A Survey of Different Methods used in Detecting Deepfakes

Dr S Manimala[1], Suraj R[2], Syed Hazim[3], Chitranjan Kumar[4], Ravi CV[5]
[1]Asst. Professor, [2, 3, 4, 5]BE Students
Department of Computer Science and Engineering, JSS Science and Technology University, Mysore- 570006

Abstract

Deepfake is a hybrid of the fake and deep-learning technologies. Deep learning is an artificial intelligence function that can be used to both build and identify deepfakes. Fake films, photos, news, and terrorist incidents are all created using Deepfake algorithms. When the number of deepfake videos and photos on social media rises, people will lose faith in the truth. Artificial intelligence breakthroughs have made it increasingly difficult to distinguish between real and counterfeit information, particularly photos and videos. Deepfake films, which are created by modifying videos using advanced machine learning techniques, are a recent invention. In the destination video, the face of an individual from the source video is replaced with the face of a second person. As deepfakes get more seamless and easier to compute, this concept is becoming further polished. Deepfakes, when combined with the scope and speed of social media, might easily deceive people by portraying someone saying things that never happened, leading to people believing imaginary scenarios, causing distress, and propagating fake news. Individuals, communities, organisations, security, religions, and democracy are all being impacted by deepfakes. In this study, we look at a number of strategies that can be used to identify deepfake videos. We employ a Transfer Learning strategy in which the system applies the feature information it learned while training on the ImageNet dataset and updates itself while training on our dataset. The trained models are used to classify counterfeit and unaltered videos. We then perform a comparative analysis on their performance metrics.

Key Words: deep learning, deepfake, artificial intelligence, detection, transfer learning

Introduction

Deepfakes is a video modification technique that was first used to replace the faces of celebrities in video recordings published to sites like Reddit. They function by substituting one person's face in an original video with the face of a second person who is inserted, resulting in similar head movement, facial expressions, lighting, and lip syncing. While thousands of images of the second person to be superimposed into the deepfake are frequently required, current study has shown that successful deepfakes can be made with a reduced number of second person images[11]. Deepfakes were first implemented using convolutional autoencoders [13]. Using an encoder, images of both subjects are reduced to lower dimensions and then rebuilt using a decoder. Both source and destination facial expressions are trained in this manner. A trained encoder of the source is mapped with a decoder trained on the target subject's face to conduct a face swap. Adding a generative adversarial network (GAN) to the decoder is an enhancement to this technology [4]. A generator and a discriminator are the two components that make up a GAN. The generator's job is to create images that seem like the source, while the discriminator assesses whether or not the image is fake. It is an iterative process, which makes deepfakes realistic as they are constantly learning. The availability of such advanced deepfake creation tools in the hands of regular researchers, as well as the possibility of their exploitation by others, has raised worries about their probable misuse. Deepfacelab[10], FakeApp, and OpenFaceSwap are GUI-based tools for creating deepfake movies that are accessible to relatively inexperienced researchers. With these techniques, it is becoming increasingly possible to manipulate video evidence for political purposes, false video evidence, and fake news. As a result, this presents a challenge for society as well as a potential for innovative entertainment, but it necessitates the development of an effective system for detecting counterfeit video.

While most detectors function admirably on a test subset derived from the same data distribution as those on which they were trained, how do they perform in a cross-dataset scenario? When a CNN trained for deepfake detection on dataset A is put to the test on dataset B, what happens? Because it is difficult to acquire direct insights into what happens within a CNN black-box model, we present a cross-dataset comparison of CNN-based deepfake detection algorithms in this study. We train the most common architectures used by competitors in the DeepFake Detection Challenge [3] and analyse how alternative training approaches affect intra-dataset and cross-dataset detection performances, rather than focusing on inventing a novel methodology suited for a single dataset. Our tests are based on datasets that are publicly available, such as the DeepFake Detection Challenge Dataset [3] and CelebDF(v2) [18]. Because video compression is frequently stronger than image compression, we focus on faces extracted from deepfake videos rather than just deepfake photos. We also perform some analysis taking into account a limited availability of training data.

