DeepFake Detection

Baran Deniz KORKMAZ, Doğukan KALKAN

Department of Computer Engineering

Bogazici University

17.02.2021

CMPE492 - Project

İnci Meliha BAYTAŞ

Abstract

The availability of large-scale public databases and the advent of advanced techniques in deep learning, in particular Generative Adversarial Networks (GANs) and autoencoders, have led into the generation of extremely realistic fake content. The generated fake images and videos may pose a threat for the privacy, democracy and national security. Considering the potential use of deepfakes for malevolent purposes, image and video forgery detection has been a trend topic in research recently. In this paper, we are going to present advanced deep learning techniques used in deepfake creation along with the latest face manipulation techniques, I.e. entire face synthesis, identity swap, attribute manipulation, expression swap. We will discuss about the need for a forgery detection system and challenges in deepfake detection. We will extend our discussion with the most recent state-of-art techniques and our motivation. Then, we present our work. First, a deep technique called Multi-Task Cascaded Neural Networks is used to detect and align face images from the publicly available dataset called FaceForensics++. Second, the extracted real and fake images are fed into a variety of neural networks. The performances of different neural network architectures are combined together in order to use in our future studies. We present extensive discussions on challenges faced so far and our future work in deepfake detection.

Keywords: deepfakes, deep learning, generative adversarial networks, autoencoders, face manipulation, FaceForensics++, face recognition, face alignment, MTCNN, convolutional neural networks

Table of Contents

Table of Contents

List of Figures

List of Tables

1 Introduction

Throughout the years, fake news has always been a threat to human society, democracy and peace. With the improvements in technology in recent years and availability of the Internet and computers and mobile devices, it has now become much easier to catch up with the news from all around the world, which at first glance might sound beneficial and helpful. However, all the improvements and advancements in technology have led to a significant issue, that is information pollution. Nowadays, it takes people two seconds to learn what has happened on the other side of the world on the Internet, which is full of information that is provided by other people. The main issue is that people may provide others with fake news, fake videos or fake audio content in order to deceive and manipulate them.

1.1 Problem Definition

Given the availability of huge data on the Internet, it is now possible to collect them in the form of image, video or audio content in order to create new and synthetic data. The improvements in the Artificial Intelligence and Neural Networks have also enabled people to produce fake content in a more decisive and easy way.

1.1.1 What is Deepfake?

Deepfakes are hyper-realistic video or audio content that depicts people saying or doing things that in reality did not happen. Neural networks are utilized in order to analyze huge sets of data to learn the characteristic features, facial expressions, voice, mannerism and so on. Deepfakes can be extremely hard to detect for humans and also, they can negatively affect the society and even politics since the consequences of fake images or videos can cause various problems.

Figure 1: Two example deepfake frames from a video where a fake Obama is talking.

1.1.2 What is Deepfake Creation?

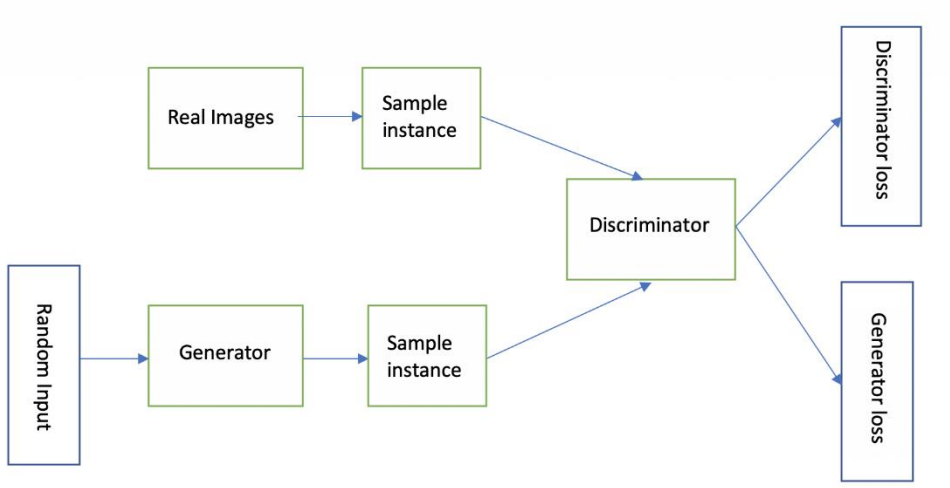

Figure 2: Overview of a simple Deepfake Creation System

Deepfake creation is a process that involves using Artificial Intelligence to create deepfakes, more specifically Generative Adversarial Networks (GANs). GANs consist of two artificial neural networks working mutually to create deepfakes of different types. These two closely related networks are called "the discriminator" and "the generator". The generator is used to create realistic videos or audio contents that are supposed to be good enough to deceive people to a point where people think that those contents are real. The discriminator on the other hand, is used to distinguish between real and

fake content. The more real-looking content the generator produces, the more the discriminator is forced to be better at distinguishing between real and fake content. In other words, these two networks drive each other to gradually improve themselves. The discriminator and the generator are trained on large datasets of images, videos or sounds.

