ISSN 1816-6075 (Print), 1818-0523 (Online)
Journal of System and Management Sciences
Vol. 12 (2022) No. 5, pp. 21-35
DOI:10.33168/JSMS.2022.0502

Deepfakes Detection using Computer Vision and Deep Learning Approaches

Chong Jun Xiong, Kah Ong Michael Goh, Tee Connie

Faculty of Information Science & Technology (FIST), Multimedia University, Jalan Ayer Keroh Lama, 75450 Melaka, Malaysia

[+] *michael.goh@mmu.edu.my (corresponding author)*

Abstract. Earlier in 2018, deepfakes had grown in popularity as programmers used cutting-edge AI techniques to make software that could swap one person's face for another. The growth of deepfakes has not slowed down with each iteration of improvement and new approaches to swap faces. In 2019, Facebook, Tiktok, and Microsoft have started to block deepfakes videos and photos that might cause consumers to believe a subject act is from a real person. Humans' capacity to distinguish between face-swapped photos is no longer taken into account while trying to find a solution. In order to combat the false information that could harm some people, techniques to detect deepfakes are crucial. The goal of this research is to examine the most cutting-edge methods now available for identifying deepfake photos and to suggest a new or superior way utilizing computer vision and deep learning techniques. On the Face Forensic ++ DeepFake Dataset, the final models may achieve an Area Under the Curve (AUC) of 0.96661.

Keywords: deepfakes, face augmentation, face detection, face manipulation, deep learning.

1. Introduction

Machine learning techniques have advanced quickly in tandem with technological growth. Machine learning was once again given a boost when people began to employ it to increase their productivity in many industrial fields. Additionally, everyone may use machine learning technology whether or not they have prior machine learning knowledge thanks to the rapid progress of computer technology, which in turn leads to a decrease in the cost of computer parts. This technology has the potential to both damage and help society when it is used maliciously. Face-swapping machine learning is one of the technologies that will be covered in this project. With sufficient computing power and pre-trained algorithms, this technology can generate amazing face-swapping outcomes. While fake news or sensational news can be readily manufactured with little planning and cause disturbances in society, in real-world settings, machine learning technology can be dangerous if it falls into the wrong hands.

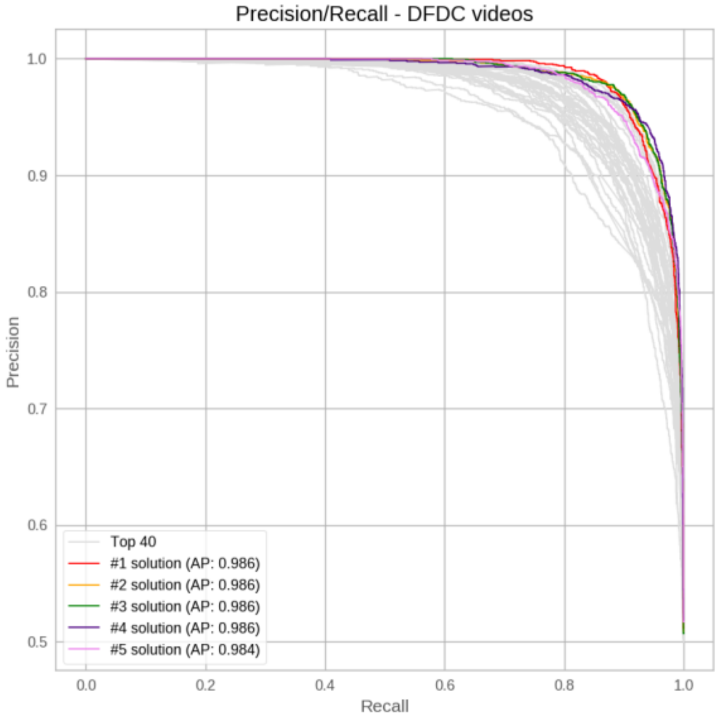
To stop such acts, a machine learning solution to identifying swapped faces is being developed, in which the system can determine which video is authentic (video without face swapping) or fake given a sequence of movies (video with face swapped). Videos of face swapping and its original form will be used as the datasets. The datasets are collected and assembled using feature extraction on the facial region of the video frame. After that, these features will be enhanced to the required format so they may be trained on certain models. The models' matrices are then recorded so that performance can be compared. For the final system, where they will forecast the output as an ensemble, the best few models are then chosen.

