AdaBoost with Totally Corrective Updates for Fast Face Detection

Jan Šochman Jiří Matas
Centre for Machine Perception, Dept. of Cybernetics, Faculty of Elec. Eng.
Czech Technical University in Prague, 166 27 Prague 6, Technická 2, Czech Rep.
{sochmj1,matas}@cmp.felk.cvut.cz

Abstract

An extension of the AdaBoost learning algorithm is proposed and brought to bear on the face detection problem. In each weak classifier selection cycle, the novel totally corrective algorithm reduces aggressively the upper bound on the training error by correcting coefficients of all weak classifiers. The correction steps are proven to lower the upper bound on the error without increasing computational complexity of the resulting detector. We show experimentally that for the face detection problem, where large training sets are available, the technique does not overfit.

A cascaded face detector of the Viola-Jones type is built using AdaBoost with the Totally Corrective Update. The same detection and false positive rates are achieved with a detector that is 20 % faster and consists of only a quarter of the weak classifiers needed for a classifier trained by standard AdaBoost. The latter property facilitates hardware implementation, the former opens scope for the increase in the search space, e.g. the range of scales at which faces are sought.

1. Introduction

Face detection has numerous applications and a range of algorithms has been proposed [8, 10, 12, 4]. In many applications, real-time performance is required. Recently, Viola and Jones [12] introduced an impressive face detection system capable of detecting faces in real-time with both high detection rate and very low false positive rates. The desirable properties are attributed especially to the efficiently computable features used, the AdaBoost learning algorithm, and a cascade technique adopted for decision making. In this paper, an improvement of the AdaBoost algorithm is proposed and its utility for cascade building in the context of face detection is shown.

The Viola and Jones detector consists of several classifiers trained by the AdaBoost algorithm [1] that are organised into a decision cascade. Each cascade stage classifier is set to reach a very high detection rate and an "acceptably" low false positive rate. Since it is trained on the data classified as a face by the previous stages, the final false positive rate is very low (equal to the product of false positive rates of all stages, see Algorithm 3) and the final detection rate remains high.

The cascade evaluation is equivalent to a sequential classification using a degenerated decision tree. When the current stage classifier labels a region in an image as a non-face, the decision process is terminated. Otherwise, the next stage classifier is run. A region is declared a face if it is accepted by all classifiers in the cascade.

Face detection is done by moving the cascade detector across the image at multiple scales and locations. A typical image contains only a small number of face regions compared to the number of regions scanned. Due to early termination of the decision process in non-face regions, only few stages of the cascade are evaluated on average [12, 5]. Hence, the speed of evaluation depends heavily on the computational complexity and rejection rates of the first few stages. The enhanced AdaBoost learning algorithm proposed in this paper produces a classifier that, for a given detection and false positive rates, is more likely to make a decision early in the evaluation of the cascade.

AdaBoost constructs the classifier as a linear combination of "weak" classifiers chosen from a given, finite or infinite, set. Its goal is to choose a small number of weak classifiers and assign them proper coefficients. The linear combination can be seen as a decision hyper-plane in the weak classifier space. Hence, AdaBoost can be viewed as an optimization procedure, that operates in the space of weak classifier coefficients, starting with a zero vector and ending with a vector with only small number of non-zero elements.

The standard (discrete) AdaBoost is a greedy algorithm, that in each step sets one zero-valued coefficient to a non-zero value. Because of its greedy character, neither the found weak classifiers nor their coefficients are optimal.

A totally corrective algorithm with coefficients updates (TCAcu) proposed in this paper differs from the standard AdaBoost in two main aspects. Firstly, the coefficients of already found weak classifiers are updated repetitively during the learning process. Secondly, in the standard Ada-

Boost, a newly added weak classifier can be shown to be "independent" in a precisely defined way of the previously added weak classifier. The TCAcu algorithm finds a new weak classifier that is independent of *all* weak classifiers selected so far. It is shown that these modifications minimise the classification error upper bound more aggressively and that shorter classifiers are found.

The term "totally corrective algorithm" was introduced by Kivinen and Warmuth [3]. However, the Kivinen and Warmuth algorithm did not update the coefficients of already found weak classifiers. The algorithm thus lost the important property of minimisation of the upper bound on the training error. Kivinen and Warmuth made no empirical evaluation of the algorithm. It was experimentally tested by Oza on several standard problems with poor results [6].

