Progress Project Report

Overview

By definition, a deepfake is an artificial video or image where a person's face in said video or image is replaced with another person's face or is manipulated in some way. As of now, deepfakes are incredibly realistic and it is difficult for an average person to distinguish between a deepfaked video and an authetnic one. Therefore, for my final project, I would like to investigate the topic of deepfake detection using machine learning. Since deepfakes are quite realistic and difficult to perceive by humans, machine learning has been a viable way to detect deepfaked images or videos.

There are four types of deep fakes I am currently investigating: face swap (two versions), expression modification, and face synthesis. There are two versions of face swap deepfakes. The first version of face swap deepfakes is a simple approach that extracts the face region of one image and places it into another. The second version of face swap deepfakes is a learning based face swap approach that uses two autoencoders with a shared encoder. The latter is more complex and produces more realistic images or videos than the former. Expression modification involves modifying the expressions of a person's face (such as whether the person is smiling or not, eye color, etc.). This is usually done by transferring the expressions of a source actor to a target actor while preserving the target actor's identity. Unlike face swaps, the original person is still there but has a different expression. Face synthesis involves creating completely artificial faces using provided data. Examples of such images can be found here: https://this-person-does-not-exist.com/en. Although the faces on this website look realistic, they are not real. Faceforensics++, the dataset I am using, provides all verisons of the deepfakes except face synthesis. I have created my own face synthesis dataset by getting a large collection of images from https://this-person-does-not-exist.com/en. Given these categories of deepfakes, I am investigating how strong or weak different models of deepfake detection models/classifiers are for the three categories.I am currently using *MesoNet* and *Deepfake scanner* as classifiers as these two are very reliable. I have also downloaded the original (unmodified) dataset from faceforensics++ and I will be applying that dataset to the two classifiers I mentioned above to test how well the classifiers can detect unmodified images or videos. All the four datasets I got from faceforensics++ (the original dataset and the three deepfake datasets) came in video format. That is, each of the four datasets had a series of .mp4 files. However, due to the differences between *MesoNet* and *Deepfake scanner* as well as the differences between the face synthesis dataset and the other 4 datasets (this is because the face synthesis dataset didn't come from faceforensics++), I have to modify each of the datasets so that the most accurate comparison can be made between *MesoNet* and *Deepfake scanner*. I am currently in the process of modifying the original datasets. I am using a couple of python scripts as well as FFMPEG to do so. Although the process of modifying the datasets is a little difficult, I am making good progress on this and by the time I finish modifying the datasets properly, I will provide each classifier with the data and gather results.

Additionally, I plan to run an experiment to see which facial feature in a deepfaked image makes it easiest for a deepfake classifier to detect that it's a deepfake. For example, I will want to see if a particular facial expression or hair length makes it easier for a deepfake classifier to see that the image is a deepfake. In order to run this experiment, I will have to generate my own

deepfakes with a particular facial feature. I will use styleGAN to generate the images because this generative adversarial network allows you to generate deepfakes with custom facial features. I am currently learning how to use StyleGAN, but I believe that I'm making good progress because I have been able to generate images of people with glasses and images of people with no glasses.

Value to User Community:

This project is relevant to the course because it is about a recent and ongoing issue in software engineering. The papers I have read so far were published in 2019 or after. I chose this topic because I find the fact that machine learning models can generate artificial yet realistic images very interesting. By doing this project, I get to expand on my interest by investigating how machine learning can distinguish between what is real and what is fake. Additionally, I believe that my work can help researchers because my data will show them how they can improve on existing models. Since my project investigates the strengths and weaknesses of current deepfake detectors, researchers may get a better understanding of how to improve existing models.

Research Questions

1: Which type of deepfake are deepfake detectors best at detecting?

Using my results from the first part of my experiment, I can answer this question.

2: In general, what are the strengths and weaknesses of video-based deepfake detectors and image-based deepfake detectors?

Since *MesoNet* is an image-based classifier and *Deepfake Scanner* is a video-based classifier, I can compare the two classifiers to answer this research question.

3: Given face synthesis deepfakes, is there a particular facial feature that makes it relatively easy for deepfake detectors to detect that images with said facial features are deepfakes?

I can answer this research question from the results of the second part of my experiment.

Demo

For my demo, I plan to do a short presentation because it will be hard to show any meaningful running code during my presentation. I could potentially demonstrate styleGAN, but I am not certain whether or not this is feasible yet because I haven't experimented with styleGAN enough.

Tell us how you will deliver the code, documentation and other software artifacts for your project

The datasets I downloaded from faceforensics++ are very large (around 30GB on average), so I will upload them to Zenodo. If there are any other datasets that are too large to upload to Github, I will also upload these to Zenodo. I will upload everything else to Github.