Take a Deeper Look

Harry Ko
Stanford University
harryxko@stanford.edu

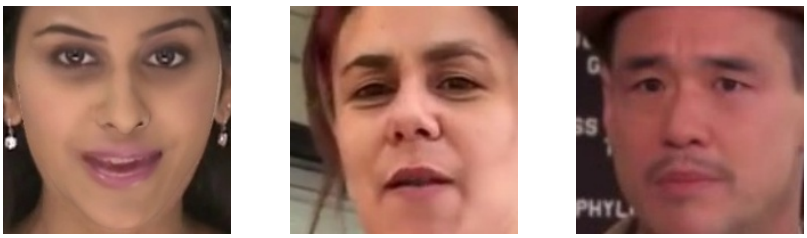
Surajit Mondal
Stanford University
sm4992@stanford.edu

Sophie Regan
Stanford University
sregan20@stanford.edu

1 Introduction

Deepfake techniques, which present realistic AI-generated videos of people doing and saying fictional things, have the potential to have a significant impact on how people determine the legitimacy of information presented online. While many are likely intended to be humorous, others could be harmful to individuals and our society. These content generation and modification technologies may affect the quality of public discourse and the safeguarding of human rights—especially given that deepfakes may be used maliciously as a source of misinformation, manipulation, harassment, and persuasion. Identifying manipulated media is a technically demanding and rapidly evolving challenge that requires collaborations across the entire tech industry and beyond.

Our project explores how deepfakes can be better detected. The input to our algorithm is an image of a face extracted from a video. We then used two different approaches: a pretrained convolutional neural network based classifier and a Support Vector Machine based classifier. The output predicted whether or not the image originated from a deepfake video or not.



(a) FakeForensics++ (b) DFDC (c) Celeb-DF

Figure 1: Deepfakes can be difficult to recognize for humans.

2 Related Work

For the past twenty years, research in virtual face manipulation has become more and more prominent. Virtual face manipulation systems have advanced to the point where they can create images that cannot be identified as fake by human observers. One such example, Face2Face can alter facial movements in videos from the internet by combining 3d model reconstruction and image-based rendering techniques [17]. In recent years, several deep learning approaches to face image synthesis have been developed. Researchers have successfully used Generative adversarial networks (GANs) to convincingly manipulate face images to appear older [2] or of a different race [11]. Further, Deep Feature Interpolation has also been used to successfully alter images along the dimensions of age, facial features, and expressions [19].

Utilizing the scientific techniques of multimedia forensics, researchers have attempted to analyze multimedia signals in images and videos in order to determine their authenticity. Early attempts focused on the integrity of images and relied on hand-crafted features. Currently, much of the research

in this domain focuses on solutions that rely on CNNs and techniques that employ both supervised and unsupervised learning.

Regarding deepfakes, some solutions have focused on specific artifacts present in videos or images, such as cue arising from colors, textures, or shapes [3, 4]. Additional research has been more general in scope, and instead focuses on capturing subtle indicators of inauthenticity. One recent approach demonstrated compelling results while also using a simple architecture [8]. They used a CNN to extract frame-level features, and then used those features to train a recurrent neural network (RNN), which learned to classify whether a video had been subject to manipulation. However, while these methods are often very successful, they may not be as useful in practical settings, such as analyzing images uploaded to social media sites. For example, when images or videos are compressed or resized, it becomes much harder to identify the traces left by manipulation. Recent research suggests that incorporating domain specific knowledge, wherein only the area around the face is used as the input, improves the overall performance of deepfake detection in comparison to approaches that use the whole image.

3 Datasets

We used three different publicly available datasets, FaceForensics++ [14], Celeb-DF [10] and Deepfake Detection Challenge (DFDC) [7] in our analysis. All datasets contain a collection of generated deepfake videos as well as corresponding original videos. We used the dlib library to extract every 10th frame from the videos and the OpenCV library to extract faces from the frames. The resulting data was manually cleaned by removing images that did not correspond to a face or corresponded to a face from the background. The train and validation split is 80/20 and the number of images for each dataset is shown in Table 1. We resized all faces to 200x200 pixels and used a 16 image batch size with shuffling enabled.

Table 1: The number of frames used after face extraction.

	Train		Val	
	Fake	Real	Fake	Real
FaceForensics++	31,765	34,997	7,429	7,409
Celeb-DF	44,862	17,842	11,215	4,460
Deepfake Detection Challenge	14,169	1,545	3,543	387

Deepfakes in the FaceForensics++ dataset are generated by four different methods, DeepFakes, Face2Face, FaceSwap and NeuralTextures which are applied to 1,000 Youtube videos. FaceSwap and Face2Face are computer graphics based approaches transferring facial landmarks, shapes and expressions from a source video to a target video. DeepFakes uses autoencoders that are trained to reconstruct training images of the source and the target face. Neural Textures uses Generative Adversarial Nets (GAN) to learn and reconstruct a target.

