Face Manipulation Detection Using Deep Learning

Aditya Kaduskar
Electrical Engineering Department
Arizona State University
Tempe, Arizona
akaduska@asu.edu

Aradhita Sharma
Electrical Engineering Department
Arizona State University
Tempe, Arizona
ashar314@asu.edu

Swaroop Pandit
Electrical Engineering Department
Arizona State University
Tempe, Arizona
spandit4@asu.edu

***Abstract*—This project describes a method for detecting face tampering in videos that is both automatic and efficient, with an emphasis on two contemporary techniques for creating hyperrealistic images: 'Deepfake' and 'Face2Face'. In most cases, picture forensics techniques aren't well matched due to the compression, which severely lowers the quality of the video data. As a result, this project takes a data driven approach. We use two networks 'MesoNet' and 'XceptionNet'. We trained and tested the networks on the Faceforensics++ dataset: A collection containing Prisitine(Real), and Forged(Fake) videos manipulated by Deepfake and Face2Face techniques. Faces were detected, cropped and labelled from these videos, and the networks were trained on these faces. Upon testing, MesoNet achieved an accuracy of 97% and XceptionNet an accuracy of 98%. To implement our system, we generated confidences for faces detected in a video input, and described them as real or fake.**

I. Introduction

Due to advancements in Applied Computer Graphics and Artificial Intelligence, it is now possible to generate realistic videos of people doing and saying fictional things. This significantly impacts how people determine the legitimacy of information presented online. Furthermore, such content may be used maliciously as a source of manipulation, harassment, misinformation and persuasion.

Faces are of special interest as subject to current manipulation techniques since they play an important role in person-to person communication. Also, Facial Feature tracking and face reconstruction are well researched fields in computer vision. Techniques are used to alter the expression of the target. One such method which we discuss is Face2Face[5], a computer graphics based approach used to transfer the expressions of one person to another in real time. On the other hand, a face can be altered to change its identity altogether, as well. Such techniques are called face-swapping techniques. Deepfake[4] is one such method which adopts deep learning for face swapping. While Computer graphics based methods work in real time, a DeepFake generator needs to be trained with facial data of the source and target faces.

The field of digital image forensics focuses on detecting image forgeries in order to control the spread of such altered content. There have been a number of methods for detecting image forgeries [6, 7], the majority of which either assess discrepancies in comparison to a regular camera pipeline or rely on the extraction of specific image modifications in the resultant image. Image noise [8] has been found to be an effective indicator for detecting splicing (copy-past from an image to another). Picture compression artifact identification [9] also provides some useful information about image alteration.

While there have been advancements in Image manipulation detection, digital video falsification detection remains an uphill battle. Furthermore, most image-based algorithms cannot be directly applied to videos, owing to the significant deterioration of frames following video compression. Current video forensic research [10] focuses primarily on video re-encoding [11] and video recapture [12, 13], but video edition remains difficult to detect.

II. Literature Review

A. Similiar Face Manipulation Techniques

Thies et. al[1], the co-creator of Face2Face, demonstrated one of the first attempts at real-time facial expression re-enactment. Their approach captures the facial performances of source and target subjects using a commodity RGB-D sensor. The expression is transferred by computing difference between the source and target expressions in parametric space, and modifying the target parameters to match the source parameters.

Bregler et. al[2] have developed a system called Video Rewrite, which takes existing footage and automatically creates new video of a person mouthing phrases they didn't say in the original footage. The system tracks points on the speaker's mouth in training footage using computer vision algorithms. The phonemes in the training data and the new audio recording are automatically labelled by Video Rewrite. Video Rewrite rearranges the mouth pictures in the training movie to fit the new audio track's phoneme sequence. When specific phonemes are missing from the training footage, Video Rewrite chooses the closest approximations. The generated mouth image sequence is then merged into the background film. The stitching procedure compensates for changes in head position and orientation between the mouth images and the backdrop film automatically.

