Deep learning-based video anonymization for security and privacy purposes

IMT4392 - Deep learning for visual computing

Jon Gunnar Fossum, Benjamin Skinstad

December, 2020

Abstract

Over the last years the topic of privacy and security has been a driving factor in the design of systems. With the introduction of GDPR, companies has been forced to consider the privacy of people they store identifiable data on, or risk heavy fines.

In this paper we propose a system that will detect and censor faces in a privacy focused manner. We constructed a system that takes in a video as input, and automatically detects and censors facial features. We based our implementation on a custom configuration that used Facebook's state of the art object detection system, Detectron2. We built upon this system, making it capable of performing facial detection, and facial censoring.

We conducted a small user study of videos our system had censored, to test how well our solution performed. Results of this study indicates that our implementation is capable of semi-reliably censor facial features to such a degree that people are not able to recognise the censored individual. Our results also indicates that our system might work very well with minor modifications done to it.

This paper focuses on demonstrating anonymization of facial features by using generic object detection methods.

Keywords: Anonymization, Detectron, Facial detection, GDPR, Object detection, Privacy, Video censorship

Preface

We want to thank Mohib Ullah for supervising the project and providing us with much needed guidance and assistance when necessary.

Contents

Figures

Chapter 1

Introduction

Data privacy and security are of utmost importance for organizations, it is important because of GDPR. In this project, we will explore deep learning-based video anonymization techniques that yield a consistent and reliable solution for anonymizing the subjects in videos and images.

With GDPR being introduced in 2018, and with many more privacy laws being introduced frequently, many companies are now looking into ways to efficiently anonymize their data. Companies like Google, have already applied facial anonymization techniques inside some of their programs, for example, as seen in figure 1.1 people depicted in Google earth have their faces automatically censored.

This project will focus on the detection and anonymization of human faces in videos and images, with core focus being on the anonymization aspect. This paper will be separated into two main parts: The first main part will focus on facial detection and anonymization while part two will focus on facial morphing and artistic anonymization of faces. The latter part will focus on making the censored faces look more human, while still keeping the identity unrecognisable.

Figure 1.1: Google Earth bluring

1.1 Task description and goal

We were given the task to anonymize videos containing humans and other identifying information like car number plates. Other than this, we were not been given any specific constraints or restrictions to adhere to, but we will, for personal reasons aim at an implementation done in either PyTorch or Tensorflow. Our supervisor requested us to look into alternative ways of facial censoring if we have time at the end of the project.

During our project, we will focus on the anonymization of human faces and facial features over complete anonymization of human bodies, facial features and other identifying information. This is to avoid over-scoping during development.

Research questions

While we were doing preliminary research we created these questions we needed to answer before creating our system:

- What are the current trends in deep learning?
- What are current state of the art object detection systems?
- How are these systems used in facial detection?

To do this project we will first try to construct a system capable of detecting faces in videos, and then we will build a second system that takes inn these faces, and censor them.
The main milestones for this project:

- Detect faces
- Censor faces
- Make the system work on videos
- Research alternative censoring algorithm to try to make the video less jarring to look at, eg. StyleGAN

1.2 Learning outcomes

Starting this project, we both had a more general technical experience in programming than knowledge about Machine/Deep learning. During our bachelor we had a more general AI focused course, which did not cover deep learning in detail, and did not have any applied deep learning projects with relation to computer vision. With this, and our research questions in mind, we wanted to learn more about these topics:

- Expand our theoretical and practical knowledge of AI systems.
- Get a better understanding of state of the art deep learning models.
- Get applied experience in designing and creating deep learning models.
- Get a better understanding of how object detection systems are used in facial detection systems.
- Gain general experience in state of the art object detection systems.

- Learn how to use PyTorch or Tesorflow in a practical project.

Chapter 2

Related work

Facial detection systems are often built upon general object detection algorithms. We are looking for an implementation that mainly focuses on facial detection, but has the capability to be more generalized in the future.

OpenCV is a python library for computer vision applications. OpenCV is widely used for object detection, and face detection systems. The library is well documented, and there are a myriad of tutorials on how to do facial detection using this library. Many other systems, like Pyimagesearch [1] is built on top of OpenCV, while other systems like Keras, can easily work in tandem with OpenCV.

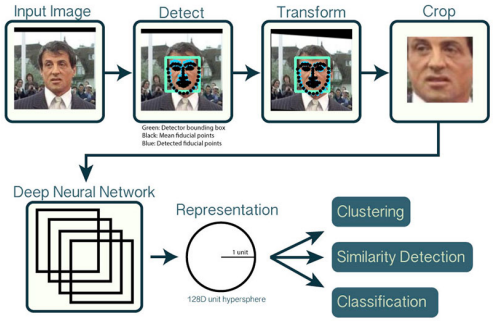

Figure 2.1: An overview of the OpenCV face recognition pipeline.

DeepPrivacy[2] A project that aims to solve the same problem as we tried to. We discovered this paper after our deadline for development had passed. Instead of blurring or morphing faces, this paper proposes a solution where a GAN fills in a blank section in an image where a face used to be.

Facepixler[3] is a website where users can upload a picture of a person, and have any faces detected in said picture blurred. The site offers automatic facial detection, and will try to censor the faces detected. The site offers manual censoring, if the user is unhappy with the automatic one.

Generative Adverserial Networks(GANs)[4] Is a fairly new framework that was proposed in 2014, and has seen a lot of success in applications that generates images of numbers, faces, animals, and more. It uses two models, one generative and one discriminative, that competes in a zero-sum two player game. In 2018, NVIDIA Labs published a GAN system named StyleGAN [5], this project was further improved in 2019[6][7].

Filmora[8] is a video editing program that has a built-in facial detection system. Filmora allows editors to automatically track and censor faces. Filmora offers a wide array of censoring methods, some of whom are very effective, while others are more artistic.

Chapter 3

Research: Object detection

While doing preliminary research on the current "state of the art" object detection models, we started by looking for the most popular available solutions. We tried to implement many of the most popular available "state of the art" object detection models without luck, but after some time, we managed to create a system running Detectron 2. Detectron 2 is a open-source object detection model created Facebook. Facebook states that Detectron 2 is the current object detection system they use on all of their platforms. Detectron 2 GitHub repository can be found here [9]. Detectron 2 is talked about in more detail in section 3.2

We will dedicate this chapter to discussion of general object detection models and Detectron 2.

