October 2019

Detecting Digitally Forged Faces in Online Videos

Neilesh Sambhu
University of South Florida

Detecting Digitally Forged Faces in Online Videos

by

Neilesh Sambhu

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Computer Science
Department of Computer Science and Engineering
College of Engineering
University of South Florida

Date of Approval:
October 17, 2019

Keywords: machine learning, deepfake, deep learning, computer vision, convolutional neural network

Dedication

This thesis is dedicated to my family, friends, and the University of South Florida faculty who have helped support me in my educational endeavors.

Acknowledgments

I want to thank Dr. Shaun Canavan for guiding me through the research process. The memories of his kind words and open mind will remain with me for a lifetime. Additionally, I want to thank Daniel Sawyer, David Sherrier, and Diego Fabiano for being helpful colleagues and providing me with the keys to becoming a more successful researcher. Lastly, I want to thank Dr. Yu Sun and Gabriela Franco for always providing me with prompt administrative assistance throughout my graduate studies.

Table of Contents

List of Tables

List of Figures

Abstract

We use Rossler's FaceForensics dataset of 1004 online videos and their corresponding forged counterparts [1] to investigate the ability to distinguish digitally forged facial images from original images automatically with deep learning. The proposed convolutional neural network is much smaller than the current state-of-the-art solutions. Nevertheless, the network maintains a high level of accuracy (99.6%), all while using the entire FaceForensics dataset and not including any temporal information. We implement majority voting and show the impact on accuracy (99.67%), where only 1 video of 300 is misclassified. We examine why the model misclassified this one video. In terms of tuning the network, we observe how changing hyperparameters affects training time for each epoch and accuracy for training, validation, and testing datasets. There are some challenges involved in obtaining consistent results with deep learning because of the randomization involved in initializing weights. We also replicate Rossler's XceptionNet [2] experiment for classifying images as originals or forgeries and examine the underlying issues with his research: using a subset of data that is not representative of the full dataset and lack of generalization because of network overfitting when using transfer learning with XceptionNet. Lastly, we explore future work, including forged audio, different network types, and new image datasets.

Chapter 1: Introduction

1.1 Motivation and Problem Statement

The increase of modern computing power into the 21st century has enabled consumer-accessible devices and electronics to perform several billions of floating-point operations per second [3]. Consequently, the ease with which one can produce digitally forged videos has improved. Moreover, as of December 2018, there are 4.1 billion Internet-connected people throughout the world [4]. Long gone are the days of physically distributed pamphlets and booklets. The majority of people have access to the internet [5]. As such, the spread of information – and, more importantly, *misinformation* – has far-reaching effects.

In news media contexts, the term “Deepfake” is part of the common lexicon [6]. However, this singular verbiage has two distinct meanings: (1) from the computer vision point of view, “Deepfake” refers to one specific technique of forging digital faces; (2) for the news publishers, “Deepfake” refers to any technique that creates a digitally forged face. In this paper, we use the term “forged video” or a slight variation of this term to refer to identify any such digitally altered video and avoid confusion. As can be seen in figure 1, human face forgeries can be very realistic and convincing. The combination of abundant computing power, a widely connected populace, and deceitful individuals provides the groundwork for pranksters; corrupt, politically motivated entities; and other cyber criminals to disseminate convincing yet falsely construed information. Motivated by this, we propose a deep learning solution for detecting digitally forged faces in online videos. The main contributions of this thesis are detailed in the next subsection.

1.2 Contributions

The main contribution of this work is a novel neural network architecture for automatically detecting forged faces in online videos. The contributions are 4-fold and can be summarized as follows.

(1) We propose a convolutional neural network architecture, along with majority voting, to detect forged faces in online videos. We do not incorporate temporal information in our model, and it is a smaller model compared to what is commonly used. We show that there is evidence to suggest a lightweight but fast model can achieve high accuracy for detecting forged faces in videos.

(2) We evaluate proposed neural network architecture in regards to accuracy levels, network size, and generalizability through lack of temporality.

(3) To the best of our knowledge, this is the first work to evaluate the entire uncompressed FaceForensics dataset [1], as other works have only used a subset of the data.

(4) We replicate and evaluate the work from Rossler et al. [1], showing some limitations of the work that helped motivate our proposed neural network architecture.

Chapter 2: Related Works

2.1 FaceForensics Dataset Use

Regarding accuracy, Rossler et al. [1] obtain 99.93% testing accuracy by using a variant of XceptionNet [2], a network architecture with 36 convolutional layers, and transfer learning on the FaceForensics dataset. However, Rossler et al. select 10 frames from each video in the corresponding training, testing, and validation sets [1]. The method by which Rossler et al. choose the 10 frames is not specified [1]. Later in this paper, we will dispute the claim of achieving 99.93% accuracy through XceptionNet.