1.1.2.1 Manipulation Techniques

In the field of Deepfake Creation, there are 4 main types of manipulation in general. Below, the manipulation techniques and the datasets created using these techniques will be explained and shown.

i. Entire Face Synthesis: This manipulation technique a new face image from scratch by using powerful GANs (e.g. StyleGAN, Pro-GAN)

Figure 3: Imagined by StyleGAN2 (Dec 2019) - Karras et al. and Nvidia.Taken from: www.thispersondoesnotexist.com

Table 1: Datasets generated by entire face synthesis. [17]

DATABASE	REAL IMAGES	FAKE IMAGES
100K-Generated Images (Karras et al. (2018))	-	100K (StyleGAN)
100K-Faces	-	100K (StyleGAN)
DFFD (2020)	-	100K (StyleGAN) 200K (ProGAN)
iFakeFaceDB (2020)	-	250K (StyleGAN) 80K (ProGAN)

ii. Identity Swap: This manipulation technique replaces the face of a person in a video with the face of another person.

a. 1st Generation

Table 2: 1st Generation datasets generated by identity swap. [17]

DATABASE	REAL IMAGES	FAKE IMAGES
UADFV (2018)	49 (YouTube)	49 (FakeApp)
DeepfakeTIMIT (2018)	-	620 (FaceSwap-GAN)
FaceForensics++ (2019)	1000 (YouTube)	1000 (FaceSwap) 1000 (DeepFake)

b. 2nd Generation

Table 3: 2nd Generation datasets generated by identity swap. [17]

DATABASE	REAL IMAGES	FAKE IMAGES
DeepFakeDetection (2019)	363 (Actors)	3068 (DeepFake)
Celeb-DF (2019)	890 (YouTube)	5639 (DeepFake)
DFDC Preview (2019)	1131 (Actors)	4119 (Unknown)

iii. Attribute Manipulation: This manipulation technique modifies some attributes of the face of a person (e.g. the color of the hair, the age of a person).

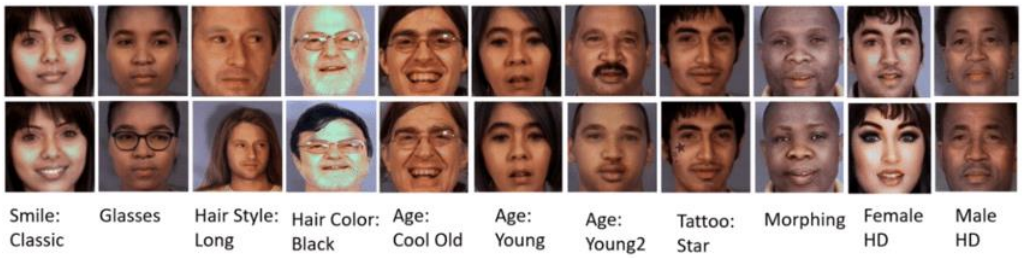

Figure 4: Examples of different manipulations from Deepfake MUCT dataset: Real Samples(first row) and Manipulated Samples by FaceApp(second row) [1].

- Diverse Fake Face Dataset (DFFD) - 18,416 - Face App, 79,960 - StarGAN (Fake Images)

iv. Expression Swap (Face Reenactment): This manipulation technique modifies the facial expression of a person.

- FaceForensics++

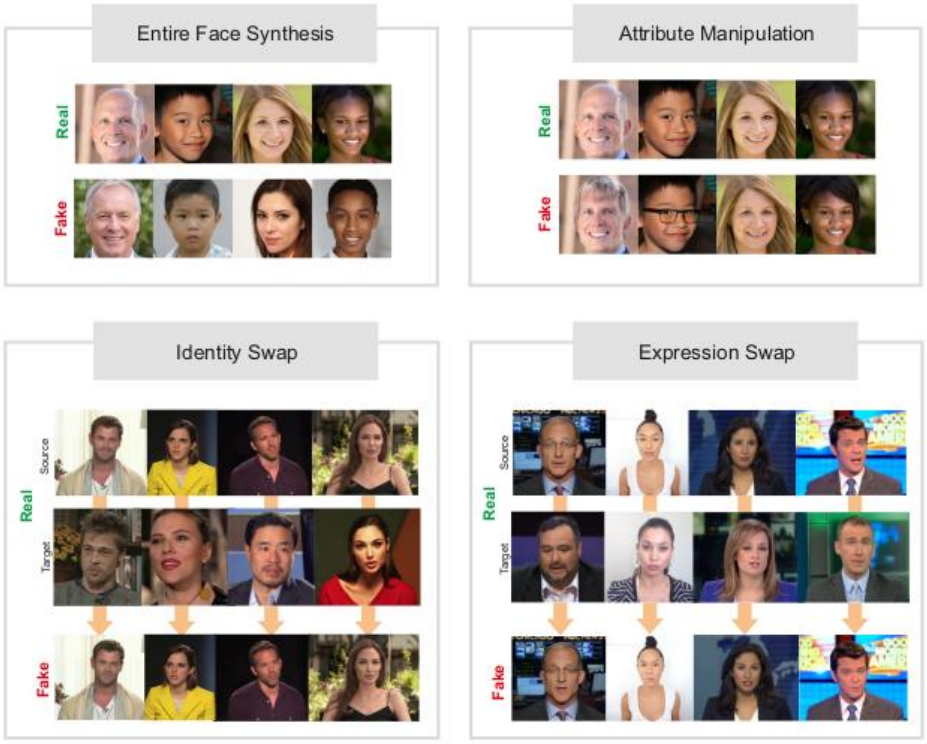

Figure 5: Summary of manipulation techniques [17].