3.1 Facial detection vs Object detection

At the start of the README file for Detectron2, it states that it "implements state-of-the-art object detection algorithms". For our purposes this sounds promising, but how well does it do facial detection?
According to the TinaFace[10] paper, the risk of choosing a generic object detection model over a facial detection model should not impose any major problems. A facial detection model is just a specialised object detection model.

3.2 Detectron 2

Detectron 2 (hereby referred to as Detectron)[9] is the current object detection system used by Facebook. Detectron is built upon years of object detection research done by Facebook, and the newest version was published October 2019. Detectron is under continuous development and sees updates weekly. Detectron offers developers the ability to easily create their own models, train using custom data sets, or use one of their many pre-trained models.

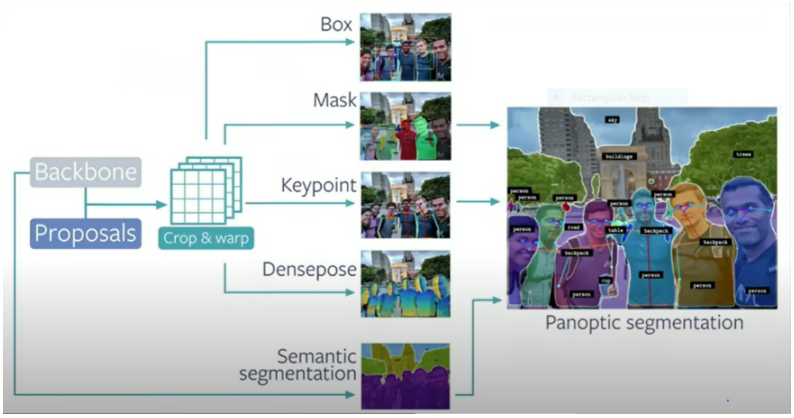

Figure 3.1: General overview of Detectron 2

Detectron support configurations for various object detection algorithms, some of these are:

- Mask R-CNN
- Faster R-CNN
- Fast R-CNN

Our system is using Detectron's built-in Fast R-CNN configuration. Our system have functionality for Mask R-CNN, but said functionality is not used in our current iteration of the system.

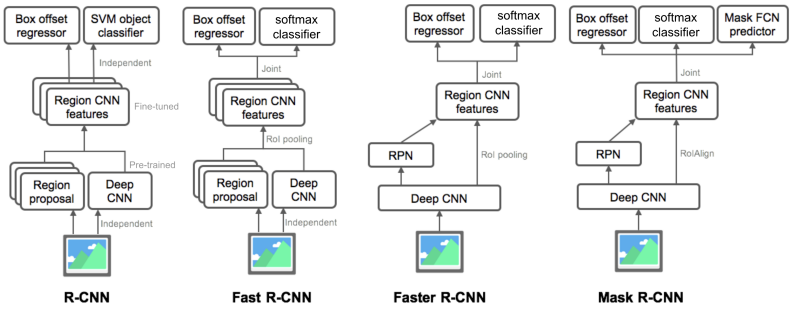

Figure 3.2: Overview of the model designs of R-CNN, Fast R-CNN, Faster R-CNN and Mask R-CNN [11]

For our system, some of the benefits of using Detectron are:

- Simplicity: Detectron is easy to set up, and the Detectron GitHub contains all necessary resources to set up projects.
- Ease of use: Before selecting Detectron, we tried to implement a wide array of other facial and object detection systems unsuccessfully. Many of the systems we tried to implement required complex setup and prerequisites. Many of these prerequisites were outdated. In contrast, the setup of Detectron was simple, and can easily be loaded into a Google colab notebook.

- Well documented: Detectron has a well-documented API, many tutorials, and an active scene of daily users. The Detectron 1 and 2 system has existed for many years, with extensive real-life testing.

Chapter 4

Methodology

While we were researching facial detection models we came across an article by Venelin Valkov where he talked about his implementation of a facial detection system using Detectron 2[12]. This article was very detailed and showed great results in terms of facial detection. We were impressed with the details of the article, and we decided to follow his setup and fit the system to our use case.

Venelin Valkov provides code for training of a custom model, but we are basing our implementation on the pre-trained weights he provides for his Detectron model. The pre-trained model was rigorously tested against a large number of images, and we were satisfied with the results to continue using these weights for our project.

Figure 4.1: Output from the Valkov model on custom dataset

For our implementation, we first changed his system to allow for the reading of custom images provided by us. The goal is to censor videos, but to make that work, we need to divide the videos into individual frames and censor them, and to then combine them back into a video. The frames are individually passed through

the prediction algorithm, and they are individually given a prediction box for each face as seen in figure 4.2.

Figure 4.2: Facial prediction with bounding box using the Valkov model

After a frame is passed to the predictor, it returns a tensor[1] containing all necessary information needed to locate all predicted faces in an frame. By accessing the variables in the tensor, we are able to find the number of predicted faces, and positions of the corners of all of the prediction boxes.

```
for x in range(numberOfBoxes):
    x1=value.pred_boxes[x].tensor.cpu().numpy()[0][0]
    y1=value.pred_boxes[x].tensor.cpu().numpy()[0][1]
    x2=value.pred_boxes[x].tensor.cpu().numpy()[0][2]
    y2=value.pred_boxes[x].tensor.cpu().numpy()[0][3]
```

Code listing 4.1: Code used to extract corner positions of each prediction box found in a tensor

After we have iterated over a frame based on the number of faces predicted, and retrieved the coordinates of the corners of the prediction boxes. We pass these coordinates into a crop function. We crop out every face predicted in the original frame while we save the corner coordinates for further use. These cropped images are passed through a blurring algorithm. The strength of this algorithm is decided based on the size of the input image. The cropped blurred faces are saved to file with a special naming convention to make it possible to link the names to the coordinates later on. We decided to save the faces to file because of various compatibility issues, with the included libraries. These libraries were causing problems with storing the images in memory and operating on them.