Afchar et al. [7] use MesoNet to classify authentic and forged videos from the FaceForensics dataset [1]. MesoNet contains 4 convolutional layers and 1 dense layer before the final fully connected layer [7]. As such, Afchar et al. achieve an accuracy of 95.3% [7]. However, Afchar et al. use lightly compressed videos (i.e. compression level of 23) and only use 300 videos from the FaceForensics dataset [1] for training [7].

In the work of Nguyen et al., they use capsule networks to solve an inverse graphics problem [8]. In this case, the problem is specific to detecting forged videos on social networks [8]. Like Rossler et al. [1], Nguyen et al. [8] use only 10 frames from each FaceForensics [1] video. However, unlike Rossler et al. [1], Nguyen et al. [8] specify these subsets of images as always being the first 10 frames of each video. Nguyen et al. [8] achieve similarly high accuracy as Rossler et al. [1] (i.e. greater than 99%). The capsule network reaches 99.13% accuracy on the standard FaceForensics dataset [1], and 99.37% accuracy when random noise is introduced [8].

As mentioned in section 1.2, several papers that use the FaceForensics dataset only use a subset of the data. One notable exception is the work of Sabir et al. [9], which uses the entire dataset. The images are compressed, but their model achieves 89.8% accuracy on the FaceForensics dataset when classifying individual frames – as opposed to majority voting.

2.2 Deep Learning with Images

With each user's visit, the online website ThisPersonDoesNotExist.com uses a generative adversarial network (GAN) [10] to create a unique and realistic face [11]. See figure 1 for example images. All four images were created by a GAN from ThisPersonDoesNotExist.com. The top two images represent typical results, which are believable and realistic. However, the GAN will sometimes create strange and unnatural artifacts in the generated images, as can be seen in the lower two images of figure 1.

Similar to ThisPersonDoesNotExist.com, the Chinese phone app ZAO allows users to generate realistic movie scenes with users' faces inserted in place of famous actors and actresses [12]. There are limitations: users can only select a handful of movie clips to have their faces inserted [12]. However, the speed and simplicity with which the app generates the videos is astonishing. Video generation takes less than 8 seconds; users need to upload only one photo of their own face, and the app is still able to create realist video renderings [12].

More generally speaking, deep learning is an important field with far-reaching implications. Hinduja et al. [13] use a convolutional neural network – the same type of network used in this paper's proposed model – to detect the emotional valence of a subject [13]. In this case, the convolutional neural network is part of a larger model that uses several input sources to generate a valence output [13]. However, the underlying concept of convolving over facial images holds true for both the work of Hinduja et al. [13] and the approach we take in this paper.

Ultimately, deep learning – and machine learning in general – are tools that enable computers to create some form of understanding of the natural world; from this understanding, the computers make decisions for people, decreasing the amount of time humans need to spend on tasks, especially those which are tedious and time consuming.