Another attempt to shorten the final classifier was proposed by Li et al. [4] and was motivated by the feature selection view of AdaBoost. In case, the weak classifiers correspond directly to the features as in the Viola and Jones face detection framework, changing one coefficient to a non-zero value effectively selects this feature [12]. Li et al. proposed FloatBoost, a modification of AdaBoost where some of already non-zero coefficients are set back to zero when it leads to a lower upper bound on the classification error. Instead of the greedy feature selection, the sequential floating forward selection (SFFS) technique [7] is used. Li et al. show that this modification leads to shorter classifiers.

The main contribution of this paper is (1) a modification of AdaBoost algorithm which leads to shorter classifiers and a speedup of classification, (2) the introduction of totally corrective algorithm to face detection training. It is shown that resulting classifier performance is comparable to the standard AdaBoost and the resulting classifier runs faster.

The paper is structured as follows. In the Section 2 the totally corrective algorithm with coefficients updates is described in the framework of the standard AdaBoost. Then, in Section 3, necessary details of the Viola and Jones work are given. Experimental results are shown in Section 4 and the paper is concluded in Section 5.

2. Totally corrective algorithm

In this section, the standard AdaBoost algorithm is described and motivation for the totally corrective step (TCS) is given. Then TCS is explained and its role in AdaBoost learning is discussed.

2.1. Standard AdaBoost

The totally corrective algorithm with coefficient updates (TCAcu) is based on AdaBoost [1] and its structure is depicted in Algorithm 1. Schapire and Singer's [9] notation is used and the algorithm differs from Schapire and Singer's

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

1. Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j$; $\epsilon_j = \frac{1}{2}[1 - \sum_{i=1}^{m} D_t(i) y_i h_j(x_i)]$
2. If $\epsilon_t \geq 1/2$ then stop
3. Set $\alpha_t = \frac{1}{2} \log(\frac{1-\epsilon_t}{\epsilon_t})$
4. Update
$$D_{t+1}(i) = \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$
5. *Totally corrective step* (see Algorithm 2)

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

Algorithm 1: TCAcu: Totally Corrective Algorithm with coefficient updates.

one only by an additional Step 5. The standard AdaBoost, i.e. Step 1 to 4, will be described first.

The goal of AdaBoost is to train a classifier using a set of examples. First, a weight $D_1(i)$ is assigned to each training example. Learning then proceeds in a simple loop. At time t, the algorithm selects a weak classifier h_t minimising a weighted error on the training set (Step 1). The loop is terminated if this error exceeds $1/2$ (Step 2). The value of α_t is computed next (Step 3) and the weights are updated according to the exponential rule (Step 4). In Step 4, Z_t is a normalisation factor which assures D_{t+1} remains a distribution. The final decision rule is a linear combination of the selected weak classifiers weighted by their coefficients. The classifier decision is given by the sign of the linear combination.

There are two properties of AdaBoost exploited in the paper. First, as has been shown in [9] the algorithm minimises an upper bound on the classification error $\varepsilon_{tr}(H)$ on the training set

$$\varepsilon_{tr}(H) \leq \prod_{t=1}^{T} Z_t = \frac{1}{2^T} \prod_{t=1}^{T} \sqrt{\epsilon_t(1-\epsilon_t)} \qquad (1)$$

This upper bound is minimised by selecting a weak classifier with the smallest weighted error ϵ_t on the training set as done in Step 1 and by setting its coefficient as done in Step 3.

Second, the re-weighting scheme assures that the up-

Initialize $\widehat{D}_0 = D_t$
For $j = 1, 2, ..., J_{\max}$

1. $q_j = \arg\max_{q=1..t} |\epsilon_q - 1/2|$.
2. If $|\epsilon_{q_j} - 1/2| < \Delta_{min}$ exit the loop.
3. Let $\widehat{\alpha}_j = 1/2 \ln((1 - \epsilon_{q_j})/\epsilon_{q_j})$.
4. Reweight

$$\widehat{D}_{j+1}(i) = \frac{1}{Z_j} \widehat{D}_j(i) \exp(-\widehat{\alpha}_j u_{q_j,i})$$

5. $\alpha_{q_j} = \alpha_{q_j} + \widehat{\alpha}_j$

Assign $D_{t+1} = \widehat{D}_j$

Algorithm 2: The Totally Corrective Step

dated distribution satisfies

$$\sum_{i=1}^{m} D_{t+1}(i) u_{t,i} = 0 \tag{2}$$

where $u_{t,i} = h_t(x_i) y_i$.