When the function is done iterating over all the faces that were predicted by the prediction algorithm, we iterate over all the saved coordinate sets. For each

[1] A Tensor in this case returns a multi-dimensional array

coordinate sets, we find the corresponding cropped face that was previously saved to file, then paste said censored face onto the original frame. For each face, we stored 4 coordinates, one for each corner of the prediction box. These were stored to guarantee that we paste the cropped face back at its correct position. Using only one corner did not guarantee the cropped face is pasted in its original scale.

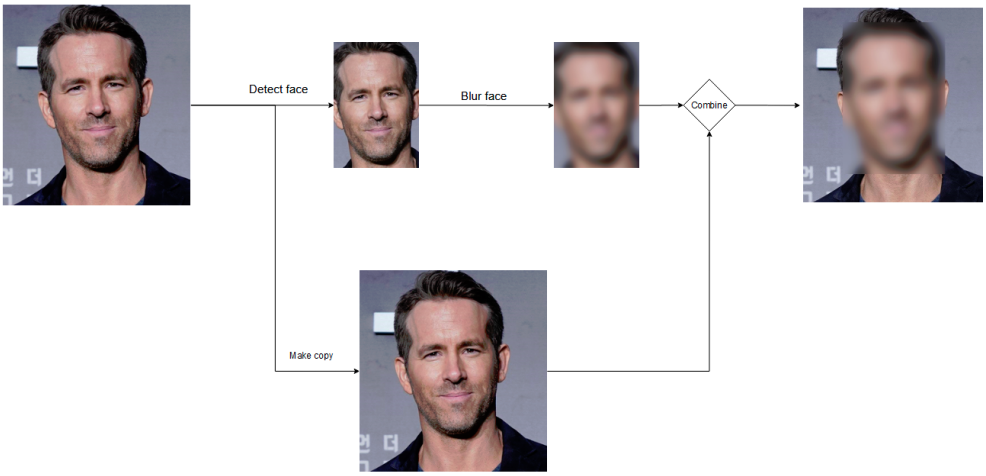

Figure 4.3: Our different system modules and how they relate

The blurring filter we used on our images, was the Gaussian blur filter provided by the Python skimage library. We used an initial strength of 15 on our images. After testing, we realised we needed to increase the blur strength on images of higher quality. This is because the blur effect was very weak on high resolution images. To combat this, we created functionality to make our blurring filter's strength scale with image size.

Figure 4.4: Censored video frame after the video has been trough our system

To make the system work on videos, we take in a video file as input, and extract all the frames of the video. These frames are saved in order as .png files in an external folder. Our program iterates over all the video frames saved in the external folder folder, and performs prediction and potential censoring on the frames. The frames are then saved away in a different folder. When the program is

done iterating over all the frames from the video, the frames are combined back into a censored video. A visualisation of our complete system architecture can be seen in figure 4.5 and an example video is included as extra material to this hand-in, or can be seen here: `https://vimeo.com/490538009`.

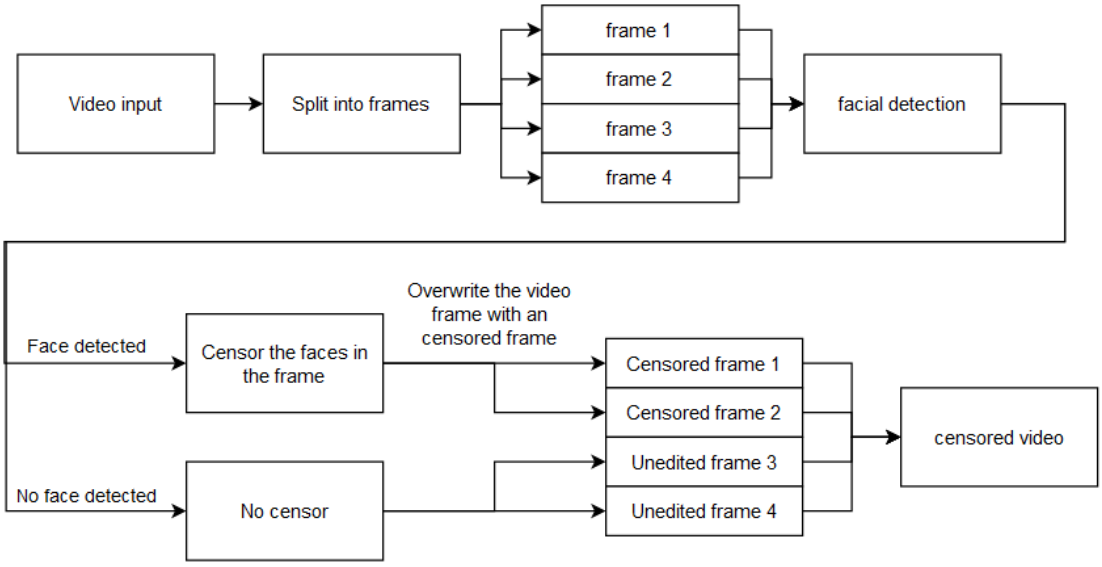

Figure 4.5: Figure showcasing how our system takes a video file input, and converts it to a censored video

Chapter 5

Experiment

After the system was created, we were able to censor any video we put into our application, but we were unsure how effective the system was at censoring the people in the videos. To study the effectiveness of our system, we designed and performed a study. What we want to know, is how effective our system currently is. To test this, we will conduct a survey containing several videos we have censored, and we want to find a way to see how well our system hides the identity of the actors in the videos.

Our hypothesis is that our system is able to censor videos in such a way that a test candidate viewing the video, will not reliably be able to identify the person removed. The null-hypothesis for this experiment is that our system will not have any effect, and the people censored will still be identifiable after the censoring process.

We assume that our system will be able to anonymize the actors in the video, but we also assume that it might not be able to reliably remove all facial features necessary for complete anonymization.

To test the effectiveness of our system, we performed a short study.
A copy of the survey we distributed can be viewed here: `https://bit.ly/3qEIWRD`

In this study, we censored 3 short videos containing 4 actors in total. The test subjects saw these 3 clips, one by one, and were then presented with a number of options with pictures containing faces. An example question can be seen in fig **5.1**. The test subjects were then asked to pick out the image corresponding to the actors who had been censored from the videos. The actor's image was hidden among other pictures of people who looked like the actor. In case the candidates had prior experience with that actor or clip we added an option to say they were familiar with said actor. Those responses are not taken into account in the statistics.