The majority of the fake videos in the Deepfake Detection Challenge (DFDC) dataset are generated with the Deepfake Autoencoder (DFAE) method. According to the creator of the dataset this choice was made to reflect the distribution of public deepfake videos. The other generation methods are three different GAN's and one computer graphics based method. All videos were shot by the creators themselves.

The Celeb-DF dataset consists of 5,639 deepfake Youtube videos. The aim of the creators was to make the deepfakes challenging to detect, as they argue that visible artifacts make existing deepfakes easy to distinguish from real videos even for humans. In particular, they focus on improvements in terms of resolution, color mismatch, face masks and flickering. The underlying method is based on an encoder-decoder model.

		Predicted	
		Fake	Real
Actual	Fake	7,387	42
	Real	38	7,371

(a) FaceForensics++ CNN

		Predicted	
		Fake	Real
Actual	Fake	3,532	11
	Real	23	364

(b) DFDC CNN

		Predicted	
		Fake	Real
Actual	Fake	10,807	408
	Real	318	4,142

(c) Celeb-DF CNN

		Predicted	
		Fake	Real
Actual	Fake	660	323
	Real	588	439

(d) FaceForensics++ SVM

		Predicted	
		Fake	Real
Actual	Fake	213	95
	Real	100	194

(e) DFDC SVM

		Predicted	
		Fake	Real
Actual	Fake	297	21
	Real	51	245

(f) Celeb-DF SVM

Table 2: Confusion matrices show type-1 and type-2 errors are mostly balanced.

4 Methods

4.1 Pretrained CNN

We implemented our convolutional neural networks method using the Xception architecture [5] as the base classifier and fine-tuned it on the real and fake video datasets. Xception is a convolutional neural net introduced in 2017 by Francois Chollet which performs well on ImageNet with a reported 79% Top-1 accuracy and 94.5% Top-5 accuracy. In addition, XceptionNet has been successfully used in previous works of deepfake detection, most notably FaceForensics++[14].

Xception stands for "Extreme Inception" and builds on the Inception architecture [16]. Inception became widely known in 2014 when GoogLeNet won the ILSVRC. Its main innovation was Inception blocks with “wider” networks instead of “deeper” networks. Every block runs several convolutions in parallel, each focused on a different, smaller region of the input image. The inception blocks are then stacked on top of each other. Xception takes this spatial separation one step further using depthwise separable convolutions. This means that each channel in each layer is treated completely independently with its own convolution matrix and spatial convolutions are run subsequently. As an illustration, for the input layer this means that patterns are identified for each RGB channel separately.

$$L = -\frac{1}{n}\sum_{i=1}^{n}[y_i(\log p_i) + (1 - y_i)\log(1 - p_i)] \tag{1}$$

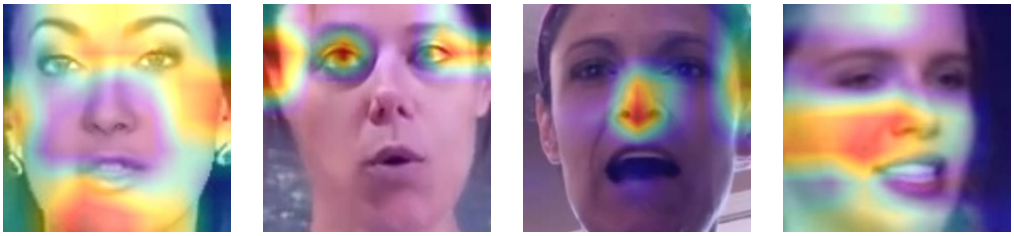
We removed the final softmax output layer of Xception and replaced it with a MaxPooling Layer followed by a binary classification output layer with sigmoid activation. The weights for the new output layer are initialized using the default Keras Xavier uniform initialization. We used the Adam optimizer instead of conventional gradient descent. The cost function is binary cross-entropy loss. As can be seen in equation 1, depending on the true label y, the loss function takes into account the predicted probability for a given example. As such, misclassifications aren't treated equally but instead the loss function penalizes predictions which have been wrong with a high probability. On the other hand though it doesn't reward predictions which have been correct with a high probability.

4.2 Support Vector Machines

A Support Vector Machine is supervised learning model that offer high accuracy compared to other classifiers such as logistic regression. It is one of the best known methods in pattern classification and image classification. It is designed to separate a set of training images into two different classes, $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$ where x_i in R^d, d-dimensional feature space, and y_i in -1,+1 (real and fake in our case), the class label, with $i = 1..n$. SVM generates the optimal hyperplane in an iterative manner, which is used to minimize error. SVM builds the optimal separating hyper planes based on a kernel function K that helps to transform the input space into a higher dimensional space which helps to separate problems that can not be solved by linear hyperplane. All images of which the feature

vector $h(x) = g(w^T x + b)$ lies on one side of the hyper plane i.e. $h(x) < 0$ are classified as class -1 and if $h(x) \geq 0$ are classified as class +1.