2. Literature Review

The approaches developed for the DeepFake Detection Challenge (DFDC) [1] competition utilising the DFDC dataset will be the major focus of this review on deepfake detection. Finding out what features are utilised as a dataset, what model is employed, and what approach is taken to obtain the result is the major objective here. The results of the top 5 models' performance in the DFDC competition are displayed in Table 1 and Figure 1 [1].

Table. 1: Top 5 models' results.

Team	Overall log loss	DFDC log loss	Real log loss
[2]	0.4279	0.1983	0.6605
[3]	0.4284	0.1787	0.6805
[5]	0.4345	0.1703	0.7039
[6]	0.4347	0.1882	0.6831
[7]	0.4371	0.2157	0.6621



Selim developed the approach that performs the best [2]. For speedier processing, the author employs Multi-task Cascaded Convolutional Networks (MTCNN) for face extraction. Due to time constraints, the author chooses to use (380x380) as the input size for training, and for the input data, he decides to use extreme data augmentation by removing a significant portion of the facial part; this can aid in the model's generalisation and also prevent the model from incorrectly predicting images with occlusions. The author trained 7 instances of the same model using various seeds and utilised the average of these 7 models as the prediction. The models used were EfficientNet B7, which was pre-trained with noisy student data, and ImageNet. Finally, the author also added a confident strategy to strengthen his final output for video inference by setting a threshold for the predictions that count as fake such that when the fake predictions are more than 11 or fake is more than 40% of the length of the given prediction list on the video, it will only return the mean of fake predictions, else if the number of predictions that is lower than 0.2 is more than 90% of the length of the prediction, the author added a confident strategy. This tactic has the potential to raise the system's confidence.

The following is a submission from WM [3]. In order to shrink the aligned faces to (320x320), the authors first extract them using RetinaFace from the provided DFDC video dataset. The mean and variance of these images are also determined for normalisation during training. After some basic data augmentation, these datasets are run through three different pipelines for prediction, two of which include either

Xception or EfficientNet B3, followed by WS-DAN and a different classifier. The pipelines also include Xception for feature extraction and Weakly Supervised Data Augmentation Network (WS-DAN) [4] for unsupervised data augmentation. The only components of the final pipeline are an Xception net and a classifier. Then, a weight of 0.2 on Xception, 0.7 on Xception and WS-DAN, and 0.1 on EfficientNet B3 and WS-DAN are used to determine the prediction of these three pipelines.

Over-fitting, according to NTechLab [5], is one of the primary issues in this competition. The author overcame this issue by utilising Mixup on aligned real-fake pairs. To do this, the source and target face images were extracted from the genuine and false films using the same bounding box at the same frame rate. The author produced a second set of training datasets called the blended into the fake dataset using some interpolation on these two photos, allowing the author to control the mixing ratio between the source and target image. Since the background remained the same after interpolation, this allowed the model to focus more on the facial region. The first two models, EfficientNet B7 trained with Noisy Student, are used in this method; they are the frame-based method with the first modal input having a zoomed face and the second model input having a full head input; both are with the size of (224x192); the third model, EfficientNet B7 with some modifications; the input for this model is a sequence of 7 frames skipped between 1/15 second; this method demonstrates the effectiveness of the method. Finally, since both models forecast poor quality frames very precisely to 0.5, the prediction is done using weights proportionate to the confidence level provided by the model.

RetinaFace is utilised by Eighteen Years Old [6] for the inference pipeline's bounding box extraction. In this competition, the authors employ a risky strategy in which their method is built using seven image-based models and four video-based models; they claim that the training of the image-based models requires eight GPUs and that of the video-based models requires sixteen. The image-based models include Xception, Efficient B3, B1, B1long, B1short, B0, and ResNet 34, whereas the video-based models include four slowfast networks. Additionally, the authors created a score fusion technique tailored specifically for the DFDC dataset.

