \frac{D_K}{L} \sum_{s=1}^{L} \hat{Z}_{WS}^s, \tag{6}$$

where random variables $\hat{Z}_{WS}^s(S_K,h)$ attain the value $e^{-y_{i(s)} h(x_{i(s)})}$ with probability $\frac{d_{i(s)}}{D_K}$; $i(s)$ is the index chosen in step s.

From Eq. (5) we can see that variance of $\hat{Z}_{QWS+}$ can be approximated by $\hat{Z}_{WS}$ since a trimming has variance equal to zero. Variance of $\hat{Z}_{WS}$ can be substituted from Eq. (6) as:

$$\mathrm{Var}[\hat{Z}_{QWS+}] \approx \mathrm{Var}[\frac{D_K}{L} \sum_{s=1}^{L} \hat{Z}_{WS}^s] = \frac{D_K^2}{L} \mathrm{Var}[\hat{Z}_{WS}^s]. \tag{7}$$

Frontal faces (S_{front})	3085 profile faces from the Internet
Google faces (S_{Google})	9771 multi-view faces from Google face search
Profile faces (S^{β}_{prof})	3384 left profile faces from 8 movies rotated by β degrees in plane from upright position
Background (S_{back})	3310 images without target concept

Table 1: Training databases used in our experiments.

The variance of the individual samples $\mathrm{Var}[\hat{Z}^s_{WS}(S_K,h)]$ depends on S_K and the assessed weak classifier h. However, the selection of the active set cannot depend on the weak classifier for efficiency reasons. Next, we assume that variance of individual samples changes slowly as a function of K, i.e. $\mathrm{Var}[\hat{Z}^s_{WS}(S_K,h)] = c_K$ and $\frac{c_K}{c_{K-1}} \approx 1$. In the following, we express the optimality condition that variance of QWS+ estimator for K must be smaller than for $K-1$ (trimming one more sample must reduce the variance):

$$\begin{aligned}
\mathrm{Var}[\hat{Z}^K] &\leq \mathrm{Var}[\hat{Z}^{K-1}] \\
\frac{D_K^2}{L} c_K &\leq \frac{D_{K-1}^2}{L+1} c_{K-1} = \frac{(D_K + d_{(K)})^2}{L+1} c_{K-1} \;,\; c_K/c_{K-1} \approx 1 \qquad (8) \\
\frac{L+1}{L} &\leq \frac{(D_K + d_{(K)})^2}{D_K^2} \qquad (9) \\
\frac{1}{L} &\leq \frac{d_{(K)}}{D_K}\left(2 + \frac{d_{(K)}}{D_K}\right) \approx \frac{2 d_{(K)}}{D_K} \qquad (10)
\end{aligned}$$

In practice, the optimal K is found iteratively. Samples are trimmed until the Eq. (10) is satisfied.

3 Experiments

This section presents large scale experiments on real data. The training is performed on several databases as shown in Table 1. The testing is performed on standard CMU-MIT frontal/profile test set and compared with other authors when possible. The detectors are compared by ROC and Precision-recall curves (PRC) .

We use the real AdaBoost [14] to train a hypothesis that is added to a modified WaldBoost classifier [16]. Thresholds for classification are given by Wald's criterion [16] but decision is not done after each weak classifier but only when the upper bound drops under some specified threshold. The reason is speedup of training with little influence on the resulting performance. Tree types of local image features are used: Haar-like features [6], Local Binary Patterns [9, 22] (LBP) and Histogram of Orientated Gradients [1] (HoG) projected into one dimension using weighted LDA [7]. All the features are localized within a reference frame and can be computed for arbitrary scale and aspect ratio. We generate a set of approximately $12,000$ Haar, $12,000$ LBP and 400 HoG features for image frame of size 28×28 pixels.

In Section 3.1 we verify our theoretical results about bias and variance for all discussed strategies. In Section 3.2 we investigate the impact of the relative active set size on the classifier performance. And finally, real profile/specific detectors will be trained in Section 3.3 and compared with other approaches.