Related Work on Deepfake Detection

Structural Similarity Index (SSIM) is a perceptual metric proposed in the work [12] that evaluates image quality degradation induced by processing such as data compression or data transmission losses, and for deepfake films, a quality degradation in frames is attributed to a less well-trained neural net. It's a comprehensive reference metric that necessitates the use of two photos taken at the same time. The SSIM is determined by luminance, contrast, and structure. When comparing the SSIM time series of an original video to its deepfakes version, discrepancies in frames owing to pixelated faces appear.

Observing the eye blinking is another approach to spot deepfakes. Blinking of the eyes is a basic biological feature that is incredibly difficult to replicate in deepfake films. Most training datasets of movies used for deepfake detection feature a small number of faces with their eyes closed, with an average rate of 4.5 blinks per second and each blink lasting 0.1-0.4 seconds. As a result, as revealed in the paper[8,] the lack of eye blinking can be a promising signal of a deepfake film.

The authors of [3] provide a sneak peek into deepfake identification issues using a publicly available dataset and two facial alteration techniques. The authors of [14] discuss the challenges and prospects of false news and the detection of fake news by presenting algorithms for detecting fake news from web services. The report [15] presents a thorough examination of current advances in deep face recognition, including databases and protocols, algorithm designs, and application scenarios. A overview of face image modification techniques, deepfake approaches, and methods to identify manipulations is presented in this paper[16].

The paper which is most similar to our work is by Luca Bondi, Edoardo Daniele Cannas, Paolo Bestagini, Stefano Tubaro, where they investigate the intra dataset and cross dataset performance of different architectures that are equipped with different training and data augmentation techniques. They investigate the model performance using Binary Cross Entropy and Triplet loss functions.[17]

Deepfake Datasets

A. DFDC Dataset

The footage in the DFDC collection was shot in non-natural contexts such as news or briefing rooms. The source data included 3,426 subjects with an average of 14.4 video searches per subject, with the majority of videos taken in 1080p, totaling 48,190 videos with an average of 68.8 seconds each.Recordings were shot in a range of natural settings without professional lighting or cosmetics to illustrate the potential harm of Deepfaked videos targeted to injure a single, maybe non-public person.
After the source films were pre-processed with an internal face tracking and alignment algorithm, all face frames were cropped, aligned, and downsized to 256x256 pixels.

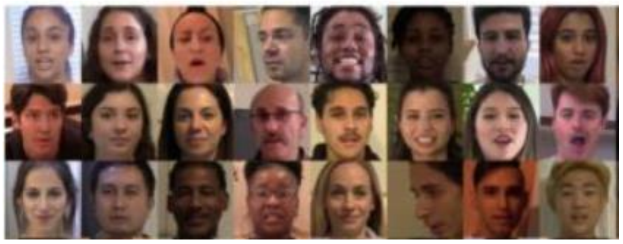


Fig 1: Sample frames of the DFDC Dataset

B. CelebDF Dataset

The Celeb-DF (v2) dataset contains actual and DeepFake generated movies of comparable visual quality to those found on the internet. The Celeb-DF (v2) dataset is significantly larger than the Celeb-DF (v1) dataset, which only contained 795 DeepFake movies. Celeb-DF now contains 590 original YouTube videos with topics of various ages, ethnic groupings, and genders, as well as 5639 DeepFake videos.

Celeb-DF's videos are created with a new DeepFake synthesis algorithm, which is crucial to the improved visual quality. Synthesized faces' low resolution has been enhanced to 256x256 pixels. This is accomplished by employing encoder and decoder models that have additional layers and dimensions.

