Monitoring COVID-19 social distancing and mask wearing with person detection and tracking

Jiachen Jiang, Zhexin Wu, Hu Sun

February 19, 2024

Executive Overview

In this work, we construct a computer vision algorithm pipeline taking a video input to monitor social distance and face covering. We successfully combined geometric, feature-based and deep learning methods to achieve satisfactory performance on real-time data.

Keywords — Homography, CNN, Classification, Instance Segmentation

1 Background and Impact

The coronavirus COVID-19 is affecting 218 countries and territories around the world, causing about 61.6 million confirmed cases along with 1.44 million deaths globally until December 1, 2020. Since effective medicines and vaccines are still under development, precautionary steps, like keeping proper social distance and wearing masks, is probably the best way to prevent the spread of the pandemic[1]. We can monitor the public on these steps using computer vision algorithms thanks to the popularity of CCTV cameras in public. In this research, we propose a series of such algorithms and integrate them into a pipeline to process video inputs.

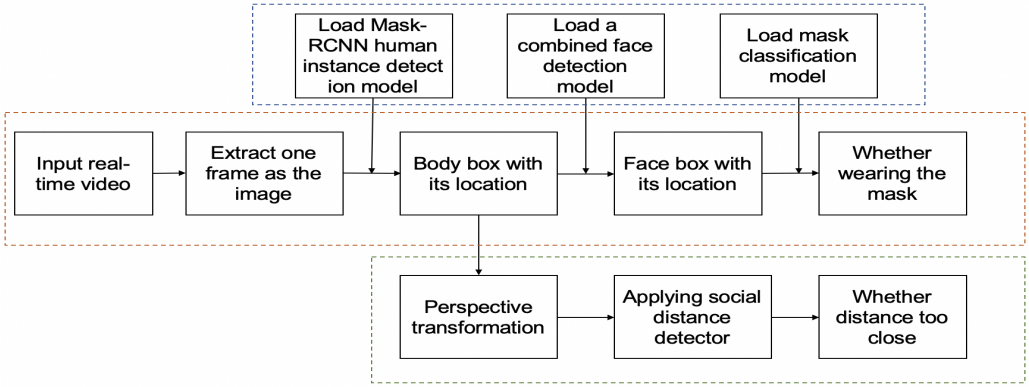


Figure 1: Overall pipeline

2 Method

Human and Face Detection

We launch our pipeline by a Mask-RCNN human instance detector[1], aiming at object instance segmentation. Training involves two classes – people the foreground and the background, resulting in robust bounding box and binary mask prediction. Then we detect the face for each human instance within corresponding bounding box. This turns out to be a trickier task for existing models [2][3], mainly because of occlusion caused by mask covering. Hence, we trained an auxiliary CNN regressor to backup for MTCNN's[3] performance. The regressor, inspired by bounding box regression[4][5][6] in loss function design and fully convolutional network[7] in structure, predicts the chin's vertical coordinate offset w.r.t the image center and proposes a bounding box in conjunction with the binary mask obtained in the previous step.

Mask Detection

With the detected bounding box for person and their face, our ultimate goal is to detect whether the person is wearing a face covering or not. For this purpose, we train a binary classification model with three major components: an image up-sampler, a feature extractor and a classifier. The image up-sampler follows the paradigm of the generator in DCGAN

[1](social distance): `https://www.cdc.gov/coronavirus/2019-ncov/prevent-getting-sick/social-distancing.html`, (mask): `https://www.nature.com/articles/d41586-020-02801-8`

[8], which contains a series of 2d transposed convolutional layers. The up-sampled images are propagated into the feature extractor which is set to be a pre-trained MobileNet_V2 [9] with the last layer removed. Finally, the output feature vector of length 1280 is passed into two fully-connected layers and output the predicted probability of wearing a mask. We train two models with the same architecture, one for low-resolution images and the other for high-resolution images.

Social distance highlighting

This part is about highlighting a 6-feet social distance contour for each individual in a video/image. By taking advantage of the arrow on the ground or other annotation in the scene, we should be able to construct a mapping between pixel distances and real distances via estimating a perspective transformation between the image and constructed standard images. Since we have only one 2D camera and it's hard to estimate the physical distance based on the monocular vision, we make two assumptions to help us measure the distance. Firstly, we assume that the camera's location and pose is invariant and the background scene would not change. Secondly, we make some annotations, like the arrow or squares, in the background scene and already know its size and shape, which provides a connection between the image and the physical world.

