A principled approach to remove false alarms by modelling the context of a face detector

Cosmin Atanasoaei[1,2]
cosmin.atanasoaei@idiap.ch
Christopher McCool[2]
christopher.mccool@idiap.ch
Sébastien Marcel[2]
marcel@idiap.ch

[1] École Polytechnique Fédérale de Lausanne (EPFL)
Lausanne, CH
[2] Idiap Research Institute
Martigny, CH

Abstract

In this article we present a new method to enhance object detection by removing false alarms in a principled way with few parameters. The method models the output of an object classifier which we consider as the *context*. A hierarchical model is built using the detection distribution around a target sub-window to discriminate between false alarms and true detections. The specific case of face detection is chosen for this work as it is a mature field of research. We report results that are better than baseline methods on XM2VTS and MIT+CMU face databases. We significantly reduce the number of false acceptances while keeping the detection rate at approximately the same level.

1 Introduction

A variety of applications like video surveillance, biometric recognition and human-machine interface systems depend on robust face detection algorithms. In the last decade there has been an increasing interest in real-time systems with high accuracy and many successful methods have been proposed [16]. Still face detection remains a challenging problem and there are improvements to be made.

It is often posed as the task of classifying a sub-window as being a particular object or not. As such it requires a method to sample an image for sub-windows and a classifier to classify the sub-window. Research to date has dealt mainly with the issue of building a robust and accurate object classifier. An object classifier tells if an object is found at a specific position and scale (referred as sub-window) in an image. For instance work by Froba et al. [2] and Viola and Jones [15] has provided significantly improved face classifiers. Different approaches have been proposed such as the pioneering work from Rowley et al. [12] or [3] based on Neural Networks. But the most successful face detection methods are based on a cascade of boosted classifiers that provide real-time performance with high accuracy [7].

There are many ways to obtain sub-windows from an image, with the sliding window approach [12] being the most well known. The sliding window approach finds all the object instances by scanning the image at different positions and scales. This can result in multiple

BMVC 2010 doi:10.5244/C.24.17

detections and false alarms as shown in Fig. 1. A merging and pruning heuristic algorithm is then typically used to output the final detections [11, 12, 15].

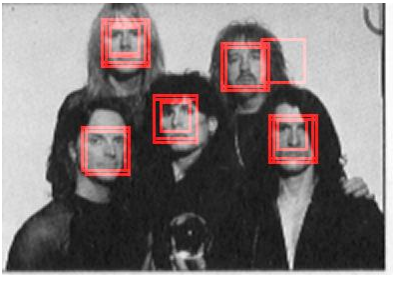


Figure 1: Typical face detections using the multiscale approach and the boosted cascade classifier described in [2] (without clustering multiple detections nor removing false alarms).

Recent work has been done to overcome the limitations of the sliding window approach by using a branch-and-bound technique to evaluate all possible sub-windows in an efficient way [6]. The authors build a model that also predicts the location of the object [1]. However, it is not clear how to use this method for different classifier types (for instance boosted cascades) or to detect multiple objects.

A different approach was recently proposed in [13] and [14] where the authors study the score distribution in both location and scale space. Their experimental results have shown that the score distribution is significantly different around a true object location than around a false alarm location, thus making possible to build a model to better distinguish the false alarms and enhance detection. This approach is motivated by the fact that the object classifier is usually trained with geometrically normalized positive samples and it does not process the *context* (area around given samples). Also, some false alarm sub-windows may have a higher score than a true detection nearby and may be selected erroneously as being final detections when using a simple heuristic merging technique.

We propose a model to enhance a given face classifier, by discriminating false detections (sub-windows) from true detections using the contextual information. Our approach was inspired from the work of [13, 14]. Similarly we investigate the detection distribution around some sub-window (which we call the *context*) in order to evaluate if it corresponds to a true detection or not.

