DeepFake Video Classifier Project Report

Keywords: Deepfake, Deep Learning, Computer Vision, Classification Model

Yunting Chiu

2021-11-16

1 Abstract

Videos may fool people nowadays and they are causing trouble. This is due to a technology called DeepFake. Deepfake is a technique that makes computer-created artificial videos in which images are combined to create new footage. Recently, Deefake technique has been widely discussed in Taiwan since a famous Taiwanese YouTuber was discovered to be responsible for producing, selling, and circulating Deepfake videos of women, mostly public figures. Based on this, techniques for solving this kind of problem have been in high demand because more and more relevant issues about misuse of Deepfake technique will be extensively expanded in the future. I have relevant experience in building a machine learning model to determine whether the input video is fake or real. Moreover, I am working on a computer vision project that analyzes deepfake videos as part of my capstone project. As a result, I feel the need to apply my pertinent academic background to this project so that this model can be tested and even improved. **The main goal of the project is to assist people in determining whether a given video was generated using the Deepfake technique or a real video by using the DeepFake Video Classifier Model.**

2 Dataset

FaceForensics++[1] is a popular and widely used database for detecting image or video forgeries. FaceForensics++ is a forensics dataset consisting of 1000 original YouTube videos that have been manipulated with four automated face manipulation methods: Deepfakes, Face2Face, FaceSwap and NeuralTextures. In this project, I experiment on `Deepfakes` subset for fake videos, and I also download original videos from the same source that the authors use to create fake videos for real videos. Both can be found from FaceForensics GitHub. The videos have a wide range of facial expressions because they are about TV reporters and journalists of various sexes, ages, and races.

In short, the Deepfake classification model is generated by 2000 videos, which includes 1000 real videos and 1000 fake videos utilizing Deepfake technology.

3 Methodology

3.1 Download the Dataset

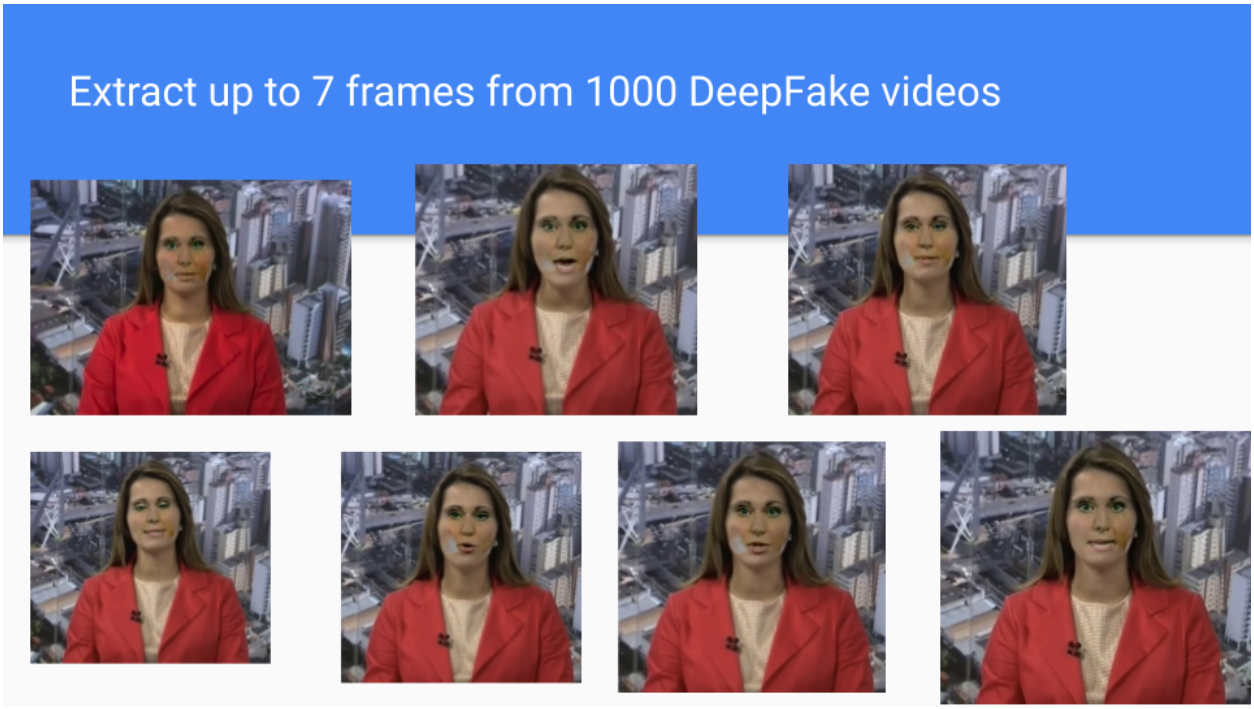
I use the Python script provided by the authors to download the whole data set from FaceForensics++. The entire data is larger than 2 terabytes. Please see `00.FaceForensics_download_script.ipynb` for the source code.