Step 1 of the AdaBoost algorithm at time $t + 1$ can be also written as

$$h_{t+1} = \arg\max_{h_q \in \mathcal{H}} \sum_{i=1}^{m} D_{t+1}(i) u_{q,i} \tag{3}$$

Employing equation (2) it is evident that the selected h_{t+1} is "maximally independent" of the mistakes made by h_t [9].

Moreover, for the weighted error ϵ_t^{t+1} of h_t where the upper index indicates that the error is measured on the weights used at time $t + 1$

$$\epsilon_t^{t+1} = \frac{1}{2}(1 - \sum_{i=1}^{m} D_{t+1}(i) u_{t,i}) = \frac{1}{2} \tag{4}$$

The weak classifier h_t is therefore equivalent to a random guess on the weights D_{t+1}.

Summarising equations (1) - (4), the AdaBoost algorithm minimises the upper bound on the classification error, selects weak classifiers with the smallest weighted error, and the selected weak classifier at time t is maximally independent of the mistakes made by the weak classifier selected at time $t - 1$.

2.2. Totally corrective step

The independence property discussed in Section 2.1 is very attractive from the feature selection point of view. A question arises whether a new weak classifier, maximally independent of *all* already selected ones, can be found. In such

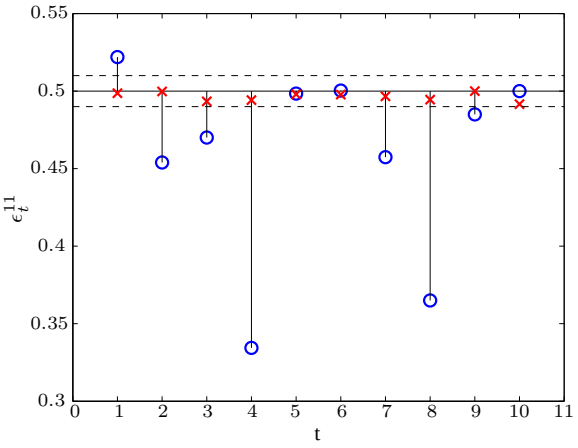

Figure 1: Weighted errors ϵ_t^{11} (eq. (4)) of weak classifiers h_t after ten iterations (T=10) for AdaBoost (circles) and TCAcu (crosses). In AdaBoost, ϵ_t^{11} for all but the last weak classifier are arbitrary. In TCAcu, all errors satisfy $|\epsilon_t^{11} - 0.5| < \Delta_{min}$. Note that the selected weak classifiers may be different for AdaBoost and TCAcu.

case, the distribution D_{t+1} must satisfy

$$\sum_{i=1}^{m} D_{t+1}(i) u_{q,i} = 0 \quad \text{for } q = 1, ..., t \tag{5}$$

where $u_{q,i} = h_q(x_i) y_i$ or equivalently $\epsilon_q^{t+1} = \frac{1}{2}$ for $q = 1, ..., t$.

There is no closed-form solution to the system of equations (5) and sometimes even an exact solution does not exist [3]. This is a consequence of the non-negativity constraint on D_{t+1}, which is a distribution. Therefore, TCS is designed as an iterative optimisation algorithm.

In AdaBoost, equation (5) holds at time t (after reweighting, Step 4) only for $q = t$. A typical situation is depicted in Figure 1. Weak classifier errors are shown after step $t = 10$ of AdaBoost. The weak classifier errors differs from 0.5 except for the last (10th) weak classifier.

Another observation can be made about the change of the upper bound. From equation (1), we see that the upper bound is reduced if the error of a newly added weak classifier differs from 0.5. The bigger the difference, the bigger the reduction of the upper bound. If follows that the upper bound can be further reduced by formally adding an already used weak classifier, if its error differs form 0.5. This addition has two important consequences.

