Ben-Gurion University of the Negev

Faculty of Engineering Science

Department of Software and Information Systems Engineering

Project in Offensive Artificial Intelligence Course

OAI Final Project - Robustness of Real Time Deepfakes

Author:
Amit Kama

Author:
Oren Shvartzman

June 23, 2022

Contents

1 Introduction

Deepfake is a general term that encompasses the use of deep learning algorithms in order to create synthetic media, in which one subject in an existing visual and/or audio content is usually replaced with another's likeness. While fraudulent content has been around for some time, recent advances in machine vision have posed a major threat to the trust and transparency of the media. Using powerful machine-learning and deep-learning techniques, deepfakes can now manipulate or generate visual and audio content that can be more easily misleading.

In recent years, deepfakes have garnered widespread attention for their uses in spreading fake news, committing financial fraud, creating pornographic materials, and many other disturbing uses. This has led to a significant need to identify and restrict their use.

Ever since the introduction of deepfakes, researchers in deep learning have increasingly focused on this area of research. In particular, they propose methods, as well as practical implementations of deepfakes is in various fields. Among other methods, in [4], Siarohin et al. propose the first order motion model for image animation. Their framework enables generating a video sequence, in which an object in a source image is animated according to the motion of a driving video, without using any annotation or prior information about the specific object to animate. According to them, once trained on a set of videos depicting objects of the same category, the method can be applied to any object of it. Based on this method, a number of real time photorealistic avatars have recently been developed, one of which we will explore in this work.

However, real time avatars are far from perfect, as they are not robust when it comes to edge cases. This includes facial gestures in the driving video, objects in the source or target media that make it difficult to identify facial boundaries, and many more.

Note that these limitations can create visual glitches and distortions that can be detected with the naked eye as well as by software, and therefore can be utilized for failures detection. This observation drives a growing amount of research dealing with those inaccuracies for dual use – correcting them in order to improve the deepfakes' credibility, or exploiting them to distinguish deepfakes from real content.

In this work, we evaluate the robustness of real time deepfakes by im-

plementing first order motion model-based avatarify and examining various edge cases on it. After demonstrating failures in the implementation, we also provide a method to utilize them for failures detection.

2 Methods

In this section we will briefly introduce the First Order Motion Model, on which the avatarify implementation whose robustness we evaluated, consists of. In addition, we will describe our proposed method for failures detection.

2.1 First Order Motion Model

In [4], Siarohin et al. propose the First Order Motion Model for Image Animation, which addresses the task of generating a video sequence so that an object in a source image is animated according to the motion of a driving video.

Once their proposed method is trained on a set of videos depicting objects in the same category, it can be applied to any object in this category, without using annotations or prior knowledge about the object to be animate. This is done using a self-supervised formulation, for detaching appearance information and motion information from each other. The method also supports complex motions, using representation consisting of a set of learned keypoints along with their local affine transformations.

The model's mathematical formulation describes motion between two frames and is efficiently computed by deriving a first order Taylor expansion approximation. Thus, motion is represented as a set of keypoints displacements and local affine transformations. Finally, a generator network combines the appearance extracted from the source image with the motion representation of the driving video. First Order Motion Model outperforms state of the art on all the benchmarks on a variety of object categories.

In the background of this method, most of the works that appear in the literature tackle image animation task by assuming strong priors on the object representation and resorting to computer graphics techniques. Such methods can be described as object-specific methods, due to the knowledge that they assume on the model of the specific object to animate.

Recent works of deep generative models have derived various techniques for image animation, such as Generative Adversarial Networks (GANs) and

Variational Auto-Encoders (VAEs) for facial expressions transfer or motion patterns between human subjects in videos. Having said that, these methods usually also use pre-trained models to extract object-specific representations.

However, these models consist of costly ground truth data annotations, and are not available in general for an arbitrary object category. Siarohin et al. address these issues with more object-agnostic approach. As they use a set of self-learned keypoints with local affine transformations to model complex motions, they call the method the First Order Motion Model.