There are significant differences between this work and that presented in [13, 14]. The first is that we extract more information from the detection distribution than just the score of the face classifier. For example we count detections within the context and we use features that describe the geometry of the detections around a sub-window. The second significant difference is that we extract features from every possible axis combination (locations x, y and scale s) and we train a classifier to automatically choose the most discriminant features.

This paper is organized as follows. Section 2 presents our proposed model that uses the output of the face classifier which we consider as the *context* to distinguish between false alarms and true detections. Finally we test our method on several face databases using a popular face classifier and we present the results in Section 3. Conclusions and future directions for work are given in Section 4.

2 Context-based model

In this section we present a model to discriminate false detections from true detections. First we describe how we *sample* around a target sub-window to build its context. Then we present the *features* we extract from the context and finally the *classifier* that uses these features to discriminate false alarms from true detections.

2.1 Sampling

We sample in the 3D space of location (x,y) and scale (s) to collect detections *around* a target sub-window $T_{sw} = (x,y,s)$. For this we vary its position and scale in all directions (left, right, up, down, smaller and bigger) and we form new sub-windows. Those sub-windows that pass the object classifier are gathered with the associated classifier output *ms*, referred to as the model score in this paper. We obtain a collection of 4D points $C(T_{sw}) = \{(x_i, y_i, s_i, ms_i)_{i=1,..}\}$ that we call the *context of the target sub-window* T_{sw}. Its parameters are the number of points to be considered on each axis (location and scale) along the positive direction, which we define as N_x, N_y and N_s respectively.

We have used two strategies for context sampling: full and axis. The *full* strategy consists of sampling by varying the location and scale at the same time. In this case the context can have at most $N_{full} = (2N_x+1) \times (2N_y+1) \times (2N_s+1)$ points. In the *axis* strategy the sampling is done just along one axis at a time. This reduces the maximum size of the context to $N_{axis} = (2N_x+1) + (2N_y+1) + (2N_s+1)$ points.

In our experiments we have used $N_x = N_y = 6$ and $N_s = 7$ with 5% increments both in scale and position [1]. This makes N_{axis} (at most 41 points) approximately 60 times smaller than N_{full} (at most 2535 points). The axis sampling approach is better suited for real time applications where building the full context may be too expensive. In our experiments this method has a small performance degradation compared to the full sampling method, but it can be many times faster.

2.2 Feature vectors

In the next step we extract a fixed number of low dimensional feature vectors from $C(T_{sw})$. The feature vectors are defined by their attribute(s) and the axis (and axes) used to obtain the attribute(s).

We use 5 attributes that capture the *global information* (counts), the *geometry of the detection distribution* (hits) and the *detection confidence* (score) obtained from the face classifier. The *counts* provide a global description of $C(T_{sw})$ by counting detections on some axis combination. The *score (standard deviation and amplitude)* describes the classifier confidence variation across position and scale changes. The *hits (standard deviation and amplitude)* capture the spread of detections on some axis. The last attribute addresses the intuition that detections can be obtained by varying more the scale or the position around a true detection than on the false alarms.

Each feature vector is computed on some axis combination (x, y and s) which gives 7 possible combinations. For example we can build sub-windows by varying all axes, just two of them (like keeping the *scale* constant and varying only the x and y sub-window's

[1] The context for a detection of size 100x100 pixels is obtained by sampling sub-windows from approximately 70x70 to 140x140 pixels and translated by at most 34 pixels.

coordinates) or just one of them (like keeping the x and *scale* fixed and moving the sub-window up and down).

More details can be found in Section 3.3, where we have visualized and investigated the discriminative properties of these features.

2.3 Classifier

The context features from the previous section are used to train a classifier to distinguish between false alarms and true detections based on their context. We build a linear classifier for each context feature (described in Section 2.3.1) and then we combine them to produce the final result (described in Section 2.3.2).

Our aim is to *automatically* select the best attributes and axes that are more discriminant. This makes the context-based model independent of the specific geometric properties of the object to detect, the type of the object classifier or the scanning procedure.