For the support vector machine approach we started by converting the images into a format for the algorithm to understand using Histogram of Oriented Gradients (HOG), a feature representation which is popular for object detection tasks. The images are divided into grids and for each pixel the HOG direction is compiled. The HOG feature vector for each image is the compilation of these block level histograms. In addition we also used global features that are invariant to global deformations. All features are then concatenated which reduces the correlation between features thus making classification more efficient.



(a) FakeForensics++ (b) DFDC-1 (c) DFDC-2 (d) Celeb-DF

Figure 2: GradCAM visualizes activations of final convolutional layer weights.

5 Experiments and Results

5.1 Pretrained CNN

We used the Keras implementation of the Xception architecture loaded with weights pre-trained on ImageNet. The reason for using a pretrained model is to make use of its existing general image classification capabilities and to avoid expending too many computational resources on training the network on basic image features which we assume are the same for both tasks. We tried several transfer learning approaches freezing the base network and only training the added top layers. However, the method that worked best was to finetune the existing weights by retraining the whole network including the base network for 15 epochs after which costs did not seem to decrease significantly any more. All training was done locally on a single GPU. We left the hyperparameters for the Adam optimizer at the default values which are α = 0.001, β_1 = 0.9 and β_2 = 0.999.

Table 3: CNN based performance for the three datasets.

	Accuracy	Loss	Precision	Recall	F1 score
FaceForensics++	99.6%	0.069	99.5%	99.4%	99.5%
Celeb-DF	95.4%	0.192	97.1%	96.4%	96.8%
DFDC	99.1%	0.028	99.4%	99.7%	99.5%

The results are summarized in table 3, figure 3 and the confusion matrices are shown in table 2. Performance is measured on the respective validation sets. The definitions of the individual metrics are shown in equations 2-4.

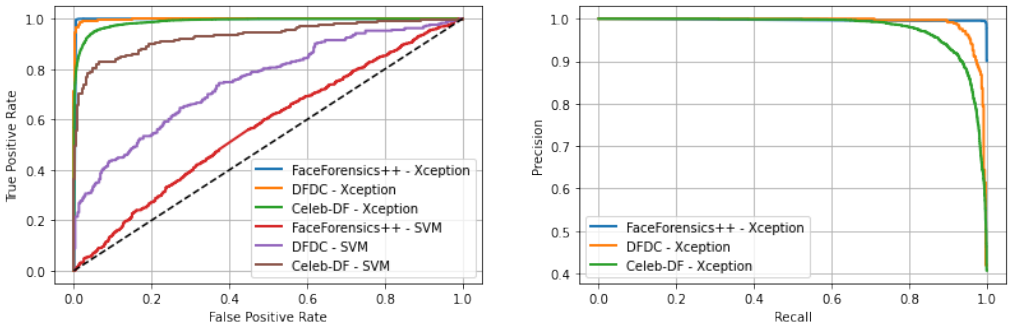
$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{2}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{3}$$

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total sample size}} \tag{4}$$

Overall, our classifier achieved accuracy rates of over 99% for two datasets and 95% for one dataset. In addition, we were able to slightly outpeform the creators of the FaceForensics++ dataset who

reported a 99.26% accuracy. The classifiers perform equally well on real and fake videos as precision and recall is roughly the same for all three datasets.



(a) AUC (b) AUPRC

Figure 3: AUC and AUPRC curves show decreasing performance.

To aid in interpretability of the results we also implemented GradCAM [6][15] which stands for Gradient Class Activation Mapping and visualizes final layer activations which can be overlayed onto the original image. The outcome is shown in figure 3. The classifier trained on FakeForensics++ focuses on the contours of the face and the chin region when deciding whether an image is fake or not. On the other hand, the classifier trained on DFDC focuses very specifically on landmarks such as eyes and nose when making predictions. The classifier trained on the Celeb-DF dataset focuses on a more broader region on one side of the face. We could observe the same mapping for the majority of the images of each dataset pointing to characteristics of different deepfake generation methods in each dataset.

5.2 Support Vector Machines

For our SVM approach we used a smaller subset of the data with the same 80/20 split for training and validation. We used the SVM module in the Scikit-Learn library which contains built in classes for different SVM algorithms. For the current classifier, we used an RBF kernel and random state 42. The results are shown in table 2, table 4 and figure 3.

Table 4: SVM based performance for the three datasets.