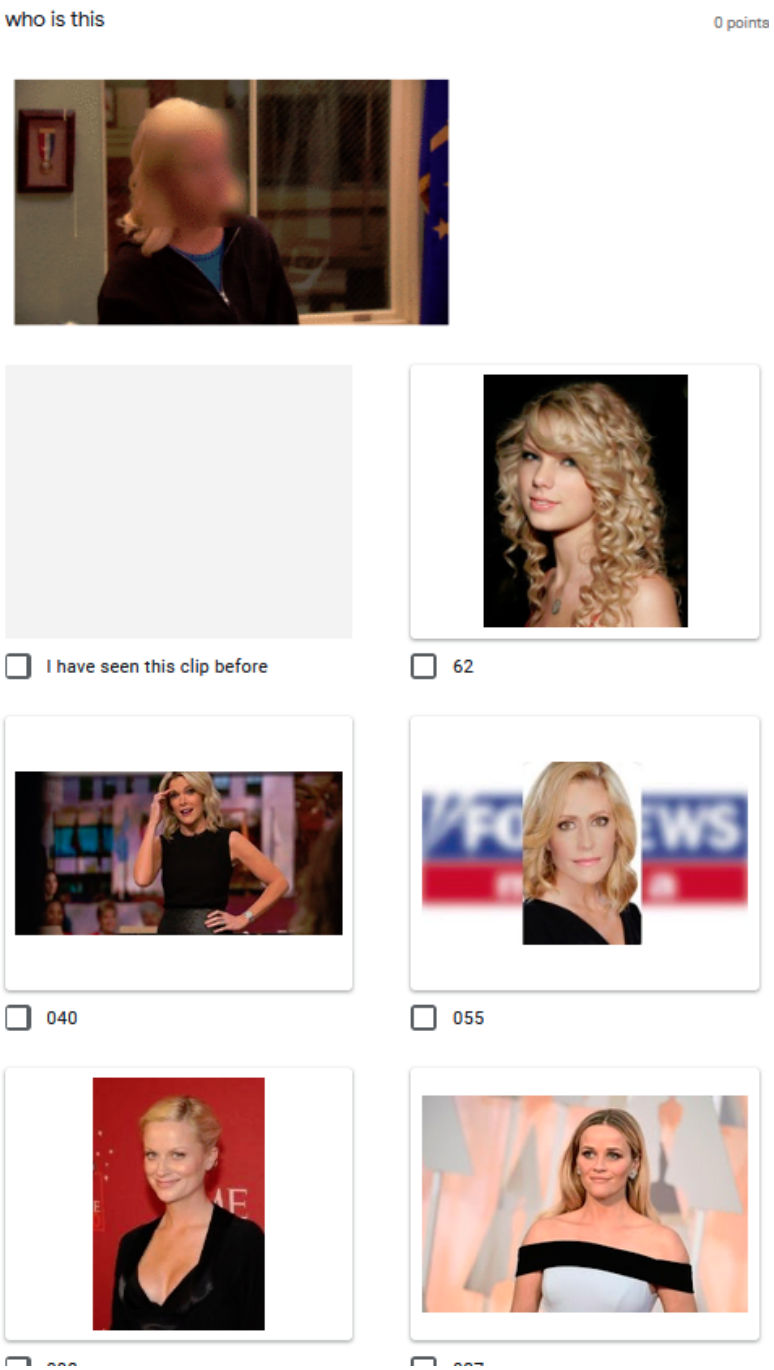

Figure 5.1: Example question from our survey

Survey Result In total we received responses form 51 candidates, split across 3 questions, who gave us in total 181 alternatives chosen.

Correct answers	70
Wrong answers	84
Knew the clip from before	27

Table 5.1: All recorded responses

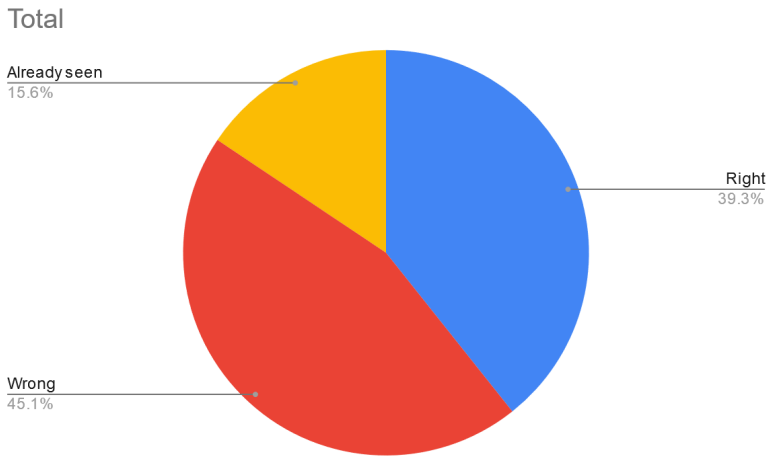

Figure 5.2: Relationship between the 3 answers

After reading over the answers, we discovered that in question 1, there were 8 candidates who only answered one option. In question 2 there was one candidate who gave multiple answer, and in question 3 there were 2 candidates who gave multiple answers. We **discard these answers for question 2 and 3 and record question 1 both with and without these outliers**. In our chart, the color RED means the candidates answered wrong, BLUE means correct, while YELLOW means they had seen the clip from before. The data with outliers removed is seen in fig **5.2**. When comparing data we disregard the "Knew the clip from before" option. This is due to any candidate who answer this option know the correct answer.