First, because of the linear combination form of the final classifier, addition of an already used weak classifier h_r, $r < t$, requires only a change of α_r coefficient, not a change of the final classifier size. A new coefficient is computed as $\alpha_r = \alpha_r^r + \alpha_r^{t+1}$, where the upper indexes express the cycle in which the coefficient was computed.

Second, a new distribution obtained by this addition satisfies equation (5) for $q = r$, but not for any other q. If equation (5) is approximately satisfied for all q the goal is reached. If not, another q is selected and h_q "virtually added". Each such addition will lower the upper bound.

TCS is formally summarised in Algorithm 2. At time t, a distribution D_t is used to initialize the algorithm. In each iteration a weak classifier is selected from the already used ones so that the absolute difference of its error and 0.5 is maximised. The standard scheme is used to find $\widehat{\alpha}_j$ and the new distribution $\widehat{D}_{j+1}$. The value $\widehat{\alpha}_j$ is added to the corresponding coefficient and the loop is repeated.

Since an exact solution may not exist, the computation is terminated if a close enough solution is found or if the maximum allowed number of iterations is reached. The final distribution is then used in cycle $t + 1$ of AdaBoost learning. A typical result of the algorithm is depicted in Figure 1.

Convergence properties of the TCS step and standard AdaBoost are the same. The only difference is in the set of weak classifiers which in TCS is limited to the already selected ones in the main AdaBoost loop.

A similar algorithm was proposed by Kivinen and Warmuth [3]. TCAcu differs from Kivinen and Warmuth algorithm in two important aspects: (1) the coefficients of weak classifiers are updated repetitively, (2) the property of minimisation of the upper bound is kept. The Kivinen and Warmuth algorithm was experimentally tested by Oza on several standard problems [6] with poor results.

3. Face detection and AdaBoost

The totally corrective algorithm was applied to the face detection problem using the framework introduced by Viola and Jones [12]. To train a classifier, Viola and Jones select from a large number of very efficiently computable features (see [12] for detailed description). Every weak classifier implements a simple threshold function on one of the features. Having such a large set of weak classifiers, AdaBoost learning is used to chose a small number of weak classifiers and to combine them into a classifier deciding whether an image is a face or a non-face.

Due to its greedy character, AdaBoost is able to cope with very large sets of weak classifiers. However, for face detection, very large training set has to be explored as well in order to build a high-quality classifier. To solve infeasibility of this problem a bootstrapping technique [11] is commonly used. Viola and Jones proposed another technique to cope with this problem.

Cascade building. Instead of training a single classifier, a cascade of classifiers is built. An image window (region) is passed to the first classifier. It is either classified as non-face or a decision is deferred and the image is passed to the second, etc. classifier. The goal of each classifier is to prune the training set for the next stage classifier of the cascade. Since easily recognisable non-face images are classified in the early stages, classifiers of the later stages of the cascade can be trained rapidly only on the harder, but smaller, part of the non-face training set.

Input: Allowed false positive rate f, and detection rate d;
final false positive rate f_{final}

$F_0 = 1$, $D_0 = 1$
Do until $F_i > f_{\text{final}}$

1. Train a classifier until $f_{\text{reached}} < f$ and $d_{\text{reached}} > d$ on the validation set
2. $F_{i+1} = F_i \times f_{\text{reached}}$
3. $D_{i+1} = D_i \times d_{\text{reached}}$
4. Throw away misclassified faces and generate new non-face data from non-face images

Algorithm 3: Building the cascade.

The cascade building is described in Algorithm 3. Inputs to the algorithm are: the desired false positive rate f, the detection rate d of the cascade stages, and the final false positive rate of the cascade. Each stage is trained until f and d are reached. Since AdaBoost is neither designed to reach low false positive rates nor high detection rates, a threshold is adjusted ex post.

In the cascaded classifier, the overall false positive and detection rates are a product of the rates of individual stages. The pruning process is asymmetric and concentrates on the non-face images. The stage false positive f is usually set to higher values. The multiplication in Step 2 guarantees an exponential reduction of the overall false positive rate. The detection rate must be set close to one to ensure that final D is high.