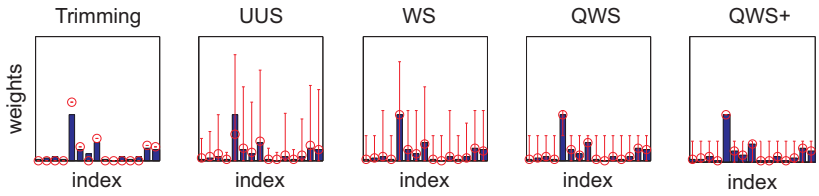

Figure 2: Pool with weights d_i (blue bars) is repeatedly sampled to obtain a distribution of the estimated weights which range (1 and 99% quantiles) is depicted by red vertical lines and expected values by circles.

3.1 Properties of sampling strategies

Estimation of weights. In this experiment, we test our claims about bias and variance of estimated weights for different strategies. We generated a pool of size $M = 15$ with randomly assigned weights d_i. The active set of size $N = 5$ was sampled 1000 times, each realization outputs a set of estimated weights $\hat{d}_i$. The expectation and variance of $\hat{d}_i$ are shown in Figure 2. The circles in the figure denote the expectation of the estimated weight. Notice that the expectation of Trimming and UUS is biased, the remaining strategies are not. The variance of the estimated weights is depicted by the red error bars. Note the reduction of variance by quasi-random sampling (QWS, QWS+). Furthermore, WS+ completely eliminates variance on large examples thus rendering the variance of all weights minimal.

Estimation of error upper bound. In this experiment, we created a pool S containing $M = 95,000$ examples with distribution of weights after the 100th iteration of training a frontal face detector. The 100th iteration was chosen because the distribution is dominated by large samples but still there are many samples in the pool allowing statistical evaluation. We first trained the optimal hypothesis h^* on whole pool S with corresponding upper bound $Z(S,h^*)$. The pool was than sampled by each strategy to obtain 1000 active sets $\hat{S}$ of size $N = 500$. Next, the optimal hypothesis was tested on each active set in order to obtain its approximated upper bound $\hat{Z} = Z(\hat{S},h^*)$. The expectation and variance of $\hat{Z}$ is shown in the left panel of Figure 3. Weighted sampling reduces significantly the variance of $\hat{Z}$ and keeps mean unbiased. Intuitively, the error is dominated by examples with high weights, weighted sampling forces these examples to be included in every realization of $\hat{S}$, hence the error in different trials is similar. These results indicate that the weighted sampling leads to an accurate estimate of the error using only a small active set.

Overfitting and generalization. In this experiment we use the same sets as defined in previous paragraph. We first trained the optimal hypothesis h^* on full pool S with corresponding upper bound $Z(S,h^*)$. The hypothesis was trained by domain partitioning approach [14]. Next we trained an approximate hypothesis $\hat{h}$ on the active sets $\hat{S}$ with training upper bound $\hat{Z} = Z(\hat{S},\hat{h})$ (Figure 3, middle) and tested them on the full pool (Figure 3, right). By comparing the two panels we see that UUS sampling overfits the training data much more than any other strategy. Further more, the variance of UUS is very high. The variance is significantly reduced by weighted sampling; quasi-random

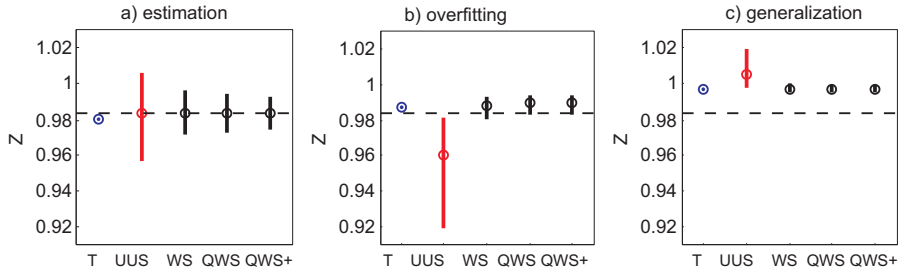

Figure 3: Accuracy of hypothesis obtained by different sampling strategies. Black horizontal dashed line represents the error of the optimal hypotheses h^* tested on the full pool. Vertical lines represent range of errors between 5 and 95% quantiles, the circles denote the expectations. *a)* The error of the optimal hypothesis is estimated on the active sets. *b)* The hypothesis is trained and tested on the active sets (overfitting). *c)* The hypothesis is trained on the active set and tested on the full pool (generalization error).