2.3.1 Context classifiers

The contextual information is used to form 35 different context features: there are $n = 5$ types (as discussed in Section 2.2) computed for each of the $m = 7$ axis combinations. For each feature vector we build a logistic linear model which we denote as $M(x,w)$, where the x is the d-dimensional sample feature vector and w is the $d+1$-dimensional parameter value. The model output is:

$$M(x,w) = \frac{1}{1+exp\left(-w_0+\sum_{i=1}^{d} x_i w_i\right)}, \tag{1}$$

where w_0 is sometimes called the bias term and the w_i terms are the weights of the inputs.

Training the model is done by minimizing the negative of the likelihood of the model output being generated from the input data. Additional L_1 and L_2 norm regularization terms are added as described in [4]. Following [10], our function to optimize is:

$$E(w,\lambda_1,\lambda_2) = \frac{\sum l(w,x^+)}{N^+} + \beta\frac{\sum l(w,x^-)}{N^-} + \lambda_1 \underbrace{\sum_{i=1}^{n}|w_i|}_{L_1} + \lambda_2 \underbrace{\sum_{i=1}^{n}|w_i|^2}_{L_2}, \tag{2}$$

where $l(w,x) = -y\ log(M(x,w)) - (1-y)\ log(1-M(x,w))$ is the negative log likelihood of the sample x using the model weights w; obviously y relates to the label of interest so it represents the positive class for the case of $l(w,x^+)$ and the negative class for $l(w,x^-)$. The log likelihoods are averaged separately over the N^+ positive samples and the N^- negative samples respectively because of the unbalanced nature of the training samples. λ_1 and λ_2 are priors for the L_1 and L_2 norms. The purpose of the L_2 norm regularization term is to avoid over fitting, while the L_1 one is to keep the model sparse hopefully by automatically selecting the most informative features.

The weight β represents the relative importance attributed to the error caused by the negative samples relative to the one caused by the positive samples. In the case of object detection (in particular face detection) it is preferred to have higher false alarms than to miss objects. This implies that β needs to penalize false rejections more than false acceptances which corresponds to $\beta < 1$. Several preliminary experiments were performed on a small sub-set of the training data and $\beta = 0.3$ was chosen as the optimal value.

There are some robust methods to optimize the non-continuously differentiable function $E(w,\lambda_1,\lambda_2)$ (for a review see [4]). We have used a simple method called *grafting* described in [10]. This method integrates well with standard convex optimization algorithms and it uses an incremental approach to feature selection that suits our needs. Jorge Nocedal's L-BFGS library [8] was used for the optimization of the error function at each step of the grafting algorithm.

Another related problem we need to solve is the choice of the λ_1 and λ_2 prior terms. For this we use a cross-validation technique on two datasets, one for training and one for tuning, as specified by each database's protocol. We first optimize the λ_1 prior term using a logarithmic scale keeping $\lambda_2 = 0$ and second we optimize the λ_2 prior term using the same logarithmic scale and keeping the already estimated λ_1 value. The criterion to choose the best (λ_1,λ_2) configuration is the Weighted Error Rate (WER) defined as:

$$WER(\beta,\tau) = \frac{\beta \times FAR + FRR}{\beta + 1}, \tag{3}$$

where FAR is the False Acceptance Rate and FRR is the False Rejection Rate computed as $FRR = 1 - TAR$; TAR is the True Acceptance Rate also referred to as Detection Rate (DR). The same weight β was used as in Equation 2.

2.3.2 Combined classifier

Each feature classifier can be considered as an expert. By combining them two benefits can be obtained: first the combined classifier should perform better and second only some (the best) experts are combined which implies that some irrelevant features can be (automatically) discarded. The combined model uses the same logistic linear model as for the context classifiers. This makes the proposed hierarchical model a non-linear mapping of the inputs, while each context classifier is kept very simple and linear.