For The Medic [7] face extraction approach, every 10 frames of the video are processed through MTCNN. The bounding boxes are then utilised to form a mask in a 3D array, where overlapping faces in the 3D array are counted as one moving face per moving person. On the previous region of interest, a second bounding box is likewise constructed, this one containing the entire face in every frame. The image-based model then makes use of these discovered bounding boxes. This technique likewise uses a large ensemble of models, 7 of which are video-based and 1 of which is image-based. Two alternative input resolutions—224x224 and 112x112—as well as four different architectures—I3D, 3D ResNet34, MC3, and R2+1D—are used in the video-based models. Two of these models were trained using cutmix augmentation, whereas the other four were not. SE-ResNeXT50 trained with cutmix augmentation is the image-based model. The final outcome is then submitted using the prediction's average.

The authors' Convolutional Cross ViT (Vision Transformer) architecture is one of the ways [8] they suggest. This approach comprises of two separate branches that the authors refer to as S-Branch and L-Branch. The S-Branch upper component is made up of a convolutional neural network that terminates with the (7x7) patch size and is then linearly projected into the S-transformer Branch's encoder. The S-job Branch's is to extract features in local areas with a patch size of seven by seven. The upper architecture of the L-Branch is similar to the S-Branch with the exception that the convolutional layer's output patch size is set to (64x64), for the same reason that different deepfake generating methods produce different artefacts that can be local or global. After receiving both transformer encoder outputs, these outputs are placed into a cross attention layer to enable direct interaction between the two outputs. After then, separate branches' multi-layer perceptrons are used to classify the outputs. According to the authors' experiment, EfficientNet B0 worked best on the convolutional layers of the Convolutional Cross ViT, demonstrating once more how effective it is at detecting deep fakes. The outcome of the authors' model is presented in Table 2, along with a comparison to other models.

Table. 2: Performance of the models on FaceForensic++.

Model	Mean (%)	Face Swap (%)	Deep fakes (%)	Face Shifter (%)	Neural Texture (%)
Convoluti-onal ViT	67	69	93	46	60
Efficient Vit	76	78	83	76	68
Conv Cross Vit Wodajo CNN	76	81	83	73	67
Conv Cross Vit EfficientNet B0	80	84	87	80	69

3. Methodology

The system flow for each video is briefly explained in the flow chart in Figure 2. Each of the target videos will first have their features extracted by the system; the low-resolution photos will be ignored. After that, each model will independently forecast each image. After classifying the results using the best ROC threshold and combining the results by obtaining the mean, the class of the video is determined using the inference technique.

3.1. Feature extraction

A maximum of 32 frames from each video are extracted for training, and a maximum of 30 frames are extracted from each video for inference. This is done because, according to a study published in paper [11], one of the authors' experiments shows a diminishing return as the number of frames extracted rises on a similar dataset, and because Google Colab has hardware limitations of 32 frames for training and 30 frames for inference. The bounding box is then extracted from the frame using RetinaFace. The bounding box and the frame number will be stored if there is just one face in the frame. These boxes are used to extract faces from both actual and false videos after receiving all of the bounding boxes from the original video. To obtain a similar frame for relatively simple data augmentation and to create a balanced dataset, the bounding box will be utilised to crop the real video and the false video at the saved frame number.

After these face photos have been extracted, a script performs a clean-up to get rid of undesired images like error images (all black images) and an image with low resolution from multiprocessing. Before the model is trained during the transformation, the image is also cropped and bordered.