Correct answer	68
Wrong answer	78
Knew the clip from before	27

Table 5.2: All recorded responses with outliers removed

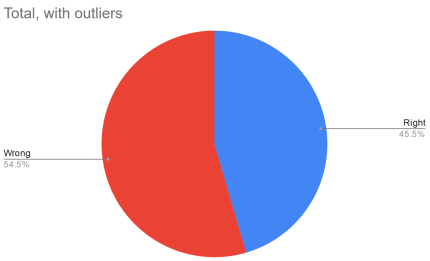

Figure 5.3: Total with outliers included

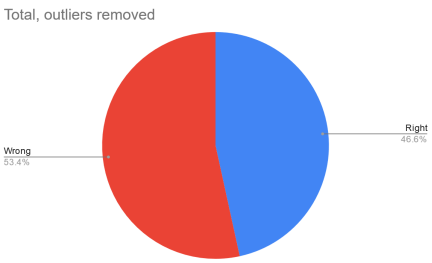

Figure 5.4: Total with outliers removed

Question 1 results Question 1 had 2 potential correct answers, the candidate was asked to provide 2 answers to this question. For Question 1 we recorded these results:

Correct answer	42
Wrong answer	38
Knew the clip from before	5

Table 5.3: All recorded responses for Question 1

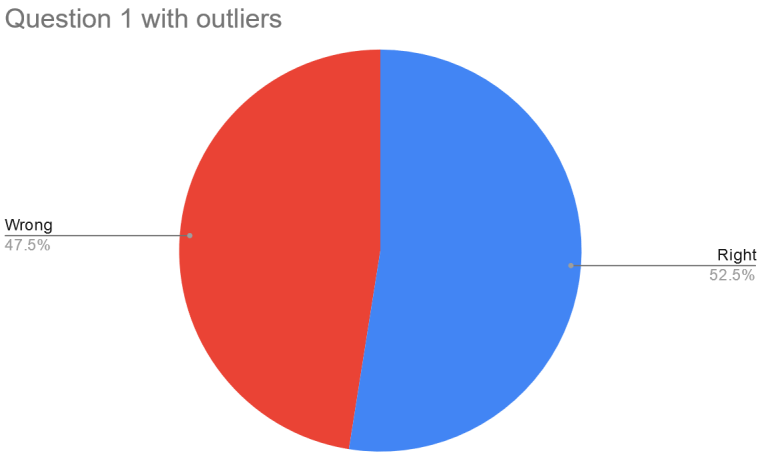

Figure 5.5: Chart of total number of answers for Question 1

As mentioned earlier, there were 8 candidates who did not give 2 answers to this question. Removing their answers we record:

Correct answer	40
Wrong answer	32
Knew the clip from before	5

Table 5.4: All recorded responses for Question 1 with outliers removed

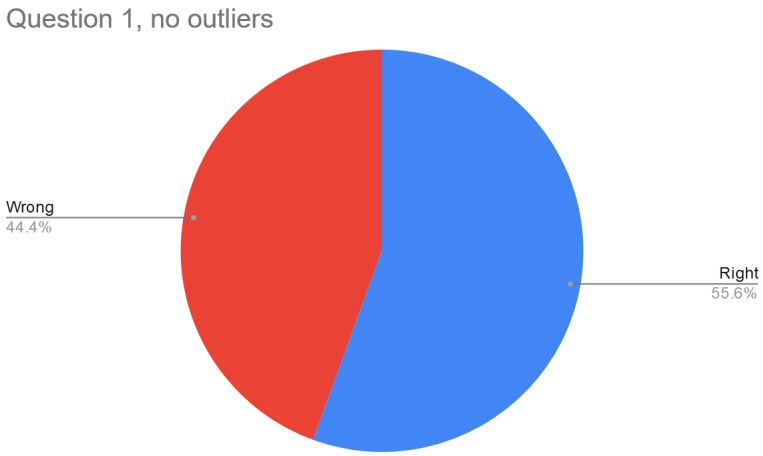

Figure 5.6: Chart of total number of answers with outliers removed for Quesiton 1

Question 2 results In Question 2, one candidate had their response removed due to the providing 2 answers while we only asked for 1.
For question 2 we record these results:

Correct answer	10
Wrong answer	26
Knew the clip from before	11

Table 5.5: All recorded responses for Question 2

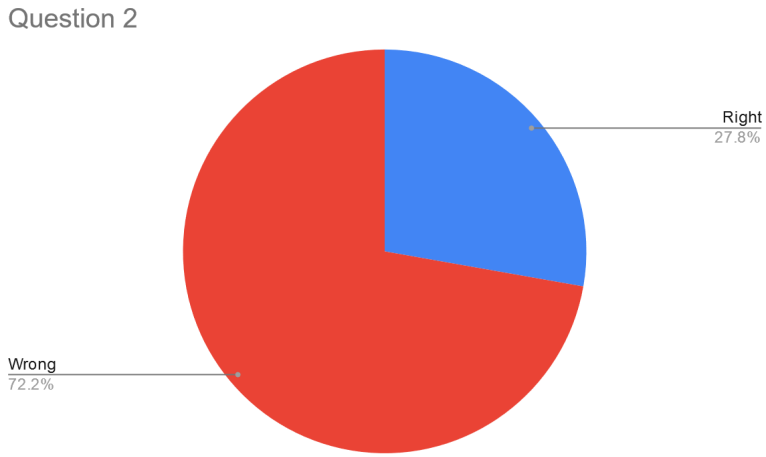

Figure 5.7: Chart of total number of answers with for Question 2

Question 3 results In Question 3, two candidates had their response removed due to them providing 2 answers while we only asked for 1.
For question 3 we record these results:

Correct answer	18
Wrong answer	20
Knew the clip from before	11

Table 5.6: All recorded responses for Question 3

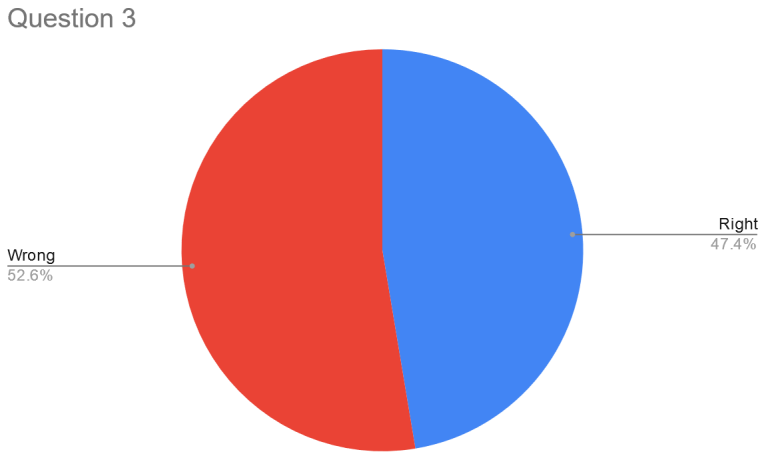

Figure 5.8: Chart of total number of answers for Question 3

Candidate feedback: As part of the survey, we gave candidates the option to provide us with direct feedback. This was optional and 3 participants gave us

feedback relevant to the survey. The 3 comments all mention that our system does not do a well enough job in censoring hair and other identifying non facial features. As seen in fig 5.9 and fig 5.10

