Sensitivity analysis for face detection

Romi Koželj, Žiga Emeršič, Luka Šajn

University of Ljubljana, Faculty of Computer and Information Science
E-mail: romi.kozelj@gmail.com, ziga.emersic@fri.uni-lj.si, luka.sajn@fri.uni-lj.si

***Abstract*—Face detection has become one of the most active research topics in computer vision for its interesting applications, such as face recognition, face tracking, facial expression analysis and video surveillance. A large number of the face related applications assume that the face region is perfectly localized or detected. In order to adopt or use a certain face detection algorithm, evaluation of its performance is needed. In this paper, we evaluate three face detection algorithms with six different methods, using three evaluation metrices and discussed common failure cases that are worth to be further investigated.**

I. Introduction

Face detection is a critical step to all facial analysis algorithms, including face alignment, face recognition, face verification, and face parsing. Given an arbitrary image, the goal of face detection is to determine whether or not there are faces in the image and obtain the location and the size of each face [1]. The challenges associated with face detection can be, in addition to variations in pose and facial expression, attributed also due to scale, occlusion, brightness conditions, noise, etc. In this paper we evaluate the accuracy of the three chosen face detectors, whose parameters are changed in different manners to show, how different conditions of images influence detection accuracy.

We compared performance of three face detectors, namely `opencv's` [2] *detectMultiScale* method, which uses Viola and Jones detection algorithm [3], `insightface` detector [4], which is minimalistic inference-focused repack of [5] and `mxnet-face` [6] detector. First detector was taken from Python *OpenCV* library and the code for the last two detectors was obtained from the corresponding authors.

For `opencv's` *detectMultiScale* method parameters *scale factor* and *min neighbors* need to be specified. Parameter *scale factor* is specifying how much the image size is reduced at each image scale. By rescaling the image, larger faces can be resized to a smaller ones, making them detectable by the algorithm. Small step for resizing means that the algorithm works slower since it is more thorough. Bigger step is used for faster detection, with the risk of missing some faces altogether. We set parameter value to 1.05, which means that we used step size 5% for rescaling the image. Parameter *min neighbors* is specifying how many neighbors each candidate rectangle should have to retain it. This parameter affects the quality of the detected faces. Higher value results in fewer detections but with higher quality. We set parameter to value 3. `Insightface` detector performs pixel-wise face localisation on different scales of faces by taking advantages of joint extra-supervised and self-supervised multi-task learning [7]. `mxnet-face` detector uses fast R-CNN network [8], [9].

We ran face detectors on images, whose quality we were changing with 6 different parameters. Then we compared predictions using 3 evaluation metrices and reported the results.

The remaining sections of our paper are organized as follows. We first do a short review of related work. Then, in the methods section, we explain how used methods and metrices work. In the results section, we show and compare the results of this methods. At last, we summarise and comment the results in the conclusion section.

II. Related work

Face detection has been studied for decades in the computer vision literature. Face detection algorithms can be categorized into four categories: cascade based methods [10], [11], part based methods [12], channel feature based methods [13], [14], and neural network based methods [15], [16]. The seminal work by Viola and Jones [11] introduces integral image to compute Haar-like features in constant time. These features are then used to teach AdaBoost classifier with cascade structure for face detection. Various later studies follow a similar pipeline. Among those variants, SURF cascade [10] achieves competitive performance. One of the well-known part based methods is deformable part models (DPM) [12]. Deformable part models define face as a collection of parts and model the connections of parts through Latent Support Vector Machine. The part based methods are more robust to occlusion compared with cascade-based methods. Aggregated channel feature (ACF) is first proposed by Dollar et al. [13] to solve pedestrian detection. Later on, Yang et al. [14] applied this idea on face detection. In particular, features such as gradient histogram, integral histogram, and color channels are combined and used to teach boosting classifier with cascade structure. Studies [15] and [16] show that face detection can be improved by using deep learning, leveraging the high capacity of deep convolutional networks.

III. Methodology

To measure performance of detectors, we were gradually changing quality of images in dataset and at each step evaluated detectors predictions. We manipulated images with changes of brightness, contrast, resolution, blurriness, noise and occlusion, separately. We were changing images in each of these manners by discrete steps and at each step ran detectors over all images in dataset and calculated three metrices for evaluation, namely *intersection over union*, *precision* and *recall*. For each metric we calculated its value for each image in dataset and averaged values over whole dataset.