Dubbing is a time-consuming operation that necessitates precise translations and precisely timed recitations in order for the new audio to roughly match the mouth action in the video. The video-to-audio match is never perfect since the sequence of phonemes and visemes in the original and dubbing languages are different, which is a primary source of visual discomfort. V-Dub[3] is a system which changes an actor's lip motion in a video to match the new audio track. Audio

analysis in combination with a space-time retrieval method is used to synthesize a new photo-realistically rendered, highly detailed 3D shape model of the mouth region to replace the target performance, which is based on high-quality monocular capture of 3D facial performance, lighting, and albedo of the dubbing and target actor.

III. Theory

A. DeepFake

Deepfake is a technique that replaces a targeted person's face in a video with the face of someone else. It emerged in 2017, used to generate face-swapped adult content. Since then, a community friendly version called 'FakeApp'[4] has been created.

The central concept is the simultaneous training of two autoencoders. Their architecture can differ depending on the size of the output, the intended training duration, the expected quality, and the resources available. An auto-encoder is a combination of an encoder network and a decoder network. The encoder's job is to lower the size of the data in the input layer by encoding it into a smaller number of variables. The decoder's purpose is to output an approximation of the original input using those variables.

The generation of Deepfake images follows the same concept as shown in the figure 1. First, the aligned faces of two people A and B are gathered. Then an auto-encoder EA is trained to rebuild the faces of A from a dataset of facial images of A, and an auto-encoder EB is trained to reconstruct the faces of B from a dataset of facial photographs of B. The idea is to share the encoding weights of the two auto-encoders EA and EB while keeping their respective decoders distinct. Post-optimization, any picture with an A face can be encoded using this shared encoder and decoded using the decoder of EB.

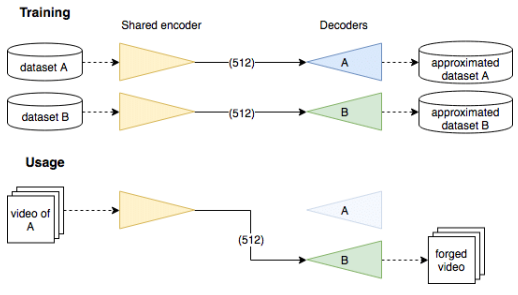

Fig. 1. Deepfake Principle

B. Face2Face

Face2Face by Thies et al.[5] is a computer graphics based method that performs photorealistic and markerless facial re-enactment in real-time. The method requires the first few minutes of a pre-recorded video of the target to generate a dense reconstruction of the face or 'blendshape'. Similarly, a blendshape of the input source is also generated. At run time, the facial expressions of both source and target video are tracked. Re-enactment is then accomplished by transferring deformation between the source and the target in a fast and effective manner. The mouth interior from the target sequence that best fits the re-targeted expression is extracted and warped to obtain an apt fit. The final image synthesis is rendered by overlaying the target shape by a morphed facial blendshape to match the facial expression of the source as shown in figure 2.

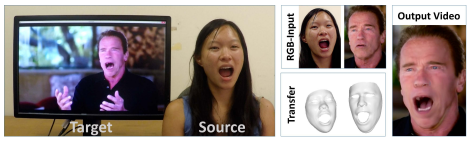

Fig. 2. Transfer of expressions from Source to Target using Face2face

IV. Proposed System

As generating deep fake techniques are getting advanced, there is a need to get better at identifying deep fakes. This project aims to develop a deep fake detection system using two approaches. Deepfake detection is a binary classification, in which classifier classifies whether a given video is real or fake. This project consists of use of two different neural network models for getting real or fake classification. For building this classification system, input of real and fake videos is given to two dense neural network models, which are MesoNet4 and Xception64 and classification results from both the models are compared.

This system is trained on 1000 real videos and 1000 fake videos which are built using DeepFake and Face2Face techniques. This dataset is obtained from FaceForensics++[14]. The dataset that is fed into the model as input is carefully chosen to be of equal number of inputs for real as well as fake videos, since the models are going to be trained to the particular dataset given as input to it.