The inputs to the combined classifier are the normalized outputs of the context classifiers. Let us define the context classifiers as $M_{k,l}(x,w)$, where k indicates the attribute type $(k = 1..n, n = 5)$ and l corresponds to the axis combination $(l = 1..m, m = 7)$. Let $\tau_{k,l}$ be the optimum threshold value of the $M_{k,l}$ model. Then the value forwarded to the combined classifier is $x_{k,l} = M_{k,l}(x,w) - \tau_{k,l}$.

This normalization has two benefits. First, the sign indicates the decision of the $M_{k,l}$ model: positive for true detections and negative for false alarms. Second, the absolute value is (empirically) proportional to the confidence of the $M_{k,l}$ model in its decision.

3 Experiments

The experimental procedure used in this paper is defined by these aspects: the databases used, the protocol for these databases, the face classifier and the methods for evaluating performance.

3.1 Databases and protocols

We have evaluated our method on two scenarios: XM2VTS [9] and MIT+CMU [12]. For each scenario a distinct training, tuning and testing image collection was provided to train, optimize parameters and evaluate the context-based model.

The XM2VTS database, split using the Lausanne protocol, contains one large centred face in each image taken in a controlled environment. There is an overlap with the identities used for the training, tuning and testing datasets, but different captures were considered.

The second scenario uses the WEB [3] database for training, the CINEMA [3] database for tuning and the MIT+CMU database for testing. This scenario is considered as the most difficult because it consists of images with multiple, sometimes very small, degraded faces or without any face, taken in different environments (indoor and outdoor).

3.2 Face classifier

Our method was tested using the MCT-based face classifier [2] implemented with the Torch3vision open-source library. We alter the performance of this face classifier by varying the threshold (θ) of the last stage. This allows us to understand if the performance of the classifier affects the performance of the context models. For each θ four context-based models have been trained: using both *full* and *axis* context sampling methods for each of the two scenarios.

The detections (and contexts) are obtained using a standard sliding-window approach. The context-based model checks each detection and the false alarms are removed. The final detections are obtained by averaging the remaining detections that overlap, which removes most of multiple detections around the same face. It should be noted that no sub-window heuristic pruning was used during scanning.

We have compared our method with the merging method implemented by Torch3vision and referred to as HMergeT. More details can be found in [11]. To label a detection as positive we used the Jesorsky measure with the threshold $\varepsilon_J = 0.25$ [5].

3.3 Analysis of context features

Preliminary experiments have been carried away to analyse if the proposed features provide enough information to discriminate between the two cases of contexts. We have plotted some of these context features in Fig. 2 on the XM2VTS training dataset. To assign a detection (and its context) to the positive or negative class we have used the Jesorsky measure with a relaxed threshold $\varepsilon_J = 0.5$. This is because a valid detection should be kept even if it does not match well the true position, but its context captures a significant part of the ground truth. Still, for face detection performance evaluation a more precise $\varepsilon_J = 0.25$ has to be used as specified in the previous section.

There is a significant difference between the negative and the positive contexts that support our intuition: around a true detection many more detections are generated than around a false alarm. This implies that just by counting detections good discriminative information is obtained. For example in Fig. 2 (a, b) more than 95% of the negative contexts have their count attribute less than 95% of the positive ones.

Also, fewer detections implies much less score variation for negative samples. It can be noticed that negative contexts are more compact around the center, while the positive are much more spread having the standard deviation much higher for the combination of two axes (see Fig. 2 (c, d)).

We have found experimentally that it is easier to visually separate the two context classes using the full sampling, for example see Fig. 2 - (a) versus (b), (c) versus d. This is expected because the full sampling gathers many more detections and it is also verified by the next set of experiments where it outperforms with a small margin the axis sampling variant.