Fig 2: Sample frames of the CelebDF Dataset

Data Preprocessing

We start by extracting frames from a video sample. This is done using OpenCV Library[1] , more specifically the VideoCapture class. We also use this Library to scale the extracted frames in order to maintain the aspect ratio between them. The scaling factor to be used while scaling, is dependant on the dimensions of the frame extracted. For example, if the dimensions of the extracted frame is greater than 1900, the scaling factor will be 0.33. On the same lines, if the dimensions of the extracted frame is lesser than 300, the scaling factor will be 2. Lastly, if the dimensions of the extracted frame is in between 1000 and 1900, the scaling factor is defined to be 0.5. These extracted scaled frames are now passed onto the Face extraction module. The face extraction process is done using Multi Cascaded Convolutional Neural Networks (MTCNN)[19] which extracts the faces from the frames with a certain confidence level. MTCNN extracts faces in three stages The MTCNN creates numerous frames in the first stage, scanning the entire image from the top left corner to the bottom right corner. P-Net (Proposal Net), a shallow, fully connected CNN, is used to retrieve information. In the second stage all the information from P-Net is used as an input for the next layer of CNN called as R-Net(Refinement Network), a fully connected, complex CNN which rejects a majority of the frames which do not contain faces. In the third and last stage, a more powerful and complicated CNN known as O-Net (Output Network) outputs the facial landmark location recognising a face from the given image/video, as the name suggests.

Fig 3 Sample of the faces extracted using MTCNN

The fig 3 shows the faces extracted using MTCNN. The next phase of this Pipeline is the splitting of the dataset into training, testing and validation set.The training set is applied to train, or fit, your model. The validation set is used for unbiased model evaluation during hyperparameter tuning and the test set is needed for an unbiased evaluation of the final model. You shouldn’t use it for fitting or validation. Splitting a dataset might also be important for detecting if your model suffers from one of two very common problems, called underfitting and overfitting.
To avoid these problems, a good rule of thumb for the ratio of splitting is 8:1:1. We follow this rule and split our dataset of the cropped faces into training, testing and validation set.

Methodology

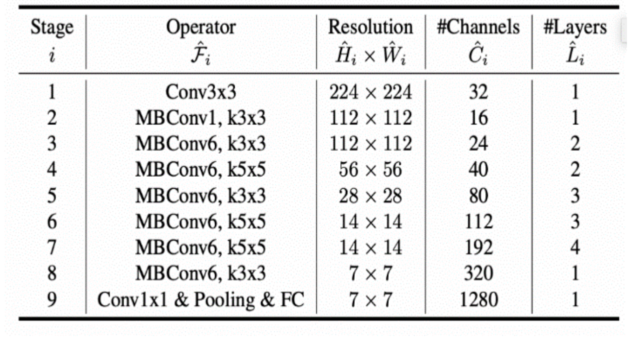
We must first develop a homogenous training and testing technique in order to successfully compare intra-dataset and cross-dataset detection results. A face detection and extraction phase is the first step in assessing whether a face in a video has been modified. Due to their prominence in the DeepFake Detection Challenge, we train EfficientNetB0 [20] and Resnet50 [24] architectures as reference CNNs, once faces are extracted and are uniform in size. Each face's likelihood of being a fake is predicted using the trained model.

After 2000 batch repetitions with no reduction in validation loss, the network is initialised with a model pre-trained on ImageNet, batch of 32 faces, Adam optimizer, initial learning rate of 10e-4 multiplied by a factor 0.1, and initial learning rate of 10e-4 multiplied by a factor 0.1. When the learning rate goes below 10-8, the training is over. The final model is the one that minimises the validation loss at each iteration. Training and validation batches are always balanced, with equal quantities of genuine and artificial faces chosen at random. The two CNN models are trained using the training sets of the two datasets, and then each model is tested against the test set of each dataset.

We trained the models on datasets that were of size 10,20,40, 50 and 80. We then observed the intra-dataset and cross-dataset performance of these models. The results of our experiments were produced using Python3 on Colab, a research based Jupyter Notebook server provided by Google. In addition to training models on individual datasets, we also trained them on combination of both datasets and evaluated their performance.

EfficientNet

The first method we investigate is EfficientNet[20]. Convolutional Neural Networks (ConvNets) are often built with a fixed resource budget and then scaled up for higher accuracy when more resources become available. In their research [20], they look at model scaling in depth and find that properly balancing network depth, width, and resolution can improve performance. While increasing individual dimensions improves model performance, they found that balancing all network dimensions, including width, depth, and picture resolution, against available resources produced the best overall results. The initial stage in the compound scaling strategy under a fixed resource constraint is to conduct a grid search to establish the relationship between different scaling dimensions of the baseline network (e.g., 2x more FLOPS). This calculates the proper scaling coefficient for each of the dimensions listed above. They then scale up the baseline network to the required target model size or computational budget using those factors. In addition to squeeze-and-excitation blocks, the base EfficientNet-B0 network is built on the inverted bottleneck residual blocks of MobileNetV2.