A. Training Workflow

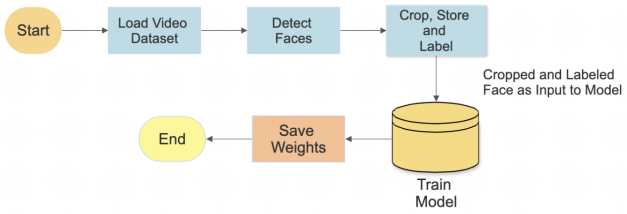

Fig. 3. Workflow of training of the model

Training workflow for achieving the deep learning solution to DeepFake detection is shown in the Figure 3. The program starts with loading the video dataset of both real and fake videos. Every video is divided into number of frames and face detection algorithm (Dlib face detection) is used to detect the faces in each frame. Dlib face detection algorithm

works on extracting HOG (Histogram of Oriented Gradients) features and Linear SVM (Support Vector Machines) detector for detecting faces. It returns bounding boxes around the faces detected in an image. The region within the obtained bounding box is cropped and this cropped face is saved separately. This saved cropped face is labelled as real or fake depending on the video from which the frame was obtained initially. All the cropped faces obtained from frames of all the videos are then given as input dataset to the models. This dataset is splitted into two parts in which 98% is used for training and 2% is used for validation. The models are trained for 10 epoch with a batch size of 32 and model weights are saved for testing the model's performance. This proposed system for training of models gives model weights as the output which are further used for deep fake detection implementation.

This classification system is built on two models to compare the performance of system when these models are implemented as classifiers. Meso4 and Xception64 models are used in this project as classifiers.

B. MesoNet

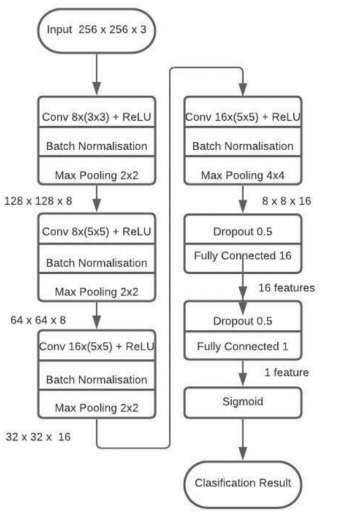

Fig. 4. MesoNet(meso4) architecture

MesoNet[15] is a convolutional neural network which is designed for Deepfake detection purposes. This network is shown in figure 4 which consists of 4 convolutional blocks followed by 1 fully connected hidden layer. Convolutional block include a convolution layer which uses ReLU (Rectified Linear Unit) activation for improving generalization, batch normalization layer for normalizing the inputs to each layer for reducing the interdependence of two layer parameters (inter covariate shift) along with preventing the vanishing gradient effect and a pooling layer after the block to reduce the dimensionality of data to speed up the computational process. Fully connected layer at the last uses dropout to avoid overfitting and improve the robustness of the model. Activation function for last layer used is sigmoid to get better output classification results of real or fake classification.

C. Xception

Xception[16] which stands for extreme inception is a 71 layer deep convolutional neural network and is a advanced version of inception v3 model and it achieved a 78.1% accuracy on ImageNet dataset. The architecture of the Xception model is shown in the figure 5 below.

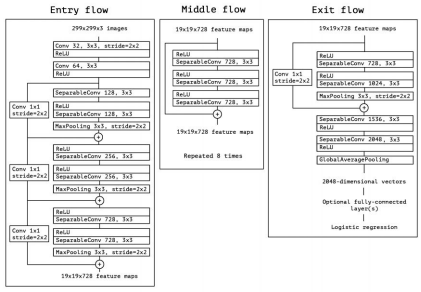

Fig. 5. Xception architecture

The Xception neural network is broken down into the following 3 components.