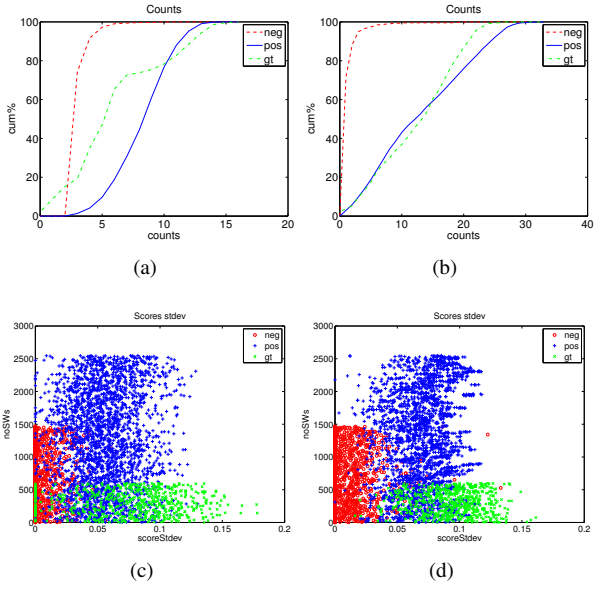


Figure 2: Distributions of various features using the full (right column) versus axis (left column) sampling on XM2VTS training dataset. Cumulative histogram of counts for two axes (y, scale) using axis sampling (a) and full sampling (b). Cloud of points of score standard deviation for 3 axes (x, y, scale) using axis sampling (c) and full sampling (d). The ground truth is represented with green, the positive class with blue and the negative with red.

3.4 Context-based model evaluation

In this set of the experiments we have evaluated how well the context-based model distinguishes between false alarms and true detections. For this we have computed and plotted the WER as shown in Fig. 3 for the two scenarios. The full (blue) and axis (green) sampling situations are plotted on the same graphic to easily compare them.

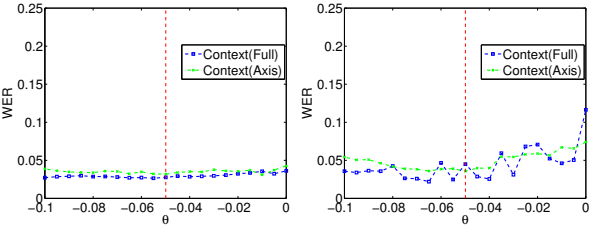


Figure 3: Context-based model's weighted error rate (WER) for the test sets of XM2VTS (a) and MIT+CMU (b). The default threshold point of the face classifier is represented with dashed red vertical line.

The full sampling context-based model performs better than the axis sampling one for the majority of different threshold values. Still this rather small increase in performance requires much larger contexts (2535 versus 41 samples, see Section 2.1) which impacts on the speed

of the overall face detection process. Even with the axis context sampling (41 samples) our context-based model manages to distinguish the false alarms from true detections.

On average both sampling models have a WER lower than 5% for the XM2VTS (Fig. 3 a) scenario. The same performance is obtained for the MIT+CMU scenario (Fig. 3 b), even though the training data is scarce (5 and 30 times less training images than for the XM2VTS scenario) and the database is much more challenging.

These results are stable across multiple threshold values of the face classifier. It is important to note that using simple logistic regressions as proposed is enough to obtain an accurate context-based model. This indicates that the features extracted from the contexts (see Section 2.2), although very simple and low dimensional, are discriminative enough.

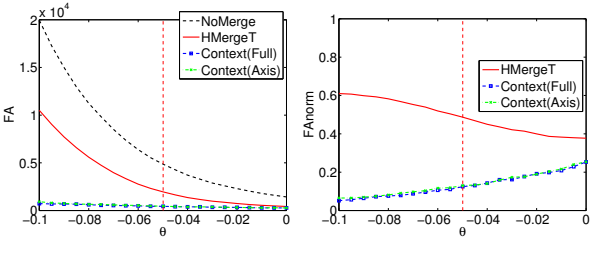
3.5 Face detection evaluation

Next we have performed experiments to assess the impact our model has on the face detection results. We studied the effect of: i) using the heuristic method HMergeT and ii) using the context-based model with either full or axis sampling, for face detection.