1.1.2.2 Consequences of Deepfakes

Deepfakes, as explained, are images, videos or audio contents that nowadays are almost indistinguishable to human eye. In general. Any other type of fake information can be abused to trick people. In this case however, the consequences can be more devastating due to the nature of the content. Deepfakes can be abused to cause political, social, religious, and economic conflicts in society. If overlooked, they can even be used to starts wars between countries. For example, in 2017, University of Washington researchers created a fake video where the former President of the USA, Barrack Obama speaks. Although the video was created by a group of researchers, the possibilities are endless. In other words, not everyone is pure and nice as the researchers. Some people might use the same technique to create videos of leaders of other countries which can cause problems between countries or even global issues.

In fact, a video of Ali Bongo Ondimba, the President of Gabon appeared on January 1, 2019. Although everyone thought that the video was fake, Gabon's military decided that the video was evidence that Bongo was not fit to be the president and launched a military coup. Although the coup was not successful, this failed coup attempt suggests that deepfakes might be an extreme threat to countries and cannot be overlooked.

1.2 Problem Solution

As explained above, although improvements in the area of Neural Networks have made the process for deepfake creation easier than ever, the same thing applies to the systems that detect these deepfakes created by Deepfake Creation Systems, which are called Deepfake Detection Systems which also utilize the powerful Deep Learning Algorithms.

1.2.1 Why Deepfake Detection

At the disposal of the ill-indented people, deepfake creation systems and deepfakes are powerful weapon to cause problems in society. Deepfakes can be used to blackmail country leaders, business people or even regular people. In the near future, deepfakes might be used to start political issues between countries, or even maybe wars. At this point, a counter defense mechanism is necessary for the peace and democracy in society. Since the first appearance of deepfakes, the studies in the area of Deepfake Detection have also rapidly increased. Deepfake Detection is a natural response to the creation of deepfakes. It plays an important role to avoid any kind of manipulation and deception caused by deepfakes.

1.2.2 Challenges

The main challenge in the field of Deepfake Detection is the generalization of the model. More specifically, there are large amount of different GANs that produce different types of deepfakes, that is

each of those GAN is specifically designed to manipulate or synthesize an image or a video in a unique way. Of course, it is possible to design a model to detect images or videos created by a specific GAN. However, it would not be feasible to design a different model for each GAN. So, the main goal of Deepfake Detection Researchers is to create a model that is successful at detecting all the fake content given to it.

2 Background

In this section, you will find detailed explanations about the motivation behind the studies related to deepfakes and the recent state-of-art studies, in particular of those which we are inspired in our study.

2.1 Motivation

Deep Learning and Neural Networks have been the two important and popular areas in the Computer Science. The improvements in these areas and the creations of large datasets have enabled computer scientists to create complex and accurate models. Deepfake Detection Models are one of them. In general, although deepfake is a relatively new term in Computer Science, it has been a hot topic since it came out. And, also it is worth noting that deepfakes can be useful in some areas, for example, in cinema. It is now quite easy but still expensive to make the actors look younger than they normally are using Neural Networks. On the other hand, deepfakes can be abused to manipulate people as well. That is why researchers pay attention to Deepfake Detection Systems and ever since Deepfake Creation Systems emerged, so did Deepfake Detection Systems thanks to numerous studies in the field. In the future, Deepfake Creation will be more popular since the available data on the Internet gets bigger and bigger. Thus, Deepfake Detection should also be an important field for the scientists for the sake of society.

2.2 Related Work

In this section, a selected set of recent studies about face forging creation and detection techniques have been presented by the authors. The studies mentioned are specifically those inspired our work.

With the advent of advanced deepfake creation techniques, I.e. generative adversarial networks, large-scale datasets containing fake samples of different manipulation techniques have been introduced. Rössler et al. released a remarkable face manipulation dataset consisting of around half a million manipulated images from over 1000 videos [2]. Later, Rössler et al. Introduced FaceForensics++ [3], which is an extension of previously released FaceForensics dataset.

The release of publicly available large-scale datasets has significantly facilitated new studies in forgery detection. Li and Lyu [4] proposed a detection system based on CNNs in order to detect the existence of artifacts, which are left during the face manipulation, around the detected face regions and the surrounded areas.

The detection systems are not only studied at the image level, but also at temporal level, along the video frames. Güera and Delp [5] proposed a deep network architecture, which is a combination of CNN and RNN, in order to detect fake videos by analyzing temporal features across frames.

Another study about manipulated video sequence detection was proposed by Sabir et al.[18]. The idea behind their study was to exploit temporal discrepancies across frames. Their proposal was similar to Güera and Delp [5]. They proposed a similar combination of CNN and RNN architectures, however they preferred an end-to-end training instead of using a pre-trained model. One another remark about this study is that they only considered low-quality videos in the analysis.