4. Experiments

The performance of TCAcu and AdaBoost was compared on the face detection problem. The training dataset, training process and obtained results are discussed next. The performance evaluation concentrates on the speed and complexity of the learned cascaded classifiers.

Training data. The data for training were collected from various sources. Face images are taken from the MPEG7 face dataset [2]. The dataset contains face images of variable quiality, different facial expressions and taken under wide range of lightning conditions, with uniform or complex background. The pose of the heads is generally frontal with slight rotation in all directions. Eyes and the nose tip are aligned in all images. The dataset contains 3176 images, one image was removed due to severe distortion.

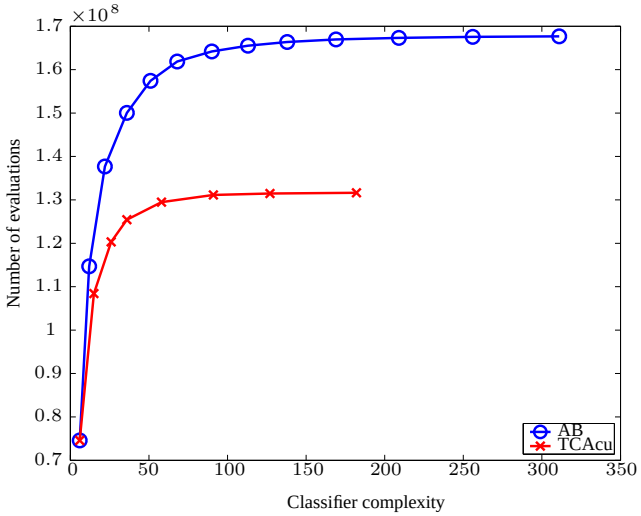

Figure 2: Selectivity comparison. Horizontal axis: the complexity of the cascaded classifier expressed by the number of weak classifiers used. Vertical axis: number of weak classifier evaluations on the MIT+CMU dataset.

Pose variability was added synthetically to the data. The images were randomly rotated by up to $5°$, shifted up to one pixel and the bounding box was scaled by a factor of 1 ± 0.05. Two datasets, training and validation, of the same size as the original dataset were created by the perturbations.

Non-face images were collected from the web. Images of diverse scenes were included. The dataset contains images of animals, plants, countrysite, man-made objects, etc.. More than 3000 images were collected and random subwindows used as non-face examples.

Training process. During the training process, the training and validation dataset are updated for each stage (cf. Algorithm 3). The non-face part of the training and validation datasets consist of 5000 randomly selected regions from the non-face images. Only regions that were not rejected by previous stages of the cascade are included. The face set remains almost the same over the whole training. The faces rejected by some of the stage classifiers are removed, but the cascade is build to ensure that these false rejects are just a small fraction of the face data.

The process is driven by the stage false positive, detection and final false positive rates. In the reported experiments, the values were set to 0.4 stage false positive rate, 0.999 detection rate and 0.0001 the final false positive rate. The final false positive rate was reached in stage eight in TCAcu and in stage ten in AdaBoost.

4.1. Results

The classifiers were tested on the MIT+CMU dataset [8]. This dataset has been widely used for comparison of face detectors [8, 10, 12]. The main objective of the experiments is to demonstrate the detection speedup in comparison with the classical Viola-Jones approach, rather than improvement of the detection rate per se. This means that we did not try to find e.g. the optimal sets of weak classifiers since this is not important for a fair comparison of AdaBoost and TCAcu.

The results for the cascades trained by the two variants of AdaBoost are summarized in Table 1. For each number of stages in the cascade, the following quantities are recorded (left to right in Table 1): the number of weak classifier forming a stage in the cascade, the total number of evaluations in each stage, and the false negative and false positive rates on the MIT+CMU dataset.

It can be observed that for both algorithms the complexity of stages increases gradually except for two small fluctuations. At the beginning the growth of TCAcu is slower and it changes after four stages. However, the complexity is not the only important factor determining the speed of face detection. Also the number of regions marked as a potential face in each stage is significant. It can be seen that TCAcu discards many more regions in early stages than AdaBoost. This early prunning influences false positive and false negative rates that are shown in the last two columns. These two rates measure the performance of the cascaded classifier. The table shows that both algorithms lead to similar false positive and false negative rates, but TCAcu converges much faster.