With average *intersection over union* we calculated average proportion of area of correctly predicted face bounding boxes against area of both predicted and ground-truth bounding boxes on images. Metric is defined as

$$\text{intersection over union} = \frac{\sum_{i=1}^{m} \left(\frac{A^i_{overlap}}{A^i_{union}} \right)}{m},$$

where m denotes number of images, $A^i_{overlap}$ denotes overlapping area of ground-truth and predicted face bounding boxes for image i and A^i_{union} denotes area of union of ground-truth and predicted face bounding boxes for image i. With average *precision* we calculated average proportion of correctly predicted bounding boxes against all predicted bounding boxes and with average *recall* we calculated average proportion of correctly predicted bounding boxes against all gound-truth bounding boxes. We calculated average *precision* as

$$\text{precision} = \frac{\sum_{i=1}^{m} \left(\frac{tp^i}{tp^i + fp^i} \right)}{m}$$

and averege *recall* as

$$\text{recall} = \frac{\sum_{i=1}^{m} \left(\frac{tp^i}{tp^i + fn^i} \right)}{m},$$

where m again denotes number of images, tp^i denotes area of true positive detections, which is in this case the same as $A^i_{overlap}$. fp^i denotes area of false positive detections for image i – area on image i detected as face, but is not a face and fn^i denotes area of false negative detections for image i – area on image i where faces are present, but are not detected.

To calculate above three metrices, we represented each image as ground-truth matrix, representing image with ground-truth face bounding boxes and as prediction matrix, representing image with predicted face bounding boxes. Matrices have the same size as original image and are composed only of values 0 and 1. Ground-truth matrix has value 0 on indexes where faces are not present and value 1 on indexes where faces are present. Prediction matrix has value 0 on indexes not predicted as face and value 1 on indexes predicted as face. To get value for $A^i_{overlap}$ (and tp^i) we summed up these two matrices and counted the number of indexes of summed up matrix, where values where bigger than 1. Similarly, to get the value of A^i_{union} we counted the number of indexes of summed up matrix, where values where bigger than 0. To get value for $tp^i + fp^i$, which represents area on image i where faces are predicted, we counted the number of indexes of prediction matrix, where values were bigger than 0. To get value for $tp^i + fn^i$, which represents area on image i were faces are present, we counted the number of indexes of ground-truth matrix, where values were bigger than 0.

When testing the effect of brightness on detection quality, we were gradually changing brightness of images from 0% to 100% by 5%. In the same way we tested effect of contrast. When testing the effect of image sizes, we first reduced size of each image to 1% of its original size and then enlarged it by 5%, until we reached 101% of the original size. To incorporate effect of blurriness, we replaced each pixel value in the image by average value of the neighbouring pixels, called kernel. We changed kernel size by 1% from 0% to 13% of the original image, where we stopped, since accuracy of the predictions for all detectors fell to 0%. Similarly we incorporate effect of noise on images, where weighted randomize matrix of the same size as the image, with values from 0 to 255, was added to each image. We changed noise percentage (weight) by 4% intervals, from 0% to 80%, where accuracy of the predictions fell to 0%. At last, we added occlusions to images. Here we were randomly masking images with black boxes, whose sizes were gradually increased by 5% from 0% to 100% of the original image. The number of boxes for each image was determined by the number of ground-truth faces on the image.

IV. Results

We found dataset with ground-truth face bounding boxes (i.e., the hand labeled bounding boxes for images that specify the faces locations) on `Dataturk` website [17]. Dataset includes 409 labeled images. Since the dataset contains a smaller amount of images, we split the images in 5 folds and calculate mentioned 3 metrices for each fold separately and average the results. To show how spread out the results are, we also calculate standard deviation over 5 folds.

Figures 1– 6 show performance of chosen three detectors evaluated with described metrices. In terms of resolution and occlusion nice hierarchy of these detectors can be seen along the whole scale of changes for all three metrices. `mxnet-face` detector performs the best, next is `insightface` detector and the worst is `opencv` detector. The same hierarchy can be observed from other quality changes, but holds only to some extent. When quality of images is changed to a grater extreme, `opencv` detector out-performs one or both other detectors. This is probably because `insightface` and `mxnet-face` detector were both trained on WIDER FACE dataset, which contains images with occlusions as stated in [18]. Also different sizes of images are probably included in this dataset. For this reason detectors trained on this dataset perform well on images with different resolutions and added occlusions. Other types of quality changes like extreme lightning conditions, high contrast, blurriness or noise, are probably not included in images in the WIDER FACE dataset, so this two detectors perform poorly when used on images whose quality is reduced in such way.

`mxnet-face` and `insightface` detectors remain in the same hierarchy order for all types of quality changes, except in terms of blurriness, where for images blurred for more than aprox. 4.5% (for all three metrices), `mxnet-face` detector shows the worst results.