	Accuracy	Precision	Recall
FaceForensics++	54.7%	57.6%	42.7%
Celeb-DF	88.3%	92.1%	82.8%
DFDC	67.6%	67.1%	66%

6 Conclusion and Future Work

In this project we investigated different approaches to deepfake detection using different datasets. We find that our convolutional neural network approach using the Xception architecture with pretrained weights produces high accuracy and outperforms Support Vector Machines even using default parameters, relatively small datasets and few iterations. This holds true for deepfakes created with different generation methods. Differences in accuracy between datasets are related to the quality of deepfakes generated. We can show that classifiers trained on each dataset focuses on different areas of the image, which might relate to weaknesses in deepfake generation in that specific dataset. One area of future research is to survey existing deepfake generation methods and to carefully design a training set to make deepfake detection generalizable across different generation methods previously unseen by the network.

7 Contributions

Harry focused on the deep learning approach, Surajit focused on the support vector machine approach and Sophie focused on related work and the introduction.

8 Code

The code for this project can be accessed on github (`github.com/hko2333/takeadeeperlook`).

References

[1] Afchar, D., Nozick, V., Echizen, I. & Yamagichi, J. (2018) *"MesoNet: a Compact Facial Video Forgery Detection Network"*, arXiv:1809.00888.

[2] Antipov, G., Moez B. & Dugelay, J. (2017) *"Face aging with conditional generative adversarial networks"*, 2017 IEEE international conference on image processing (ICIP).

[3] Carvalho, T., Faria, F., Pedrini, F., Torres, R. & Rocha, A. (2015) *"Illuminant-based transformed spaces for image forensics"*, IEEE transactions on information forensics and security 11.4: 720-733.

[4] Carvalho, T., Riess, C., Angelopoulou, E., Pedrini, H.& Rocha, A. (2013) *"Exposing digital image forgeries by illumination color classification"*, IEEE Transactions on Information Forensics and Security 8.7 (2013): 1182-1194.

[5] Chollet, F. (2017) *"Xception: Deep Learning with Depthwise Separable Convolutions"*, arXiv:1610.02357v3.

[6] Chollet, F. (2018) *"Deep Learning with Python"*, Manning.

[7] Dolhansky, B., Bitton, J., Pflaum, B., Lu, J., Howes, R., Wang, M. & Canton Ferrer, C. (2020) *"The DeepFake Detection Challenge Dataset"*, arXiv:2006.07397v3.

[8] Güera, D. & Delp, E. (2018) *"Deepfake video detection using recurrent neural networks"*, 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS).

[9] Jiang, L., Wu, W., Li, R., Qian, C. & Loy, C. (2020) *"DeeperForensics-1.0: A Large-Scale Dataset for Real-World Face Forgery Detection"*, arXiv:2001.03024.

[10] Li, Y., Yang, X., Sun, P., Qi, H. & Lyu, S. (2020) *"Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics"*, arXiv:1909.12962v4.

[11] Lu, Y., Tai,Y. & Tang, C. (2018) *"Attribute-guided face generation using conditional cyclegan"*, Proceedings of the European conference on computer vision (ECCV).

[12] Mirsky, Y. & Lee, W. (2020) *"The Creation and Detection of Deepfakes: A Survey"*, arXiv:2004.11138v3.

[13] Qi, H. Guo, Q. Xu, F. Xie, X. Ma, L. Feng, W. Liu, Y. & Zhao,J. (2020) *"DeepRhythm: Exposing DeepFakes with Attentional Visual Heartbeat Rhythms"*, arXiv:2006.07634v2.

[14] Roessler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J. & Niessner, M. (2019) *"FaceForensics++: Learning to Detect Manipulated Facial Images"*, arXiv:1901.08971v3.

[15] Selvaraju, R., Cogswell, M., Das, A., Vedantam, R., Parikh, D. & Batra, D. (2019) *"Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization"*, arXiv:1610.02391v4.

[16] Szegedy, C, Wei, L., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. & Rabinovich, A. (2014) *"Going deeper with convolutions"*, arXiv:1409.4842.

[17] Thies, J., Zollhoefer, M., Stamminger, M., Theobalt, C., & Niessner, M. (2016) *"Face2Face: Real-time Face Capture and Reenactment of RGB Videos"*, Proceedings of the IEEE conference on computer vision and pattern recognition.

[18] Tolosana, R., Vera-Rodriguez, R., Fierrez, J., Morales, A. & Ortega-Garcia, J. (2020) *"DeepFakes and Beyond: A Survey of Face Manipulation and Fake Detection"*, arXiv:2001.00179v3.

[19] Upchurch, P., Gardner, J., Pleiss, G., Pless, R., Snavely, N., Bala, K. & Weinberger, K. (2017) *"Deep feature interpolation for image content changes"*, Proceedings of the IEEE conference on computer vision and pattern recognition.

[20] Verdoliva, L. (2020) *"Media Forensics and DeepFakes: an overview"*, arXiv:2001.06564.