Face Detection Using Viola-Jones Object Detection Framework

Algot Johansson algotj
Daniel Felczak danfel
Eric Guldbrand ericgul
Oskar Grönqvist oskgro
Rikard Helgegren rikhel

Abstract

This paper describes our implementation of Viola and M. Jones (2001) object detection algorithm applied to faces, along with a discussion of previous work in the field and some of the ethical aspects of face detection and recognition in society. Our implementation does not yield as good results as that of Viola and M. Jones (2001), possibly due to insufficient training data or too low training time, but still manages decently on scanning a real image with many faces.

1 Introduction

The widely used face detection algorithm presented by Viola and M. Jones (2001) is still used today (Parande 2019) thanks to its speed on low performance devices (Viola and M. Jones 2001). In order to explore this algorithm it is re-implemented and tested.

For medical use (Sapiro, Kimmel, and Caselles 1995) and in astronomy (Perret et al. 2010), object detection is a commonly used tool. However, time is not the main concern in these areas as long as the accuracy is good. In contrast, for self driving cars, video games and robots that interact with its surroundings, object detection in real-time and high accuracy is a necessity.

1.1 Background

When Viola and M. Jones (2001) object detection framework was presented it was groundbreaking. Previous solutions to face-detection existed, however, none of the algorithms could detect an object with high accuracy and in a short enough time for it to be real-time. With Viola and M. Jones (2001) algorithm, it was possible to have real-time face-detection due to the algorithms speed of processing, that is, its capability of analyzing and detecting faces in an image at up to 15 frames per second. The success of this method can largely be attributed to the use of specific Haar-features (Papageorgiou, Oren, and Poggio 1998), the calculation of integral images as defined by Viola and M. Jones (2001) for efficiently processing image information, the AdaBoost-algorithm (Freund and Schapire 1997) and the cascading method (Viola and M. Jones 2001; Amit, Geman, and Wilder 1997; Fleuret and Geman 2001).

A feature is a template covering an X by Y rectangle, showing which pixels should be added and subtracted in order to calculate a so-called feature value for an image. This is used instead of analyzing each pixel to increase speed and to encode domain knowledge that is hard to learn otherwise (Viola and M. Jones 2001).

An integral image is a method to represent a grey-scale image by adding its pixel values but keeping it the same size (Viola and M. Jones 2001). More precisely, each pixel, or 2-dimensional position, in the integral image is the sum of all pixel values above and to the left of the same pixel in the grey-scale image. This process is beneficial as many tasks consist of calculating the pixel sum of a given rectangle inside the original grey-scale image. The result is that only four operations have to be made in order to calculate the sum of the rectangle instead of the $m \times n$ calculations for a given $m \times n$-rectangle necessary when using the original grey-scale image.

Adaptive Boosting, AdaBoost, (Freund and Schapire 1997) is a boosting algorithm that is often used in machine learning algorithms (Viola and M. Jones 2001; Ripley 2007; Dietterich 2000). AdaBoost works by creating a strong classifier by combining many weak and somewhat inaccurate classifiers. Weak classifiers are assigned weights based on performance and the strong classifier classifies an image based on a weighted majority answer from all its weak classifier.

A cascade classifier is created from multiple strong classifiers. Each strong classifier represents a single node (or layer) in a degenerate decision tree and is made to have a high detection rate but not necessarily a very low false positive rate. Only images classified as a face will be passed on to the next strong classifier. As an example, 10 strong classifiers with detection rate 0.99 and a false positive rate of 0.3 will have a final detection rate of $0.99^{10} \approx 0.90$ and the final false positive rate will be $0.3^{10} \approx 6 * 10^{-6}$.

The advantage of a cascade classifier over a single strong classifier is that for most images, fewer weak classifiers will have to be evaluated since most images will be rejected in the first or second strong classifier.

Data refill was implemented late in development. This process was described by Viola and M. J. Jones (2004) as part of their cascade training algorithm but not named by itself. It allows the number of negative samples in training data to remain constant for several layers. See section 3 for more details.

There are two measurements in particular that will be referred to in this paper: *detection rate* and *false positive rate* of a classifier. Detection rate measures the proportion of actual faces it correctly labeled as faces. False positive rate measures the proportion of non-faces it incorrectly labeled as faces. The quality of a given cascade is entirely determined by these two rates.