1) The entry flow: The entry flow takes input images of size 299x299x3 and has 2 blocks of convolutional layer followed by a ReLU layer at the input. The 2 blocks are followed by a series series of separable convolutional layers, ReLU and maxpooling layers with a interconnects between layers as shown in the figure 6 below. The entry flow produces 728 filters of size 19x19.

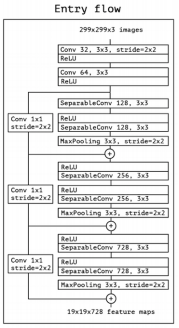

Fig. 6. Entry flow of Xception architecture

2) The middle flow: The middle flow consists of 8 set of repeating blocks. The input and output size of these blocks are 19x19x728 and are kept the same as the blocks should be repeating one after the other. The repeating blocks consists of series of ReLU and separable convolutional layers. There are no max pooling layers in the middle flow as the size of the input and the output are the same.

3) The exit flow: The exit flow consists on 4 layers of separable convolutional and ReLU layers with interconnect

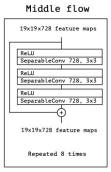

Fig. 7. middle flow of Xception architecture

and max pooling layer as shown in the figure 8 below. The exit flow is then followed by a global averaging layer with 2048 vector output and is followed by a fully connected layer. To get the confidence level of each of the class, the output layer is a logisitic regression layer which gives the probability of output being in each class.

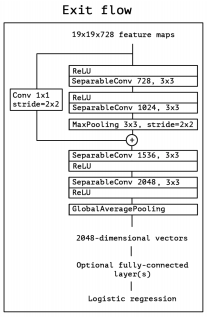

Fig. 8. Exit flow of Xception architecture

There are 2 distinguishing features in the Xception model which make the model more simpler and training more effective. They are:

a) Separable convolutional layers: The separable convolutional layers greatly reduces the complexity of the network by changing a 2D convolution layer into 2 1D convolution layers. In the case of a mxn convolutional filter, to get a output pixel value we have to perform mxn multiplications. But for a mxn separable convolutional layer the number of multiplications reduces to (m+n).

b) Interconnects between different layers: It can be seen in the figure 5 that all the 3 blocks have interconnects between layers which will improve the training accuracy. It can also be seen that some of the interconnects have convolutional layer with certain stride, this is done to match the dimensions of the inter-connecting layers where the main path of the neural network will have a maxpooling layer which reduces the size the output.

V. Implementation

Flowchart of implementation of this project is shown in figure 9. The Deepfake detection system takes an unknown video as input which is to be classified as real or fake. From this video, faces present are detected using dlib face detection frame by frame. The face detected in each frame is surrounded

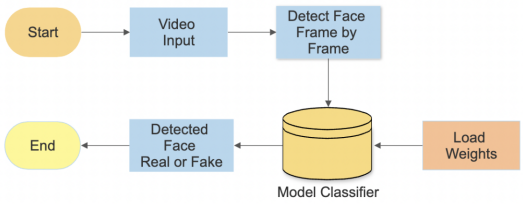

Fig. 9. Implementation of our detection system

by a bounding box, which is given as input to Meso4 and Xception64 classifiers. The weights of the two models saved after the training process are loaded into the classifiers and based on these weights, classifier classifies the cropped image of face as real or fake. So, the output of the Deepfake detection system is classifier output of real or fake considering each frame independently. Hence, with these steps, the system can predict whether the given input is real video or forged video obtained using Deepfake techniques. Accuracy obtained from both the models is discussed in results section.

VI. Results

A. Mesonet model

Mesonet model was trained in keras using a dataset of 20000 training images and 5000 testing images. The model was trained using adam optimizer with a learning rate $\alpha = 0.001$ and momentum $\beta = 0.99$. The model was trained for 10 epochs and the training and test accuracy of the model is shown in figure 10. In figure 10, the training and validation accuracy is plotted against the number of epochs . In figure 11 is a plot between loss being updated after every epoch during the training process of mesoNet after obtaining the feedback during back propagation. From the above plots, it can be seen that accuracy of MesoNet4 or Meso4 model is increased to approx 95% for training and validation dataset and loss of the model is decreased to approx 5% for training and approx 1% for validation dataset.