Although many studies have been published for deepfake detection, it is a remark that deriving a robust solution is still a challenge. The proposed architectures struggle with achieving generalization as they perform lower results with 2nd generation databases.

3 Methodology

3.1 Image Preprocessing

In general, image preprocessing in the study of Deepfake Detection is not entirely necessary, but when it is applied, we can usually observe a great amount of improvement in the accuracy of the models. Image preprocessing might consist of several steps, such as resizing, rescaling, noise removal etc. In our project, we needed three steps: face detection, face alignment and face cropping.

The reason why we need to apply these three steps is that we need consistent data points (images) to feed to our model. The frames in our data set are taken from the real and fake videos. The frames are not always in a good shape, which means there is unnecessary information (noise) that can affect the performance of our models in a bad way. In other words, we do not need any additional information other than the face, we should eliminate unnecessary information in the images.

Thus, since our aim is to detect fake faces with ideally high accuracy, we need to apply these three steps to our data.

Figure 6: An Example for face detection, face alignment and face cropping

In our project, we use Multi-task Cascaded Convolutional Networks [6] for face detection and alignment. MTCNN is able to outperform other face-detection networks while retaining real-time performance.

MTCNN consists of three closely-working deep convolutional networks, namely Proposal Network(P-Net), Refinement Network(R-Net) and Output Network(O-Net).

Before these stages, the image is resized to different scales to have a collection of images that will be fed into the P-Net.

i. Stage 1: The input is fed into the P-Net, which will eliminate some of the images in the collection of images created prior to this stage by using Non-Maximum Suppression (NMS).
ii. Stage 2: The remaining images are fed into the R-Net. In this stage, the images are further analyzed and the ones with the lower confidence levels are eliminated.
iii. Stage 3: The remaining images are fed into O-Net. In this stage. The network provides the facial landmarks of the face as an output.

In our project, we used MTCNN through a library called DeepFace [7] created by Şefik İlkin Serengil. We use functions to detect, align and crop the faces in the images in our dataset. MTCNN is used in these functions for detection and alignment. The given output is in the form of 224x224x3. Although this is the output form, before feeding to our model, we adjust the shape according to the input shape of our model.

3.2 Proposed Solutions

This section presents the details about our solution approaches, the proposed network architectures and learning parameters. As mentioned previously, there are different media resources that are manipulated, i.e. audio, image, video. Therefore, there are different solution approaches for the detection problem in terms of classification. Many studies regard the detection problem as a binary classification. However, one-class classification approach is also adopted in some of the recent studies [8].

We regard the problem of deepfake detection as a binary classification problem as it provides the most essential insight into the problem. At the initial stages, we aimed at detecting the forgery video sequences by applying a combination of convolutional neural network (CNN) and long short term memory (LSTM). However, we realized that the detection of video sequences requires much complicated work that is not ideal to begin with. Therefore, we target the detection of forgery images with the approach of binary classification. This way, we are able to compare our results due to the availability of plenty of studies regarding the problem just as we did.

3.2.1 Network Architectures

This section introduces the network architectures we have developed in order to detect fake images successfully. In the following section, you will be able to see the details related to the learning parameters. Please note that these learning parameters are fixed for all proposed network architectures as we want to compare their results.

The collaborators plan to conduct a long-term project which will be explained in detail in **5.2 Future Work.** In the context of undergraduate project, our main goal is to produce a benchmark which will form a baseline for our future studies. Therefore, we decided to build and evaluate different kind of architectures utilizing different learning paradigms. In that sense, we have trained a basic CNN architecture from scratch, applied transfer learning via VGG19, and finally extracted features via ArcFace, a state-of-art face recognition model, and fed the features into a detection network.

3.2.1 4-Layer Convolutional Neural Network from Scratch

In the architecture of a 4-layer Convolutional Neural Network (CNN), each layer applies convolution, downsampling, and a rectification by the activation function Rectified Linear Unit (ReLU). The training from scratch of a 4-layer convolutional neural network is one of the very basic attempts that can be tested since this architecture is one of the most basic and effective designs for common problems.

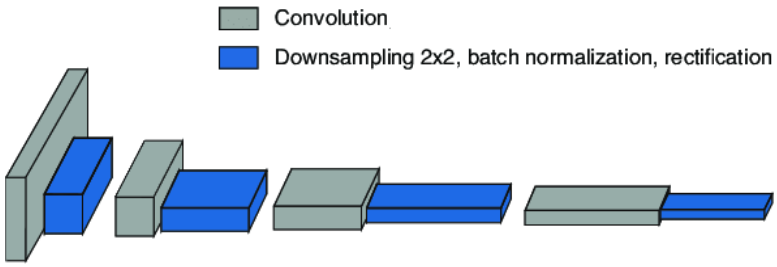

Figure 7: Architecture of 4-layer CNN. [19]