From all types of quality changes, `opencv` detector shows better results for average *recall* metric in comparison to the other two metrices, since line jumps up to the other two lines. This can be due to the lower value for *min neighbors* parameter, which we specified to value 3. If parameter would be higher, detections would have higher quality and less incorrect predictions of the faces would be made. For that reason the area of images where faces are predicted would be lower, which would improve values for average *intersection of union* and average *precision*. Although as observed from figures showing the results for average *recall*, described hierarchy would still hold for all types of quality changes, except maybe for contrast and noise, where `opencv` detector out-performs `insightface` detector on almost whole scale.

Figures show low values for metrices even in cases when detectors perform best. The reason for such low values is that hand labeled bounding boxes are much wider and higher than detectors predict, even though, in this cases, detectors detect faces perfectly.

V. Conclusion

We tested performance of three detectors on images whose quality was changed in terms of brightness, contrast, resolution, blurriness, noise and occlusion. In terms of resolution and occlusion nice hierarchy of these detectors is observed, `mxnet-face` detector performing the best and `opencv` detector performing the worst. In terms of other types of quality changes this hierarchy holds only to some point, where then `opencv` detector out-performs one or both other detectors. `mxnet-face` detector, which overall shows the best results, performs poorly on images blurred for more than aprox. 4.5%.

For detectors that need training on images, quality of these images need to be changed in various ways, to improve detectors performance. Since `opencv` detector shows better results for average *recall* metric than for the other two metrices, future work should be done on testing `opencv` detector performance with different parameter values and evaluating results. To increase values of metrices for all three detectors and show more realistic results, future work should be done on excluding outer margins for hand labeled bounding boxes, as they are to big, or try to evaluate detectors on some other dataset as well.

References

[1] Ming-Hsuan Yang, D. J. Kriegman, and N. Ahuja, "Detecting faces in images: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 34–58, Jan 2002.

[2] "https://docs.opencv.org/2.4/modules/objdetect/doc/cascade_classification.html."

[3] M. Hassaballah, K. Murakami, and S. Ido, "Face detection evaluation: A new approach based on the golden ratio," *Signal Image and Video Processing*, vol. 7, 03 2013.

[4] "https://github.com/kiselev1189/insightface-just-works."

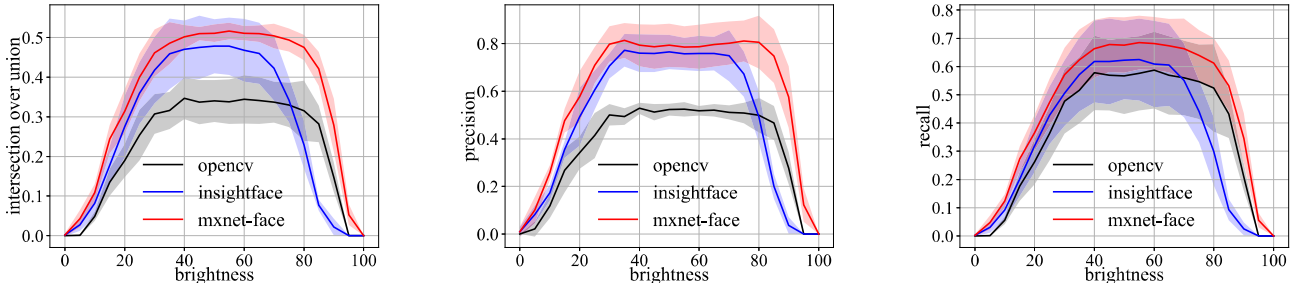

Figure 1: Plots of average intersection over union (left), average precision (middle) and average recall (right) for 5 folds of overall 409 images along the brightness of these images for `opencv`, `insightface` and `mxnet-face` detector. Lines show mean values and shaded areas show standard deviations over folds for each detector. Values on x axis represent brightness percentage of the images.