2 Previous and related work

Rowley (1999) presented a face detection algorithm that used a neural network based algorithm. In short, the algorithm consisted of running a 20x20 sized window across an input image of some arbitrary size, then for each window, a pre-processing step is taken which ultimately serves to increase the contrast in the sub-image to further ease detection for the neural network. The processed 20x20 window is then passed as an input to their neural network. In the neural network multiple receptive fields are used to detect if sub-images contain a face or not. The receptive fields analyze the image at several different scales and locations. After an entire image is processed, an arbitration stage is run, eliminating overlapping detections. Their algorithm could process a 320x240 pixel image in less than 5 seconds in 1998, on an SGI Indy which had a microprocessor operating at 100 Mhz. Additionally, their detection rates varied between 80 percent and 90 percent with an acceptable false positive rate. In comparison, the Viola-Jones face detection algorithm performs about 15 times faster at a higher rate of accuracy.

3 Implementation

Images from pre-existing datasets (*CBCL Face Database #1* n.d.; Huang et al. 2007) are cropped and scaled down in advance to 19x19 pixels so that they all show similar face regions. This size is used to speed up training without too much of an effect on detection accuracy (Viola and M. Jones 2001). Non-face data was taken from existing datasets (*CBCL Face Database #1* n.d.) and by crawling the web and a stock-photo site. Some non-face data used may contain faces anyways since they were not curated, but later non-face data collected for the data refill process was curated by us to ensure no human faces were present. However, some animal faces were left in.

After collection and pre-processing, upon starting traing each image has its integral image calculated. Feature values are pre-calculated for as many images as memory allows. When feature values are pre-calculated for all training images, training time is cut in almost half.

The cascade training algorithm presented by Viola and M. J. Jones (2004) is used to build a cascaded classifier. In each step, the AdaBoost algorithm (Viola and M. Jones 2001) is used to create an increasingly complex strong classifier until the cascade requirements for that layer are met. These requirements consist of a maximum false positive rate per layer, and a minimum detection rate per layer. Each time a strong classifier completes training, the current cascade classifier is used to remove all negative samples

from the training data that are correctly classified. The current cascade classifier is then used to perform data refill by scanning large pictures containing no faces, selecting each sub-window it (incorrectly) classifies as a face and adding it to the negative samples in training data. This allows the number of negative samples to remain constant until refill data becomes depleted. Finally, when the cascade reaches a target maximum false positive rate, the algorithm terminates. In addition to the Viola and M. J. Jones (2004) version of the algorithm, our algorithm is also allowed to terminate once the amount of available negative samples is low enough. This happens if there is not enough data to be used for data refill.

In addition, a detector that uses a trained classifier to scan and mark faces in any real image was implemented. This detector applied to an example image can be seen in Figure 3.

Functions to allow for easy saving and loading of trained classifiers were also implemented.

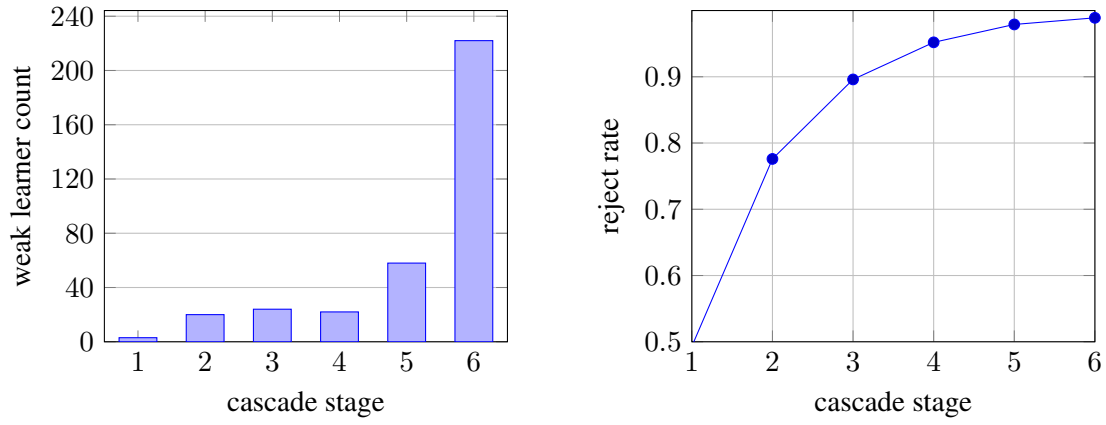
During development, especially early, several unit tests were written to ensure the correct function of elementary parts, in particularly for integral image creation and feature value calculation.