The network was also trained by using batch normalization. Batch normalization is a commonly used technique in deep learning applications that standardizes the inputs to a layer for each mini-batch. This operation provides a more stabilized learning process and it can drastically diminish the required training time to train deep networks.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 224, 224, 16)      448
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 112, 112, 16)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 112, 112, 32)      4640
_________________________________________________________________
batch_normalization (BatchNo (None, 112, 112, 32)      128
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 56, 56, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 56, 56, 64)        18496
_________________________________________________________________
batch_normalization_1 (Batch (None, 56, 56, 64)        256
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 28, 28, 64)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 28, 28, 64)        36928
_________________________________________________________________
batch_normalization_2 (Batch (None, 28, 28, 64)        256
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 14, 14, 64)        0
_________________________________________________________________
flatten (Flatten)            (None, 12544)             0
_________________________________________________________________
dense (Dense)                (None, 1024)              12846080
_________________________________________________________________
dense_1 (Dense)              (None, 32)                32800
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 33
=================================================================
Total params: 12,940,065
Trainable params: 12,939,745
Non-trainable params: 320
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 224, 224, 64)      1792
_________________________________________________________________
batch_normalization (BatchNo (None, 224, 224, 64)      256
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 112, 112, 64)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 112, 112, 128)     73856
_________________________________________________________________
batch_normalization_1 (Batch (None, 112, 112, 128)     512
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 56, 56, 128)       0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 56, 56, 256)       295168
_________________________________________________________________
batch_normalization_2 (Batch (None, 56, 56, 256)       1024
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 28, 28, 256)       0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 28, 28, 512)       1180160
_________________________________________________________________
batch_normalization_3 (Batch (None, 28, 28, 512)       2048
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 14, 14, 512)       0
_________________________________________________________________
flatten (Flatten)            (None, 100352)            0
_________________________________________________________________
dense (Dense)                (None, 128)               12845184
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 129
=================================================================
Total params: 14,400,129
Trainable params: 14,398,209
Non-trainable params: 1,920
```

Figure 8: The architectures of 4-layer CNN implementations. The version on the righthand-side includes batch normalization layers.

In the figure above, you can see the detailed network architectures of two different versions of 4-layer convolutional network.

3.2.2 Transfer Learning with VGG-19

Transfer learning is a method in machine learning that focuses on applying gained knowledge while solving a problem on a different but related problem. The training of complex models with large datasets requires advanced facilities. Such complex networks trained for general problems can later be adapted into more specific problems by configuring the output layer. Therefore, it is a commonly used technique in image classification problems due to the availability of pretrained networks. You can use the pretrained network directly by configuring the output layer for the desired output. Moreover, you can also apply fine-tuning where you enable training of a predetermined number of last layers from the pretrained network in order to provide a better adaptation into the problem.

One commonly used example of complex convolutional neural network architecture is VGG19 [9] which provides a great success in image recognition. It takes fixed size 224x224 RGB images as input. Its neural architecture consists of a stack of convolutional layers followed by three fully connected layers. The final fully connected layer outputs the softmax values belonging to 1000 classes.

VGG19 has a pretrained version trained on ImageNet, a large dataset consisting of 1.4M images and 1000 classes. In our study, we configure the output layer of VGG-19 starting from the first fully connected layer so that it outputs a single sigmoid value that represents the probability of class prediction. Then, we apply fine-tuning starting from the last stack of convolutional layers of VGG19. By doing so, we target adapting a pretrained network, which is commonly used in image classification task, into our problem in order to evaluate the performance of transfer learning on deepfake detection.

3.2.3 Feature Extraction by ArcFace and Detection Network

One final idea is to use a state-of-art deep face recognition model to extract feature embeddings from images and to feed the resulting feature vectors into a basic detection network that outputs either the image represented as the feature is real or fake.

Deng and Guo et al. (2019) proposed an Additive Angular Margin Loss (ArcFace) to obtain highly discriminative features for face recognition [10]. The experimental results suggest that ArcFace loss improves the performance of commonly used backbone networks in face recognition. Some network architectures that are used as a backbone network for ArcFace loss are the variants of ResNet. Serengil et al. [11] has released a face-recognition library that provides the pretrained weights of ResNet34 whereby the ArcFace is used as the loss function.

Similar to VGG19, ResNet34 is composed of a stack of convolutional layers, but in a different configuration. It takes fixed size 112x112 RGB images as input. In overall, its network architecture consists of stack of 5 convolutional layers followed by a fully connected layer which outputs a feature vector of sized 512.

The extracted feature vectors are then fed into a basic detection network consisting of 2 fully connected layers, as you can see the figure showing the architecture below.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 64)                32832
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 65
=================================================================
Total params: 32,897
Trainable params: 32,897
Non-trainable params: 0
```

Figure 9: Architecture of Detection Network.

3.2.2 Learning Parameters

This section describes the details related to the learning parameters including loss function, optimizer, batch size, and learning rate. Please note that these learning parameters are fixed for all proposed network architectures as we want to compare their results.

The preferred loss function is binary cross-entropy loss as it is commonly used in binary classification problems. The binary cross-entropy loss is computed by the formula given below:

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

Binary Cross-Entropy / Log Loss

Figure 10: The formula of binary cross-entropy loss.

Adaptive Moment Estimation (ADAM) is used as the optimizer which is a method for stochastic optimization. ADAM combines the two stochastic gradient descent approaches which are Adaptive Gradients and Root Mean Square Propagation. In the most recent studies, ADAM is commonly preferred as the optimizer. Finally, the mini-batch size is set as 32 and the learning rate is 10^{-4}.