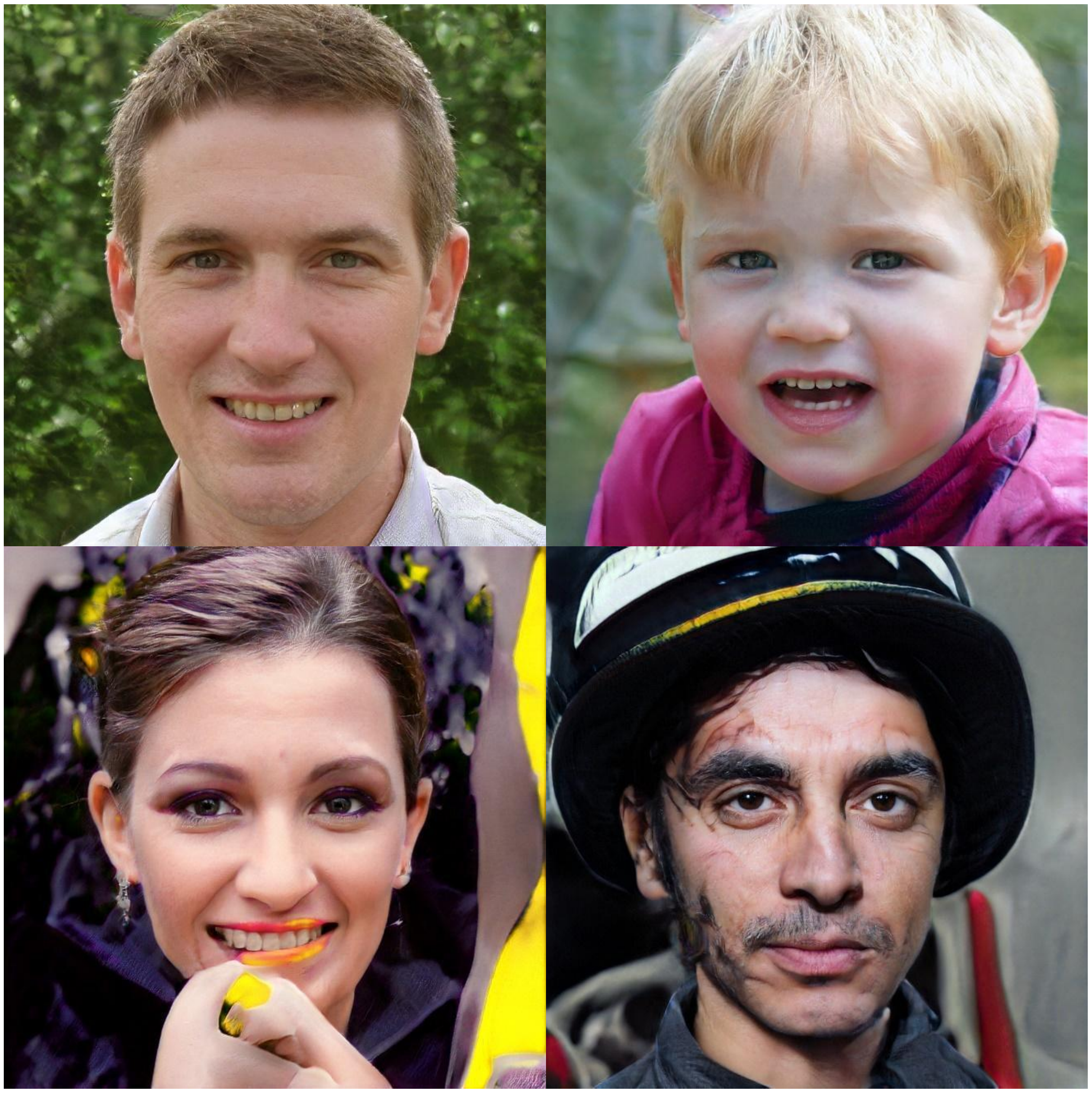

Figure 1 Images created from ThisPersonDoesNotExist.com [11].

Chapter 3: Dataset

3.1 FaceForensics

The dataset consists of 1004 videos from online news and other media sources like YouTube [1]. The creators of the FaceForensics database created all the face forgery videos (called "altered" videos) from the parent videos (called "original" videos). As such, there is perfect class balance in the dataset between altered and original videos. The FaceForensics dataset includes 736,270 samples of training data sourced from 704 videos; 151,052 samples of validation data sourced from 150 videos; and 155,490 samples of testing data sourced from 150 videos. The aforementioned frame count values are split evenly between altered and original videos. All frames are cropped and centered around the face. Additionally, all frames are resized to 128 ×128 pixels before being fed into any network.

Chapter 4: Experimental Design

4.1 Neural Network Implementation

In this paper, we propose a convolutional neural network [14] that has 5 layers and provides 99.6% accuracy in distinguishing between the entirety of the original and altered test videos of the FaceForensics dataset. The model is trained on individual frames and provides classification labels ("original" or "altered") for individual frames. See figure 2 for an outline of how the network is constructed. Because the model is a binary classifier and has one neuron for the final dense node, the classification label is presented as a decimal value between zero and one, with anything less than 0.5 representing an original video and anything greater than or equal to 0.5 representing an altered video. Note that the batch normalization and flattening are simply operations that transform the data and do not represent layers with neurons and corresponding weights, as the convolutional and dense layers do.

Coding in Python [15], we used a sequential model in Keras [16] to build the network. The model was compiled using the following hyperparameters: a binary cross-entropy [17] loss function, the Adam optimizer [18] with a learning rate of 0.001, and accuracy as the metric. Hyperparameters for fitting the model include a batch size of 128. We set the model to train for a maximum of 10 epochs. However, the early stopping callback in Keras monitors the validation accuracy and stops training when the difference in validation is less than 0.01. Later we will discuss the fine tuning and why we chose this network in particular and its hyperparameters.

During the model creation process, we ran into two major file I/O issues: (1) normalization of floating-point values and (2) image read-in time. In an attempt to avoid any vanishing or exploding gradients [19], we normalized the input values by dividing each RGB pixel by 255. Unfortunately, this caused issues with the convolutions resulting in the network to output predictions of NaN (not a number). We fixed this issue by removing the division operation. Regarding image read-in time, reading in all the pre-cropped PNG images from the FaceForensics dataset into numpy arrays took over 10 hours. Later, we stored the pre-processed images as numpy arrays (*.npy) on hard disk. This reduced subsequent image read-in times to 15 minutes. By storing the RGB values as unsigned 8-bit integers instead of 32-bit integers, we brought read-in time down to 10 minutes.