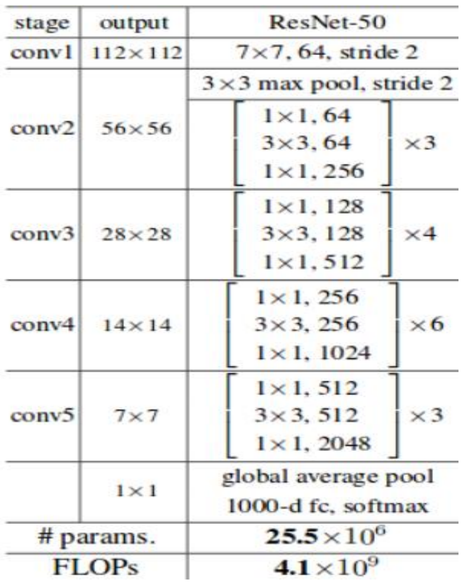
Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

Table I EfficientNetB0 Architecture

The architecture of the baseline EfficientNetB0 is shown in table I. MBConv stands for MobileNet Convolution[21] k stands for the kernel followed by the kernel size. FC stands for Fully Connected Layer. This compound scaling method consistently improves model accuracy and efficiency for scaling up existing models such as MobileNet[21] and ResNet compared to conventional scaling methods We then scale up the baseline network to obtain a family of models, called EfficientNets.

ResNet

The vanishing gradient problem was the primary reason for the development and proposal of ResNets[24]. The error is calculated and gradient values are determined during the backpropagation stage. The weights are modified after the gradients are transmitted back to hidden layers. The gradient determination process is repeated until the input layer is reached, after which it is sent back to the next concealed layer. As a result, the weights of the first layers will either update slowly or remain unchanged. In other words, the network's initial layers will not be able to learn successfully. As a result, deep network training will not converge, and accuracy will begin to deteriorate or saturate at a specific value.



stage	output	ResNet-50
conv1	112×112	7×7, 64, stride 2
conv2	56×56	3×3 max pool, stride 2
		$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3	28×28	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$
conv4	14×14	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$
conv5	7×7	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	1×1	global average pool 1000-d fc, softmax
# params.		$\mathbf{25.5} \times 10^6$
FLOPs		$\mathbf{4.1} \times 10^9$

Table II ResNet50V2 Architecture

The ResNet design is made up of a skip connection that bypasses some layers in the middle. As shown in table II, we trained ResNet50, a variation of the ResNet architecture with 48 Convolution layers, 1 MaxPool layer, and 1 Average Pool layer. We have one layer thanks to a convolution with a kernel size of 7 * 7 and 64 distinct kernels, all with a stride of size 2. Following that, we have max pooling with a stride size of 2. There is a 1 * 1,64 kernel in the next convolution, followed by a 3 * 3,64 kernel, and finally a 1 * 1,256 kernel. These three layers are repeated three times in total, giving us nine layers in this phase. Following there is a kernel of 1 * 1,128, followed by a kernel of 3 * 3,128, and finally a kernel of 1 * 1,512. This phase was performed four times, giving us a total of 12 layers. Then there's a 1 * 1,256 kernel, followed by 3 * 3,256 and 1 * 1,1024 kernels, which are repeated six times for a total of 18 layers. Then a 1 * 1,512 kernel was added, followed by two more 3 * 3,512 and 1 * 1,2048 kernels, for a total of nine layers. After that, average pooling is performed, followed by a fully connected layer of 1000 nodes, and finally, a softmax function is used.

