DEEPFAKE DETECTION GENERALIZATION VIA KNOWLEDGE DISTILLATION

A THESIS SUBMITTED TO THE GRADUATE DIVISION OF THE UNIVERSITY OF HAWAIʻI AT MĀNOA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

COMPUTER SCIENCE

MAY 2023

By

Cristian Flores

Thesis Committee:

Kyungim Baek, Chairperson
Martha Crosby
Peter Sadowski

Keywords: knowledge distillation, deepfake

ACKNOWLEDGMENTS

I would like to express my gratitude to all that have supported me in my academic career. In particular, I would like to thank Dr. Baek for offering support and insight throughout the process of creating this work, as well as all the members of my thesis committee, Dr. Crosby and Dr. Sadowsky, and the ICS Graduate Chair, Dr. Suthers. I would also like to express my gratitude towards UH MĀNOA and the Department of Information and Computer Sciences for providing the opportunity to continue my studies and for allowing me to serve as a Graduate Assistant. Finally, I would like to thank my family and friends for their continued support throughout my education and life.

ABSTRACT

Two ongoing challenges in the field of deepfake detection are a lack of generalized models and a loss in performance when analyzing highly compressed videos. This work attempts to address these problems, through the use of knowledge distillation (KD) using heterogeneous teachers. In a typical class setting, a student learns from several teachers, each an expert in their domain. This process is analogous to the use of KD with heterogeneous teachers. A contribution of this work is the creation of a KD pipeline that can effectively utilize the knowledge of heterogeneous teachers to train a student model. This pipeline introduces the Winner-takes-all method for utilizing the knowledge of all teachers during training time. The three primary goals of this work are to show that a student model trained utilizing the proposed KD pipeline is a generalized learner, demonstrate that knowledge from its teachers was retained, and maintain a sufficiently high level of performance despite using a shallower, more compressed model compared to its teacher(s). The results indicate that all three goals were met, with the benefits of using the KD pipeline varying from marginal to moderate.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Deepfake videos can be defined as artificially created or altered videos that attempt to alter the face, expressions, mouth movements, and/or head position of person in a target video with corresponding properties from a separate source. As the term is used today, a deepfake manipulation may also include altered audio. In 2017, the original github repo for DeepFake[1] was created, leading to it becoming an easily accessible technology [8]. Since then, deepfake techniques have been utilized on various media for a drastically different motivations. For example, in 2020, the ad agency Craft Worldwide released various versions of a commercial where the featured actor Snoop Dogg advertised an app, which utilized different names across different markets in the world. Through the use of deepfake technology, the ad agency ended up saving time and money by altering the original recording rather than re-recording the commercial [16]. Deepfake technology lends itself well to industries that regularly utilize human models, such as film producers, virtual reality models, and video game developers. However, there exist more harmful and potentially dangerous uses of deepfake technology. Deepfake videos have been used as a defamation tool [2, 9], and are viewed as a continuing threat in politics and elections [4, 10]. The potential damage deepfakes can cause motivates the continued research into detection techniques.

Numerous methods to generate deepfake videos have been created over the years [18, 15, 29, 30]. Likewise, many deepfake detection models have been created, some utilizing generic image classification models [12, 5], while others utilized more specialized architectures [19, 1, 6]. It has been shown that deepfake detection models are able to perform well on individual manipulation methods, but lack the robustness for generalizability [19, 6, 23]. Generalization has been a topic of focus for some literature [14, 21], but there exist drawbacks for such methods as well. A common shortcoming is a drastic drop in performance when analyzing low quality videos. Low quality videos, as they are compressed, tend to lose

[1]The proper noun 'DeepFake' refers to the manipulation technique. In lowercase, 'deepfake' refers to any media altered using any deepfake manipulation technique.

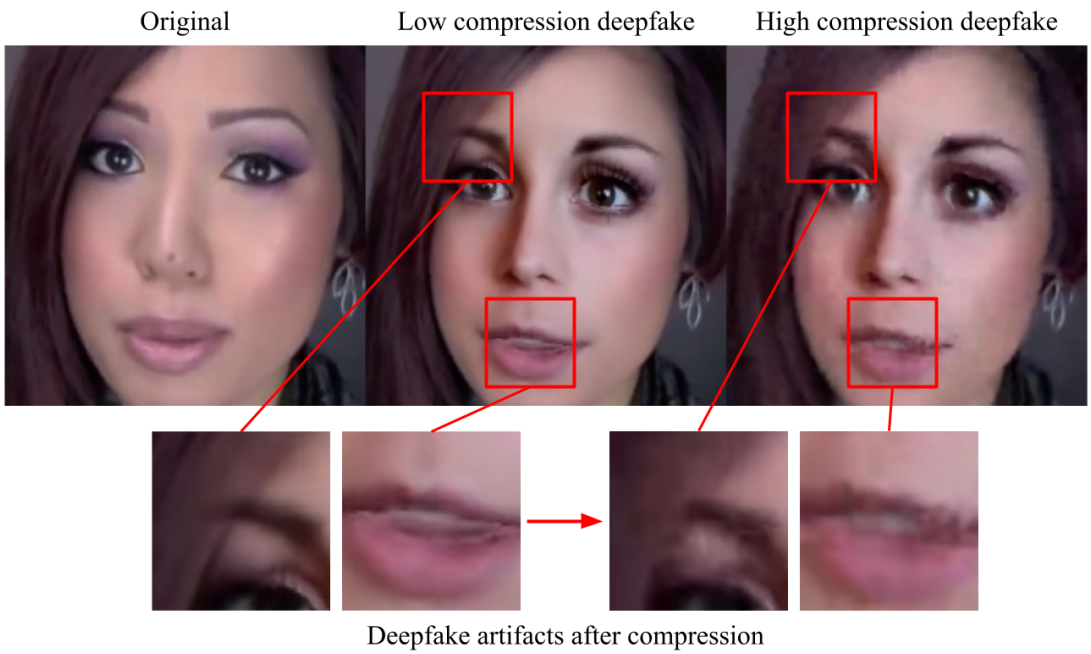

Figure 1.1: Deepfake generated using FaceSwap, faces from FaceForensics++ dataset [23]

information useful for detecting deepfakes. As an example, see Figure 1.1. The image contains a frame from an unedited video on the left, in addition to two frames from deepfakes generated using the unedited video. Both deepfake images are compressed, although to different degrees. Note that compression is applied after a deepfake is generated, as opposed to compressing the original video and then manipulating it. In the low compression deepfake image, the subject's eyebrow can be seen overlapping onto their hair and an object with defined edges appears in their mouth. On the high compression image, these deepfake artifacts become blurrier, and as a result, blend into the subject. Given that videos are compressed on most common media platforms, possibly numerous times over, an ideal model should be able to address both the generalization and low quality problems simultaneously.

In this work, it is proposed that by training a model using a group of heterogeneous teachers via knowledge distillation, both the generalization and low quality problems can be addressed. Knowledge distillation (KD) is a training method where a student model is learned from teacher model(s) [3, 13]. The idea of using heterogeneous teachers is inspired from the human learning process. Typically in school, students learn subjects from a variety

of teachers, each an expert in their respective domain. Furthermore, knowledge of one subject may help improve a student's understanding of another.

The concept of creating artificial neural networks (ANNs) inspired by the human brain, towards the pursuit of human-like AI, is a common characteristic of connectionism, as well as the belief that human cognition may be approximated using ANN's. With the imagery of a student learning from multiple teachers in mind, applying KD to the problem of deepfake detection is thus a natural connectionist experiment. Phrased another way, the goal of this work is to create domain specific ANNs that attempt to become better generalized learners. For a more thorough discussion on connectionism, see [7].

A stated above, a student learns from *expert* teachers. As such, a key component of this paper's methodology is to find expert heterogeneous teachers. Using these heterogeneous teachers, it is possible to train a student model via KD. The student model is evaluated under the following research questions to determine the extent of success, or lack thereof, of KD in regards to generalization, knowledge transfer, and model compression.

- **RQ1:** In comparison to the teachers, and an additional baseline model trained under similar circumstances, does the student model generalize well? In this context, is the student model more accurate at identifying deepfake videos generated using both seen and unseen (during training) deepfake manipulation techniques?

- **RQ2:** How well is knowledge transferred from the teacher models to the student model? In a mathematical sense, given that a teacher is able to identify a deepfake, what is the probability that the student will also successfully detect it?

- **RQ3:** Knowledge distillation was initially created as a compression technique. How does varying the depth and parameters of a student model affect its performance?

Through experiments where a student model was trained using heterogeneous teachers, the research questions were addressed. A baseline model was created that was trained directly using the same data as the student model, but without the assistance of the teacher models during training. The contributions of this paper can be summarized as follows:

A KD pipeline is introduced with the capacity to distill knowledge from n heterogeneous teachers, suited for the task of deepfake detection. It has been shown that KD can improve generalization [17]. As such, a motivation for creating this pipeline comes from the need to create a generalized student, in order to address **RQ1**. Given that the focus of **RQ2** centers around knowledge transfer, a method for effectively processing and distilling the knowledge from n heterogeneous teachers is proposed, named the 'Winner-takes-all' method. Three student models were trained using the pipeline, with the students varying in number of parameters and layers in order to address **RQ3**. Finally, an evaluation of these student models is provided. Through the evaluation, it is shown that student models trained under the proposed KD pipeline generalize better than the baseline models, the 'Winner-takes-all' method effectively passes knowledge from the teachers to the students, and that student models with as little as 50% of the number of parameters of one of the teachers can perform well with minimal performance losses.