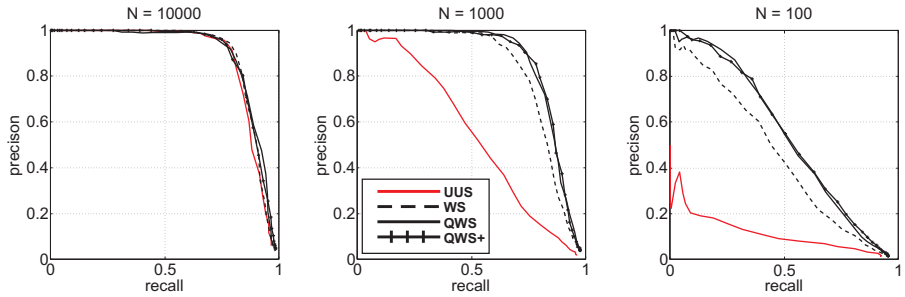

Figure 4: Dependence of classifier performance on the relative active set size and the sampling strategy.

weighted sampling further reduces the variance just slightly in this case. This apparently insignificant difference in the variance reduction proves to be of importance, when full detector training in evaluated.

3.2 Influence of relative active set size

In this experiment, we investigated the influence of relative active set size on classifier performance for different sampling strategies. We used the following pool: 10.000 positive samples generated from S_{front} and 56 million background patches generated from S_{back}. This pool was approximated in training by active sets of three different sizes $N \in \{100, 1.000, 10.000\}$. For each size of the active set, four classifiers were trained using different sampling strategy (UUS, WS, QWS, QWS+). Trimming was also tested but its performance was far worse than any other strategy so we do not compare it with the others. The classifiers were trained up to the length 200, resampling of the active set was performed after each weak classifier.

Figure 4 shows the resulting PRC curves on CMU-MIT dataset. The performance of all sampling strategies decreases with smaller active set size. For sufficiently large

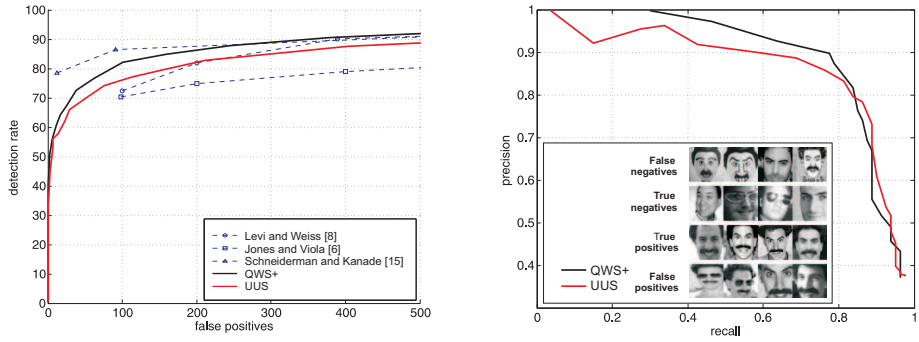

Figure 5: *Left:* ROC curves on CMU-MIT profile database (368 profile faces in 208 images). *Right:* PRC for specific face detector applied on the results of Google face search.

active sets ($N = 10.000$) there is no difference in performance. For smaller active sets ($N = 1000$) the UUS sampling performs significantly worse than the others. WS is in this case slightly worse than QWS and QWS+. For extremely small active sets ($N = 100$) the difference increases. This clearly demonstrate the benefit of QWS and QWS+. The difference between QWS and QWS+ is in this case on a noise level only. The active set size has also impact on training efficiency. While training with $N = 10.000$ takes 3 hours, training with $N = 100$ requires 1 hour on the same machine.

3.3 Real detectors

Face profile. In order to compare our system with other approaches a profile detector was trained with the following setting. The pool of $41,000$ positive and 424 million negative patches was generated from S^0_{prof}, S^{15}_{prof}, S^{30}_{prof} and S_{back}. Two left-profile classifiers of length 2000 were trained with QWS+ and UUS strategy. The right profile detectors were obtained by flipping of the model.