To compare the speed of the cascades trained by TCAcu and AdaBoost, the number of weak classifiers evaluated on MIT+CMU dataset was measured. All regions have to be evaluated by the first stage classifier. The number of evaluations is consequently a product of the number of regions and the length of the first stage classifier. The same holds for the second (and higher) stage classifier, but only regions not rejected by the first (previous) stage(s) are evaluated. Summing the numbers evaluations of the first and the second stage gives the number of evaluations of the two-stage cascade classifier. The result for all lengths of the cascade and for both algorithms is depicted in Figure 2.

Figure 2 demonstrates two important phenomena. First, the complexity of the cascades with comparable false negative and false positive rates is up to four times smaller for the TCAcu algorithm (six-stage TCAcu vs. eleven-stage AdaBoost). Second, the number of evaluations needed in AdaBoost is higher by 20 % than in TCAcu.

5. Conclusions

A new extension of the AdaBoost algorithm was proposed and compared with the state-of-the-art Viola and Jones face detection algorithm. The proposed TCAcu algorithm finds the final classifier by aggressive minimisation of the up-

Number of stages	Stage classif. length		Number of evaluations		False negatives		False positives	
	AB	TCAcu	AB	TCAcu	AB	TCAcu	AB	TCAcu
1	6	6	12431151	12431151	0	0	3930473	3682307
2	10	9	4009205	3757632	0	0	1598933	1054019
3	14	11	1643072	1083123	0	1	795262	492881
4	15	10	823246	512004	2	5	415751	173341
5	17	22	435483	183962	4	16	189902	44287
6	22	33	201982	49814	11	39	92226	6488
7	23	36	100887	9052	17	83	46499	1584
8	25	55	52867	3173	26	143	22966	183
9	31		27504		35		11262	
10	40		14818		53		5070	
11	47		7568		59		2193	
12	55		4194		73		905	
13	49		2602		84		426	

Table 1: Comparison of AdaBoost (AB) and TCAcu performance.

per bound on the training error and produces a significantly shorter classifier. The obtained results are comparable to the Viola and Jones method in terms of detection and false positive rates. The classifier trained by the novel method was about 20 % faster and consists of only a quarter of the weak classifiers needed for a classifier trained by standard AdaBoost.

The algorithm can be applied with other weak classifiers suitable for face detection and in conjunction with FloatBoost-like feature selection techniques. The reduction of the number of weak classifiers can be important in areas where the weak classifiers are expensive to compute or to implement, e.g. on smart cards or other special purpose hardware.

Acknowledgments

The authors were supported by The Grant Agency of the Czech Republic under project GACR 102/02/1539 and by The Grant Agency of the Czech Technical University under project CTU 0307313 and by The Ministry of Education of the Czech Republic under project Kontakt ME 678.

References

[1] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *ECCLT*, pages 23–37, 1995.

[2] ISO/IEC JTC 1/SC 29/WG 11 Moving Picture Experts Group.

[3] J. Kivinen and M. K. Warmuth. Boosting as entropy projection. In *CLT*, pages 134–144, ACM, New York, July 1999.

[4] S. Li, L. Zhu, Z. Zhang, A. Blake, H. Zhang, and H. Shum. Statistical learning of multi-view face detection. In *ECCV*, page IV: 67 ff., 2002.

[5] R. Lienhart, A. Kuranov, and P. Vadim. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *DAGM*, Magdeburg, Germany, September 2003.

[6] Nikunj C. Oza. Boosting with Averaged Weight Vectors. In *Multiple Classifier Systems*, pages 15–24, 2003.

[7] P. Pudil, J. Novovicova, and J. Kittler. Floating search methods in feature selection. *Pattern recognition letters*, 15:1119–1125, 1994.

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

[9] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, pages 37(3): 297–336, 1999.

[10] H. Schneiderman and T. Kanade. A statistical method for 3d object detection applied to faces and cars. In *CVPR*, pages I:746–751, 2000.

[11] K. K. Sung and T. Poggio. Learning human face detection in cluttered scenes. In *CAIP*, pages 432–439, 1995.

[12] P. Viola and M. Jones. Robust real time object detection. In *SCTV*, Vancouver, Canada, 2001.