4 Results and Discussion

This section presents the detailed information related to the experiments conducted and the challenges faced during our own work.

4.1 Experiments

This section introduces the FaceForensics++ dataset used in our study and the experimental results achieved by the proposed methods.

4.1.1 Dataset

FaceForenics++ dataset [3] has been released by Andreas Rössler et al. in order to facilitate the new studies in the uprising field of deepfake detection due to the raising concerns given the spread of extremely realistic manipulated media content. It is the extended version of previously released FaceForensics dataset. It contains manipulations based on the classical computer graphics-based methods Face2Face [12] and FaceSwap [13], as well as learning-based approaches DeepFakes [14], NeuralTextures [15] and FaceShifter [16]. It is a large-scale dataset composed of video sequences of 1000 videos with real sources and manipulated versions per each manipulation technique. In overall, it contains 1.8 million real and manipulated images. Finally, FaceForensics++ dataset allows 3 different video quality options which are raw, c23 (HQ), and c40 (LQ).

In our study, we have worked with the images extracted from high-resolution videos, I.e compression quality of c23. We have formed a subset of FaceForensics++ dataset by running a set of scripts. Below, in the figure, you can see the pipeline we set up to bring FaceForensics++ dataset into the form that we want to use in the training of our models.

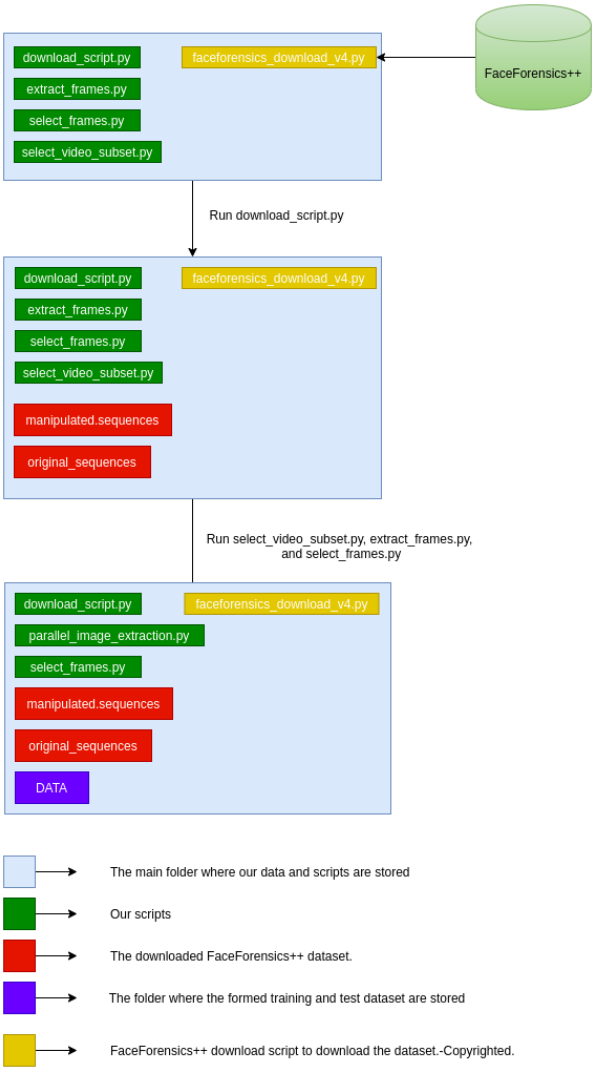

Figure 11: Generation of training dataset.

After downloading the FaceForensics++ dataset by using the download script provided by the owners, we separate a subset of videos equally from each manipulation technique, which are Deepfakes, Face2Face, FaceSwap, FaceShifter, and NeuralTextures. To acquire a balanced dataset, we randomly choose 200 manipulated videos from each manipulation alongside with a total number of 1000 real videos used to create the aforementioned manipulated sequences. At this stage, we end up with 1000 video sequences belonging to each class of real and fake content. We extract the varying number of frames within the video sequences. Our observations concluded that the number of frames vary approximately between 150 to 1100 for each video.

Therefore, we determined a specific number of frames that we are going to extract from each video in a way that we preserve the characteristics of videos without simply grabbing every frame located to avoid poor generalization. Finally, we pick 50 frames randomly per each video sequence and form our dataset that we are going to use in the training. We must also note that, as mentioned previously in **3.1 Image Preprocessing**, we apply face detection and alignment by MTCNN before adding frames into our final dataset.

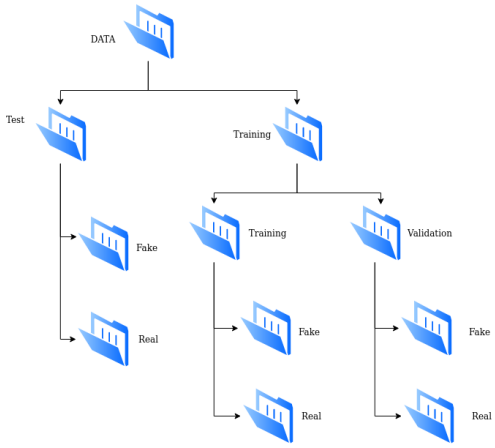

Figure 12: Directory structure of dataset.