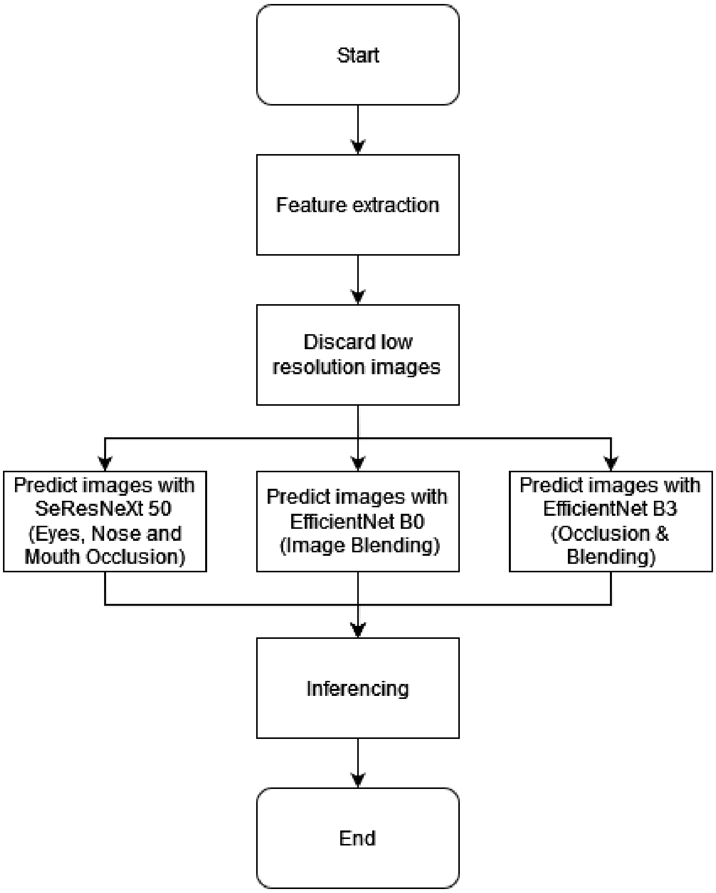

Start
Feature extraction
Discard low resolution images
Predict images with SeResNeXt 50 (Eyes, Nose and Mouth Occlusion)
Predict images with EfficientNet B0 (Image Blending)
Predict images with EfficientNet B3 (Occlusion & Blending)
Inferencing
End

Fig. 2: Flow chart of the inferencing system.

3.2. Eyes, nose and mouth occlusion

The occlusion of the facial region is one of the data augmentation techniques used in this research. The primary objective of this data augmentation technique is to ensure that the trained model does not excessively rely on a particular image feature in order to make any predictions [10] for this specific classification project scenario. For example, if the dataset is heavily loaded with images of people without glasses, the model will most likely classify the image as fake because the eyes are one of the key features used to distinguish between real and fake for this project. The code is created as a component of the PyTorch data augmentation function, which enables real-time adjustments during training because the augmentation code will be randomly generated.

Using RetinaFace, the left eye, right eye, mouth, and nose must all be located before continuing with the rest of the code. The estimated range of occlusion is then calculated using these landmarks in order to properly occlude the desired area. The landmarks provided for the eyes region are immediately on the iris of the eyes, therefore to block the entire eye, a 20 percent increment is added to the distance for each landmark when drawing the line that occludes the desired area. For drawing a line to cover the nose, the central point of the eyes is also taken into consideration. The provided landmarks are used directly for the mouth portion. The width/length of the photos, which were first resized to the target input size of 224 or 300 and multiplied by 0.1, are then used to compute the thickness of the line.

The landmarks of the nose, eyes, and mouth from RetinaFace will be out of alignment if image scaling and bordering are done before the Occlusion augmentation, as seen in Figure 3.

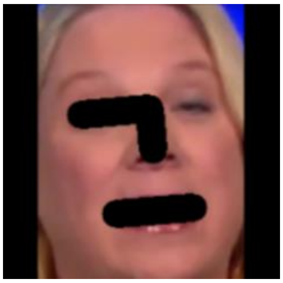


Fig. 3: Occluded, resized and bordered image.

3.3. Image blending

This project also combined fake and genuine photographs as a form of data augmentation. The purpose of this enhancement is to increase the number of training scenarios. As illustrated in Figure 4, by combining the real and false datasets, some of the artefacts on the fake dataset are able to blend in, generating a more convincing image and also expanding the number of training sets.

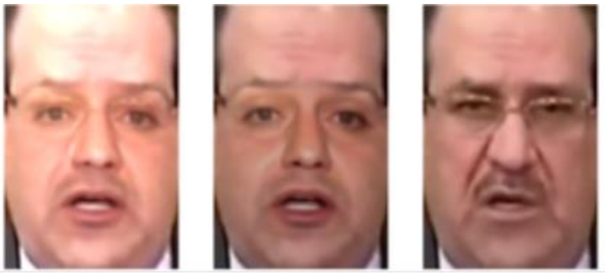


Fig. 4: Left - blended image, middle - fake image, right - real image.

A constant of 0.4 is used to blend the image from the previous one into the fictitious one. Whereas the fake image is missing details like the imprecise spectacle lenses, the longer groove behind the nose, and the contour of the beard, the blended image reveals these details.