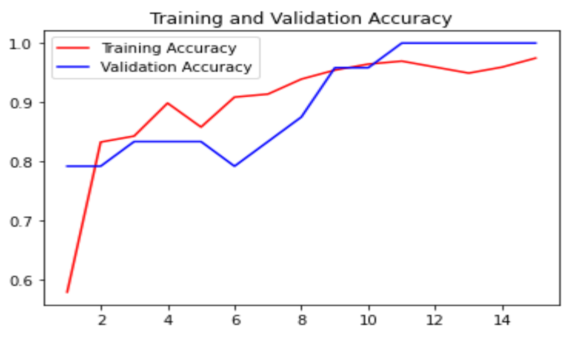


Fig 4(a)

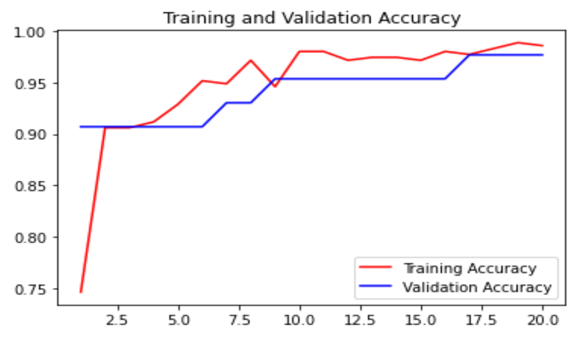


Fig 5(a)

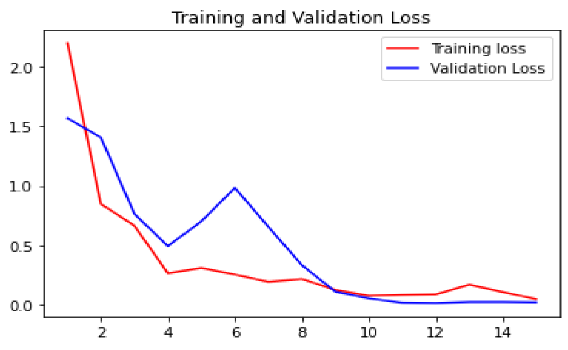


Fig 4 (b)

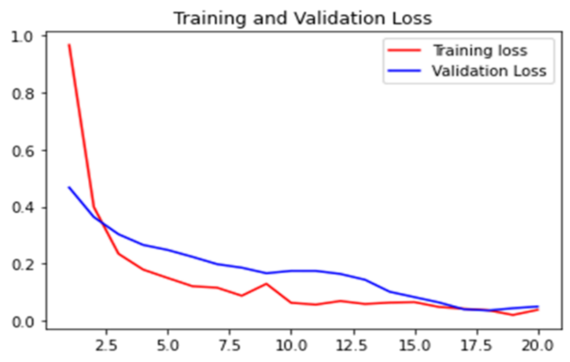


Fig 5(b)

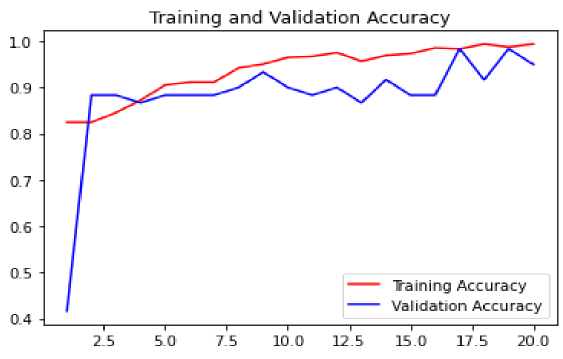


Fig 4 (c)

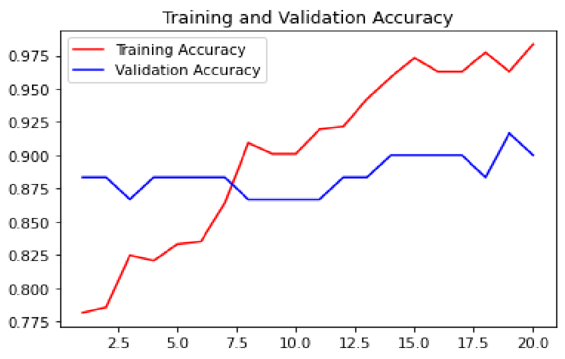


Fig 5(c)