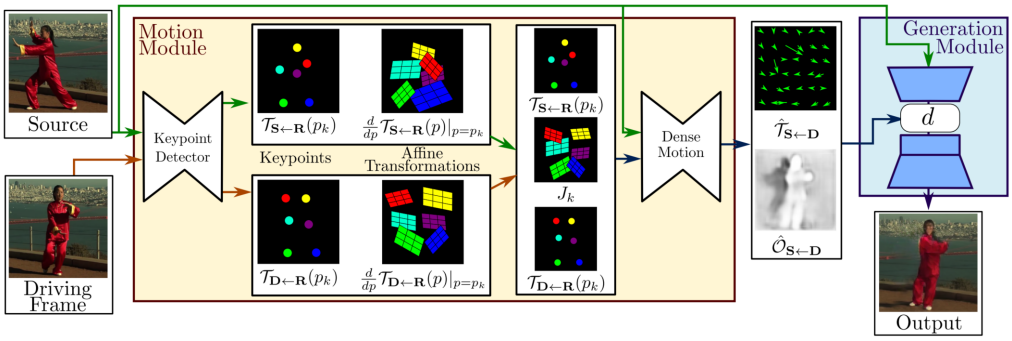

Figure 1: Overview of the First Order Motion Model approach. The model gets as inputs a source image S and a frame of a driving video frame D. Using unsupervised keypoint detector the motion model extracts first order motion representation consisting of keypoints displacements and local affine transformations with respect to a reference frame R. Using the motion representation a dense motion network generates dense optical flow $\hat{T}$ from D to S and occlusion map $\hat{O}$ from D to S. Lastly, using the source image and the dense motion network's outputs, a generator renders the target image. [4]

2.2 Failures Detection Method

At the basis of our method for failures detection is the idea according to which we can detect the avatarify failures using anomaly detection methods. To address the task of detecting avatarify failures in an output video, we simplified it to the problem of detecting avatarify failures in a target image. To do so, the method starts with dividing the video into several frames. For each frame we extract the facial landmarks, calculate Euclidean distance

between all the points. We then normalize and run the results through the local outlier factor model with 20 neighbors. Note that we extract the facial landmarks using Dlib [1, 2], a modern toolkit which contains machine learning algorithms and tools for creating complex software to solve real world problems. Among many major features that Dlib provides, there is a high quality face recognition capability. The test is done using the Yale Face Database [3], which contains 165 grayscale images in GIF format of 15 individuals, when for each subject, there are 11 images, one for each different facial expression or configuration: center-light, w/glasses, happy, left-light, w/no glasses, normal, right-light, sad, sleepy, surprised, and wink. This is in practice using novelty local outlier factor, since the dataset contains only ordinary faces.

The test includes making prediction on each frame of the video. If one or more frames are considered anomalies, we declare the video to be deepfake.

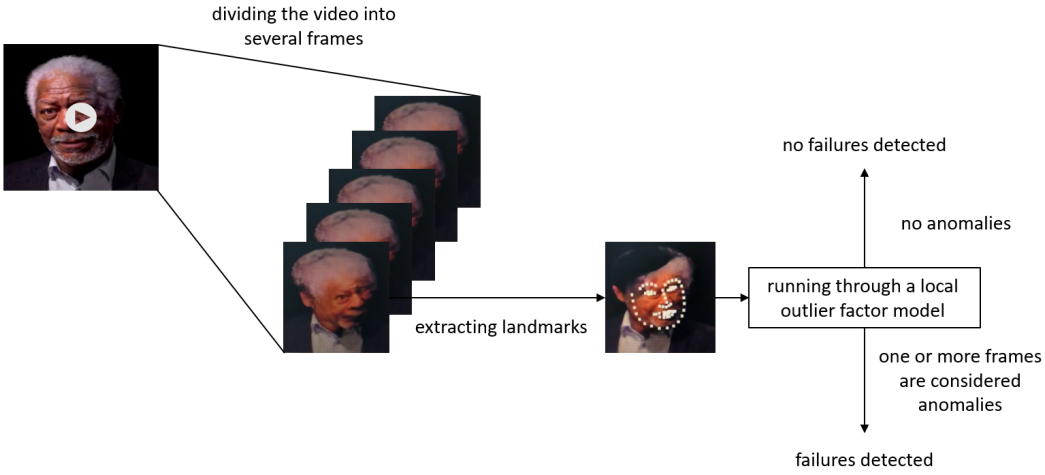

Figure 2: Overview of our proposed failures detection method.

3 Experiments and Results

In this section we introduce the experiments and results. A detailed explanation of the visual glitches and distortions that we managed to create and

detect is provided, as well as the results of our proposed method to utilize them for failures detection.

3.1 First Order Model Robustness

In order to evaluate the robustness of real time deepfakes, we started by implementing avatarify and then examining edge cases on it. We implemented the First Order Motion Model mentioned in 2.1.

Since we focused on the implementation of avatarify which work in a way that a face in a source image is animated according to the motion of a driving video, we tried to test its limitations by passing driving videos with various facial gestures and head tilt as input. This includes winks, tongue out, smiles, eyebrow raise, different head tilts, etc.

From the output videos, it was clear that the avatarify does not work on certain artifacts. For example, avatarify is not able to create videos with a tongue out at all, as well as creating a proper dental structure. However, even when it fails to copy certain facial gestures to a source image perfectly, the outputs still look relatively believable.

The more significant limitation we identified in the implementation, is the creation of a video that includes head tilts, or more precisely, that the extraction of the keypoints from the faces in the input videos is more complex.