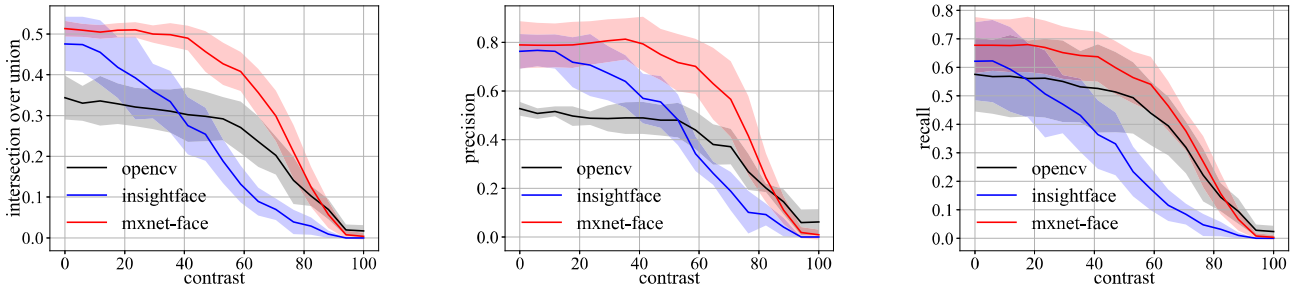

Figure 2: Plots of average intersection over union (left), average precision (middle) and average recall (right) for 5 folds of overall 409 images along the contrast of these images for `opencv`, `insightface` and `mxnet-face` detector. Lines show mean values and shaded areas show standard deviations over folds for each detector. Values on x axis represent percentage of contrast enforced to the images.

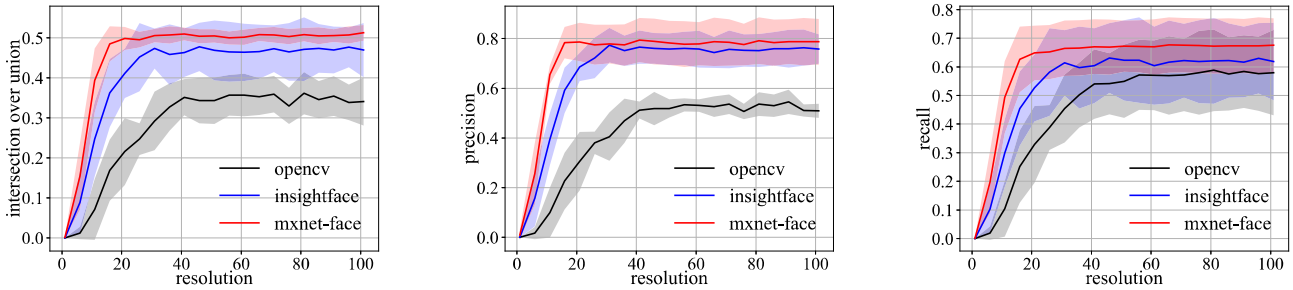

Figure 3: Plots of average intersection over union (left), average precision (middle) and average recall (right) for 5 folds of overall 409 images along the resolution of these images for `opencv`, `insightface` and `mxnet-face` detector. Lines show mean values and shaded areas show standard deviations over folds for each detector. Values on x axis represent size of images in percentage of original size of the images.

[5] "https://github.com/deepinsight/insightface."

[6] "https://github.com/tornadomeet/mxnet-face."

[7] J. Deng, J. Guo, Y. Zhou, J. Yu, I. Kotsia, and S. Zafeiriou, "Retinaface: Single-stage dense face localisation in the wild," *ArXiv*, vol. abs/1905.00641, 2019.

[8] R. Girshick, "Fast r-cnn," 04 2015.

[9] H. Jiang and E. Learned-Miller, "Face detection with the faster r-cnn," pp. 650–657, May 2017.

[10] J. Li and Y. Zhang, "Learning surf cascade for fast and accurate object detection," pp. 3468–3475, June 2013.

[11] P. Viola and M. Jones, "Robust real-time object detection," 2001.

[12] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

[13] P. Dollar, Z. Tu, P. Perona, and S. Belongie, "Integral channel features," pp. 91.1–91.11, 2009.

[14] B. Yang, J. Yan, Z. Lei, and S. Z. Li, "Aggregate channel features for multi-view face detection," *IEEE International Joint*