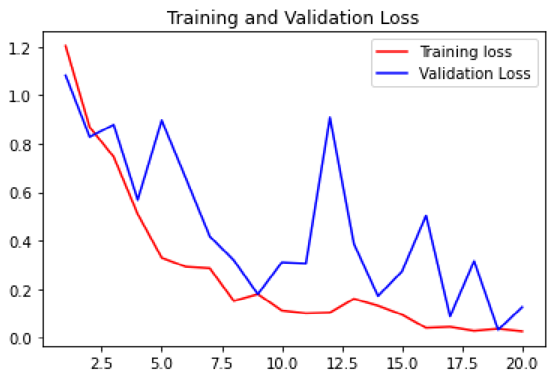


Fig 4(d)

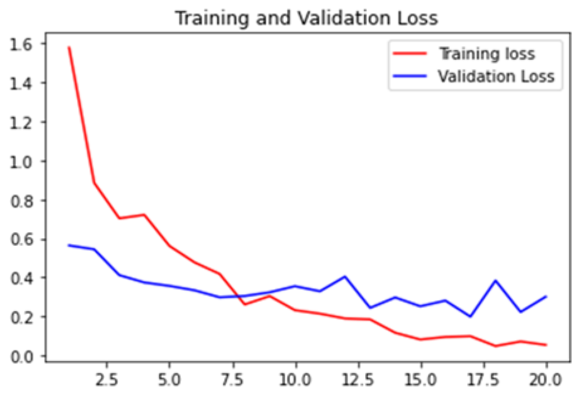


Fig 5(d)

Results for ResNet50

The training and validation graphs for ResNet50 trained on the DFDC dataset are shown in Figures 4(a) and (b). The architecture identifies the images with reasonable accuracy and achieves the optimum value with less computational resources because ResNet50 was initialised with ImageNet pretrained weights during training. The training and validation graphs for ResNet50 trained on the CelebDF dataset are shown in Figure 4(c) and 4(d). In order to compare the architecture trained on CelebDF dataset with the model built on DFDC, an equal number of samples and epochs were configured for the architecture trained on CelebDF dataset. The model trained on DFDC early stops on the epoch-15 because validation loss which was the metric monitored did not improve whereas for the model, trained on CelebDF did not stop training even after 20 epochs.

Sample Size	Training Accuracy	Validation Accuracy
10	96%	90%
20	97%	95%
40	99%	97%
50	99%	98%
80	98%	96%

Table III Results obtained for ResNet50

ResNet50 architecture, as seen anecdotally in the above table, delivers a validation accuracy of 90% as a constant curve when trained on 10 samples randomly chosen from CelebDF and DFDC datasets. If the validation accuracy graph remains constant, then the model is overfitting. To deal with the problem of overfitting, two methods are used. The first step is to minimise the network's complexity, and the second is to increase the amount of data used to train the model. We've gradually added additional data while monitoring ResNet's performance. We set the callback patience value to 4 throughout the training procedure. This means that if the model is found to have obtained constant validation loss for 4 epochs, then the training is terminated. This is done as a precaution in order to avoid overfitting.

Trained\Tested	DFDC	CelebDF
DFDC	96.06%	88%
CelebDF	68.6%	98.6%
DFDC+CelebDF	80%	99%

Table IV Cross Dataset and Intra-dataset performance obtained for ResNet50

As seen in Table IV, we're interested in both intra-dataset and cross-dataset detection performance. The intra-detection accuracy of DFDC and CelebDF is nearly same, while the cross-dataset accuracy is significantly variable. Based on the information presented above, an architecture trained on a single dataset may not be suitable for detecting deepfake samples from a different dataset. As a result, ResNet50 models trained on samples from both datasets outperform other models in both intra-dataset and cross-dataset circumstances.

Results for EfficientNetB0

Fig 5(a) and (b) show the training and validation graphs obtained for EfficientNetB0 trained on the DFDC dataset. Since EfficientNetB0 has also been initialized with ImageNet pretrained weights during training. Fig 5(c) and 5(d) show the training and validation graphs obtained EfficientNetB0 trained on the CelebDF dataset. Equal number of samples and epochs were configured for the architecture trained on CelebDF dataset in order to compare with model trained on DFDC. Both the models do not employ early stopping because validation loss keeps improving.