3 Prototype

3.1 Human and Face Detection

Human Instance Segmentation

We follow a transfer learning strategy: to begin with, we use the pre-trained model on COCO. Since we essentially aim at pedestrian segmentation, we configure the number of classes to 2, i.e. foreground (pedestrians) and background. The resnet50 backend is kept and we changed a new prediction head. The training process follows up naturally, since a well-suited dataset–*PennFudan Dataset for Pedestrian and Segmentation* is available, designed for the very same task. We trained our model on this dataset for 20 epochs with SGD optimizer on Google Colab's GPU, and already it is able to supply highly satisfactory human instance segmentation results.

Face Detection: A Combined Solution

We propose a model based on MTCNN. Despite of its flexibility in view-point variation and high accuracy [2] on uncovered faces, MTCNN fails constantly by making a wrong choice when there's something else having more "faceness" (bag, hat, fist, & etc) than a face with mask. Thus, we propose an algorithm to supply a reasonable backup when MTCNN fails, and to accept MTCNN's face proposal when it's with high probability and also at a proper location (decided by IoU score w.r.t. the backup).

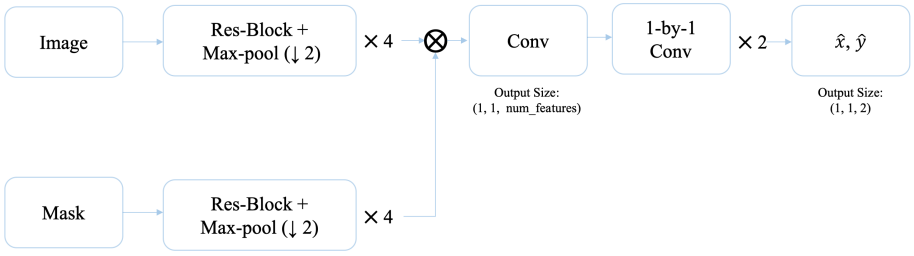


Figure 2: Structure of CNN regressor: ResNet [10] is the backbone of convolutional layers. We collected a subset of LFW (for unmasked faces) and masked face images from the Internet. The whole dataset consists of about 1000 images with a split of about 0.5[3]. Then we segment out individual human instances and manually labeled the location of chins. We trained the network on about 3000 human instances for 50 epochs, with Adam optimizer and on Google Colab's GPU. The final average y axis error is less than 5 and 6 pixels on training and validation set respectively. Note that we assign 0.9 weight to the y coordinate error and 0.1 to x since we only care about the former. However, x error can guide to a more rapid convergence compared to a network predicting only the y coordinate.

3.2 Mask Detection

Model training is based on a combination of two datasets[4]. Additionally, we include more data from several videos we capture in real life where we slice people's faces with our face detection algorithm and append label manually. All data are augmented with random rotation and down-sampling, leading to a dataset with 18145 images with masked person and 16575 images with unmasked person. All images are categorized as low-res or high-res images, depending on whether

[2] in sense of likelihood of a face and how closely it's cropped

[3] source: `https://drive.google.com/file/d/1Rnt3YWavImTdzazRy6mi1lzf_PJjtm1C/view?usp=sharing`

[4] source: `https://github.com/prajnasb/observations/tree/master/experiements/data;source:\url{https://github.com/X-zhangyang/Real-World-Masked-Face-Dataset}` The first online dataset is automatically generated with a "mask generator" program which is able to wear a mask for people in images who originally do not have a mask on, the size of the dataset is about 1400. The second is the Real-World Masked Face Dataset (RMFD) containing $90,000+$ images of people with/without masks.

the height or width is smaller than 40 pixels. Then all images are resized to have size 56×56 or 112×112, depending on the original image sizes.

The low-res and high-res images are then used for training the low-res and high-res mask detection model. Both models are trained on Google Colab's GPU with 15 epochs and an adam optimizer with decaying learning rate. Both models achieve validation accuracy $> 90\%$.