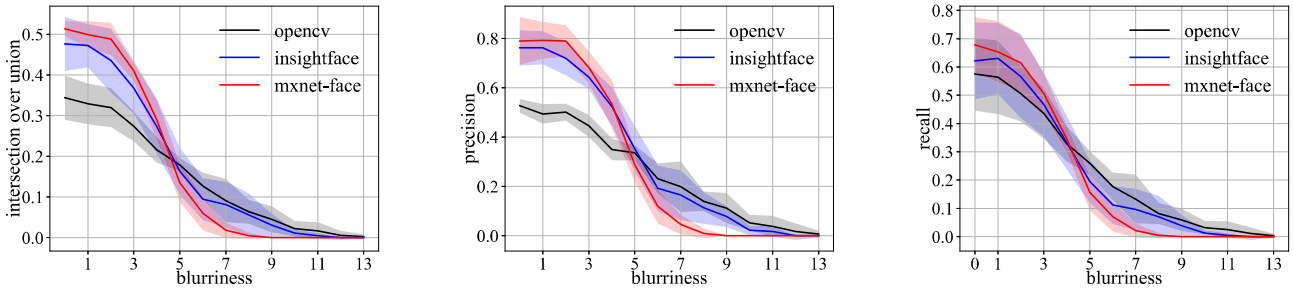

Figure 4: Plots of average intersection over union (left), average precision (middle) and average recall (right) for 5 folds of overall 409 images along the blurriness of these images for `opencv`, `insightface` and `mxnet-face` detector. Lines show mean values and shaded areas show standard deviations over folds for each detector. Values on x axis represent kernel size in percentage of original size of the images.

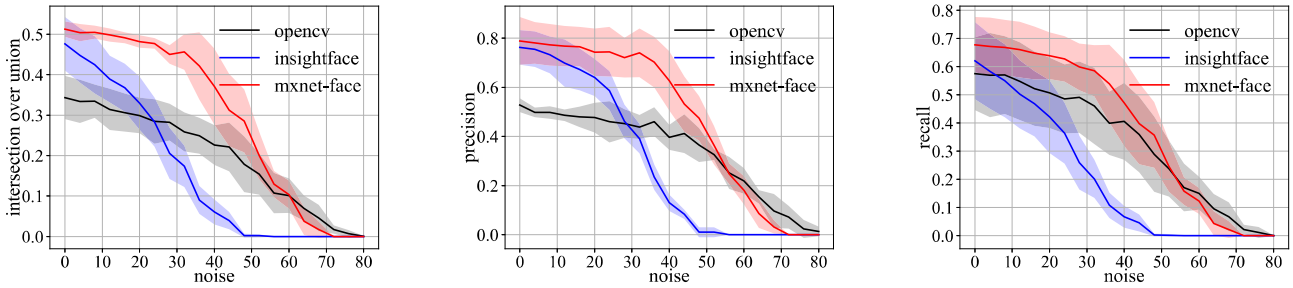

Figure 5: Plots of average intersection over union (left), average precision (middle) and average recall (right) for 5 folds of overall 409 images along the noise of these images for `opencv`, `insightface` and `mxnet-face` detector. Lines show mean values and shaded areas show standard deviations over folds for each detector. Values on x axis represent percentage of noise added to the images.

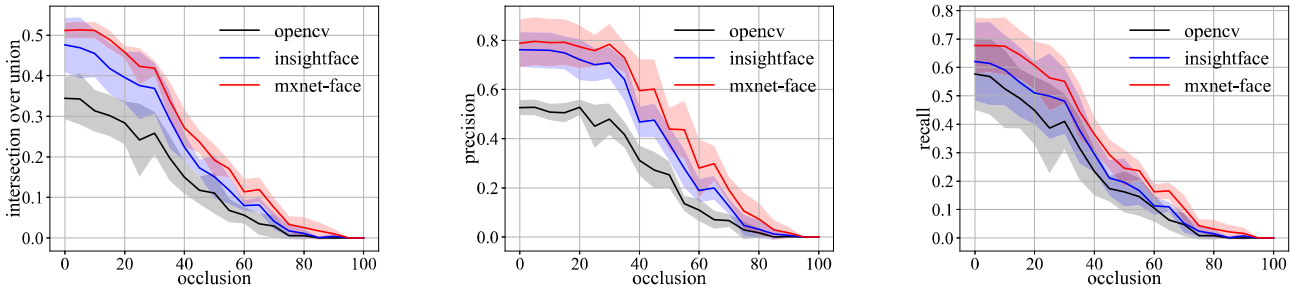

Figure 6: Plots of average intersection over union (left), average precision (middle) and average recall (right) for 5 folds of overall 409 images along the occlusion of these images for `opencv`, `insightface` and `mxnet-face` detector. Lines show mean values and shaded areas show standard deviations over folds for each detector. Values on x axis represent black box sizes inforced to the images in percentage of original size of the images.

Conference on Biometrics, pp. 1–8, 2014.

[15] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection," pp. 5325–5334, June 2015.

[16] S. Yang, P. Luo, C. C. Loy, and X. Tang, "From facial parts responses to face detection: A deep learning approach," pp. 3676–3684, 12 2015.

[17] "https://dataturks.com/projects/devika.mishra/face_detection."

[18] S. Yang, P. Luo, C. C. Loy, and X. Tang, "Wider face: A face detection benchmark," pp. 5525–5533, June 2016.