4.2 XceptionNet Replication

In the publishing of the FaceForensics database, Rossler et al. [1] use transfer learning on XceptionNet pre-trained on ImageNet. Rossler et al. [1] freeze the first 36 layers of XceptionNet during training on the FaceForensics images. These layers represent the first 4 blocks of XceptionNet [2]. Additionally, Rosler et al. [1] replace the last layer with a dense layer with two nodes to represent the classification of an original or altered image. Rossler et al. [1] then train the network for 10 epochs on a subset of the FaceForensics dataset. Hyperparameters include using Adam optimizer with a learning rate of 0.001 and a batch size of 64 [1]. The training data of Rossler et al. [1] come from 10 frames from each of the 704 videos in the training dataset. Because each video has its original and corresponding forged complement, each video provides 20 frames of training data. In like manner, the validation and testing data of Rossler et al. [1] come from 10 frames from each of the 150 validation and 150 testing videos. As mentioned previously, Rossler et al. [1] achieve an accuracy of 99.3% across the test data subset.

In this paper, we replicate the XceptionNet experiment of Rossler et al. [1] with one change: instead of using a subset of data (10 frames manually selected from each video), we use all frames of the videos. With this approach, we achieve validation and test accuracies of both 50%. See table 1 for further details during each epoch of training. See table 2 for the confusion matrix on the test data.

There is strong evidence to suggest the XceptionNet experiment overfits: (1) there is a stark difference in training and validation accuracy in the first epoch, (2) training accuracy marginally improves while validation accuracy remains constant and does not change through the course of training, and (3) the confusion matrix for the test data has a strong bias towards class 0 (the original class). Given our partial replication of the experiment of Rossler et al. [1], it is reasonable to conclude one or both of the following: (1) the selection process by Rossler et al. [1] of 10 frames from each video was not representative of the entire videos or (2) the XceptionNet architecture overfits to the training data and is not able to generalize.

4 ×	1 Convolutional Layer (4 filters, 3 × 3 window size)
	Batch Normalization
1 ×	Flatten
1 ×	Dense Layer (1 neuron, Sigmoid activation)

Figure 2 Proposed network architecture.

Table 1 Training output for XceptionNet experiment on full dataset.

Epoch number	Training time (seconds)	Training loss	Training accuracy	Validation loss	Validation Accuracy
1	6385	0.0223	0.9907	8.0590	0.5000
2	6337	0.0043	0.9987	8.0590	0.5000
3	6222	0.0033	0.9991	8.0590	0.5000
4	6210	0.0025	0.9993	8.0590	0.5000
5	6233	0.0038	0.9991	8.0590	0.5000
6	6285	0.0020	0.9995	8.0590	0.5000
7	6288	0.0017	0.9995	8.0590	0.5000
8	6273	0.0013	0.9996	8.0590	0.5000

Table 1 (Continued)

Epoch number	Training time (seconds)	Training loss	Training accuracy	Validation loss	Validation Accuracy
9	6252	0.0011	0.9997	8.0590	0.5000
10	6230	0.0014	0.9997	8.0588	0.5000

Table 2 Confusion matrix on testing data for XceptionNet experiment on full dataset.

	Predicted original	Predicted altered
Ground truth original	77745	0
Ground truth altered	77744	1

Chapter 5: Results

5.1 Frame-Based Results

The proposed model (see Section 4.1) achieves an accuracy of 99.6% on the entirety of the testing dataset from Rossler et al. [1]. Table 3 provides details on the output from each epoch during training. Table 4 outlines the confusion matrix for the model.

5.2 Video-Based Results

Given the same model used in section 5.1, we can provide classification labels on entire videos in the test dataset through the process of majority voting. Recall that there are 150 videos in the test dataset, and each original video creates a complementary forged video.

5.2.1 Majority Voting

With majority voting, we achieve a marginally higher accuracy of 99.67%. See table 5 for the detailed confusion matrix. By allowing each frame to have a vote in labeling the video, some noise is removed and none of the original videos are predicted as altered. However, one altered video is predicted as original.

5.2.2 Failure Case

The mispredicted video is an altered video that the model predicted as original. It contains 543 frames. Of the 543 frames, 288 are classified as original and 255 are classified as altered. This is a split of 53% original to 47% altered. This indicates that this single mispredicted video accounts for 52% of the 557 altered frames that the model predicts as original, as seen in table 4. Additionally, the 6% difference in the distribution of original and altered frames indicates that the model did not have a high confidence for labeling this video. See figure 3 for a more detailed breakdown of the model's frame predictions.

As shown in figure 4, the frames are visually very similar. Upon closer inspection, we can see in both frames the existence of unnatural lines along the sides of the face near the ears. Looking very closely, we can also see an unnatural curvature along the chin. With figure 5, we see the original and forged complements of the same video frame. Here we can more clearly see how a forged face differs from the original. Again, the edges along the sides of the face and chin have a slightly unnatural appearance. Additionally, the shadows cast by the noses differ.