To speed up training, two important decisions were made that significantly reduced training time but could potentially be a source of inaccuracy for the final classifier. The effects of these decisions received cursory testing to ensure no obvious accuracy loss, but due to the time involved in training, there was not enough time to properly compare the differences. In addition, step 2 of the AdaBoost algorithm (as described by Viola and M. Jones (2001)) was parallelized. This cut training time by at least a factor of 4 but should not have affected classifier accuracy in any way.

The first decision is to use only $1/4$ of all possible features. This is done by only using features with their upper left corner on even coordinates ((0, 0), (0, 2), (2, 2), ...) instead of every coordinate.

The second decision is that to, when training a weak classifier and trying to find the optimal threshold, only trying every 100th threshold instead of every single one. Since the possible thresholds are sorted by value and there are as many thresholds as there is data (about 14000), it is reasonable to assume that even when just trying every 100th, the best one found will still be close to the optimal. However, it is possible that this does not work as well when the remaining amount of data drops too low. In our best implementation, data dropped to about $4000/1000$ positive/negative samples on layer 6. Performance was not improved further and the final classifier was recovered from auto save of the first 5 layers. While there are more sophisticated ways than skipping to every 100th, logarithmic search for instance, we did not implement that due to project time constraints.

4 Results



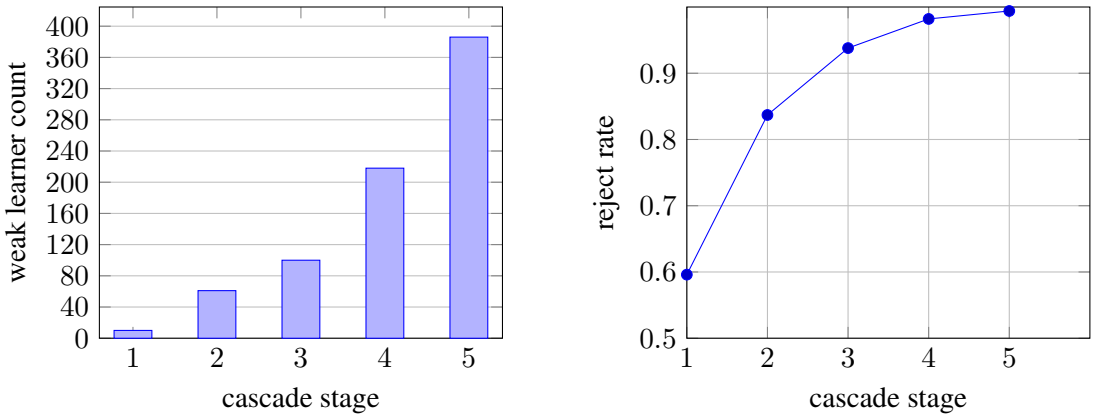
(a) *Number of weak learners on each stage.*

(b) *Total reject rate at each stage.*

Figure 1: *Weak learner count and total rejection rate per stage in a cascade classifier trained before implementation of data refill with* 94.59% *detection rate and* 1.08% *false positive rate on test data. Trained on* $4000 + 1000$ *positive samples and* $10000 + 2000$ *negative (training + validation). Tested on* 7302 *positive and* 50000 *negative samples.*

Multiple classifiers were trained using slightly different parameters and training data. Figure 1 shows

weak learner count and total reject rate (the fraction of negative samples correctly classified as negative) for each stage of one cascade classifier. This classifier was trained before implementation of data refill and has a 94.59% detection rate and a 1.08% false positive rate on test data. It was trained with a target max false positive rate of 1%, which it was unable to reach before running out of negative samples, a max false positive rate per layer of 50% and a minimum detection rate per layer of 99%. It was trained on 4000 positive samples and 10000 negative samples in the AdaBoost algorithm and on an additional 1000 positive and 2000 negative samples for validation during the cascade building process. It was then tested on 57302 samples, 7302 positive and 50000 negative.



(a) *Number of weak learners on each stage.*

(b) *Total reject rate at each stage.*

Figure 2: *Weak learner count and total rejection rate per stage in the best cascade classifier trained, after implementation of data refill. It has a* 97.45% *detection rate and* 0.63% *false positive rate on test data. Trained on* 4000 *positive samples and fed new negative samples that the classifier-in-training classified as positive, after each layer. Tested on* 6426 *positive and* 50000 *negative samples.*

Figure 2 shows weak learner count and total reject rate for each stage of the best cascade classifier trained. This classifier was trained with data refill and was allowed to scan 626 (roughly 500 by 500 pixel) images for negative samples after every stage. This cascade classifier has a 97.45% detection rate and a 0.63% false positive rate on test data. It was trained for 12 hours with a target max false positive rate of 0.1%, which it was unable to reach before running out of negative samples despite data refill, a max false positive rate per layer of 40% and a minimum detection rate per layer of 99.5%. It was trained on 4000 positive samples and 10000 negative samples on each layer before data started running low. It was then tested on 56426 samples, 6426 positive and 50000 negative. This classifier was used by the detector in Figure 3.