3.2 Extract Video Frames and Save to Images

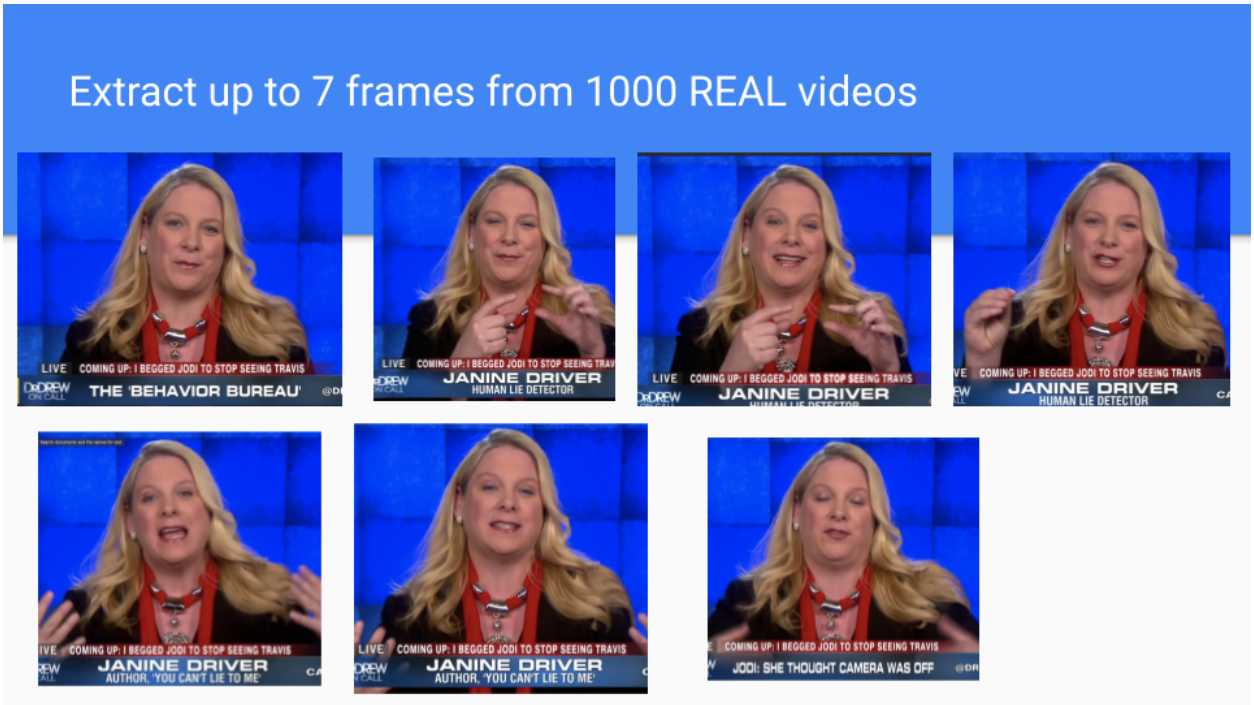
According to the relevant academic work, Abdali et al.[2] did the similar Deepfake classification model using FaceForensics++. Since all videos have constant frame rate 30 fps, they extract up to 7 frames from

each video and save them as images, so I use the same method based on the author's idea. To be more specific, I extract 1 frame in every 30 frames totally extract 7 images in one video. In this case, I can capture different facial expressions in a single video. Now I have 1000 Deepfake videos and 1000 original videos. After executing the code, I have 14000 images (observations), which includes 7000 "fake" images and 7000 "real" images. I mainly use `cv2` and `imageio` to capture features form the given videos. Please see `01.capture_videos_to_images.ipynb` for the source code.