CHAPTER 2
RELATED WORKS

2.1 Deepfake Generative Models

There exist a variety of methodologies to generate deepfakes. Some early pioneering works utilized autoencoders [8, 15], while other more recent works incorporate GANs (generative adversarial networks) [18]. This work focuses on deepfake videos generated using the following methods: DeepFake, FaceSwap, Face2Face, NeuralTextures, and FaceShifter. Methods can be classified as either face swapping or face reenactment (sometimes referred to as puppeteering), depending on how facial features are inserted into a target video. At a high level, face swap methods focus on extracting a face from a source image, and inserting it over the face of a target image. Facial reenactment methods, in contrast, create an internal model of a person and manipulate, or 'puppeteer' the expressions of the model. An illustration of the deepfake techniques utilized in this paper can be seen in Figure 2.1. The leftmost column, as its column name states, is the target video. The second column is the source video. For all the deepfake columns, frames are generated by applying features from the source video and inserting them into the target video. The extracted features vary depending on the manipulation method. The NeuralTextures method, for example, only extracts mouth movements from the source video, while the FaceSwap method extracts the full source face. Some artifacts can be observed from different methods. For example, the Face2Face technique tends to overcompensate its color blending, making the subject's lips a bright pink. In the last FaceSwap frame, it appears the blending on the eyes is flawed, as both the eyelid and eye of the subject can be seen overlapping over each other.

2.1.1 DeepFake

The term deepfake comes from the generation method called DeepFake [8]. The underlying functionality of this method comes from the use of image autoencoders. As seen in Figure 2.2,

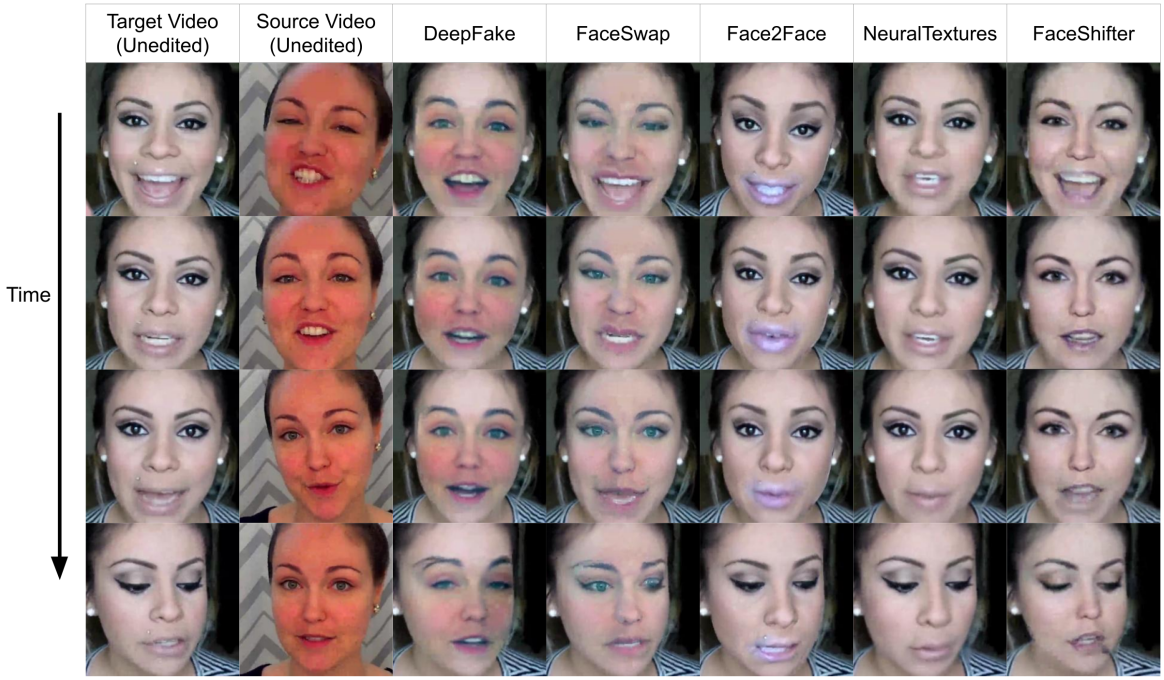

Figure 2.1: Frames from deepfake videos generated using all manipulation techniques from FaceForensics++ [23]

the autoencoder pipeline begins by extracting image features via the encoder. These features, which may include things like eyes, nose, mouth, expression, etc., are represented in an intermittent latent space. A decoder then recreates the original image using the latent space representation. A well trained autoencoder is able to compress the features into a small intermittent representation, and fully recreate the original image with little to no loss in quality. In order to generate a deepfake, two autoencoders are trained using images of a source and target person. Once trained, the output of the source encoder is sent to the target decoder, as shown in the bottom of Figure 2.2. A characteristic of face swapping methods, like DeepFake, is that they generally extract and insert faces on individual frames.

2.1.2 FaceSwap

The idea of extracting features from one image and inserting them onto another became the foundation for other deepfake generation techniques, such as the FaceSwap method

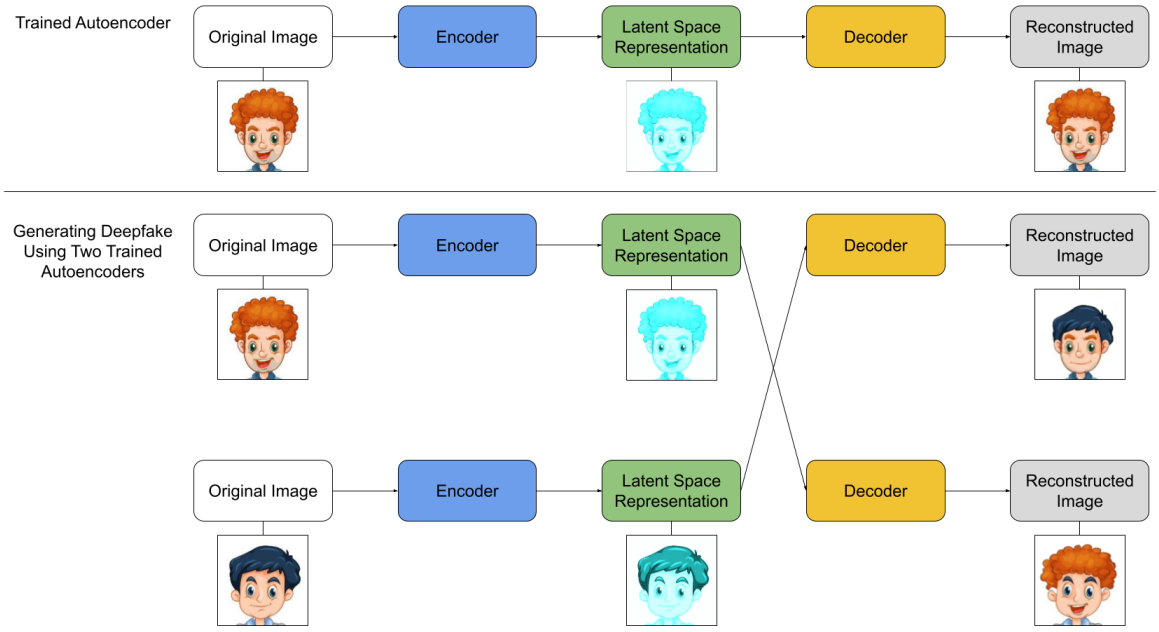

Figure 2.2: Top: Image autoencoder, Bottom: Deepfake generation using autoencoders

[15]. The FaceSwap method can be thought of as a more general autoencoder, where the encoder component maps features onto a 3d template model. The 3d template model can be thought of as a blank mask, where the main task of the encoder component is to find facial landmarks, such as mouth and eyes, and 'fit' the mask properly on the source image face. The 3d template model is then inserted and blended into the target image. In contrast to the DeepFake method, the more sophisticated encoder allows for out-of-the-box functionality, requiring little to no training to fine tune the encoder component. While this method is known for being lightweight, other methods that are fine-tuned to specific source/target pairs may produce results that are more convincing. Both FaceSwap and DeepFake are considered face swapping methods.

2.1.3 Face2Face

Face2Face, along with other facial reenactment methods, is characterized by its ability to generate whole deepfake videos as opposed to individually extracting faces from each source

frame and inserting them into the target frame [30]. The Face2Face method requires a source and target video. In similar fashion to FaceSwap, a 3d model of a face is extracted from the source video. However, the Face2Face face model is generated in the context of other frames from the source video. For example, the frames where the subject's face is angled the most towards the right and as well as the left are selected to more accurately create the side views on the 3d model. Illumination, pose, and expression are also identified and extracted in both the target and source videos. Finally, the 3d face model is inserted in the target video using the illumination, pose, and expressions of either target or source video as reference, along with blending.

2.1.4 NeuralTextures

The NeuralTextures method can be used to insert expressions from a source video to a target video, but as is implemented by the authors, does not extract and insert faces [29]. NeuralTextures, nonetheless, is classified as a facial reenactment method, with an architecture based on two trained components: a neural texture and a deferred neural renderer. A neural texture can essentially be thought of as encoded data for a specific image. In the context of deepfakes, a neural texture contains high level feature maps of the subject, conceptually similar to the 3d face model from the Face2Face method. As per the authors, during training time, features can be extracted effectively even from noisy videos. Given the neural texture from a target video and the expressions from a source video, a trained neural renderer is able to reenact the the expressions and mouth movements onto the target video's subject.