The detector was tested on a 1200 by 868 pixel group photo from the Solvay conference and the result with rectangles drawn around all sub-windows where a face was detected can be seen in Figure 3. It took 8.5 seconds on a 3.5 GHz processor to scan the entire image with a fixed sub-window scale. This detector used the best cascade classifier, shown in Figure 2. In comparison but not shown, a detector using the earlier classifier shown in Figure 1 detected less actual faces but many more hands, leaves and part of the wall.

5 Ethical aspects

Due to the fact that any facial recognition system today will need some form of face detection in order to deduce if an image contains a face, the technology discussed in this essay is highly relevant to the increased use of facial detection software used today in modern hardware such as smartphones and smart surveillance. The following example describe software which is actively used today and we will touch briefly on the ethical implications of further developments of such technology.

The relevant example of facial recognition software, as we see it, being actively used today, is the wide application of facial recognition systems existing in China. They use so called smart surveillance systems

Figure 3: *Demonstration of the detector on a group photo from the Solvay conference using the best achieved cascade classifier. It has a tendency of miss-classifying areas of high contrast, especially ties. However, in comparison to earlier classifiers, it does not detect hands.*

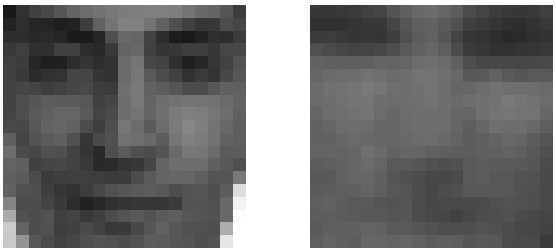
which are cameras with facial recognition technology. The intended use for this technology, according to the Chinese government, is to detect active suspects for criminal cases, find reported missing people and provide security for its citizens. By using technology, the cities police force have been able to locate suspects on warrants and subsequently made arrests. Megvii, a company responsible for developing one of the systems, reported that since 2016, with the help of the facial recognition software, roughly 1000 arrests have been made in Hangzhou, a city in China with 8 million citizens residing in its urban area (Denyer 2018).

We believe the largest ethical aspect of this sort of technology is the introduction of a new level of intrusion into peoples lives. As seen in China, with the help of a ubiquitous network of cameras employing facial recognition software while utilizing the massive modern computational power available today, they track and store explicit data about individuals. For 2020, China plans to have implemented a sophisticated social credit system (Marr 2019), which will include a social credit score on all the citizens of China. This data will be publicly available for both public and private companies to access. This means that civilians in China will have their lives publicly available. This will have an effect on the individuals lives, without their consent, and further, is a breach of privacy, according to todays right to privacy in many countries. Further, such data, used by private companies and the government, can be used for multiple instances, e.g., targeted advertising for political campaigns. Although, this sort of software can and will provide many benefits, but, the potential for misuse is large, as seen in China, which have developed their software without the citizens consent and without transparent discussions.

6 Conclusion

Initially the training and testing data differed in quality as demonstrated in Figure 4. This caused all classifiers to perform well on training data but very poorly on test data. To mitigate this, new sets were created by combining initial training and testing data and re-partitioning them into new sets. For training

the final classifier, none of the original positive test samples were used. While not a primary result of this study, this clearly showed us the importance of using good data.



(a) *A face from the training data*

(b) *A face from the test data*

Figure 4: *Example of the difference between the test data and the training data in the original data set. As can be seen the image from the training data is sharper. Most of the faces in the test-data were found to be lacking in quality.*

For the cascade classifier shown in Figure 2, a 97.45% detection rate and 0.63% false positive rate may seem quite good, but applying this to face detection in real images is another matter. Figure 3 is marked with rectangles where the classifier has classified a sub-window as a face. Some are faces, but the classifier has a tendency to prefer areas of high contrast, such as ties rather than actual faces. Clearly, this classifier is not good enough for real-world use. In contrast, Viola and M. Jones (2001) achieved a 95% detection rate and a false positive rate of 1 in 14084 but even so they later pointed out that for real applications, the false positive rate must be closer to 1 in 1000000 (Viola and M. J. Jones 2004).