3.2.1 Seven Different Facial Expressions from a Fake Video



3.2.2 Seven Different Facial Expressions from a Real Video



3.3 Facial Extraction using MTCNN

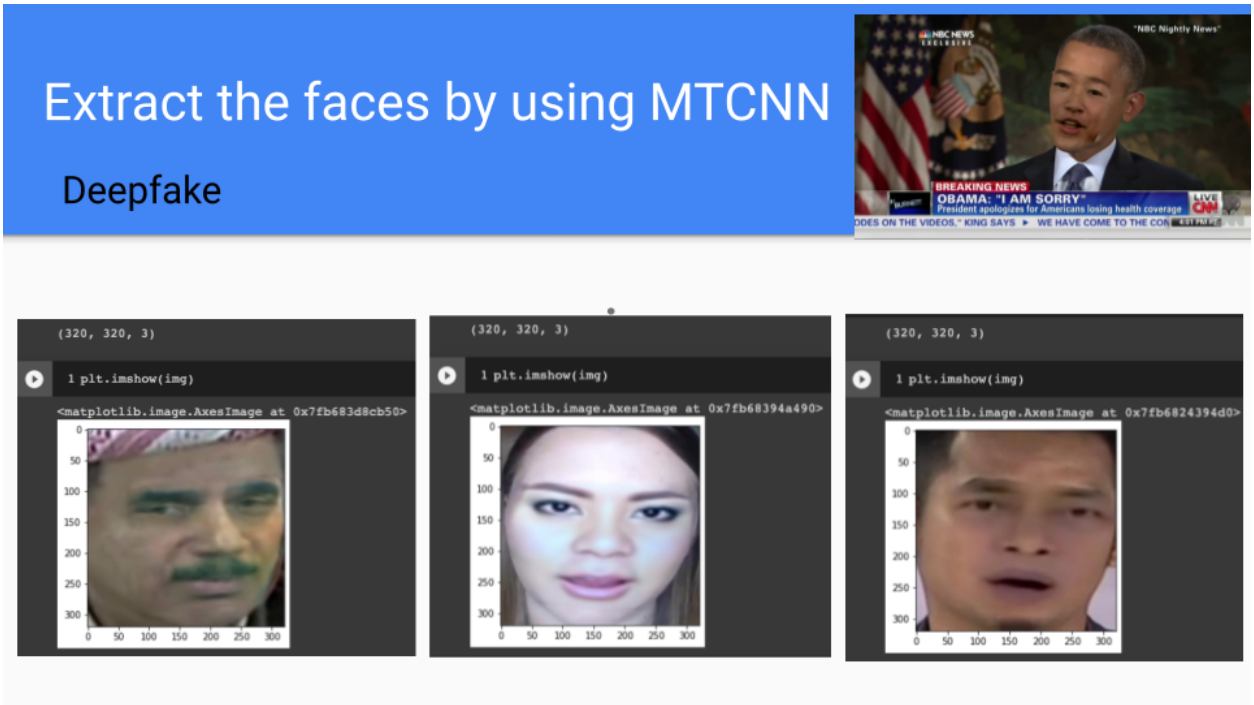
MTCNN (Multi-Task Cascaded Convolutional Neural Networks)[3] is a type of neural network that recognizes faces and facial landmarks in images. It was published by Zhang et al. in 2016. MTCNN is a good face detector that provides exactly pixel positions of precise nose, mouth, left eye, right eye, and face boundary. Thanks for the prior research, I can now simply install a `MTCNN` package to capture the facial features.

If we read one image, the MTCNN model returns three index: `box`, `confidence` and `keypoints`.