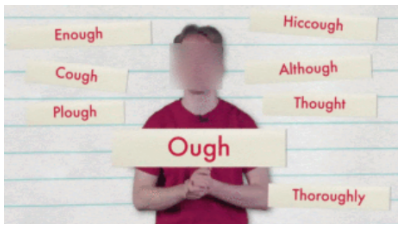

Figure 5.9: Example Question image

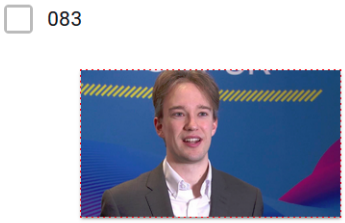

Figure 5.10: Unblured correct answer

Discussion of the results In our survey, there were a varying number of alternatives per question. The first question has 2 correct choices out of 10. That's 40% chance of getting at least one correct, and 2.2% chance of getting both actors correctly, assuming a fully random guess. How to measure that up against our results became a bit unclear after we tallied the results from question 1 as two entries.

The second question has 9 actors to chose from. That is an 11.1% chance of guessing correctly, assuming a random guess. In our result we found that 27.8% of the participants were able to identify the correct actor. Meaning that our participants did better than a random guess by a factor of 2.5.

The third question has 8 actors to chose from. That is a 12.5% chance of guessing correctly, assuming a random guess. In our result we found that 47.4% of the participants were able to identify the correct actor. Meaning that our participants did better than a random guess by a factor of 3.8.

Some candidates indicate that the system does not do a good enough job in removing all identifying facial features, As seen in figure 5.10, the hair is not properly censored, making it possible to identify the man in the video as option 083. However, 20 out of 38 candidates, still answered this question incorrectly.

Problems with the survey:

- The lookalikes were picked by us, the lookalikes were mostly found using Google image search. We are under the impression that lookalikes we picked, strongly affects the results, even so we believe the results we got proved that our system has an effect. We believe that is the reason as to why there is a discrepancy in the results recorded between the different questions.
- In post survey talks with some of the participants, it seems like some of them were confused on what we were asking for. Regarding question 1, some of the candidates responded that they did not realise we asked for 2 answers, not just 1. After looking at the data we discovered this affected 8 candidates

in Question 1, we have recorded the statistics both with and without these outliers for transparency purposes.
- The survey was distributed online with no easy way for candidates to contact us, but prior to running our main experiment, we had ran 3 pilot studies. Based on the feedback from the pilot studies we are confident that the majority of the candidates had little to no problem answering this survey correctly.
- We made all the questions optional, meaning not all candidates answered the questions. The affected 3 candidates. We recorded the other answers from these candidates.
- We forgot to limit the amount of answers that the candidates can choose for each question, meaning that some picked "I have seen this before" and answered with one or more actors. This affected 2 candidates, we recorded all of these as "I have seen this before" and disregarded their correct answer. 3 candidates gave multiple answers to the Question 2 and 3, these were disregarded.
- To minimize the chance of random guessing, we should have included a confidence meter, where the candidates are asked to answer how confident they are with their answer.

Survey Conclusion: Our system shows promise. The results indicate that our system is able to censor videos where over half of our test group are unable to recognise the person in the video. We believe this shows that our system could be able to properly censor videos with great results after some tweaking. Our results indicate that the system has an effect on the videos, and based on these results we believe our finding is of statistical significance. **We reject the null hypothesis.**

Next iteration Due to Covid-19 and time constraints, we were not able to run another experiment with different parameters. If we were to run another experiment, we would like to test different filters and compare its effectiveness, and look into ways to potentially remove a larger area of the head and see its effectiveness in comparison to our initial experiment. An example of this can be seen in **??**

Chapter 6

Continuation of our results

Having succeeded in face detection and applying a blurring filter over the faces detected, we were encouraged to find ways to cover or change the face into another unrecognizable face. We were directed to look into StyleGAN[5], and face morphing for inspiration and learning how they work. Unfortunately we hit a few hurdles during development, making it so we were unable to create a working prototype. Even so, our research is documented below:

The StyleGAN project became a popular topic after Nvidia released their code on GitHub in 2019 and seemed to have increased researchers' interest in GANs as a result. The model can generate very realistic faces of different ethnicities that can be hard to distinguish from real ones.

Figure 6.1: Image taken from whichfaceisreal[13], the person on the left is created by StyleGAN[5]

StyleGAN has sparked several projects that feature StyleGAN or some variation of the model. Examples of this can be seen at Thispersondoesnotexist[14] where each time you reload the page a new made up person shows up, and at the website whichfaceisreal[13] where you tasked to identify the real face, when its presented next to a AI generated face. As seen in fig 6.1.
There's currently a weakness with the generated models we came across and that is that the background is either blank or heavily distorted. Wall patterns are wavy, clothes are not symmetrical, and body parts or other faces are sometimes disfigured.

In figure **6.2** we can see an example of a GAN trying to make a hat of some kind, but blends it into the background and tries to make some sort of wall pattern half way down. (picture taken from thispersondoesnotexist.com)

Figure 6.2: Generated image with headgear and background blending together

After seeing some good results by other projects we decided to see if we could create our own implementation. We spent some time figuring out how StyleGan worked and tried to create our own model, but we ran out of time and decided to try a pre-trained implementation instead. After figuring out how to generate faces with the pre-trained model, we then tried to morph generated faces with the detected faces found by our other system. We were instructed to look into one such morphing tool, but found out that it required common key points to be found between the faces. Unfortunately we were already running late as it was and it was decided that we had to drop this experiment. Potentially, if we found a way to efficiently make these key points, and morph the faces, we could merge the result into our detected faces. An artistic rendition of our planned system can be seen in figure **6.3**