2.1.5 FaceShifter

FaceShifter is a face swapping method that utilizes the power of GANs at its core [18]. To summarize the contributions of the authors, FaceShifter utilizes a multi-stage pipeline. To generate a deepfake, an encoder extracts features from source and target images, which can then be forwarded to a trained generator. The output of the generator is finally processed through a component named the Heuristic Error Acknowledging Refinement Network

(HEAR-net), which as the authors state, addresses anomalies that may exist in the images relating to occlusion. Occlusion in this context refers to instances where a subject's face may be partially covered, such as by glasses or a veil. An advantage of FaceShifter is that little to no training is required to create deepfakes, similar to FaceSwap. However, the more complex architecture makes it unsuitable for real-time face swapping.

2.2 Knowledge Distillation

Knowledge distillation (KD) at its essence involves passing knowledge from one model to another. An early pioneering work of KD, proposed by Hinton et al. [13], proposed to train a student model by transferring knowledge from a larger, cumbersome teacher model, a process analogous to that of a human teacher teaching a student. As can be seen in Figure 2.3, the KD pipeline begins by inputting an image x to both the teacher and student models. Both models produce an output known as logits, denoted by z, which can be used to calculate a class probability distribution by applying a softmax function to these logits. Under KD, a temperature parameter is added to the softmax function (see Equation 2.1, where T is the temperature value set to a static value t during training time for both student and teacher, z_i is the i^{th} logit, and q_i is the probability of the i^{th} class), which can generate 'softer' probability distributions. 'Softer' probability distributions amplify probabilities from less likely classes.

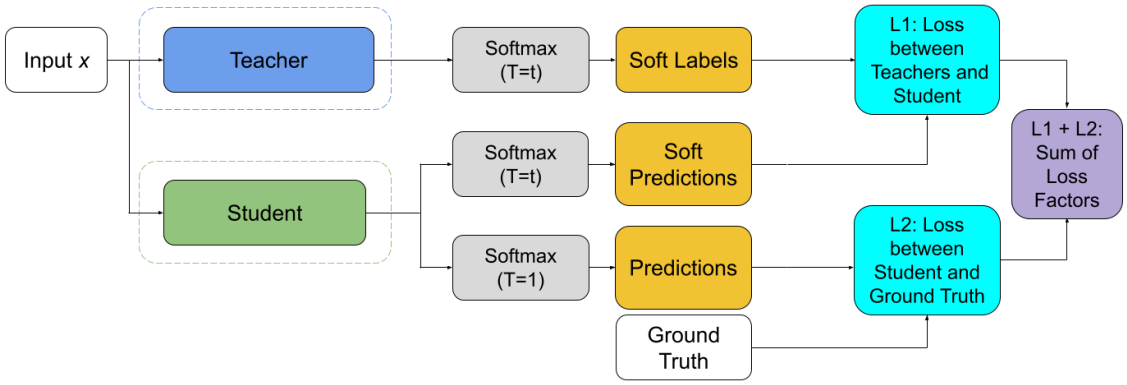

Figure 2.3: Knowledge distillation with a single teacher

$$q_i = \frac{exp\,(z_i/T)}{\sum_j exp\,(z_j/T)} \tag{2.1}$$

Continuing down the pipeline, the outputs from the softmax functions are classified as either a soft label or a soft prediction. A more traditional binary classification label takes either the value 0 or 1. Soft labels, in contrast, may be any value from $[0, 1]$, inclusive. The output from the student model is considered as a soft prediction. Using the soft labels from the teacher model as a target, loss between them and the student's soft predictions may be calculated, generating loss factor one, denoted as $L1$. Loss factor two ($L2$) on the other hand, is calculated using the ground truth as a target, along with the student's predictions. At the end of the pipeline, the two loss factors are combined to generate a final loss value.

2.2.1 Soft Labels

The utility of 'softer' class probability distributions comes from the amplification of less likely classes. To highlight their utility, consider the following probability distributions for four classes:

$$\begin{bmatrix} .01 & .02 & .92 & .05 \end{bmatrix} \begin{bmatrix} .02 & .03 & .80 & .15 \end{bmatrix}$$

In this example, consider the left hand side to be a 'regular' probability distribution, while that on the right hand side is a 'soft' distribution. A key concept of KD is that classification information relating to incorrect classes is valuable during the training process. Although both distributions above would indicate that the third class is the prediction, the right hand distribution shows that the fourth class is significantly more likely than classes one and two. Given a picture of a car, a 'truck' class resembles the true label 'car' far more than a 'dog' or 'apple' class. Hinton et al. [13] state that soft labels contain more information than hard targets for training a model. As a result, the student model can often be trained using less data than the teacher model, and also utilize a higher learning rate, which can lead to faster convergence.

CHAPTER 3
METHODOLOGY

3.1 KD using Heterogeneous Teachers

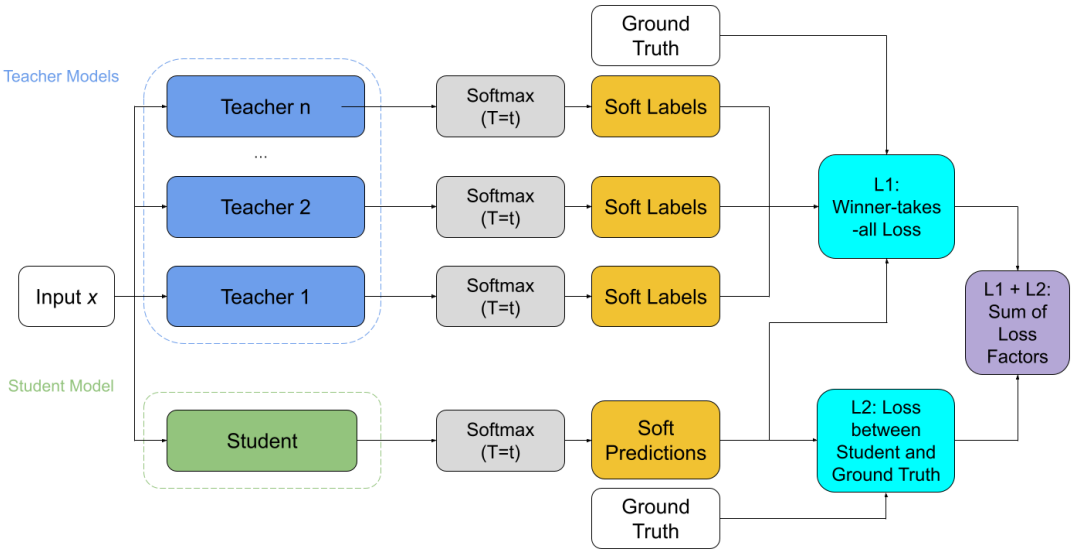

Figure 3.1: Knowledge distillation with multiple teachers

To review, a high level motivation of this project was to train a neural network in a more human-like manner, such as the case where a human student learns from various teachers. This situation is analogous to training a student model with multiple heterogeneous teacher models via KD. Research on KD with multiple teachers is a subject that has been explored and shown to be effective [13, 25, 11, 22]. Likewise, some research has been conducted on applying KD to the deepfake detection problem with promising results [17, 14]. However, to the best of my knowledge, no literature exists that addresses deepfake detection and also incorporates KD using heterogeneous teachers. As such, I propose to use the pipeline shown in Figure 3.1 as a basis to incorporate KD using heterogeneous teachers to the deepfake detection problem. The diagram shares many similarities with Figure 2.3, but with some modifications. Specifically, the pipeline has been generalized such that it can include n

teachers. In a similar manner to KD using a single teacher, all teachers and students set their temperature T value to the same static value t. However, due to the inclusion of more teachers, loss factor one, $L1$, is now calculated using what I deem the Winner-takes-all method. In the last stage of the pipeline, the final loss value is generated by adding loss factors one and two.

3.2 Distill Knowledge from Multiple Teachers

Several methodologies have been proposed to most effectively distill knowledge from multiple teachers over the years. Some methods use the teacher logits, much like in Hinton et al.'s work [13]. Others distill knowledge through intermittent features, which in the case of convolutional neural networks (CNNs), almost always refers to activation patterns found around the convolutional layers. Generally speaking, methods that use teacher logits are computationally lighter than those that use intermittent features. This is an important constraint if computational resources are limited, which is the case for this work. However, using intermittent features to distill knowledge can be more effective at passing knowledge from teacher(s) to student. A brief summary of different distillation algorithms follows, providing context for the need to create and utilize the Winner-takes-all method.

3.2.1 Logit Average and Noisy Logits

A simple use of logits was explored in [25], where the logits of multiple teachers are averaged and input through the softmax function with a temperature parameter to create soft targets. Rather than simply averaging, however, the authors determined that the variance between teacher logits could be utilized for regularization. To illustrate this concept, consider the following example: the averaged logits produced a probability for a binary classification problem of 0.8, while the variance in logits could produce a 0.1 change in probability. During training time, a soft target value of 0.8 ± 0.1 would randomly be selected. Selecting a random value is the equivalent of adding noise to the soft targets, and by extension, the loss, which

should have a similar effect as L2 regularization, as per the authors. This methodology was evaluated under the pre-requisite that all teacher models were homogeneous experts. In contrast, however, the teacher models in this work are heterogeneous. Heterogeneous teachers may have drastically different predictions for a given input, resulting in a large variance. A variance that is too large may be detrimental to the training of the student model.