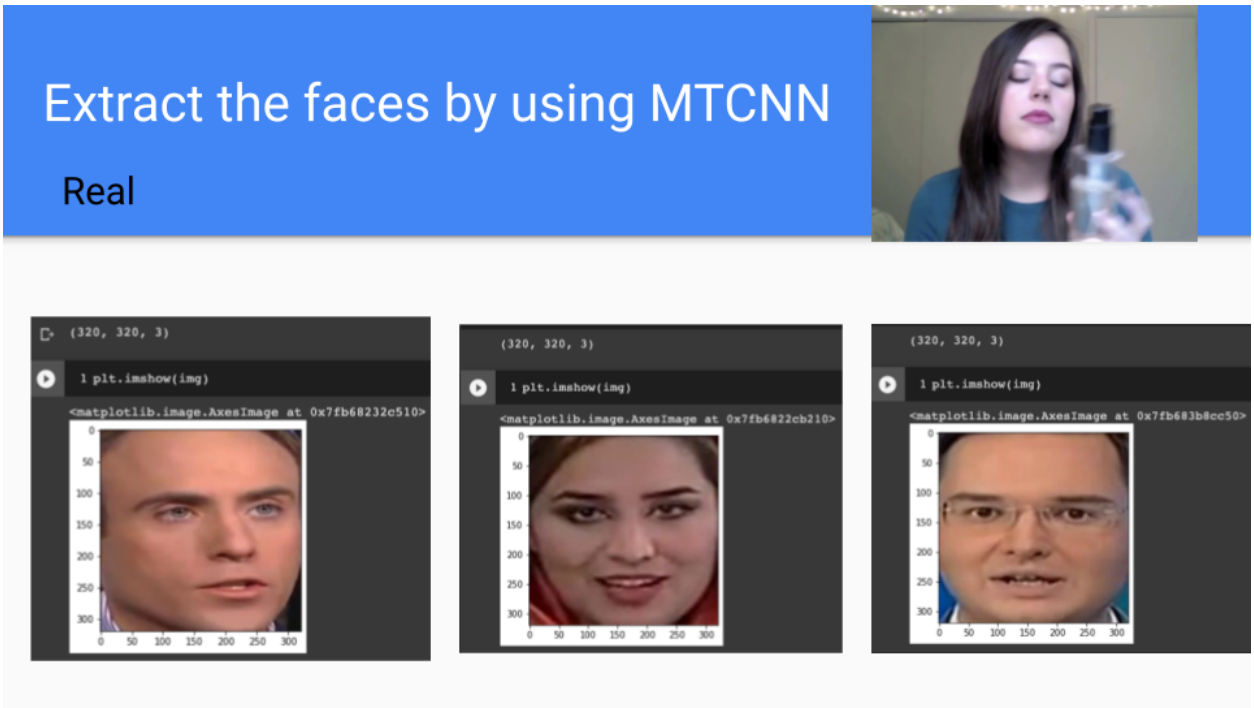
- The bounding box is formatted as `[x, y, width, height]` under the key `box`.
- The `confidence` is the probability for a bounding box to be matching a face.
- The `keypoints` are formatted into a JSON object with the keys `left_eye`, `right_eye`, `nose`, `mouth_left`, `mouth_right`. Each `keypoint` is identified by a pixel position (x, y).

However, how do we deal with images in which the face shape is unclear? Based on this, I add one more condition to set `if confidence > 0.9` in the facial extraction function, which means that if MTCNN does not have 90 % confidence to identify an input image that includes a face, I will remove it because it is a "outlier". Finally, I capture 6984 out of 7000 facial images in the fake set, and capture 7000 out of 7000 facial images in the real set. The capture rate of Deepfake images is 99.77%, and the capture rate of original images is 100%. I also unify the image size to 320 x 320 x 3 using `PIL` package. In conclusion, now I have 13984 observations, which includes 6984 fake images and 7000 real images.

3.3.1 Example of Fake Images



3.3.2 Example of Real Images



4 References

1. Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., & Nießner, M. (2019). Faceforensics++: Learning to detect manipulated facial images. In *Proceedings of the IEEE/CVF International Conference*

on Computer Vision (pp. 1-11).

2. Abdali, S., Vasilescu, M. A. O., & Papalexakis, E. E. (2021). Deepfake Representation with Multilinear Regression. *arXiv preprint arXiv:2108.06702*.

3. Zhang, K., Zhang, Z., Li, Z., & Qiao, Y. (2016). Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters, 23*(10), 1499-1503.