5.3 Neural Network Fine Tuning

We use classification results for the individual frames as the primary tool for tuning the network. We focused on changing the batch size and number of filters when designing the network, keeping in mind real-world constraints like graphics memory limits and maintaining reasonable processing times. Changing the number of layers will impact the testing accuracy. However, this effect is marginal; the proposed network modified to contain only 2 convolutional layers will still achieve a testing accuracy of 97.8%. We also tried progressively changing the number of filters for each layer in the convolutional section (i.e. convolutional layer i contains 2^i filters, where $0 \leq i \leq 3$). Again, this model was promising but still underperformed the proposed network by achieving 99.0% testing accuracy.

From sections 5.3.1 and 5.3.2, we provide experimental evidence to suggest the following three guidelines for a successful network architecture: (1) moderate batch size (e.g. 64 or 128), (2) few convolutional layers (e.g. 2-4), and (3) small number of filters (e.g. 2-8) per layer. The second guideline is arguable the most surprising; following AlexNet, Krizhevsky et al. [20] achieved a top-5 error of 17% in the 2012 ImageNet Large Scale Visual Recognition Challenge. AlexNet has five convolutional layers as well as three fully connected layers [20]. Each of the convolutional layers has at least 256 filters [20]. The ImageNet dataset contains over 15 million labeled images within 22,000 categories [20]. Given the success of AlexNet and the size of the FaceForensics dataset, we would expect a deep network with a high number of filters for each convolutional layer to be highly successful. However, the opposite holds true. This may be due to the fact that AlexNet needed to look for high-level deep features within ImageNet, whereas our proposed model needed to look for low-level features only (i.e. rendering issues at the pixel level).

5.3.1 Impact of Batch Size

We trained the following networks in table 6 very similarly to the proposed network. The model was compiled using the following hyperparameters: a binary cross-entropy loss function, the Adam optimizer with a learning rate of 0.001, and accuracy as the metric. Regarding hyperparameters for fitting the model, the batch size is a variable we change. We set the model to train for a maximum of 10 epochs. However, the early stopping callback in Keras monitors the validation accuracy and stops training when the difference in validation is less than 0.01. Compared to the proposed model, each network in this set contains only 1 convolutional layer with 4 filters and a 3×3 window size. Networks with an asterisk next to the batch size indicate an additional batch normalization operation. This batch normalization takes place before the convolutional layer. As previously mentioned in section 4.1, we attempted to avoid vanishing or exploding gradients [19].

As we can see in table 6, batch sizes of 256 and higher tend to have poor validation and testing accuracies. While a batch size of 512 may provide a faster training time, the reduced classification makes the tradeoff not worthwhile. Additionally, attempting a batch size of 2048 will result in out-of-memory errors on our GPU. There is a negative correlation between batch size and training time per epoch, with the exception of a batch size of 1024. This exception is likely due to an increased overhead in transferring the data from main memory to GPU device memory.

5.3.2 Impact of Number of Filters

We trained the following networks in table 7 very similarly to the proposed network. The model was compiled using the following hyperparameters: a binary cross-entropy loss function, the Adam optimizer with a learning rate of 0.001, and accuracy as the metric. Hyperparameters for fitting the model include a batch size of 1. We set the model to train for a maximum of 10 epochs. However, the early stopping callback in Keras monitors the validation accuracy and stops training when the difference in validation is less than 0.01. Compared to the proposed model, each network in this set contains only 1 convolutional layer with 4 filters and a 3 × 3 window size.

As shown in [21], we chose a small batch size in an attempt to maximize the quality of the model and allow the best possible chance for generalizability. However, given this dataset, we have evidence to suggest that a larger batch size (e.g. 64 or 128) actually provides better results: the same 4-filter network in the first row of table 7 modified with a batch size of 128 provides a classification accuracy for the testing dataset of 80.08%. For reference, this modified network is the same as the one listed in the second row of table 6.

When generating the results for table 7, we start with the largest number of filters at 1024 and progressively move down by powers of two. Because the large batch sizes (i.e. from 256 to 1024) take between 17.7 and 68.7 hours for each epoch, it is impractical to allow these networks to run beyond one epoch and let the early stopping callback in Keras take effect. While we do not have testing accuracy for these experiments, it is reasonable to infer from the low training and validation accuracies that the testing accuracy would be about 50%. Given the long training times and poor results, we skip intermediary powers of two and begin testing 8 filters.

Table 3 Training output for frame-based experiment.

Epoch number	Training time (seconds)	Training loss	Training accuracy	Validation loss	Validation Accuracy
1	723	0.0575	0.9814	0.1664	0.9702
2	719	0.0105	0.9977	0.0852	0.9869
3	720	0.0033	0.9986	0.0632	0.9905

Table 4 Confusion matrix on testing data for frame-based experiment.