One major difference between our classifier and that of Viola and M. Jones (2001) seems to be that our had significantly less tweaking of values and a much more limited data pool. With more time, the improvements seen by the tweaking that was done between the classifier in Figure 1 and Figure 2 alone suggests that more tweaking and better data would yield better results.

Figure 3 also demonstrates another point, the difficulty of choosing a window size. All features are originally 19 by 19 pixels, so to find faces in an arbitrary image it might be necessary to either try all possible scales of the sub-window, or to carefully select a good scale. This somewhat diminishes the use of this method on real images.

Another question arises if scaling is chosen. Should the image be scaled down, or should the detector be scaled up? Viola and M. Jones (2001) used scaling of the detector "because the features can be evaluated at any scale with the same cost" (Viola and M. Jones 2001).

Furthermore, the detector sliding window converts every sub-window to an integral image before testing whether it is a face or not. A possible improvement would be to convert the entire image to an integral image rather than calculating an integral image for every sub-window. This might reduce the detection time significantly from its current 8.5 seconds.

The decisions to reduce the number of features and the number of thresholds tested, explained in section 3, proved to be very useful since it was necessary to retrain the classifier many times, using multiple variations of training parameters and data sets. This would not have been possible if training took 4 to 400 times as long. A subject for future studies might be to document what effect these shortcuts actually have for the accuracy of the classifier or how far they can be taken without much accuracy loss.

7 Contribution

All authors contributed to writing the code and the essay, however, Algot Johansson and Eric Guldbrand were mainly responsible for implementing the algorithms, Rikard Helgegren focused on writing tests and performing manual calculations to verify particularly critical implementation steps. Oskar Grönqvist collected data to augment our existing datasets and worked on using a classifier to detect many faces in one image. Daniel Felczak contributed more to the essay.

References

Viola, P. and M. Jones (2001). "Rapid object detection using a boosted cascade of simple features". IEEE, p. 511.

Parande, A. (2019). URL: https://medium.com/datadriveninvestor/understanding-and-implementing-the-viola-jones-image-classification-algorithm-85621f7fe20b.

Sapiro, G., R. Kimmel, and V. Caselles (1995). "Object detection and measurements in medical images via geodesic deformable contours". *Vision Geometry IV*. Vol. 2573. International Society for Optics and Photonics, pp. 366–379.

Perret, B. et al. (2010). "Connected component trees for multivariate image processing and applications in astronomy". *2010 20th International Conference on Pattern Recognition*. IEEE, pp. 4089–4092.

Papageorgiou, C., M. Oren, and T. A. Poggio (1998). "A General Framework for Object Detection". *ICCV*.

Freund, Y. and R. E. Schapire (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences* **55**.1, 119–139.

Amit, Y., D. Geman, and K. Wilder (1997). Joint induction of shape features and tree classifiers. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 11, 1300–1305.

Fleuret, F. and D. Geman (2001). Coarse-to-fine face detection. *International Journal of computer vision* **41**.1-2, 85–107.

Ripley, B. D. (2007). *Pattern recognition and neural networks*. Cambridge university press.

Dietterich, T. G. (2000). "Ensemble methods in machine learning". *International workshop on multiple classifier systems*. Springer, pp. 1–15.

Viola, P. and M. J. Jones (2004). Robust Real-Time Face Detection. *International Journal of Computer Vision* **57**.2, 137–154. ISSN: 1573-1405. DOI: 10.1023/B:VISI.0000013087.49260.fb. URL: https://doi.org/10.1023/B:VISI.0000013087.49260.fb.

Rowley, H. A. (1999). *Neural network-based face detection*. Tech. rep. CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF COMPUTER SCIENCE.

CBCL Face Database #1 (n.d.). www.ai.mit.edu/courses/6.899/lectures/faces.tar.gz and http://cbcl.mit.edu/software-datasets/FaceData2.html.

Huang, G. B. et al. (2007). *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*. Tech. rep. 07-49. University of Massachusetts, Amherst.

Denyer, S. (2018). *China's watchful eye*. URL: https://www.washingtonpost.com/news/world/wp/2018/01/07/feature/in-china-facial-recognition-is-sharp-end-of-a-drive-for-total-surveillance/?noredirect=on&utm_term=.2eb7c4367557.

Marr, B. (2019). URL: https://www.forbes.com/sites/bernardmarr/2019/01/21/chinese-social-credit-score-utopian-big-data-bliss-or-black-mirror-on-steroids/#4cec43dc48b8.