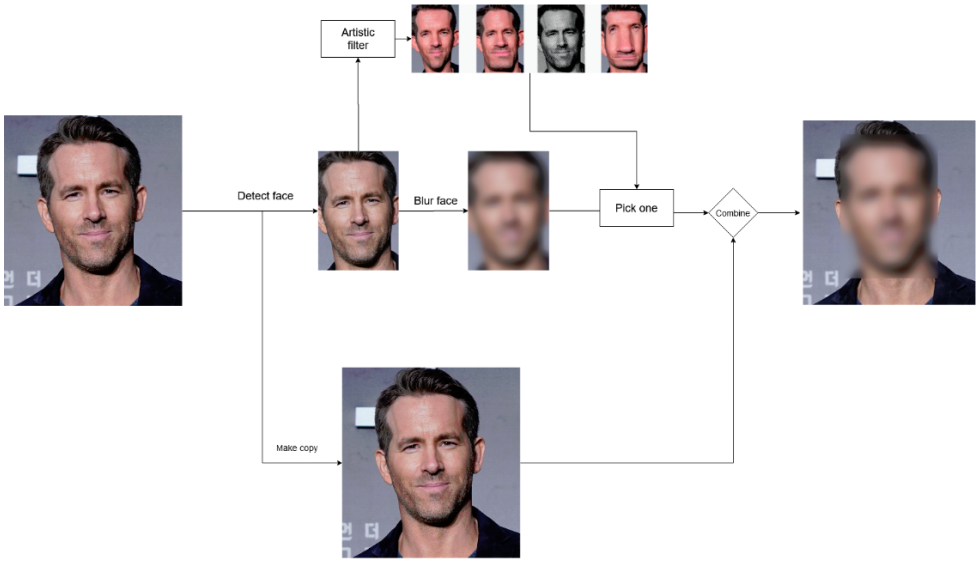

Figure 6.3: Artistic rendition of the planned system

Chapter 7

Future work and conclusion

7.1 Future work

Even if we were able to create a system capable of automatically censoring faces, there is still functionality that can be added, improved on or require more testing. We could only test our system once, since due to Covid it was very difficult to find more experiment groups to run tests on. Some parts of our system that can be improved on or expanded are mentioned here:

Optimization Our system is capable of automatic video censoring, but it is not particularly fast or optimized. Like previously stated, optimization was not prioritized during development.

Overlap and accuracy: The pre-trained weights we are using are good enough for our proof of concept system, but they are not accurate enough to always produce a reliable result. We discovered problems with large crowds when using this weight set, and we discovered that it will sometimes find a face within a face. We have fixed most of this overlapping problem, but not always. We encourage looking into better weight sets or training your own for this purpose. The overlapping faces problem is not a problem when it comes to censoring, but it could become a problem if the system is used for a more artistic or general facial detection purpose.

Improved censor filters: During our experiments and development process, we used the default Gaussian blur filter provided by Matlab. This is a process-heavy filter, taking extremely long time to blur pictures, especially pictures of larger size. We did not have the time, or access to enough testing subjects to run multiple experiments with different blur functions. We do not know what the best filters are for censor purposes, nor the filters that are better for optimization purposes.

Figure 7.1: Pixelation filter applied to an example image. Facepixler was used on this image.

Streamline the censor process: Currently our code is not streamlined to censor videos. It contains a large amount of debugging and dead code. The code should be refactored to streamline the creation of censored videos and pictures.

7.1.1 Expansion of the current system

Optimize area to include hair and face edges: As previously mentioned, the system does not censor hair or face edges. To combat this we could train the network on heads instead of faces, but a more effective method could be to scale the censored area to capture more of the face. More testing is required to know how much increase is needed before optimal results are found.

Figure 7.2: In this example the censor box was increased in size with 25%, effectively covering the entire face of the subject. No other changes to the system were done to this image.

Real time: The system only works on pre-recorded videos and pictures. A future instalment could include the possibility for live censoring of video feeds by using a web camera. Connecting this system to a CCTV system, or a web camera, we could automatically censor the recording, making the recording censored with no uncensored copies being made.

Run multiple experiments with different filters: Due to Covid-19 and time restraints, we were not able to run more than 1 experiment on our system. That experiment showed us potential changes we could do to our system to improve censorship. Getting hold of test groups in the current climate is difficult and is something we leave as a task for another time.

Non-square censoring: Our system is only capable of censoring square images. This is due to the square-shaped tensors we get from our predictor. This square is passed to a cropping function that also only takes in squares. It could be interesting to do pixel segmentation based censoring, but this was not something we dedicated resources to implement. This would entail the usage of masks etc, something that neither of us had experience with. For our censor purpose, there could also be shapes, like circles or ovals, that could be better for facial censoring.

7.2 Conclusion

The objective of this project was to find ways to anonymize people and other identifying features. This project was mainly focused on anonymization from a privacy and security perspective. In this paper, we mainly focused on the detection and censorship of human faces, and facial features. Due to us wanting to focus on one specific field at first we did not include other identifying features, e.g. full body or car number plates.

In this paper, we successfully demonstrate that it is possible to design a system, using a general object detection architecture, which is capable of detecting faces in videos, and perform custom censoring of the faces detected. Even so, after running tests on our system, our results are not satisfactory enough. We propose several changes to our system that can potentially improve it in the future. Due to external factors we were not able to incorporate these before project end. We also propose several other use-cases for this face detection system, some of which are more of an artistic nature than of privacy related concerns.

We have in this paper proved that object detection can be used for automatic censoring of faces and other facial features in videos, with relation to privacy.

Bibliography

[1] A. Rosebrock, *Opencv face recognition*, https://www.pyimagesearch.com/2018/09/24/opencv-face-recognition, 2018.

[2] H. Hukkelås, R. Mester and F. Lindseth, *Deepprivacy: A generative adversarial network for face anonymization*, 2019. arXiv: 1909.04538 [cs.CV]. [Online]. Available: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2632578.

[3] facepixler, *Facepixler home page*, https://www.facepixelizer.com/.

[4] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, 'Generative adversarial nets,' in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence and K. Q. Weinberger, Eds., vol. 27, Curran Associates, Inc., 2014, pp. 2672–2680. [Online]. Available: https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

[5] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen and T. Aila, 'Analyzing and improving the image quality of StyleGAN,' in *Proc. CVPR*, 2020. [Online]. Available: https://github.com/NVlabs/stylegan2.

[6] T. Karras, S. Laine and T. Aila, *A style-based generator architecture for generative adversarial networks*, 2019. arXiv: 1812.04948 [cs.NE].