In order to prevent the real data from changing for this code, the augmentation is only performed on the fictitious labelled training data. To avoid any undesirable output, this data augmentation is likewise performed before the occlusion augmentation. To prevent excessive blending from creating overlapped, meaningless images, the blending constant is also randomised between 0 and 0.4.

3.4. Inferencing

With a few small modifications, an inference pipeline is created for the inference component. First, each frame's bounding box is captured without having to compare identities. Then face extraction is done using the bounding box. The identical steps of squaring and eliminating low resolution are then applied to these faces. After then, each video's associated photos are saved in a different folder. The chosen 3 models then predict each of these folders separately because it would take too much computing resources for Google Colab to run all 3 models simultaneously. The predictions are then saved to a list, which will be used to determine whether the video is authentic. First, using the optimum threshold discovered during ROC curve testing, these predictions will be labelled as either 0 (false) or 1 (genuine).

The method is divided into two sections depending on whether each video's image count is less than 30 or exceeds that number. There are many identities present in one or more frames when there are more than 30 photos. The models employed in this research do not have 100% sensitivity to recognise fraudulent photos, which is the biggest issue in this case. The number of identities is determined by dividing the total number of photos by 30 for this reason. The threshold is computed by multiplying the number of identities by 2, which serves as a buffer for the incorrectly predicted image. The number 10 is used to denote that if any identity's prediction has a false count of more than or equal to 10, the entire video will be categorised as fake. When there are fewer than 30 images, the buffer will not be used because there aren't enough images. The image count is multiplied by 10/30 to determine the threshold in this case. Finally, the fake image count for each video is used to determine the inference's outcome. The video will be labelled as false or real depending on whether the number of fraudulent images exceeds the determined threshold. However, this approach resembles the approaches used by the majority of DFDC competitors, who experiment with many variables to find the optimal value.

3.5. Dataset

Face Forensic (FF++), Section 4.5 The training dataset was the DeepFake dataset. 1000 videos each for the original and deepfake classes make up the dataset, which has a downloaded quality of c23. For each class, there are 17021 face photos in the final training dataset. There is no need for additional balancing because the dataset is perfectly balanced. The dataset is divided into training, testing, and validation datasets using a 6:2:2 ratio.

4. Experiment

4.1. Data augmentation

The SEResNeXt-50, Efficient Net B0, and Efficient Net B3 models were employed in the experiment. According to Table 3, each model is trained separately with occlusion, blending, and occlusion & blending.

Version \ Models	SEResNeXt	b0	b3
No Augmentation	Acc:50.71 Sens:100 Spec:0.92	Acc:51.20 Sens:99.85 Spec:2.07	Acc :50.78 Sens:100.0 Spec:0.96
Occlusion	Acc:85.56 Sens:71.26 Spec:100.0	Acc:99.41 Sens:98.98 Spec:99.85	Acc:96.34 Sens:93.34 Spec:99.35
Blending	Acc:95.73 Sens:98.54 Spec:92.88	Acc:97.58 Sens:98.48 Spec:96.66	Acc:98.19 Sens:99.30 Spec:97.08
Occlusion & Blending	Acc:95.11 Sens:98.86 Spec:91.32	Acc:98.52 Sens:98.77 Spec:98.26	Acc:96.25 Sens:96.05 Spec:96.46

Table. 3: Result of testing for all best performing model (Acc:Accuracy, Sens:Sensitivity, Spec:Specificity).

The best AUC reading from Figure 5 is chosen as the model to utilise, and the final prediction output is then displayed after obtaining the average prediction result. Figure 6 shows the great performance of SeResNeXt 50 with each of the data augmentation method, it performs best with the occlusion augmentation. However, from Figure 7 and 8 we can see the performance of the models decrease with the increase in the models' size, this might be caused by the input image being compressed, as most of the training data does not meet the target resolution for the model due to resource constraints; but, by ensembling each of the best performing model they are provide a better classification.