The resulting ROC curves are displayed in the left panel of Figure 5. QWS+ has detection rate by 5% higher than UUS for 100 false positives. On the same level of false positives, our detector performs better than the results in [8] and [6] and slightly worse than [15]. The average number of evaluated weak classifiers is 5.6 for QWS+ and 7.2 for UUS which shows positive influence of weighted sampling on the classifier speed.

Specific face detector. Google face search was used to collect 650 images returned for query "borat". This set contained 350 images of Borat (fictional Kazakhstani journalist) the remaining images contained other faces or background. The Borat images were split into training (250) and testing (100) set, all of the non-borat images became a part of the testing set. A specific face detector was trained and evaluated on the collected data.

The resulting precision recall curve is presented in the right panel of Figure 5. Also in this task, the performance of QWS+ is superior to UUS. Our specific face detector successfully finds approximately 80% of faces with 90% precision which shows a scope for possible improvement of Google's face search. Interestingly, the false positives contain some paintings of Borat and the Borat's protagonist Sacha Baron Cohen.

4 Conclusions

In this paper, two improvements of standard weighted sampling (WS) are introduced. We design quasi-random weighted sampling (QWS) which reduces the variance of error estimate but does not guarantee unique samples. Next, we introduce a new strategy based on quasi-random weighted sampling + trimming (QWS+) which fixes this drawback. We provide the theoretical proof of the variance minimization of QWS+.

The superiority of our strategies is demonstrated on large-scale learning experiments as well on learning problem with a small dataset. Different commonly used sampling strategies are extensively evaluated in the context of frontal and profile face detection.

Various characteristics of the learning process are improved: reduction of the required active training set, better precision/recall curves and speed of the resulting classifier. In future work the proposed strategy will be extended to other learning methods.

Acknowledgment. This research was supported by EU VIDI-Video IST-2-045547 and UK EPSRC EP/F0034 20/1 grants (ZK, KM) and by Czech Science Foundation project 201/06/1821 (JM).

References

[1] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *CVPR*, 2005.

[2] B. Efron and R. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall/CRC, 1993.

[3] F. Fleuret and D. Geman. Stationary features and cat detection. Technical report, IDIAP, 2007.

[4] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, 2000.

[5] H. Grabner and H. Bischof. On-line boosting and vision. *CVPR*, 2006.

[6] M. Jones and P. Viola. Fast multi-view face detection. Technical report, Mitsubishi Electric Research Lab, 2003.

[7] I. Laptev. Improvements of object detection using boosted histograms. *BMVC*, 2006.

[8] K. Levi and Y. Weiss. Learning object detection from a small number of examples: the importance of good features. *CVPR*, 2004.

[9] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *PAMI*, 24(7):971–987, 2002.

[10] Nikunj .C Oza. Online bagging and boosting. *International Conference on Systems*, 2005.

[11] C. P. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. *ICCV*, 1998.

[12] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C*. Cambridge University Press, 1992.

[13] HA Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *PAMI*, 20(1):23–38, 1998.

[14] R. Schapire and Y. Singer. Improved boosting using confidence-rated predictions. *Machine Learning*, 1999.

[15] H. Schneiderman and T. Kanade. Object detection using the statistics of parts. *IJCV*, 2004.

[16] J. Sochman and J. Matas. Waldboost: learning for time constrained sequential detection. *CVPR*, 2005.

[17] K.K. Sung and T. Poggio. Example-based learning for view-based human face detection. *PAMI*, 20(1):39–51, 1998.

[18] V.N. Vapnik. *Statistical learning theory*. Wiley New York, 1998.

[19] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *CVPR*, 2001.

[20] B. Wu, H. Ai, C. Huang, and S. Lao. Fast rotation invariant multi-view face detection based on real adaboost. *AFGR*, 2004.

[21] R. Xiao, L. Zhu, and H.J. Zhang. Boosting chain learning for object detection. *ICCV*, 2003.

[22] L. Zhang, R. Chu, S. Xiang, S. Liao, and S.Z. Li. Face detection based on multi-block lbp representation. *ICB*, 2007.