3.2.2 Logits per Mini-batch

Another strategy that utilizes logits was proposed in [11]. Rather than utilizing all teacher logits to create soft targets during training, only a single teacher's logits would be used at a time. However, during training time, a new teacher would be randomly selected after every mini-batch, which is defined as a pre-defined fraction of a full batch. Much like the previous method, however, it is most effective when all teachers are experts in the field, which makes it ineffective for this work.

3.2.3 Intermittent Features

One of the strategies explored in [22] utilizes intermittent features as opposed to logits as a source of knowledge for the student. The method begins by selecting a student architecture that is exactly the same as the teacher. During training time, at an arbitrary convolutional layer l_s, the student modifies its weights such that the activation of its own layer more closely resemble the activation of the corresponding teacher layer, l_t. A similar strategy can be used for multiple teachers, so long as all teachers have the same architecture as the student. In the multiple teacher case, the student's layer l_s attempts to match the activation patterns of all teachers at their corresponding layer, l_t.

3.2.4 Auxiliary Network and Intermittent Features

The authors of [22] also proposed a strategy to work around this constraint that would allow for the student to learn from heterogeneous teachers, both in terms of knowledge and

architecture. The primary idea is to use what the authors refer to as an auxiliary network, which acts as a bridge between a teacher and the student. The process begins by training an auxiliary network for each teacher network, where the architecture of each auxiliary network matches that of the student model. Since each auxiliary network has a single teacher, they can be trained utilizing KD and logits, or any strategy found to be most effective. Finally, given n auxiliary networks with the same architecture as the student, the student is able to learn the teacher's intermittent features, as described above. This method in theory is appropriate for this work's needs, but requires large computational resources, rendering it inadequate.

In summary, current methods to distill knowledge from multiple teachers to a single student have been shown to be effective, but due to different requirements, fail to be a viable choice for this work. The need for an appropriate technique motivates the creation of what I deem the Winner-takes-all method.

3.2.5 Winner-Takes-All Method

The Winner-takes-all method was created in order to address all constraints of this work. First, this work utilizes heterogeneous teachers by design. The teachers are heterogeneous in knowledge, where each is an expert in a subset of the overall domain of knowledge. Additionally, the teacher architectures are also heterogeneous. Limited computational resources forms the last constraint. Methods that use intermittent features to distill knowlege tend to be far more computationally intensive than those using logits, which essentially necessitates the use of logits for this work.

As shown in Figure 3.1, the Winner-takes-all loss is used to generate loss factor one, $L1$. The method takes all teachers' soft targets as input, in addition to the student's soft predictions and the ground truth. During training time, only a single teacher's soft targets are actually used to calculate loss, hence the name. The 'winner' is the teacher with the probability most similar to the ground truth, which can be found by using Equation 3.1, where the winner's soft targets are denoted as $\hat{y}_{best_i}$, the ground truth as y_i, and each teacher's

soft targets as $\hat{y}_{TeacherN_i}$ for some input i. All teacher soft targets are compared to the ground truth, where the teacher with the lowest value in the set is selected as the winner. The winner's soft target returned by the 'MinTeacher' function and the student's soft prediction, denoted by $\hat{y}_i$, are used to calculated the log loss between the two to generate $L1$ for an input i, as shown in Equation 3.2.

$$\hat{y}_{best_i} = MinTeacher(|y_i - \hat{y}_{Teacher1_i}|,\ |y_i - \hat{y}_{Teacher2_i}|,\ ...,\ |y_i - \hat{y}_{TeacherN_i}|) \tag{3.1}$$

$$L1_i = -(\hat{y}_{best_i}) \ln(\hat{y}_i) - (1 - \hat{y}_{best_i}) \ln(1 - \hat{y}_i) \tag{3.2}$$

In essence, during training time, the Winner-takes-all method eliminates contributions from teachers that are more likely to contribute incorrect or less accurate information to the student. More specifically, by selecting heterogeneous experts to be teachers, it is likely that all but one of the teachers are incorrect. The goal of the Winner-takes-all method is to learn deepfake detection techniques from all teachers. For example, one teacher may focus on identifying features on the mouth region, while another focuses on the eye region. The distilled student model would in turn learn to search both the mouth and eye regions for deepfake artifacts.

CHAPTER 4
EXPERIMENT AND RESULTS

4.1 Data

The FaceForensics++ [23] dataset is a common test benchmark to evaluate deepfake detection methods, including [14, 21, 19, 6]. It contains a large amount of deepfake videos generated using the DeepFake, Face2Face, FaceSwap, FaceShifter, and NeuralTextures methods, as well as unedited videos, labeled as 'Pristine' by the authors. Note that FaceShifter was not originally part of FaceForensics++. It was added a few years after being published, possibly motivated by a surge in GAN-based deepfake generation techniques. For this reason, the FaceShifter method was not utilized throughout the training and selection process for teacher models. It was, however, utilized as a test set for the student model.

One reason for the widespread use of FaceForensics++ is that it provides a highly controlled dataset. Essentially, across all manipulation techniques, each target video uses the same corresponding source video. To better illustrate this property of the dataset, consider the DeepFake video labeled '001_870'. In the dataset, there are 1000 unedited videos, each with an index from 0-999. DeepFake video '001_870' is generated by using pristine (unedited) video 001 as a target and pristine video 870 as the source for features. Face2Face video '001_870' is generated using the same target and source videos. The trend continues with the other manipulation techniques. Note that all videos are available at various compression levels, but only the low quality videos (c40 compression) are utilized in this work (6000 videos). See Table 4.1 for a breakdown of the used data.

Manipulation Technique					
DeepFake	Face2Face	FaceSwap	FaceShifter	NeuralTextures	Pristine
1000	1000	1000	1000	1000	1000

Table 4.1: Summary of videos per manipulation technique in FaceForensics++ low quality dataset (c40 compression)

In this work, for all models, every 11^{th} frame from a video was selected both training trime, as well as inference. The videos varied in length from 8-30 seconds, so the number of frames per video, in turn, also varied. In terms of data splits, 40% of the videos were used for testing, 42% for training, 9% for validation, and the remaining 9% was used as a secondary test set to evaluate the performance of the teacher models. Note that these splits occur at the video level, rather than at a per frame level.

4.2 Model Architectures

The selection of teacher models was based on the objective of creating heterogeneous teachers, both in terms of knowledge and architecture. In the context of image classification, the architecture of a model can influence the features it learns, so the architecture choice would in turn also impact the knowledge of the teacher. The FaceForensics++ [23] authors created and evaluated several deepfake detectors utilizing several models, which helped guide the selection process. Further research on their models, along with similar resources also helped guide the selection process. As can be seen in Figure 3.1, the KD pipeline in theory should work for any number of teachers. The FaceForensics++ dataset contains data generated using five different manipulation techniques. For this work, given that there are five manipulation techniques included in FaceForensics++, the number of teachers is set to three. Setting the number to three allows for the performance of knowledge transfer to be verified, while also leaving two unseen manipulation methods available to be used as a test for generalization on data generated using unseen techniques. Ultimately, the three selected teacher models for the experiment were ResNet, XceptionNet, and MesoInceptionNet (sometimes shortened to MesoNet). The strengths and weaknesses of the models are discussed in the following sub-sections.

4.2.1 ResNet

ResNet [12] is a common CNN architecture available in both Pytorch and TensorFlow. Crucial to its functionality are its stacks of residual blocks. A residual block is composed of several convolutional layers and a component referred to as a skip connection. Skip connections enable the network to skip internal layers. In practice and by design, ResNet models tend to skip over layers that focus on information that hurts the performance on the model. As such, the network has a tendency for learning macro level details, and sometimes disregarding micro level details that are not immediately useful. A drawback, however, is that these micro level details may have *eventually* become integral to the performance of the model. See Table 4.2 for brief overview of the model.

4.2.2 XceptionNet

XceptionNet [5] is another popular CNN architecture, with the original paper having over 10,000 citations. The main components of XceptionNet are pointwise and depthwise separable convolutions, in addition to more traditional convolutional layers. XceptionNet is a good general network, although as per the authors, due to its pointwise convolutional layers, the model should be able to learn micro level details more effectively than ResNet. See Table 4.2 for a brief overview of the model.

4.2.3 MesoInceptionNet

MesoInceptionNet [1] can be thought of as a middle ground between XceptionNet and ResNet. This network's design has far fewer layers and parameters than the other two models. However, this is by the author's design, as the low number of layers in conjunction with the kernel sizes in the convolutional layers allow the network to focus on what is referred to as mesoscopic details. Mesoscopic details are defined by the authors as details smaller than macro level, but larger than micro level ones. This architecture was designed specifically to detect deepfakes, with surprisingly good results considering its shallow nature.

See Table 4.2 for brief overview of the model.

Although the three model architectures will remain black boxes after training, insight from the authors allow users to infer what each architecture may learn. In summary, ResNet is potentially more adept at detecting macro level details, XceptionNet for detecting micro level details, and MesoNet for mesoscopic level details.

	ResNet50	XceptionNet	MesoInceptionNet-4
# of Parameters	25,557,032	22,855,952	28,525
# of Layers	50	36	4
Specialization	Macro level details	Micro level details	Mesoscopic level details
Training Data	Face2Face	NeuralTextures	FaceSwap

Table 4.2: Overview of teacher models. Bottom row refers to manipulation technique with which each teacher was trained.