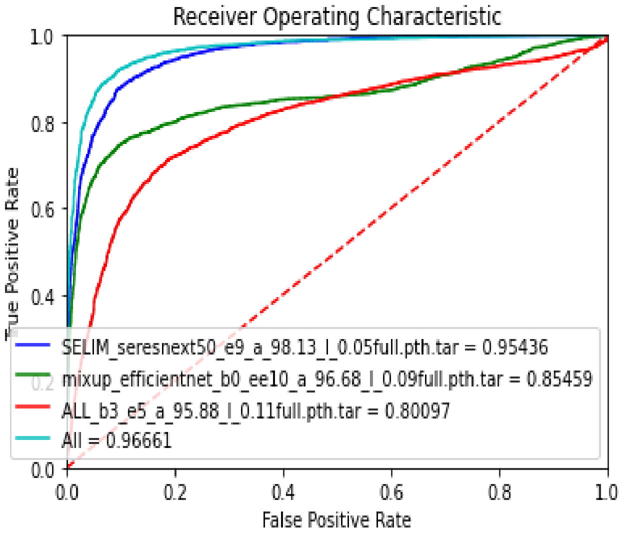


Fig. 5: ROC Curve and AUC reading of combining best models (Occlusion SEResNeXt-50, Blending Efficient Net B0, All Efficient Net B3).

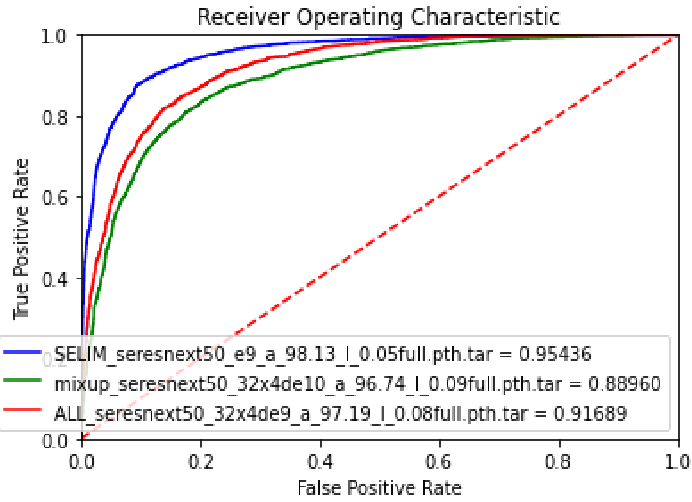


Fig. 6: ROC Curve and AUC reading of SeResNeXt-50 (SELIM = Occlusion, Mixup = Blending, ALL = Occlusion & Blending .

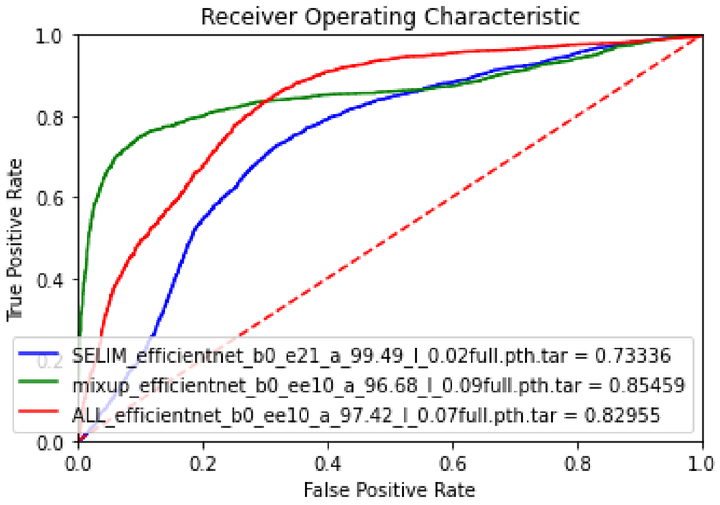


Fig. 7: ROC Curve and AUC reading of Efficient Net B0 (SELIM = Occlusion, Mixup = Blending, ALL = Occlusion & Blending).