Sample Size	Training Accuracy	Validation Accuracy
10	98%	91%
20	97%	92%
40	98%	97%
50	97%	90%
80	97%	97%

Table V Results obtained for EfficientNetB0

From the above table, it can be inferred that EfficientNetB0 architecture trained on 10 samples randomly chosen from CelebDF and DFDC datasets, provides a validation accuracy of 91% as a constant curve indicating the overfitting of the model. To overcome the problem of overfitting, we have incrementally added more data and observed the performance of EfficientNetB0. During the process of training, we defined the callback patience value to be 4. This means that if the model is found to have obtained constant validation loss for 4 epochs, then the training is terminated. This is done as a precaution in order to avoid overfitting.

Trained\Tested	DFDC	CelebDF
DFDC	95.3%	87.5%
CelebDF	70.7%	95.2%
DFDC+CelebDF	83%	97%

Table VI Cross Dataset and Intra-dataset performance obtained for EfficientNetB0

In terms of the baseline B0, we were curious about the intra- and cross-dataset detection performance. As shown in Table VI, the intra-dataset detection accuracy of DFDC and CelebDF is nearly same, while the cross-dataset accuracy is significantly varying similar to the case as observed in ResNet50. Based on the information presented above, an architecture trained on a single dataset may not be suitable for detecting deepfake samples from a different dataset even in the case of EfficientNetB0. Hence, in both intra-dataset and cross-dataset circumstances, EfficientNetB0 trained on samples from both datasets tends to perform better.

Conclusion and Future Work

As deepfakes become more common, automated solutions are becoming increasingly important in combating them. All societal features and implications of these high-impact systems should be investigated by practitioners.On racially conscious datasets balanced by gender and race, we examined the predictive performance of popular deepfake detectors.

In this paper, we explored the performance of two state-of-the-art architectures in the domain of video classification. We conducted this research in order to answer the question “Which architecture performs better using limited computational resources when provided with less amount of data?” From the results we obtained, we can conclusively ascertain that ResNet50 is the better choice. As the number of samples used for training is increased, ResNet50 tends to achieve better performance results on real world data and also uses less computational resources as it achieves the desired accuracy in earlier stages of training and employs early stopping, terminating the training process.

We can't compare our results directly to the data in the entire DFDC findings because the best performing algorithms at the Deepfake Detection Challenge [3] obtained an accuracy of 82.56 percent across a considerably bigger test dataset than we used here. We believe that our results help us to understand how deepfake video detection can be accomplished using modern architectures.

A more extensive comparison of many alternative state-of-the-art architectures, such as InceptionV3[25] and XceptionNet[26], could be included in future research. Further research can also produce results that can help answer the question of which architecture can be used in a real-world application where the data isn't structured or labelled at all. In addition, our entire study relied on supervised learning. The notion of Unsupervised Contrastive Learning[27] can also be utilised to detect deepfakes via unsupervised learning. Recurrent Neural Networks, in a similar vein, can be utilised to solve this video categorization problem[5].

Since deepfake media is being shared at an alarming pace on social media, the scope for research into this subject is fast expanding. Future work may potentially result in changes to a current architecture's fundamental parameters and hyperparameters. Specifically, seeing how performance changes when alternative optimisers, loss functions, and the metric being watched are used. Well-funded research could also lead to the development of a whole new architecture that outperforms previous systems. This new architecture might have a new layer orientation, a new kernel function, less training constraints and a better performance on real world data.

Acknowledgements

Firstly, we would like to express our sincere gratitude to Dr. M. P. Pushpalatha, HOD of Dept. of Computer Science and Engineering, JSS STU for providing an excellent environment for our education and encouraging us throughout our stay in college. We extend our heartfelt gratitude to our guide Dr. Manimala, Asst. Professor of Dept. of Computer Science and Engineering, JSS STU who has supported us throughout our project with her patience and knowledge whilst allowing us the room to work in our own way.

References

[1] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. O’Reilly Media, Inc., 2008.

[2] Francois Chollet et al. Keras. https://keras.io, 2015. Last Accessed: 21 December 2020.

[3] Brian Dolhansky, Russ Howes, Ben Pflaum, Nicole Baram, and Cristian Canton Ferrer. The deepfake detection challenge (DFDC) preview dataset. *arXiv preprint arXiv:1910.08854*, 2019.