	Predicted original	Predicted altered
Ground truth original	77675	70
Ground truth altered	557	77188

Table 5 Confusion matrix on videos in testing dataset.

	Predicted original	Predicted altered
Ground truth original	150	0
Ground truth altered	1	149

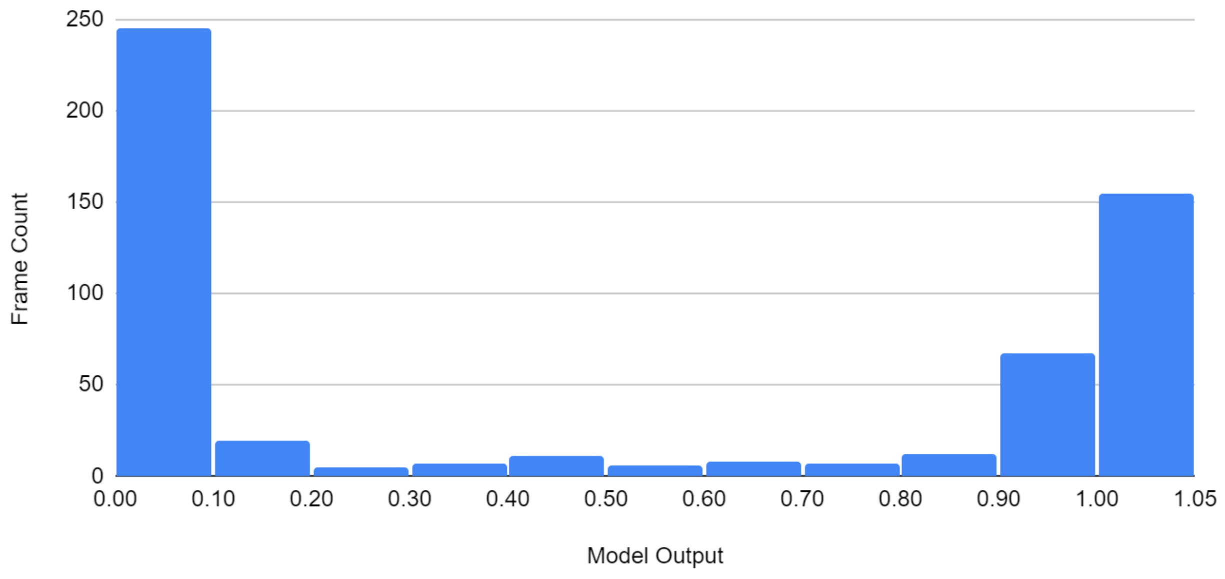

Figure 3 Histogram of model predictions for frames in mispredicted video.

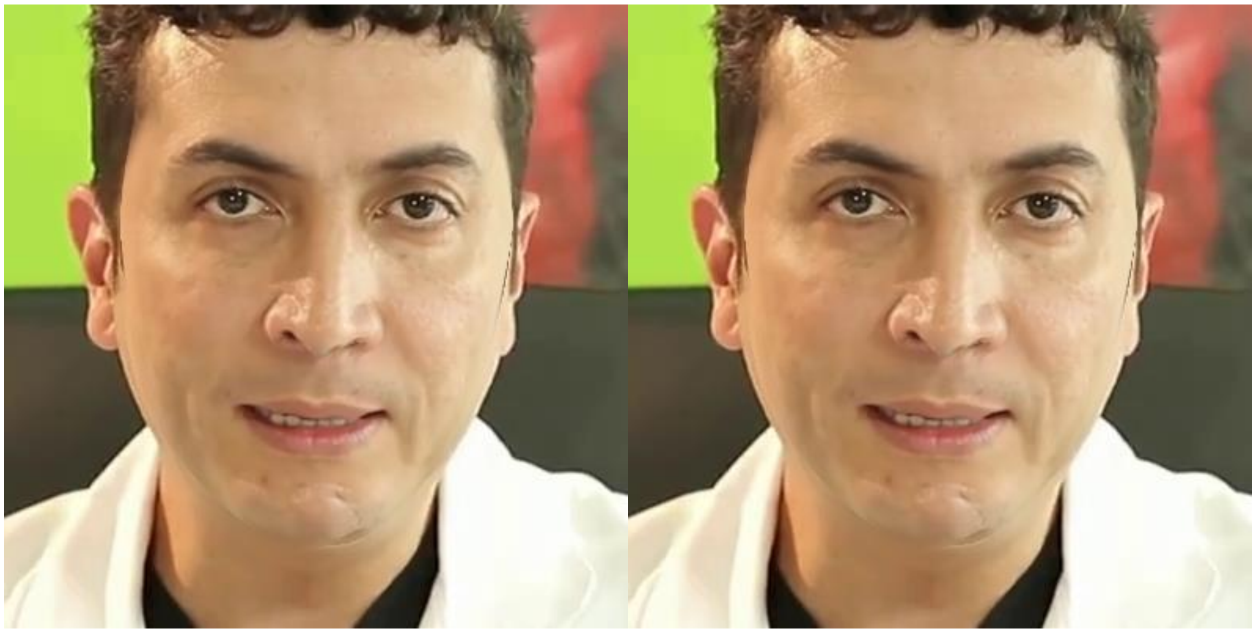

Figure 4 Frame from mispredicted video with model output of 0 (left) and frame from mispredicted video with model output of 1 (right).