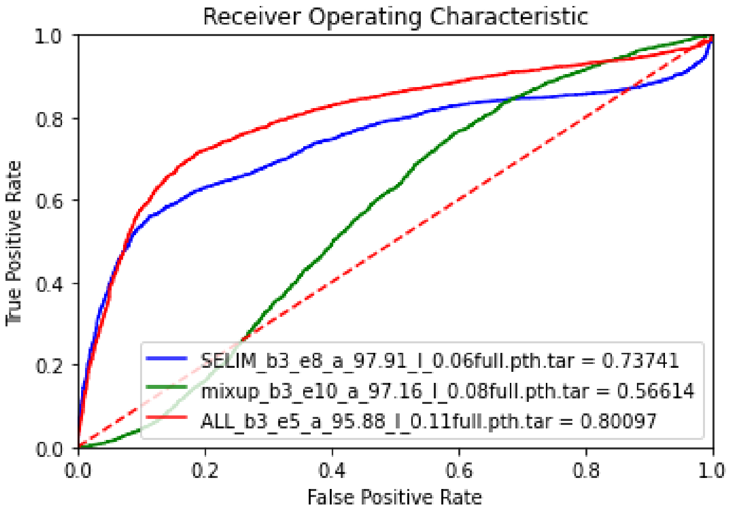


Fig. 8: ROC Curve and AUC reading of Efficient Net B3 (SELIM = Occlusion, Mixup = Blending ,ALL = Occlusion & Blending).

The AUC of the model is increased by obtaining the mean of the output. Despite the fact that the Efficient Net B3 and Efficient Net B0 AUC readings do not appear to be as good as SEResNeXt-50, they did help by raising the AUC score. Following selection of the ideal model combination, the ideal threshold is obtained when it yields the highest difference between the true positive rate and false positive rate, which in this case is 0.7856. The suggested model can outperform the other models displayed above using the same dataset, as indicated in Table 4's comparison with the other deepfakes models.

Table. 4: Comparison with the other models.

Models	AUC on FF++
EfficientNet B4,B4ST, B4Att, B4AttST [11]	0.9444
SeResNeXt 50 (Occlusion)	0.9544
EfficientNet B0 (Blending)	0.8546
EfficientNet B3 (Occlusion & Blending)	0.8010
Ensemble of SeResNeXt 50, EfficientNetB0, EfficientNet B3 (Ours)	0.9666

There is still potential for improvement in the performance of the Efficient Net models despite the fact that both of the employed data augmentation techniques can enhance the performance of the individual model.

4.2. Inferencing methods

The best-performing threshold was used as the initial technique to classify the model output. In order to extract faces from each face above the area of 20000, 30 frames from each movie will first be extracted. According to the best-performing threshold, these images will be predicted by the models, and the final output will either be 1 (actual) or 0 (fake); these fake counts of the final output will be counted. Any video that has a bogus count of more than 10 will be regarded as a fake video. This approach was found to be unreliable since more false counts would come from videos when several identities were being inferred. The incorrect model predictions are to blame for this. According to the experiment's findings, the models frequently generate many incorrectly classified outputs; as a result, when a video is being tested and contains multiple identities, both the total number of images that must be evaluated and the number of incorrectly classified outputs increase.

The recommended solution, which is obtained by multiplying with the identity ratio with a buffer range of 2, states that in order to address the issues raised above, a buffer of incorrectly categorised photos is included. By using this approach, the models' error tolerance can be improved, and the resulting inference is more trustworthy. The buffer image is not calculated for videos with image extraction of fewer than 30 photos, though, because these videos are more likely to contain just one identity than videos with image extraction of more than 30 images.

4.3. Real world scenarios

When the inference pipeline was evaluated using a real-world scenario, issues emerged. Since some of the facial characteristics or facial angles are missing from the training dataset, this problem is primarily due to the dataset that is being used. Figure 9 depicts a woman with an extreme face angle and a male with more facial hair than usual. The models' predictions demonstrate that these were not often occurring events on the training dataset.

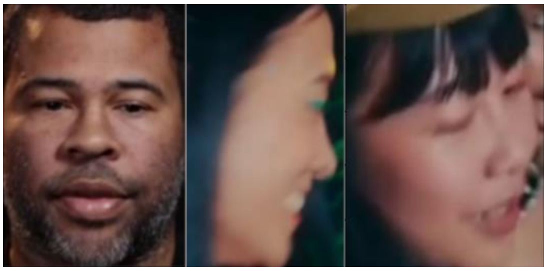


Fig. 9: Real images that were predicted as fake.

Figure 10 demonstrates how videos that have been enhanced by a skilled user can potentially be an issue. A perfected deepfake video can be quite convincing when tested on a real-world context. Frame-based models appear to be performing poorly in these well-edited movies; but, a 3D convolutional neural network may be able to detect the spatial changes that occur when the face moves and produce better results.