[7] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen and T. Aila, *Analyzing and improving the image quality of stylegan*, 2020. arXiv: 1912.04958 [cs.CV].

[8] Filmora, *Filmora documentation page*, https://filmora.wondershare.com/video-editing-tips/blur-face.html.

[9] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo and R. Girshick, *Detectron2*, https://github.com/facebookresearch/detectron2, 2019.

[10] Y. Zhu, H. Cai, S. Zhang, C. Wang and Y. Xiong, *Tinaface: Strong but simple baseline for face detection*, 2020. arXiv: 2011.13183 [cs.CV].

[11] L. Weng, *Object detection for dummies part 3: R-cnn family*, https://lilianweng.github.io/lil-log/2017/12/31/object-recognition-for-dummies-part-3.html.

[12] V. Valkov, *Face detection on custom dataset with detectron2 and pytorch using python*, `https://towardsdatascience.com/face-detection-on-custom-dataset-with-detectron2-and-pytorch-using-python-23c17e99e162`, 2019.

[13] *Which face is real main webpage*, `https://www.whichfaceisreal.com/`.

[14] *This person does not exist main webpage*, `https://thispersondoesnotexist.com/`.

Appendix A

Additional Material

Timestamp	Score	what persons are seen in this video?	who is this	who is this?	General feedback:
11/19/2020 19:33:47	2 / 2	044, 005	055	083	
11/19/2020 19:42:41	0 / 2	099	089	083	
11/19/2020 19:46:42	0 / 2	099, 056	098	091	
11/19/2020 20:11:41	0 / 2	099, 005	040	083	
11/19/2020 20:15:41	0 / 2	005, 074	098	091	
11/19/2020 20:19:22	0 / 2			033	
11/19/2020 20:39:57	0 / 2	099, 062	I have seen this clip before	024, 083	
11/19/2020 21:02:40	0 / 2	099	098	083	
11/19/2020 21:02:48	2 / 2	044, 005	098	091	
11/19/2020 21:04:42	0 / 2	099	I have seen this clip before	091	
11/19/2020 21:08:29	0 / 2	009, 044	098	033	
11/19/2020 21:35:30	0 / 2	009, 062	098	024	
11/19/2020 21:44:44	0 / 2	099, 044	I have seen this clip before	083	Did it! Happy now?! Good censor, but the hair is too distinct on some off them
11/19/2020 21:47:55	0 / 2	062	098	091	
11/19/2020 21:54:16	0 / 2	079, 005	055	033	
11/19/2020 21:55:31	0 / 2	099, 005	089	083	
11/19/2020 21:59:55	0 / 2	I have seen this clip before		091, 007	
11/19/2020 22:08:42	0 / 2	099, 044	089	I have seen this clip before	
11/19/2020 22:21:08	0 / 2	062	098	083	
11/19/2020 22:39:05	0 / 2	099, 044	089	083	
11/19/2020 22:48:26	0 / 2	I have seen this clip before	098	I have seen this clip before	This was an interesting quiz
11/19/2020 22:48:34	2 / 2	044, 005	089	I have seen this clip before	
11/19/2020 22:49:48	0 / 2	099, 044	040	033	
11/19/2020 22:52:00	0 / 2	044, 056	098	023	
11/19/2020 22:54:36	2 / 2	044, 005	I have seen this clip before	091	
11/19/2020 23:06:22	0 / 2	044	031	091	
11/19/2020 23:06:23	0 / 2	I have seen this clip before	I have seen this clip before	023	Nice
11/19/2020 23:17:21	0 / 2	079, 005	I have seen this clip before	083	
11/19/2020 23:41:15	0 / 2	005, 056	027	033	
11/19/2020 23:46:37	0 / 2				Kjent ittj att nånn..
11/19/2020 23:52:37	0 / 2	099, 079	089	I have seen this clip before	
11/20/2020 0:15:26	0 / 2	099	040	I have seen this clip before	Too many celebs per video. I got confused
11/20/2020 1:48:43	0 / 2	099, 005	031	I have seen this clip before	First two were pretty good. instantly recognised that classic red tshirt and hand gesture of the third tho heh
11/20/2020 1:49:30	0 / 2	055, 044	027	033	
11/20/2020 6:48:52	0 / 2	055, 044	62	083	
11/20/2020 7:31:12	0 / 2	005	055	083	
11/20/2020 8:35:40	0 / 2	099, 044	098, 031	083	Var veldig vanskelig å se hvem de er.
11/20/2020 8:36:12	0 / 2	099, 005	027	083	
11/20/2020 9:20:44	2 / 2	044, 005	I have seen this clip before, 098	I have seen this clip before, 083	
11/20/2020 9:37:56	2 / 2	044, 005	055	091	Her ble det i grunn vill gjetting.
11/20/2020 10:28:26	0 / 2	079, 005	089	083	
11/20/2020 14:06:37	0 / 2	099, 044	I have seen this clip before, 098	I have seen this clip before, 083	I have honestly not seen the actual clip in question #2, but I am assuming that the clip is from Parks and recreation. Good job, I like that you have provided multiple pictures of actors th
11/20/2020 14:27:15	0 / 2	I have seen this clip before	031	I have seen this clip before	
11/20/2020 23:21:08	0 / 2	099, 044	I have seen this clip before	083	
11/21/2020 17:03:48	0 / 2	044, 079	089	083	
11/22/2020 18:49:14	2 / 2	044, 005	I have seen this clip before	I have seen this clip before	
11/23/2020 8:51:47	0 / 2	099, 056	031	091	
11/23/2020 10:26:25	2 / 2	044, 005	040	083	
11/23/2020 10:33:56	0 / 2	099, 044	089	083	
11/23/2020 10:40:41	2 / 2	044, 005	055	091	
11/25/2020 0:33:20	0 / 2	I have seen this clip before	I have seen this clip before	I have seen this clip before	

	Color coding:			
	Counted as right			
	Counted as outlier	8 answered with only 1 actor out of 2	1 answered with two options	2 answered with two options
	Counted as right and wrong	24 answered partially right		
	Not counted	2 did not answer	3 did not answer	1 did not answer
			2 answered right and admitted to seeing the clip before	
	White is counted as wrong			