4.3 Teacher Models

As seen in Table 4.1, the ResNet [12], XceptionNet [5], and MesoInceptionNet [1] teacher models were trained using the Face2Face, NeuralTextures, and FaceSwap datasets, respectively. The selection of teacher models was based on an evaluation of each architecture trained on each manipulation technique, as well as insight provided by the authors of FaceForensics++ [23] and of the model architectures. A detailed discussion on the training and selection process follows.

4.3.1 Training Candidate Models

Given the need to create heterogeneous teacher models, the training process for each candidate teacher was as follows: Each candidate model was trained using data from a single manipulation technique. For example, a ResNet50 model was trained and validated using only the DeepFake and unedited data, a ResNet50 model using Face2Face and unedited data, and so forth. For evaluating the models, however, all available manipulation techniques

were used, such that in the example Face2Face ResNet50 model, data from all manipulation techniques was utilized for testing.

Both the ResNet and XceptionNet models were pre-trained using the ImageNet dataset [24]. Variations of these models were trained using randomly initialized weights, but were not utilized as they were outperformed by the pre-trained models. The MesoInceptionNet models used randomly intialized weights. All candidate teacher models were trained using a grid search for hyperparameter optimization, which included: learning rate, momentum, weight decay, batch size, and temperature. The FaceForensics++ authors provided implementation details for their own models, including hyperparameter values which aided in narrowing the grid search, as computational resources for this work were limited.

4.3.2 Evaluating Teachers

A summary of the evaluation results for all candidate teacher models is shown in Figure 4.1. An interesting result is that the MesoInceptionNet model, despite having an architecture that was specialized for the task of deepfake detection, performed worse than the other two more general models in terms of expertise. Expertise in this context means performance on the same manipulation technique with which the model was trained. For example, the DeepFake MesoInceptionNet model scored lower than 80% in terms of accuracy, while its two counterpart models scored higher than 90%. It can be said, however, that the MesoInception models learned more general information on deepfake detection, at least in comparison to the ResNet and XceptionNet models. Consider the FaceSwap MesoInceptionNet model, where although it performed worse than its counterpart models on FaceSwap data, its accuracy was higher on all other manipulation techniques. Evaluation results helped guide the model selection process, which is discussed in the following section.

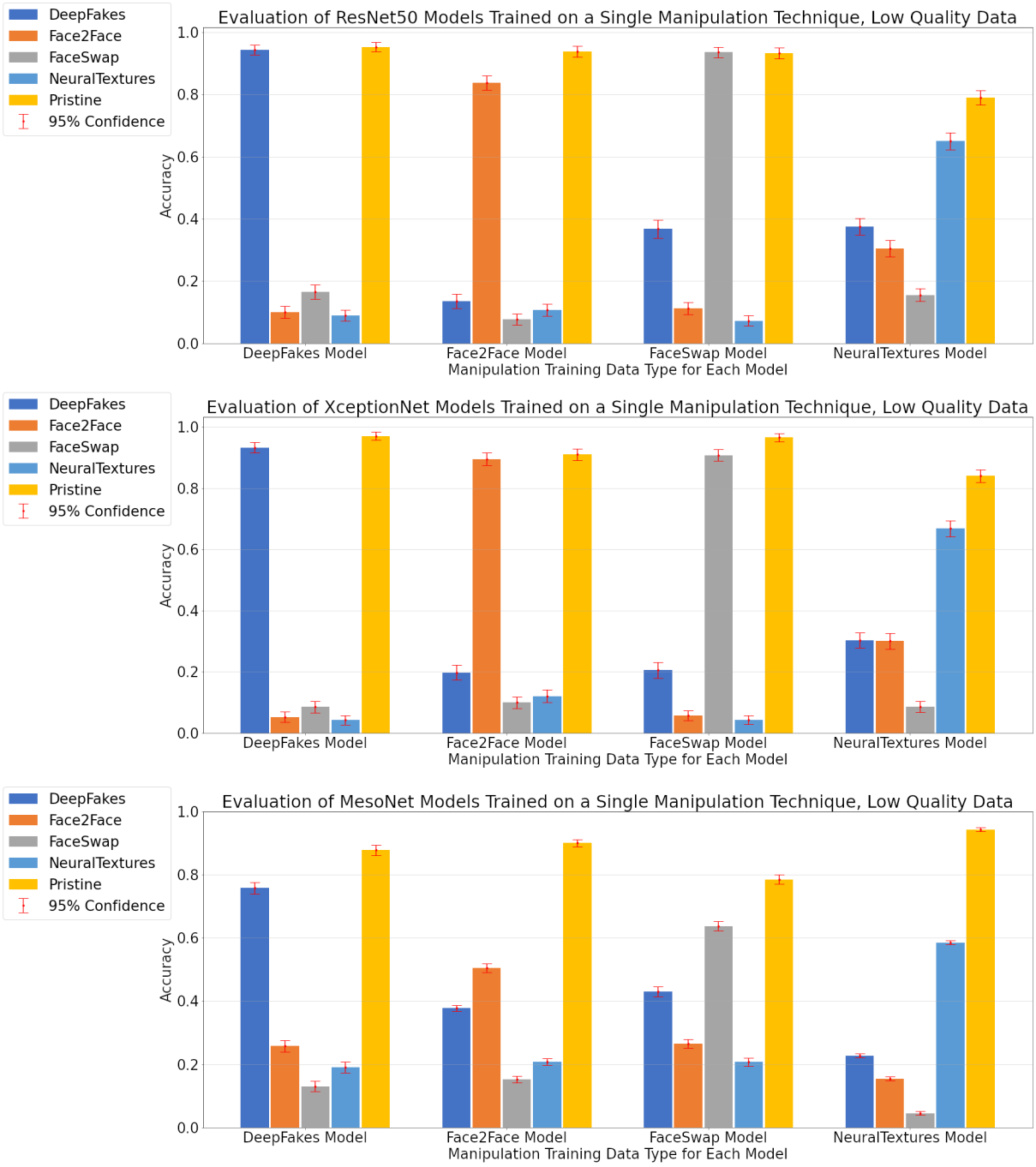

Figure 4.1: Evaluation of Candidate Teacher Models

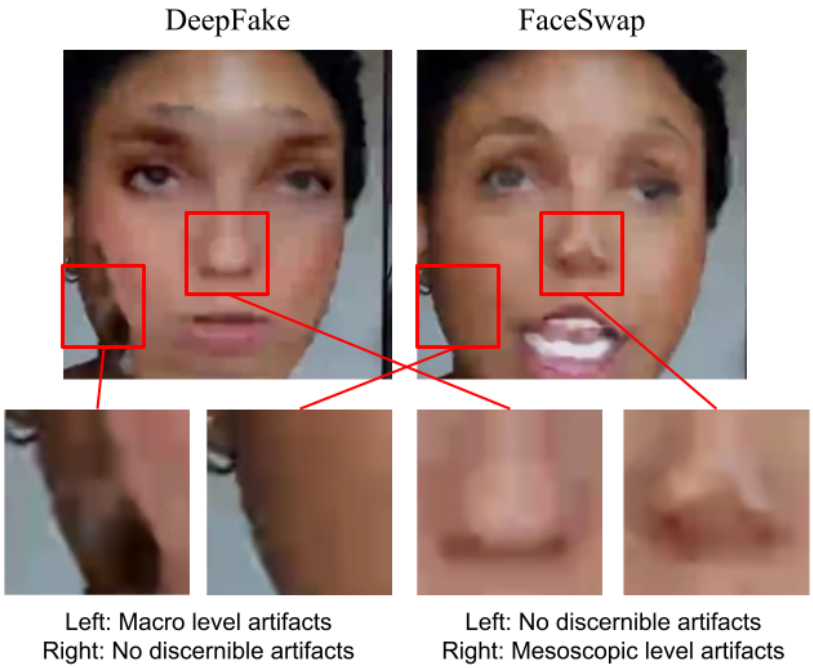

Figure 4.2: Analysis of artifacts produced by different generation techniques

4.3.3 Candidate Model Selection

MesoInceptionNet Model

Selecting the most suitable MesoInceptionNet candidate required a complete analysis of model architectures and the candidate evaluations. Ultimately, the FaceSwap candidate was selected as the most suitable teacher model. The MesoInceptionNet candidates, as mentioned above, were worse experts than their counterparts, but better general learners. Furthermore, as discussed earlier, the architecture was designed to learn mesoscopic details. The learned mesoscopic details may in part explain the generalized learning, but without further research, can not be verified. Rather than selecting the best expert model, which was the DeepFake candidate, the FaceSwap candidate was chosen, partly to take advantage of the additional general knowledge. Another significant reason to avoid selecting the DeepFake model was that deepfakes generated using this technique had a tendency to exhibit macro-level details. As seen in Figure 4.2, the DeepFake image on the left is poorly blended, leaving behind macro level artifacts. The FaceSwap image on the right, in contrast, has a smaller

artifact on the tip of the woman's nose. A goal of this work is to train a student model with heterogeneous teachers, and by extension, teach it heterogeneous knowledge. As such, the primary motivation for selecting the FaceSwap model is that it is most suitable for teaching the student mesoscopic feature information.

XceptionNet

When only considering the candidate's performance, selecting the best expert may seem to be the best choice. Selecting the best expert is a bit arbitrary, as corresponding XceptionNet and ResNet candidate models have similar performance. However, in the full context of the work, the NeuralTextures XceptionNet candidate is the most suitable. As discussed previously, the NeuralTextures method focuses on transferring mouth movements from a source to target video. Thus, the available area for analysis is relatively small in comparison to the full face of a subject. As such, XceptionNet's architecture, which in theory performs better at detecting micro level details than the other two model architectures, is best suited for this task.