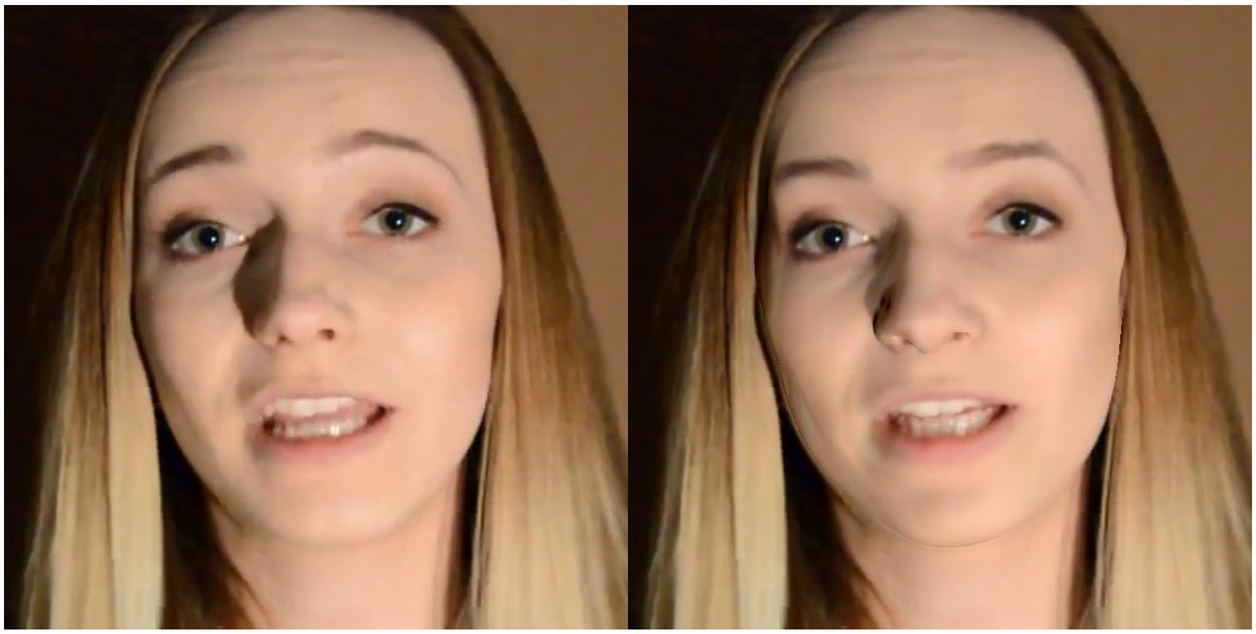

Figure 5 Frame from a correctly predicted original video (left) and frame from a correctly predicted altered video (right).

Table 6 Impact of batch size on training time and accuracy.

Batch Size	Time Per Epoch (seconds)	Epoch 1 Training Accuracy	Epoch 1 Validation Accuracy	Testing Accuracy
64	346	0.8394	0.8137	0.8051
128	328	0.8463	0.8145	0.8009
256	301	0.8176	0.5450	0.5627
512	297	0.7545	0.6870	0.6987
512*	367	0.8127	0.7638	0.7445
1024	332	0.6843	0.6096	0.5796

Table 7 Impact of number of filters on training time and accuracy.

Number of Filters	Time Per Epoch (seconds)	Epoch 1 Training Accuracy	Epoch 1 Validation Accuracy	Testing Accuracy
4	3794	0.7029	0.8080	0.6919
8	4250	0.4999	0.5002	0.5000
256	63666	0.5002	0.5002	N/A
512	124090	0.5002	0.5000	N/A
1024	247255	0.5002	0.5000	N/A

Chapter 6: Challenges with Deep Learning

With nearly any technology comes inherent obstacles. In machine learning, network weights are randomly initialized [22]. As such, some initializations naturally lead to better results than other initializations. With sufficiently large number of epochs for training, these network weights will converge to the same values [23]. However, as the number of epochs increases, the network loses its generalizability and increases its tendency to overfit to the dataset. Therefore, a reasonable goal when training a deep learning model is to find a network whose weights result in a sufficiently highest testing accuracy (i.e. greater than 99%) with the minimum number of epochs. Testing accuracy is a much better metric than either training accuracy or validation accuracy because the testing data are not ever examined during the training process. Therefore, a high classification accuracy on the testing dataset provides some evidence to suggest a reasonable model. When the testing, training, and validation accuracies are all similarly high, this provides strong evidence to suggest a model with a high degree of generalizability.