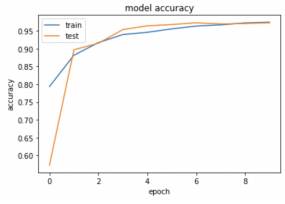

Fig. 10. Mesonet training and testing accuracy vs epoch

The classification scores obtained for MesoNet4 is a loss of 0.0674 and accuracy of 0.9741 for training, and a loss of 0.0966 and accuracy of 0.9725 for validation dataset input.

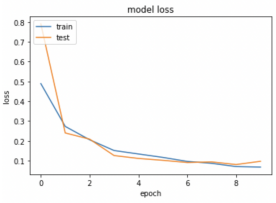

Fig. 11. Mesonet training and tesing loss vs epoch

B. Xception model

The Xception model was trained for 5 epochs with 50000 images in each epoch. The learning rate $\alpha = 0.001$ and the optimizer used was adam with momentum $\beta = 0.9$. The testing dataset contained 10000 image in which the deepfakes were generated by the same method as the training dataset. The testing accuracy obtained after 5 epochs of training is 98.1%. In figure 12, The training and testing accuracy of mesonet and Xception models are plotted.

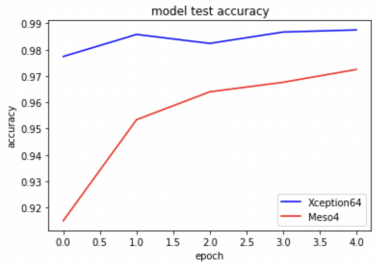

Fig. 12. Model Accuracy comparison between mesonet and Xception models

In figure 13, the loss/cost during the testing is plotted and is decreasing after every epoch, this validates that the models is not being overfitted.

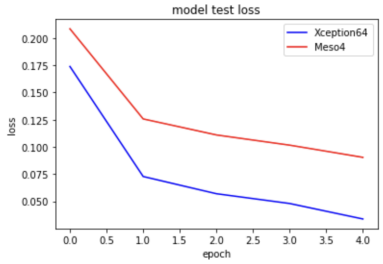

Fig. 13. Testing loss comparison between mesonet and Xception models

To test the applicability of the trained model against deepfakes obtained by generated by other methods, the validation dataset also included the deepfakes generated by neuraltexture and face-shifter methods. The Xception model gave a validation accuracy of 77.1% and the mesonet model gave a validation accuracy of 62.3%. This result is tabulated in the table 1.

Model trained	Validation accuracy(in %)
Xception model	77.1
Mesonet model	62.3

TABLE I
VALIDATION ACCURACY OF MODELS WHEN TESTED AGAINST OTHER DATASETS

VII. Conclusion

The Xception model performed significantly better compared to the mesonet model in detecting deepfakes. Though there is no significant difference in testing accuracy of Xception and mesonet model training, Xception model performed significantly better than mesonet in validation accuracy with other datasets. The Xception model gave a accuracy of 77.1% and the mesonet gave a accuracy of 62.3% when the validation set included deepfakes generated by neural-texture and face-shifter methods.

The future work of the deepfake detection would be to use temporal data and audio data to increase the detect-ability of deepfakes. 2 subsequent frames can be given to the neural network to take advantage of temporal disconnect between deepfakes generated in 2 subsequent frames. Audio input can also be traced to mouth movements in the video and can be used to detect deepfakes. The present system can detect deepfakes in both photos and videos and the Xception model detects deepfakes with an accuracy of 77.1%.