For this we analysed the TAR and the number of false alarms (FA) both parametrized by the threshold of the face classifier: $TAR = TAR(\theta)$ and $FA = FA(\theta)$ respectively. We omit the threshold of the context-based classifier in this parametrization because it is automatically optimized on the tuning dataset (see Section 2.3) and it is not varied during experiments. In our case the threshold to vary is θ, which is **not** the discriminative threshold. Indeed it just provides different context distributions to train our context-based model. We have chosen these two criteria (TAR and FA) instead of ROC curves, because the significant decrease in FA (see Fig. 4) makes the ROC curve very skewed to the left.

The aim of any multiple detection clustering algorithm is to remove as few as possible true detections and remove as many as possible false alarms. This motivates the comparison of our approach and the baseline with the face classifier without any merging. Let use define the TAR and the FA of the face classifier without any merging (NoMerge as in Fig. 4) as TAR_n and FA_n respectively. Then we report normalized TAR and FA like:

$$FAnorm(\theta) = \log\left(1 + \frac{FA(\theta)}{FA_n(\theta)}\right), \quad TARnorm(\theta) = \frac{TAR(\theta)}{TAR_n(\theta)}. \tag{4}$$

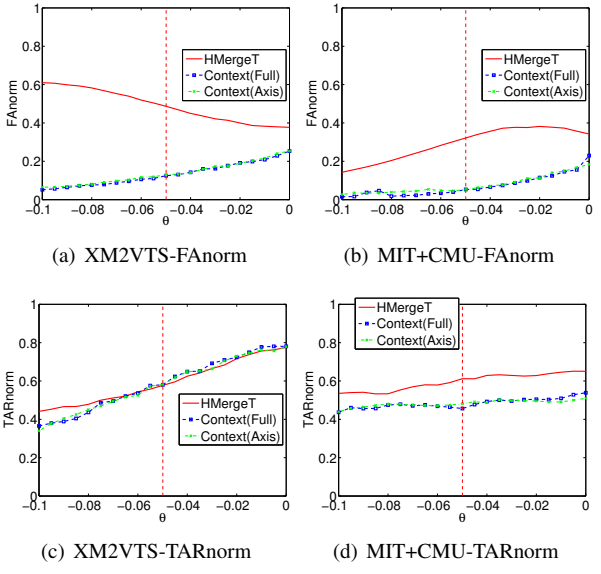


(a) XM2VTS-FA (b) XM2VTS-FAnorm

Figure 4: The normalized FA (b) on the XM2VTS scenario. The default threshold point is represented with dashed red vertical line.

First we analysed the logarithmically normalized FA plots presented in Fig. 5 (a & b) for the two scenarios. As expected the number of FAs is greatly reduced, with at least an order of magnitude compared to the baseline HMergeT. The significant decrease in the number of

FAs demonstrates that our proposed method successfully discriminates false alarms from true detections. Another important observation is that there is no significant difference between the full sampling and the axis sampling methods in the number of FAs. This indicates that a context with fewer samples (thus faster to evaluate) can be designed to have similar results.



(a) XM2VTS-FAnorm (b) MIT+CMU-FAnorm

(c) XM2VTS-TARnorm (d) MIT+CMU-TARnorm

Figure 5: Normalized FA (top row) and normalized TAR (bottom row) plots on XM2VTS (a, c) and MIT+CMU (b, d) scenarios.

Second we evaluated the impact on the normalized TAR as presented in Fig. 5 (c & d). The TAR decreases slightly for the XM2VTS scenario (up to 5%) and more accentuated for the MIT+CMU scenario (up to 10%).

We conclude that overall our system performs well compared to the baseline, the drop in TAR being justified by the exponential decrease in the FAs. Overall we found no significant performance difference between the full and the axis sampling methods.