Chapter 7: Future Work

In the future, we plan to work on detecting forged audio. The Lyrebird AI research team has already performed excellent work at developing a system for duplicating human speech [24]. Being able to combat audio forgery would provide an additional modality for detecting forged videos.

Regarding the image content, we plan to explore various different network types in the future. The proposed model lacked temporality. Incorporating a recurrent neural network (RNN) like a long short-term memory network (LSTM) [25] may pose promising [26]. When implementing majority voting, instead of voting taking place simply across multiple frames, the network can vote based on the decisions of several networks. For example, different networks can be trained for different numbers of epochs (e.g. 1, 5, and 10 epochs). By voting across networks with various training, we may be able to reduce overfitting, increase generalizability, and further increase accuracy. When fine tuning the networks in the future, we can try adding additional layers (e.g. supplementary dense layers before the final dense layer) as well as different operations to transform the data or neurons (e.g. batch normalization before the convolutional layers, dropout, etc.). Lastly, we can use compressed data to simulate online videos on social media more closely, especially those designed for mobile viewing.

References

[1] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, andM. Nießner. Faceforensics: A large-scale video dataset for forgery detection in human faces. *arXiv preprint arXiv:1803.09179*, 2018.

[2] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[3] Processing Power Compared. Experts-Exchange.com. [Online: Accessed September 22, 2019].

[4] Internet Stats & Facts for 2019. HostingFacts.com. [Online: Accessed September 22, 2019].

[5] World Development Indicators. Google.com/PublicData. [Online: Accessed September 22, 2019].

[6] When seeing is no longer believing: Inside the Pentagon’s race against deepfake videos. CNN.com. [Online: Accessed September 23, 2019].

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

[8] H. Nguyen, J. Yamagishi, and I. Echizen. Capsule-forensics: Using capsule networks to detect forged images and videos. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2307–2311. IEEE, 2019.

[9] E. Sabir, J. Cheng, A. Jaiswal, W. AbdAlmageed, I. Masi, and P. Natarajan. Recurrent convolutional strategies for face manipulation detection in videos. *Interfaces (GUI)*, 3:1, 2019.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[11] P. Wang. ThisPersonDoesNotExist.com. [Online: Accessed September 22, 2019].

[12] J. Porter. Another convincing deepfake app goes viral prompting immediate privacy backlash. TheVerge.com. [Online: Accessed September 22, 2019].

[13] S. Hinduja, M. T. Uddin, S. R. Jannat, A. Sharma, and S. Canavan. Fusion of hand-crafted and deep features for empathy prediction. In *2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019)*, pages 1–4. IEEE, 2019.

[14] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, 8(1):98–113, 1997.

[15] Python. Python.org. [Online: Accessed October 10, 2019].

[16] Keras: The Python Deep Learning library. Keras.io. [Online: Accessed October 10, 2019].

[17] Loss Functions. ML-Cheatsheet.ReadTheDocs.io. [Online: Accessed October 10, 2019].

[18] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization.arXiv preprint arXiv:1412.6980, 2014

[19] R. Pascanu, T. Mikolov, and Y. Bengio. Understanding the exploding gradient problem. *CoRR, abs/1211.5063*, 2, 2012.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[21] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

[22] Weight Initialization Techniques in Neural Networks. TowardsDataScience.com. [Online: Accessed October 10, 2019].

[23] Stochastic Gradient Descent Convergence. Coursera.org. [Online: Accessed October 10, 2019].

[24] A. Bresbisson. Lyrebird AI. Descript.com. [Online: Accessed October 3, 2019].

[25] F. A. Gers, J. Schmidhuber, and F. Cummins. Learning to forget: Continual prediction with lstm. 1999.

[26] X. Tu, H. Zhang, M. Xie, Y. Luo, Y. Zhang, and Z. Ma. Enhance the motion cues for face anti-spoofing using cnn-lstm architecture. *arXiv preprint arXiv:1901.05635*, 2019.

About the Author

Neilesh Sambhu was born and raised in the Tampa Bay area. Ever since elementary school, Neilesh had a passion for engineering and computers. As such, the University of South Florida, a Preeminent State Research University, was a perfect fit for Neilesh. With the advent of self-driving vehicles, Neilesh knew that he wanted to work in the computer vision field. Knowing both the cognitive and commercial value of a graduate degree in this field, he decided to pursue a master’s degree in computer science. Neilesh looks forward to working with Dr. Shaun Canavan and Dr. Sudeep Sarkar in his future doctoral studies at the University of South Florida.