As seen above, the resulting dataset has been split into training and test sets with the ratio of 80% to 20% respectively. These subsets contain frames that are entirely separate from each other as they are extracted from distinct video sequences. Then, the training set has been split into two subsets of training and validation with the ratio of 80% to 20% respectively once more in order to meet the requirements of learning paradigm in machine learning

The size of training, validation, and test sets are ~64K, ~16K, and ~20K respectively in a balanced order, i.e. the number of real and fake images are approximately equal.

4.1.2 Results

This section presents the experimental results obtained from the proposed solutions mentioned in **3.2 Proposed Solutions**. The proposed networks have been trained with the dataset generated described in **4.1.1 Dataset**. The dataset contains approximately evenly balanced real and fake images of ~100K images. The training, validation, and test sets contain ~64K, ~16K, and ~20K images respectively. Recall that the learning parameters are constant for each model in order to compare the results. The loss function is binary cross-entropy loss. Adam optimizer is used for training, learning rate is 10^{-4} and mini-batch size is 32.

In the table below, you can see the results obtained from our experiments for each model.

Table 4: Experimental results.

Model	Accuracy
4-Layer CNN v1	51.63%
4-Layer CNN v2 (w/Batch Normalization)	56.31%
VGG-19	60.98%
ResNet34 with ArcFace	49.48%

We observe that the proposed solutions provide accuracy levels that are barely above 50%. 4-Layer CNN architecture as the first attempt provides barely higher accuracy when batch normalization is applied. Transfer learning with VGG-19 provides the best results when we applied fine-tuning in order to customize further for our own problem. Finally, the feature extraction via ResNet34 with ArcFace loss provides the worst accuracy levels surprisingly.

The results suggest that we should re-evaluate our model architectures and training configurations. The lower accuracy results may occur due to lack of model complexity, poor generalization, complex dataset, and problems with training configurations that may cause problems such as overfitting. It is certain that providing a robust model for different manipulation techniques is an extremely challenging problem, given that our training dataset contains 5 different manipulation techniques in overall.

Below, you can see the figures showing the accuracy trends during the training for our proposed architectures.

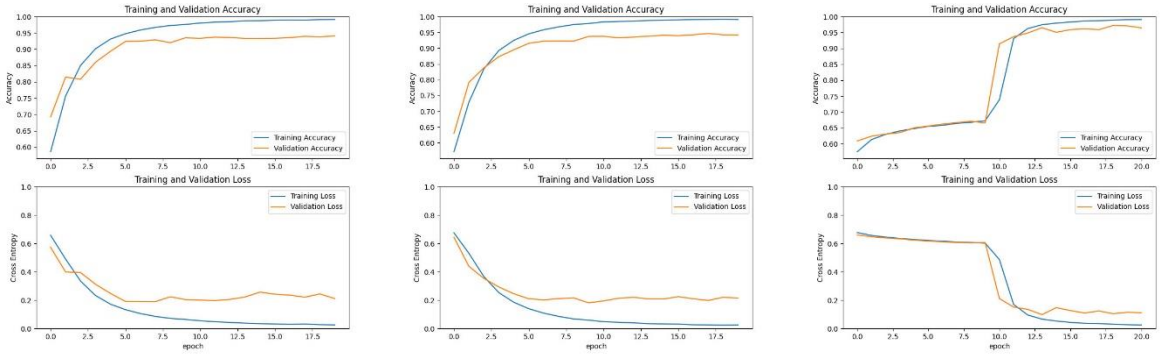

Figure 13: Accuracy and loss trends from our experiments. Starting from the left, the network architectures are 4-Layer CNN, 4-Layer CNN with Batch Normalization, and VGG-19.

The figure indicates that there might be an overfitting problem during training, since the training accuracy levels are close to 1.0. We can apply regularization techniques to avoid overfitting such as early stopping.

4.2 Challenges

In general, all projects bring about some challenges depending on the size of the project. In our case, in order to analyze and create Deepfake Detection Models, we need to train our proposed models with large datasets. So, the first challenge was to have a large dataset in order to maximize the performance of our models.

Another challenge was to create an input pipeline that manipulates the FaceForensics++ dataset to our desire. For this reason, we created several scripts to make the process easier and modular as explained in **4.1.1 Dataset**.

The biggest challenge which has not been resolved yet in our project was to find a GPU Environment to test our models on. We requested GPUs from several providers such as Google Cloud and AWS, but they could not provide us with a GPU. The charge for GPUs was also high, so we could not afford a GPU on these platforms mentioned.

In the beginning, our plan was to analyze videos and create a model to detect the fake videos rather than just images. Compared to studying on images, working with videos was considerably more difficult and also creating an input pipeline and organizing the dataset kept us busy for a long time. Additionally, there are three versions of videos in FaceForensics++ dataset, namely Raw, High Quality and Low Quality. Since we did not have enough memory in our devices, we chose to work with low quality videos, which was still large enough to occupy a big chunk of memory in our devices. Working with low quality videos for Deepfake Detection made the process more difficult for us. Instead, we drew our attention to working on high quality images taken from the videos.