[4] Ian Goodfellow. NIPS 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[5] David Guera and Edward J Delp. Deepfake video detection¨ using recurrent neural networks. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, 2018.

[6] Sepp Hochreiter and Jurgen Schmidhuber. LSTM can solve¨ hard long time lag problems. In *Advances in neural information processing systems*, pages 473–479, 1997.

[7] Marissa Koopman, Andrea Macarulla Rodriguez, and Zeno Geradts. Detection of deepfake video manipulation. In *The 20th Irish Machine Vision and Image Processing Conference*, pages 133–136, 2018.

[8] Yuezun Li, Ming-Ching Chang, and Siwei Lyu. In ictu oculi: Exposing AI generated fake face videos by detecting eye blinking. *arXiv preprint arXiv:1806.02877*, 2018.

[9] Artem A Maksutov, Viacheslav O Morozov, Aleksander A Lavrenov, and Alexander S Smirnov. Methods of deepfake detection based on machine learning. In *IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, pages 408–411. IEEE, 2020.

[10] Ivan Petrov, Daiheng Gao, Nikolay Chervoniy, Kunlin Liu, Sugasa Marangonda, Chris Ume, Jian Jiang, Luis RP, Sheng´ Zhang, Pingyu Wu, et al. Deepfacelab: A simple, flexible and extensible face swapping framework. *arXiv preprint arXiv:2005.05535*, 2020.

[11] Simranjeet Singh, Rajneesh Sharma, and Alan F. Smeaton. Using GANs to synthesise minimum training data for deepfake generation. In *Proceedings of the 28th Irish Conference on Artificial Intelligence & Cognitive Science Dublin, Ireland http://ceur-ws.org/Vol-2771/,* pages 193– 204, December 2020.

[12] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The 27th Asilomar Conference on Signals, Systems & Computers*, volume 2, pages 1398–1402. IEEE, 2003.

[13] Yifei Zhang. A Better Autoencoder for Image: Convolutional Autoencoder.2018

[14] L. O. Alvaro Figueira, The current state of fake news: challenges and opportunities, in ICHSCIST, Barcelona, 2017

[15] W. D. Mei Wang, Deep Face Recognition: A Survey,2019.

[16] R. V.-R. J. F. A. M. J. O.-G. Ruben Tolosana, DeepFakes and Beyond A Survey of Face Manipulation and Fake Detection, pp. 1-15, 2020

[17] Luca Bondi, Edoardo Daniele Cannas, Paolo Bestagini, Stefano Tubaro. Training Strategies and Data Augmentations in CNN-based DeepFake Video Detection Computer Vision and Pattern Recognition, arxiv:2011.07792

[18] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, Siwei Lyu Celeb-DF,Computer Vision and Pattern Recognition (cs.CV) Arxiv:1909.12962

[19] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, Yu Qiao Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks. Computer Vision and Pattern Recognition (cs.CV). arxiv:1604.02878

[20] Mingxing Tan, Quoc V. Le EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. Computer Vision and Pattern Recognition (cs.CV). arxiv:1905.11946

[21] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications Computer Vision and Pattern Recognition (cs.CV).arxiv:1704.04861

[22] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen. MobileNetV2: Inverted Residuals and Linear Bottlenecks. Computer Vision and Pattern Recognition (cs.CV).arxiv:1801.04381

[23] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, Quoc V. Le. MnasNet: Platform-Aware Neural Architecture Computer Vision and Pattern Recognition (cs.CV) . arxiv:1807.11626

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Deep Residual Learning for Image Recognition Computer Vision and Pattern Recognition (cs.CV) . arxiv:1512.03385

[25] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe Jonathon Shlens, Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. Computer Vision and Pattern Recognition(cs.CV) . arxiv:1512.00567

[26] Francois Chollet. Xception: Deep Learning with Depthwise Separable Convolutions. Computer Vision and Pattern Recognition(cs.CV). arXiv:1610.02357

[27] Sheldon Fung, Xuequan Lu, Chao Zhang, Chang Tsun Li DeepfakeUCL: Deepfake Detection via Unsupervised Contrastive Learning. Computer Vision and Pattern Recognition(cs.CV). arXiv:2104.11507