VIII. Contributions

A. Aditya Kaduskar

Worked on the mesonet implementation in keras, training and testing in keras, literature survey on various deepfake creation techniques and architectures to detect it and abstract and literature survey sections in the report and presentation

B. Aradhita Sharma

Worked on the mesonet implementation in keras, training and testing in keras,literature survey on various deepfake creation techniques and architectures to detect it and Mesonet architecture and mesonet results in the report and presentation

C. Swaroop Pandit

Worked on the faceforensics++ data preprocessing (downloading the video dataset and getting the frames and extracting faces), Xception model implementation in pytorch, training and testing in pytorch,literature survey on various deepfake creation techniques and architectures to detect it and Xception architecture, results and conclusion in the report.

Acknowledgment

We would like to thank Prof. Pavan Turaga and teaching assistant Ankita Shukla for providing us with a wonderful opportunity to do this project. We would also like to thank Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, Matthias Niebner for providing FaceForensic++ dataset which was used in this project.

REFERENCES

[1] Justus Thies, Michael Zollh¨ofer, Matthias Nießner, Levi Valgaerts, Marc Stamminger, and Christian Theobalt. Real-time expression transfer for facial reenactment. ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH Asia 2015, 34(6):Art. No. 183, 2015.

[2] Christoph Bregler, Michele Covell, and Malcolm Slaney. Video rewrite: Driving visual speech with audio. In 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '97, pages 353–360, 1997.

[3] Christoph Bregler, Michele Covell, and Malcolm Slaney. Video rewrite: Driving visual speech with audio. In 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '97, pages 353–360, 1997.

[4] Fakeapp. https://www.fakeapp.com/.

[5] Justus Thies, Michael Zollh¨ofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2Face: Real-Time Face Capture and Reenactment of RGB Videos. In IEEE Conference on Computer Vision and Pattern Recognition, pages 2387–2395, June 2016.

[6] H. Farid. A Survey Of Image Forgery Detection. IEEE Signal Processing Magazine, 26(2):26–25, 2009.

[7] J. A. Redi, W. Taktak, and J.-L. Dugelay. Digital image forensics: a booklet for beginners. Multimedia Tools and Applications, 51(1):133–162, 2011.

[8] T. Julliand, V. Nozick, and H. Talbot. Image noise and digital image forensics. In Y.-Q. Shi, J. H. Kim, F. P´erez-Gonz´alez, and I. Echizen, editors, Digital-Forensics andWatermarking: 14th International Workshop (IWDW 2015), volume 9569, pages 3–17, Tokyo, Japan, October 2015.

[9] M. Barni, L. Bondi, N. Bonettini, P. Bestagini, A. Costanzo, M. Maggini, B. Tondi, and S. Tubaro. Aligned and nonaligned double jpeg detection using convolutional neural networks. Journal of Visual Communication and Image Representation, 49:153–163, 2017.

[10] S. Milani, M. Fontani, P. Bestagini, M. Barni, A. Piva, M. Tagliasacchi, and S. Tubaro. An overview on video forensics. APSIPA Transactions on Signal and Information Processing, 1, 2012

[11] W. Wang and H. Farid. Exposing digital forgeries in video by detecting double mpeg compression. In Proceedings of the 8th workshop on Multimedia and security, pages 37–47. ACM, 2006.

[12] W. Wang and H. Farid. Detecting re-projected video. In International Workshop on Information Hiding, pages 72– 86. Springer, 2008.

[13] J.-W. Lee, M.-J. Lee, T.-W. Oh, S.-J. Ryu, and H.-K. Lee. Screenshot identification using combing artifact from interlaced video. In Proceedings of the 12th ACM workshop on Multimedia and security, pages 49–54. ACM, 2010.

[14] Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J. and Nießner, M., 2019. Faceforensics++: Learning to detect manipulated facial images. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 1-11).

[15] Afchar, D., Nozick, V., Yamagishi, J. and Echizen, I., 2018, December. Mesonet: a compact facial video forgery detection network. In 2018 IEEE International Workshop on Information Forensics and Security (WIFS) (pp. 1-7). IEEE.

[16] Chollet, F., 2017. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1251-1258).