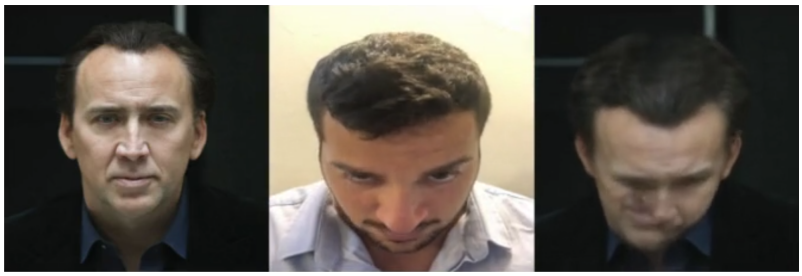

Figure 3: The output consists of a source image of Nicolas Cage and a driving video which includes head tilt. It can be seen that the avatarify failed to generate a reliable output.

The implementation, including the experiments and results can be found using the following link:
https://colab.research.google.com/drive/1oI4jPx9cB26OsCerUs2JfxyGkffBWM5v

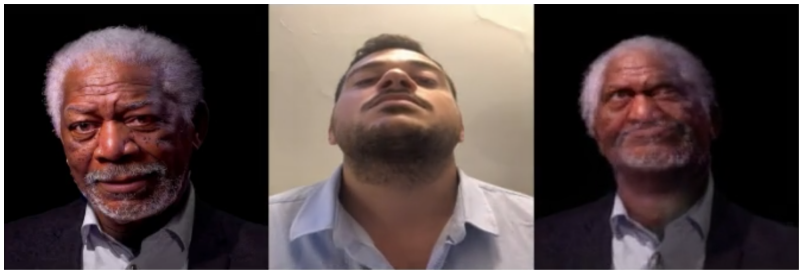

Figure 4: The output consists of a source image of Morgan Freeman and a driving video which includes head tilt. It can be seen that the avatarify failed to generate a reliable output.

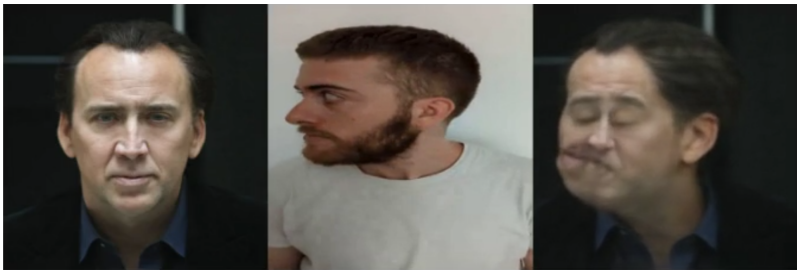

Figure 5: The output consists of a source image of Nicolas Cage and a driving video which includes head tilt. It can be seen that the avatarify failed to generate a reliable output.

3.2 Failures Detection

In order to test our proposed method, we fed it with two videos – authentic video and a deepfake of Morgan Freeman, derived from the authentic video as a driving video using the First Order Motion model. When we ran the method on each of the two videos, we found that indeed no failures were detected in the original video, while in the deepfake they were.

Note that in some cases the face in the target image is distorted enough that it is not possible to identify the face, and therefore to determine the landmarks. This fact can be exploited for further improvements in the failure detection task.

Figure 6: An example of the facial landmarks extraction. Using the local outlier factor model, the left image's landmarks considered anomalies, while in the right image no failures was found.

4 Discussion

As the impact of social media on our world grows moment by moment, along with the negligible cost and lack of need for experience enabling almost anyone to create high quality deepfakes, this emerging technology poses a serious threat to society.

For this reason, in academia and industry there is an extensive practice in this technology, and they offer many developments, both for improving the quality of deepfake applications and conversely, mechanisms for dealing with this phenomenon.

In this work we experimented with an implementation of real time deepfakes, the first order motion model-based avatarify, proposed by Siarohin et al. [4], and evaluated its robustness by examining various end cases on it.

We have learned that avatarify implementations are not robust, and that various facial gestures and artifacts can cause significant failures in the target videos. It seems that most failures caused by head tilt and rotations in the driving video, as well as gestures which include objects that the model does not recognize in the source image, such as tongue or teeth.

After demonstrating failures in the implementation, we also provided a method to utilize them for failures detection. We were able to successfully demonstrate our method on two videos – authentic video and a deepfake. No failures were detected in the original video, while in the deepfake they were.

References

[1] Dlib documentation. http://dlib.net/.

[2] Dlib github. https://github.com/davisking/dlib.

[3] Yale face database. http://vision.ucsd.edu/content/yale-face-database.

[4] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. First order motion model for image animation. *CoRR*, abs/2003.00196, 2020.