3.3 Social distance highlighting

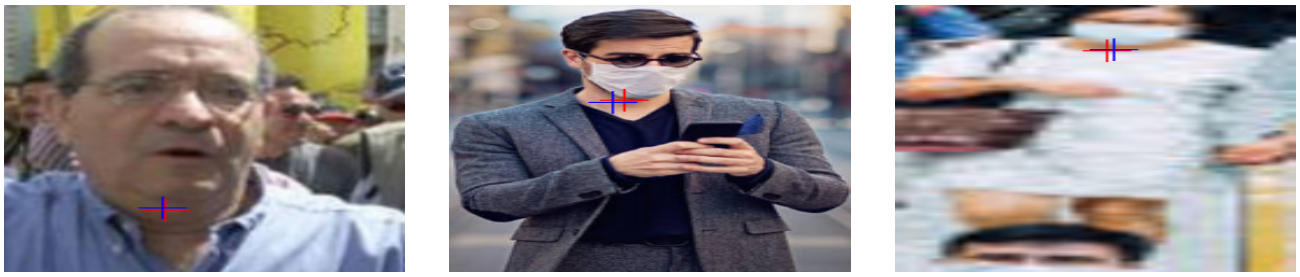
We construct a standard image of bird-view with the annotation together with a 6-feet circle on it. Then we use SIFT corner detector to find the matching points between the standard image and the scene image. Based on these corresponding point pairs, we calculate the perspective transformation matrix from the standard image to the scene image $\mathbf{H}$ and from the scene image to the standard image $\mathbf{H}^{'}$ using the Linear Least Square Method.

We project those points on the 6-feet circle to the real scene using the homography matrix $\mathbf{H}$ and get an eclipse, which indicates the safe distance of the individual. When drawing the eclipse around the person, we need to consider mainly three problems: 1) since different people have different distances from the camera, the eclipse needs to be scaled based on the person's height. 2) to make the eclipse display a strong three-dimensional effect, we remove those points on the eclipse covered by the human body based on the human shape mask in previous section. 3) when two individuals are close to each other within 6-feet distance, the eclipse's color would change from green to red, warning unsafe distance. To get physical distance among people, we project all people's location to the standard image using homography matrix $\mathbf{H}^{'}$. And in the new plane, the distance between people in terms of pixels is in direct proportional relation of physical distance.

4 Results

Human and Face Detection

The robustness of human instance segmentation and flexibility of the adaptive face detector is demonstrated in downstream tasks. We show the flexibility of CNN regressor in Figure 3.



(a) (b) (c)

Figure 3: Validation images of high, medium and low resolution. Ground truth is in red and prediction is in blue. Note that y coordinate is considerably accurate.

Mask Detection

We test our model on a video captured in real-life, where any face slices with either height or width smaller than 40 pixels are considered as low resolution, and high resolution otherwise. In Figure 4 we showed 2 sampled frames, one with a desirable result (left) and the other with an undesired result (right). Green (red) colored boxes mean the probability of wearing (not wearing) a mask is greater (smaller) than 50%. The current model is far from being perfect, though achieved decent performances on the train and validation set (accuracy $> 90\%$). It turns out that it is extremely difficult to predict low resolution images. The model also experiences difficulty in figuring out the images of peoples' profiles, possibly due to the lack of profile training data. We provide a complete video showing the performance of our model on our Github page[5], together with other results such as using only the high resolution model for prediction.

Social distance highlighting

Based on the specific scene, we notice that there is some grid on the ground and we know its real height and width of each box. So we construct a image of the standard grid as shown in Figure 5(a). Its corresponding part in the real scene are indicated by the green line in Figure 5(b)

In summary

The final testing result containing all three part is shown as Figure 6[6]:

Acknowledgements

We'd like to thank Qinyi Li of Sichuan University, China, who provided insightful suggestions on video making and technical help on video filming.

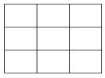
[5]Downloadable at: `https://drive.google.com/file/d/1iW7f8yECsVSmMMXhyE-auSbDuK8YDO2v/view?usp=sharing`

[6]All codes are available at: `https://github.com/husun0822/Social_Distance_Mask_Detection`

(a) a sample frame good result

(b) a sample frame with bad result

Figure 4: Two sampled result frames for testing our model. Green boxes indicate the probability of wearing a mask > 0.5, the read ones mean the opposite.



(a) the standard annotation

(b) the annotation in the scene

Figure 5: The standard annotation and related annotation in the scene. Finding homography matrix by matching points of the two images.

Figure 6: The final result of the project in the specific scene.

References

[1] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017.

[2] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I, 2001.

[3] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multi-task cascaded convolutional networks. *CoRR*, abs/1604.02878, 2016.

[4] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013.

[5] Ross B. Girshick. Fast R-CNN. *CoRR*, abs/1504.08083, 2015.

[6] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[7] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *CoRR*, abs/1605.06211, 2016.

[8] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[9] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.