ResNet

Finally, the two remaining choices for ResNet were Face2Face and DeepFake. Both candidates were relatively good experts, with their accuracy exceeding 80%. Ultimately, the Face2Face candidate was selected. According to a study conducted as part of FaceForensics++ [23], humans were able to identify low quality DeepFake videos at an accuracy of about 75%, but could only identify Face2Face videos at a rate of 40%. This discrepancy likely occurs due to the quality of the generative methods. Recall that DeepFake is one of the oldest mainstream generative methods. More recent methods, like Face2Face, would naturally improve over past methods. As such, the Face2Face candidate was selected, as features learned from this manipulation technique may help the student model generalize better on other new methods.

4.4 Student and Traditional Models

4.4.1 Student Architecture

With the teacher models trained, the next step is to create a student model. The following are some conventional strategies for selecting a student architecture for the KD process.

1. The student model has the exact same architecture as its teacher. If compression is not desired, then selecting the same architecture may be beneficial, as it simplifies the process of learning intermittent features [22].

2. The student architecture is based on one of the teacher architectures, but simplified in some way. The process for creating a simpler architecture is case dependent, but utilizing a shallower variation of the teacher architecture works well [20]. For example, ResNet50 is a more shallow variation of ResNet101.

3. A good general architecture can be a good choice if there are several different teacher architectures, but the ability to find a generalist varies largely by domain [27]. For image classification tasks, there are several convolutional neural networks that are suitable candidates, including ResNet and XceptionNet.

Inspired by strategies 2 and 3 from above, the selected architectures were ResNet34, ResNet18, and ResNet14. ResNet is a good general model, and given that the ResNet teacher model utilizes the ResNet50 variation, utilizing the 34, 18, and 14 variations abides by strategy 2 from above. The justification for utilizing these exact variations for the student models is that ResNet is a popular architecture, with these variations being built into the machine learning framework, PyTorch. Additionally, training multiple student models at various compression levels is necessary to address **RQ3**. A summary of the parameters of the ResNet variations is shown in Table 4.3.

ResNet50	25,557,032
ResNet34	21,797,672 = 85.3% of ResNet50
ResNet18	11,689,512 = 45.7% of ResNet50, 52.6% of ResNet34
ResNet14	8,029,634 = 31.4% of ResNet50, 68.6% of ResNet18

Table 4.3: Number of parameters of ResNet variations

4.4.2 Training the Students

The student models were trained according to the pipeline in Figure 3.1. The data used to train the student models is the union of the data used to train the teacher models, or in summary, the Face2Face, FaceSwap, NeuralTextures, and Pristine training videos. The data was shuffled, such that the students could encounter any of the listed manipulation techniques during training time. A thorough grid search was used for hyperparameter optimization, which included the learning rate, momentum, weight decay, batch size, and temperature. For the temperature value, in all cases, a value of 2 was found to be the most optimal. Another hyperparameter added to the grid search was the rate of contribution, R, from the loss factors in proportion to each other. For example, when $R = 2$, it indicates that the total loss is $(L1 \times 2) + L2$, while setting $R = 1/2$ sets the total loss equal to $L1 + (2 \times L2)$. A discussion on the evaluation of the student models follows in the next few sections.

The training dataset contained three times as many manipulated videos as unedited ones (Pristine). To address the imbalance, the Pristine dataset was upsampled. Specifically, the data was sampled at a higher rate of 3, which is the equivalent of replicating the videos by a factor of 3. However, to mitigate the effects of overfitting, during training time, a series of random transformations were applied to each Pristine video frame: rotation, horizontal flip, and zoom-in/zoom-out.

4.4.3 Traditional Models

In order to directly address **RQ1**, three baseline models were created. The baseline models can be referred to as 'Traditional', due to the manner in which they are trained. The three traditional models were created using the same ResNet architectures as the student

models. Also, like the student models, the traditional models were trained on the same Face2Face, FaceSwap, NeuralTextures, and Pristine training videos, using the same upsampling method for the Pristine videos. The primary difference is that the traditional models are trained without teacher models, using the labeled data directly. A grid search was used for hyperparameter optimization. In short, the training conditions for the traditional and student models were the same, with the only difference being that all student models utilized KD from heterogeneous teachers. As a result, the direct effects of KD can be more easily observed.

4.5 Results

4.5.1 Generalization

Evaluating whether a student model trained using heterogeneous teachers via KD could generalize well was the focus of **RQ1**. To this end, both the student and traditional models were evaluated for accuracy using test data from all available methods: DeepFake, Face2Face, FaceShifter, FaceSwap, NeuralTextures, as well as the Pristine dataset. Another way to view the test data is by grouping the manipulation techniques into a seen or unseen group. All models were trained using data generated using Face2Face, FaceSwap, and the NeuralTextures methods. As such, the 'seen' group consists of data generated by these methods. The 'unseen' group consists of data from the FaceShifter and DeepFake manipulation methods. The results of all tests can be seen in Figure 4.3, Figure 4.4, and Figure 4.5. For all result figures in this chapter, 95% confidence intervals are shown. These intervals are generated using the Pytorch extension for statistical analysis, Torchmetrics. The confidence intervals are generated using 1000 bootstraps, with each bootstrap scored based on the mean squared error.

In regards to 'seen' methods, the student models outperformed the traditional models at the ResNet34 and ResNet18 levels. As an example, the ResNet34 student had accuracy gains of more than 10% for the Face2Face and NeuralTextures methods, in comparison to the

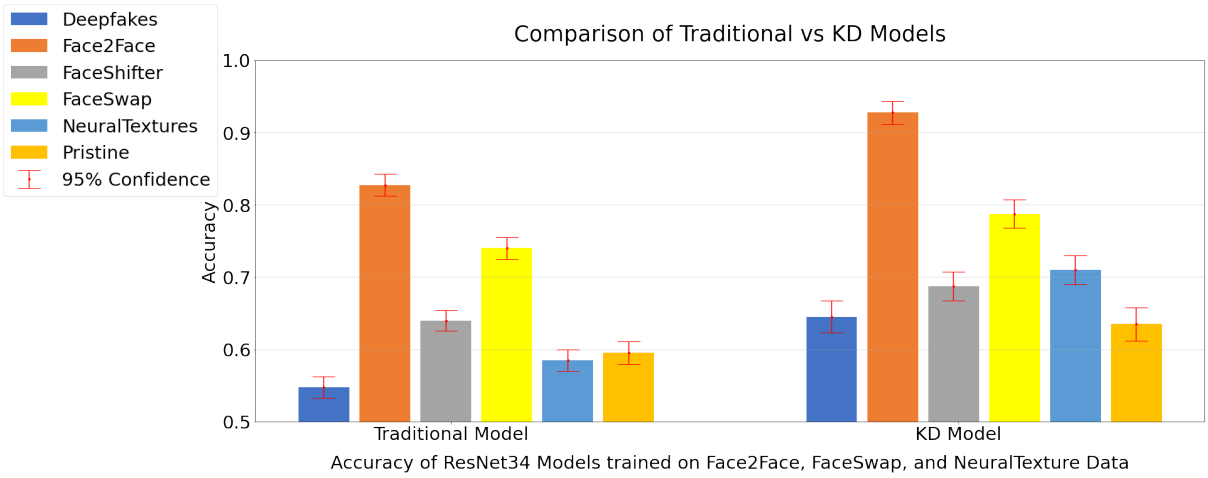

Figure 4.3: Accuracy Comparison of Traditional vs KD Model ResNet34

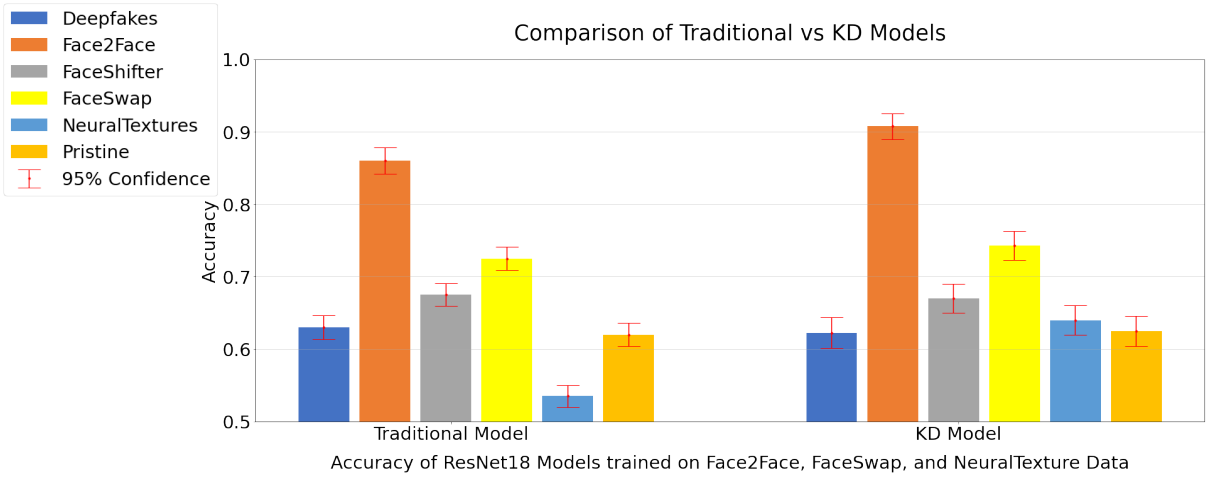

Figure 4.4: Accuracy Comparison of Traditional vs KD Model ResNet18

traditional model. At the ResNet14 level, however, both models peformed poorly, with the only significant difference in performance occuring on the NeuralTextures data, where the student model performed almost 10% better than the traditional model. Grouping together the 'seen' methods, these trends become more visible, as can be seen in Figure 4.6.