Practically, our input was still frames taken from videos but in the actual code, we had to adjust our input so that each data point would be a collection of frames taken from only one video.

5 Conclusion and Future Work

5.1 Conclusion

With the availability of advanced deep learning techniques and large-scale databases, malicious use of media content is highly possible. The requirement for the forgery detection systems is an undisputable fact. Therefore, the authors have studied the detection of deepfakes for the entire semester. We aimed at developing network architectures based on different learning techniques such as training from scratch, transfer learning, and feature extraction. At the moment, we can conclude that we must review our model architectures and training configurations. In the larger scope, our work will constitute a baseline for our future studies in deepfake detection, as we have created an efficient input pipeline along with different network architectures that can form a basis for future improvements.

5.2 Future Work

With the advances in artificial intelligence, safety and security concerns raise due to the likelihood of malicious use of technology. One example is malicious spoof content that try to deceive a

security system. In that sense, there are many recent studies which attempt to tackle with the detection of AI-generated media. Therefore, we can conclude that this is one of the trend topics in the literature.

Initially, the authors aim at obtaining better results with further improvements on the network architectures. In a broader context, the authors target building an efficient system that combines the face detection and fake detection systems in a way that it automatically detects and verifies the face within a frame.

6 Acknowledgements

We would like to express our great appreciation to our supervisor Assistant Professor İnci Meliha Baytaş. We also would like to thank to the researchers who collaborated in the development of FaceForensics++ for sharing their work.

7 References

[1] Z. Akhtar, M. R. Mouree and D. Dasgupta, "Utility of Deep Learning Features for Facial Attributes Manipulation Detection," 2020 IEEE International Conference on Humanized Computing and Communication with Artificial Intelligence (HCCAI), Irvine, CA, USA, 2020, pp. 55-60, doi: 10.1109/HCCAI49649.2020.00015.
[2] Rössler A, Cozzolino D, Verdoliva L, Riess C, Thies J, Nießner M. Faceforensics: a large-scale video dataset for forgery detection in human faces. arXiv preprint. arXiv:1803.09179 (2018).
[3] Rossler A, Cozzolino D, Verdoliva L, Riess C, Thies J, Nießner M. Faceforensics++: learning to detect manipulated facial images. In: Proceedings of the IEEE international conference on computer vision. 2019.
[4] Y. Li and S. Lyu, “Exposing DeepFake Videos By Detecting Face Warping Artifacts,” in Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2019.
[5] D. Güera and E. Delp, “Deepfake Video Detection Using Recurrent Neural Networks,” in Proc. International Conference on Advanced Video and Signal Based Surveillance, 2018.
[6] K. Zhang, Z. Zhang, Z. Li and Y. Qiao, "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks," in IEEE Signal Processing Letters, vol. 23, no. 10, pp. 1499-1503, Oct. 2016, doi: 10.1109/LSP.2016.2603342.
[7] https://github.com/serengil/deepface Accessed 17.02.2021.
[8] H. Khalid and S. S. Woo, "OC-FakeDect: Classifying Deepfakes Using One-class Variational Autoencoder," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 2020, pp. 2794-2803, doi: 10.1109/CVPRW50498.2020.00336.
[9] Simonyan, Karen & Zisserman, Andrew. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv 1409.1556.

[10] J. Deng, J. Guo, N. Xue and S. Zafeiriou, "ArcFace: Additive Angular Margin Loss for Deep Face Recognition," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 2019, pp. 4685-4694, doi: 10.1109/CVPR.2019.00482.
[11] S. I. Serengil and A. Ozpinar, "LightFace: A Hybrid Deep Face Recognition Framework," 2020 Innovations in Intelligent Systems and Applications Conference (ASYU), Istanbul, Turkey, 2020, pp. 1-5, doi: 10.1109/ASYU50717.2020.9259802.
[12] Justus Thies, Michael Zollhofer, 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.
[13] Faceswap. https://github.com/MarekKowalski/FaceSwap/ Accessed 17.02.2021.
[14] Deepfakes. https://github.com/deepfakes/faceswap/ Accessed 17.02.2021.
[15] Justus Thies, Michael Zollhofer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. ACM Transactions on Graphics 2019 (TOG), 2019.
[16] Lingzhi Li, Jianmin Bao, Hao Yang, Dong Chen, Fang Wen. FaceShifter: Towards High Fidelity and Occlusion Aware Face Swapping. arXiv:1912.13457 (2019).
[17] Tolosana, Ruben & Vera-Rodriguez, Ruben & Fierrez, Julian & Morales, Aythami & Ortega-Garcia, Javier. (2020). DeepFakes and Beyond: A Survey of Face Manipulation and Fake Detection.
[18] E. Sabir, J. Cheng, A. Jaiswal, W. AbdAlmageed, I. Masi, and P. Natarajan, “Recurrent Convolutional Strategies for Face Manipulation Detection in Videos,” in Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2019.
[19] Berardino, Alexander & Ballé, Johannes & Laparra, Valero & Simoncelli, Eero. (2017). Eigen-Distortions of Hierarchical Representations.