4 Conclusion

This paper has presented a new method to enhance object detection by removing false alarms in a principled way with few parameters. We have evaluated the performance of our method on several popular face databases using a well known face detector to study the effect of two sampling methods - full and axis. It was found that our system reduces the FA exponentially while keeping the TAR at similar level as the baseline approach. The full sampling method has a slightly better performance but it needs many more samples, while the axis sampling version is a trade-off between performance and speed.

There are several advantages to using our method. First our algorithm can be initialized with any sub-window collection, which can be obtained using some sliding window approach or a totally different approach. Second it can work on top of any object classifier - there are no restrictions regarding its score values, its type or the features used. Further

improvements can be envisaged including the use of higher dimensional context-based features, different feature classifiers (such as SVM and AdaBoost) or more efficient sampling methods. Other work could also examine the use of contextual information to improve the accuracy of detections and even to recover miss-aligned detections.

Acknowledgement: The authors would like to thank the FP7 European MOBIO project (IST-214324) and the Hasler Foundation (CONTEXT project number 10040) for their financial support.

References

[1] Matthew B. Blaschko and Christoph H. Lampert. Learning to localize objects with structured output regression. In *European Conference on Computer Vision (ECCV)*, volume 5302, pages 2–15, 2008.

[2] B. Froba and A. Ernst. Face detection with the modified census transform. *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 91–96, 2004.

[3] Christophe Garcia and Manolis Delakis. Convolutional face finder: A neural architecture for fast and robust face detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(11): 1408–1423, 2004.

[4] Su in Lee, Honglak Lee, Pieter Abbeel, and Andrew Y. Ng. Efficient L_1 regularized logistic regression. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2006.

[5] Oliver Jesorsky, Klaus J. Kirchberg, and Robert Frischholz. Robust face detection using the hausdorff distance. In *AVBPA '01: Proceedings of the Third International Conference on Audio- and Video-Based Biometric Person Authentication*, pages 90–95, London, UK, 2001. Springer-Verlag. ISBN 3-540-42216-1.

[6] Christoph H. Lampert, Matthew B. Blaschko, and Thomas Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.

[7] R. Lienhart, A. Kuranov, and V. Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *25th Pattern Recognition Symposium*, pages 297–304, Madgeburg, Germany, 2003.

[8] Dong C. Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989.

[9] K. Messer, J. Matas, J. Kittler, J. Lüttin, and G. Maitre. Xm2vtsdb: The extended m2vts database. In *In Second International Conference on Audio and Video-based Biometric Person Authentication*, pages 72–77, 1999.

[10] Simon Perkins, Kevin Lacker, James Theiler, Isabelle Guyon, and André Elisseeff. Grafting: Fast, incremental feature selection by gradient descent in function space. *Journal of Machine Learning Research*, 3:1333–1356, 2003.

[11] Yann Rodriguez. *Face Detection and Verification using Local Binary Patterns*. PhD thesis, Ecole Polytechnique Fédérale de Lausanne, 2006.

[12] Henry A. Rowley, Shumeet Baluja, and Takeo Kanade. Neural network-based face detection. *IEEE Transactions On Pattern Analysis and Machine intelligence*, 20:23–38, 1998.

[13] Hiromasa Takatsuka, Masayuki Tanaka, and Masatoshi Okutomi. Spatial merging for face detection. In *SICE-ICASE International Joint Conference*, pages 5587–5592, 2006.

[14] Hiromasa Takatsuka, Masayuki Tanaka, and Masatoshi Okutomi. Distribution-based face detection using calibrated boosted cascade classifier. In *ICIAP '07: Proceedings of the 14th International Conference on Image Analysis and Processing*, pages 351–356, 2007.

[15] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1:511–518, 2001.

[16] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Comput. Surv.*, 35(4):399–458, 2003. ISSN 0360-0300.