In terms of 'unseen' methods, the impact of KD appears to become diminished as the model compression rate increases. A performance gain of 10% occurs at the ResNet34 level on the DeepFake data. However, at the ResNet14 level, the opposite occurs, with the traditional model outperforming the student model by more than 10% on FaceShifter data. Figure 4.7

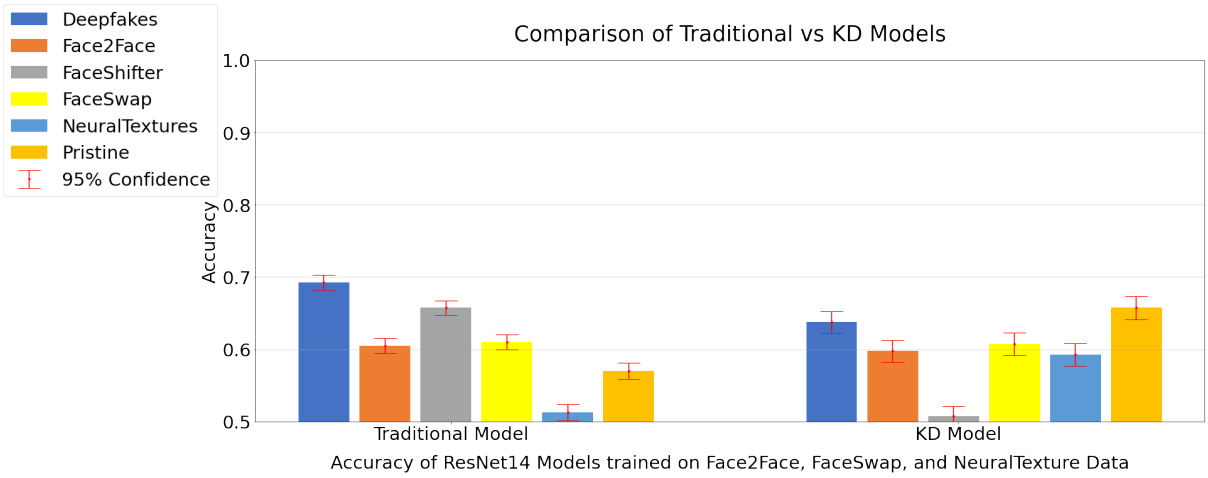

Figure 4.5: Accuracy Comparison of Traditional vs KD Model ResNet14

shows the 'unseen' methods grouped together for a higher level view.

An evaluation of the Pristine data is shown in Figure 4.8. In general, the student models outperformed the traditional models, with the students maintaining a baseline performance of at least 0.6. In contrast to the other evaluations, the specificity did not seem to be correlated to the model compression.

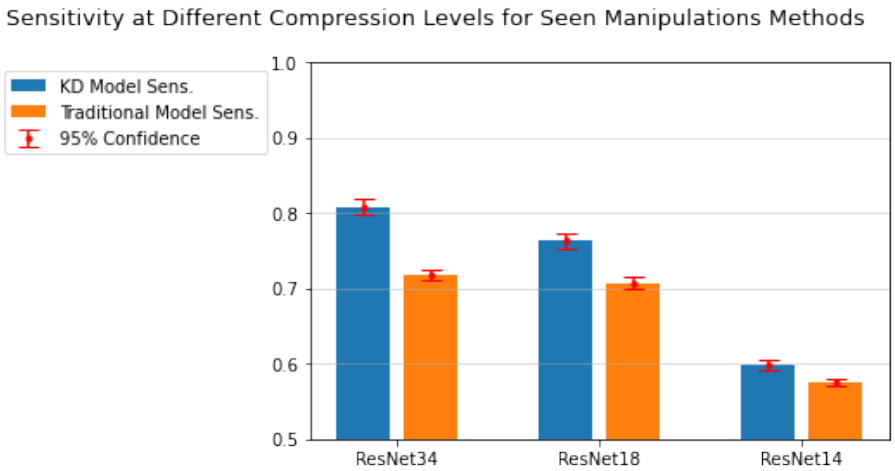

Figure 4.6: Sensitivity Across Different Compression Levels, Seen Methods

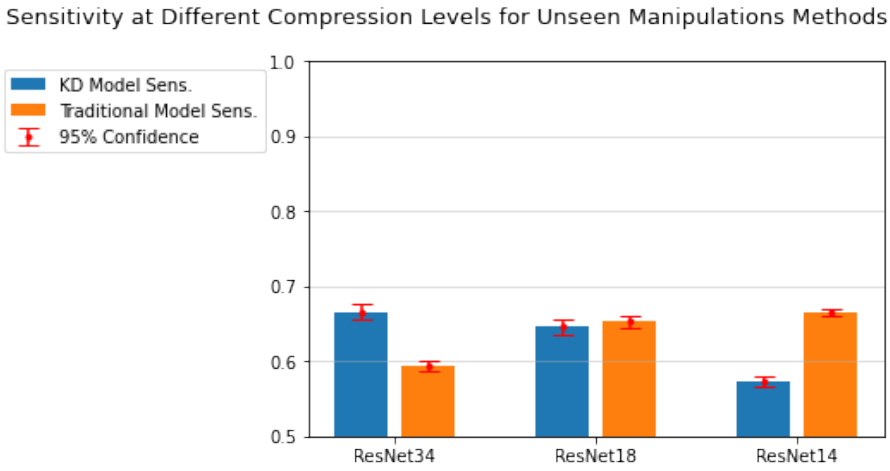

Figure 4.7: Sensitivity Across Different Compression Levels, Unseen Methods

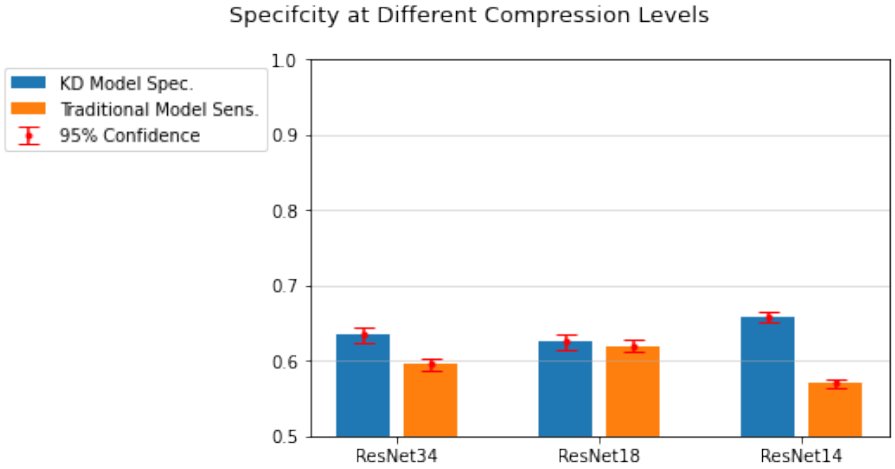

Figure 4.8: Specificity Across Different Compression Levels

4.5.2 Knowledge Transfer

To understand the definition of knowledge transfer, as it pertains to **RQ2**, consider the human student and expert teachers example from before. How are students typically evaluated? A teacher would likely test a student on material covered in their class. Assuming the student learned all strategies from their teacher, they should have the capability to solve test problems on similar material. In this case, the strategy is to give an identical 'test' to both student and teacher, and compare their performance. If the teacher is not capable of

answering a question correctly, then it is assumed they did not pass the capability to solve the problem to the student either. Stated as a conditional probability statement, it can be asked "What is the probability that the student correctly identifies a deepfake, given that the teacher identified it correctly?"

	ResNet Face2Face	XceptionNet NeuralTextures	MesoInceptionNet FaceSwap
ResNet34	.964	.793	.850
ResNet18	.955	.720	.827
ResNet14	.633	.668	.688

Table 4.4: Probability that a student model identified a deepfake video correctly, given that the teacher model also identified it correctly.

	ResNet Face2Face	XceptionNet NeuralTextures	MesoInceptionNet FaceSwap
ResNet34	.870	.638	.809
ResNet18	.903	.591	.8
ResNet14	.620	.524	.681

Table 4.5: Probability that a traditional model identified a deepfake video correctly, given that the teacher model also identified it correctly.

The results of this analysis can be seen in Table 4.4. A perfect score of 1.0 would indicate that the student successfully detected all the deepfakes that the teacher detected in the corresponding category. The top row indicates the teacher model, along with the 'test' given to both teacher and student models. For example, the ResNet34 student identified 85% of deepfakes that its teacher, the MesoInceptionNet model, identified. For the sake of comparison, the 'test' was also given to the traditional models. The results can be seen in Table 4.5. Across all values, the traditional models performed worse than the student models.

4.5.3 Compression

The previous tests also serve as a proxy to address **RQ3**. In the generalization tests, the ResNet34 and ResNet18 models have somewhat similar performance, giving evidence that

the depth and parameters are not a substantial bottleneck until reaching the ResNet14 compression level. A similar trend can be observed in the knowledge transfer test, where performance is maintained at the ResNet34 and Resnet18 compression levels, but also falls at the ResNet14 level.

CHAPTER 5
DISCUSSION AND CONCLUSIONS

5.1 Discussion

RQ1

The results from Figure 4.3 and Figure 4.4 suggest that KD produces a more generalized student model, so long as the student model is not too shallow. The ResNet34 student model achieved better generalization than the baseline traditional model for both seen and unseen data. As an extension of **RQ1**, is there evidence that KD could help generalize the knowledge from *any* heterogeneous teachers? Performing the experiment utilizing different teacher candidate models could help answer the question. As it stands, however, the answer is difficult to determine, as the conditions for this set of teachers were highly optimized. The teacher models were selected based on insight from other research, architecture knowledge, and an understanding of the deepfake generative models. Utilizing a selection of arbitrary teachers would likely diminish the performance of the student models, but may potentially increase performance in comparison to more traditional training methods that do not utilize KD.

RQ2

The results from Table 4.4 and Table 4.5 indicate that knowledge was successfully transferred from the teachers to the student model, given that the student model is not too shallow. The traditional models were trained independently from the teacher models, so its results on this test function as a baseline. Comparing the two results, it can be seen that the benefit of utilizing KD varies, but generally has moderate results in the range of 5-15% over the traditional models.

A challenge in the knowledge distillation process is that detection techniques learned from one teacher may affect learning techniques from another. In this work, the XceptionNet

teacher likely guided the student to focus on micro level details. In contrast, the ResNet teacher more likely guided the student to learn macro level details. Ideally, a student would learn and retain all knowledge. However, the mixed guidance may also inhibit the distillation process from all teachers to some extent. In short, a potential reason that the student models did not retain all knowledge from the teachers is that the knowledge itself was contradictory to some extent.

RQ3

A common trend across all tests is that the ResNet34 and ResNet18 student models perform somewhat similarly, with ResNet14 student typically having far worse performance. This trend is not exclusive to models trained using KD, as the traditional models also drop in performance at the same compression level, as can be seen in Figure 4.5 and Table 4.5. It can be inferred that ResNet14 architecture is too shallow to effectively learn to identify deepfakes. Recall the number of parameters in the models, as shown in Table 4.3. This study reinforces the results from Hinton et al. that KD is useful for model compression [13]. ResNet18, which contains less than half the number of parameters as the teacher architecture, ResNet50, is able to perform almost as well as the larger ResNet variations.

5.2 Conclusions

From a high level perspective, is it beneficial to perform the KD procedure from this work, when considering the necessary resources? If teacher models are available, the added cost of using the teachers for KD may be outweighed by the moderate performance gains. However, if teacher models are not available, then the cost of training expert teachers may be too high, especially if computational resources are limited. Note that this conclusion is in regards to the KD procedure from this work.

As seen in the results, the best student models peaked at 65% specificity. Although the proposed methods in this work offer a potentially better alternative to more traditional training methods, generalized deepfake detection on highly compressed videos remains a chal-

lenge. A potential issue with this work is the amount of knowledge that is transferred from teacher to student. Knowledge is passed exclusively through the loss factor. As discussed, other implementations of KD pass layer-level details from the teacher to student, leading to a more direct transfer of knowledge. Although the benefit of KD in this work varied from marginal to moderate, the field of KD as a whole can not be dismissed as ineffective for the task of deepfake detection. In conclusion, the results of this work do not support the use of these methods for real-world applications, but rather, motivate further research into the use of KD to address both the generalization and compression challenges in deepfake detection.

5.3 Future Works

The methods and experiments have significant potential to be refined and extended in future works. Other standard analysis techniques could be applied to this study to better understand the resulting student models of this work. Some analysis metrics that could provide further insight into the results include F1 and F2 scores, AUROC figures and scores, and a confusion matrix for a simpler view of the overall results. Additionally, generating saliency maps from both teacher and student models could offer insight into what features are being analyzed in images. At the very least, saliency maps could offer a sanity check to ensure all models are focusing on reasonable areas of an image.

Exploring the videos that the students and teachers incorrectly classified may also be worthwhile. In 2021, researchers found that models trained using ImageNet may contain biases stemming from ImageNet itself, including racial bias [28]. Researchers have developed tools to help identify some biases, such as Deepface [26]. Deepface is a facial analysis framework that can detect the faces of subjects in images and identify attributes, including race. Given that two of the teacher models in this work were pretrained using ImageNet, it is possible biases were transferred to the student models.

Another area of research that may be extended is the distillation methodology itself. The knowledge distillation method detailed in [22] is also suitable for heterogeneous teachers. Due to the fact that it passes knowledge to the student using intermittent features, as

opposed to only utilizing the loss factor like in this work, the potential performance gains are more substantial. The high computational cost of the methodology, however, makes it more difficult to implement. Another possible area of research is to develop a KD process that utilizes intermittent features, can transfer knowledge from heterogeneous teachers, and does not rely on training auxiliary networks, so as to reduce the required computational resources.

BIBLIOGRAPHY

[1] Darius Afchar, Vincent Nozick, Junichi Yamagishi, and Isao Echizen. Mesonet: a compact facial video forgery detection network. pages 1–7, 2018.

[2] Rana Ayyub. I was the victim of a deepfake porn plot intended to silence me. *Huffington Post*, 2018. Accessed: 2022-01-06.

[3] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541, 2006.

[4] Bobby Chesney and Danielle Citron. Deep fakes: A looming challenge for privacy, democracy, and national security. *Calif. L. Rev.*, 107:1753, 2019.

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

[6] Davide Cozzolino, Justus Thies, Andreas Rössler, Christian Riess, Matthias Nießner, and Luisa Verdoliva. Forensictransfer: Weakly-supervised domain adaptation for forgery detection. *arXiv preprint arXiv:1812.02510*, 2018.

[7] M.R.W. Dawson. *Mind, Body, World: Foundations of Cognitive Science*. Online access: Center for Open Education Open Textbook Library. Athabasca University Press, 2013.

[8] deepfakes. deepfakes. Available at `https://github.com/deepfakes/`. Accessed: 2022-04-27.

[9] Rebecca Delfino. Pornographic deepfakes—revenge porn's next tragic act–the case for federal criminalization. *Available at SSRN*, 2019.

[10] Steven Feldstein. How artificial intelligence systems could threaten democracy. *The Conversation*, 2019.

[11] Takashi Fukuda, Masayuki Suzuki, Gakuto Kurata, Samuel Thomas, Jia Cui, and Bhuvana Ramabhadran. Efficient knowledge distillation from an ensemble of teachers. In *Interspeech*, pages 3697–3701, 2017.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[14] Minha Kim, Shahroz Tariq, and Simon S Woo. Fretal: Generalizing deepfake detection using knowledge distillation and representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1001–1012, 2021.

[15] Marek Kowalski. Faceswap. Available at `https://github.com/MarekKowalski/FaceSwap/`. Accessed: 2022-04-27.

[16] Vejay Lalla, Adine Mitrani, and Zach Harned. Artificial intelligence: Deepfakes in the entertainment industry. Number 2, pages 12–17. World Intellectual Property Organization, 2022.

[17] Binh M Le and Simon S Woo. Add: Frequency attention and multi-view based knowledge distillation to detect low-quality compressed deepfake images. *arXiv preprint arXiv:2112.03553*, 2021.

[18] Lingzhi Li, Jianmin Bao, Hao Yang, Dong Chen, and Fang Wen. Faceshifter: Towards high fidelity and occlusion aware face swapping. *arXiv preprint arXiv:1912.13457*, 2019.

[19] Lingzhi Li, Jianmin Bao, Ting Zhang, Hao Yang, Dong Chen, Fang Wen, and Baining Guo. Face x-ray for more general face forgery detection. In *Proceedings of the*

IEEE/CVF conference on computer vision and pattern recognition, pages 5001–5010, 2020.

[20] Tianhong Li, Jianguo Li, Zhuang Liu, and Changshui Zhang. Few sample knowledge distillation for efficient network compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14639–14647, 2020.

[21] Aakash Nadimpalli and Ajita Rattani. On improving cross-dataset generalization of deepfake detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 91–99. IEEE Computer Society, 2022.

[22] Nikolaos Passalis, Maria Tzelepi, and Anastasios Tefas. Heterogeneous knowledge distillation using information flow modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[23] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics: A large-scale video dataset for forgery detection in human faces. *arXiv preprint arXiv:1803.09179*, 2018.

[24] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[25] Bharat Bhusan Sau and Vineeth N Balasubramanian. Deep model compression: Distilling knowledge from noisy teachers. *arXiv preprint arXiv:1610.09650*, 2016.

[26] Sefik Ilkin Serengil. Deepface. Available at `https://github.com/serengil/deepface`. Accessed: 2023-04-05.

[27] Chengchao Shen, Mengqi Xue, Xinchao Wang, Jie Song, Li Sun, and Mingli Song. Customizing student networks from heterogeneous teachers via adaptive knowledge amalgamation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3504–3513, 2019.

[28] Ryan Steed and Aylin Caliskan. Image representations learned with unsupervised pre-training contain human-like biases. In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 701–713, 2021.

[29] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *Acm Transactions on Graphics (TOG)*, 38(4):1–12, 2019.

[30] Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. volume 62, page 96–104, New York, NY, USA, dec 2018. Association for Computing Machinery.