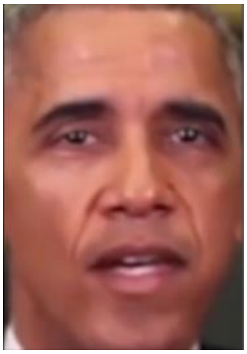


Fig. 10: Fake image detected as real.

The model could be able to differentiate the artefacts better with a larger dataset and a more extensive face-swapping technique. Nevertheless, in order for this method to be a trustworthy means to determine the veracity of the video, a regular update on the dataset is required to keep up with the development of face swapping techniques.

The trained model performed noticeably worse than the filtered image dataset when tested on a face picture dataset that had not been filtered. This results from the photos being resized before training since low-resolution photographs are unable to offer the model with many texture features, which means the information provided by the real and fake head images will not differ significantly. The area is around 40% of the training resolution of (224 × 224) and 20% of the training resolution of so images that are lower than the area pixel of 20000 are discarded (300 x 300). This is further demonstrated by the inference experimentation, which shows that when the area is set to 15000, 17500, and 20000, the models are unable to accurately anticipate the input image. Although threshold areas 15000 and 17500 may harvest more input data than threshold area 20000 during this test, the algorithms are unable to successfully distinguish even the training set from real-world circumstances.

5. Future Work

This project displays average-sized models that were trained with adequate data augmentation and constrained computing resources. A larger dataset and a sufficiently large model may be able to increase performance, but this will also result in a greater demand for computing resources. We are able to generate a sizable result on deepfake identification. More sorts of data augmentation techniques that can help the models emphasise the artefacts of deepfake video can also be investigated, as can a better iteration of the occlusion method that occluding more facial features that mimic real-life settings. In order to achieve a more dependable performance on inferencing, a better face extraction approach can also be performed during the inference stage. This strategy eliminates side faces or odd angles that are not present in the training set.

Since most real-world photographs are not set to be larger than 20000 pixels, a study on lower resolution deepfake detection can also be conducted. As higher quality photos may be downscaled and just a smaller model is required to analyse the lower resolution images, this study can also help enhance the performance of the future model on predicting deepfake detection.

A more effective solution than an image-based detection system is required, though, to reliably determine whether a video is real in a situation where it is needed. For example, to detect facial motion, we can use a 3D convolutional neural network trained on the deepfake video's spatial artefact, or we can use an audio-video relation model that can recognise when audio and video are synchronised.

6. Conclusion

In conclusion, the major goals of this research are to build a pipeline for deepfake detection and evaluate its effectiveness. RetinaFace is used to extract faces from the downloaded FF++ video collection. Multiprocessing accelerates the majority of the extraction processes. The extracted dataset is then enhanced using a variety of image enhancement techniques, resizing, and almost attaining the appropriate input for the models' training. Then, using a variety of data augmentation techniques,= the 3 chosen models—SEResNeXt-50, Efficient Net B0, and Efficient Net B3—are trained. The results are then compared using a variety of metrics and are documented for future reference. The inferencing pipeline then employs the top-performing models determined by the metrics. To find the best method, the inference strategy is also tested with various outcomes. With a result of 0.9661 AUC generated from the ROC

Curve, the final product comprises of SEResNeXt-50 with Occlusion, Efficient Net B0 with Blending, and Efficient Net B3 with Occlusion and Blending.

It is crucial for these systems to operate properly, especially at this time when wars are raging, to stop fake news from spreading and causing unintended harm to any living thing or even a nation. This deepfake detection system may be used to determine whether a video is genuine or not.

Acknowledgments

This project is supported by the TM R&D (Project no. RDTC/221054) and MMU IR Fund (Project ID MMUI/210108).

Reference

Andreas, R., Davide, C., Luisa, V., Christian, R., Justus, T. & Matthias, N. (2019). FaceForensics++: Learning to detect manipulated facial images.

Azat, D. (2020). Deepfake-detection-challenge. https://github.com/NTech-Lab/deepfake-detection-challenge.

Bonettini, N., Cannas, E. D., Mandelli, S., Bondi, L., Bestagini, P. & Tubaro, S. (2021). Video face manipulation detection through ensemble of cnns. In *2020 25th International Conference on Pattern Recognition (ICPR)*, 5012-5019. IEEE.

Davide, C., Nicola, M., Claudio, G. & Fabrizio, F. (2021). Combining efficientnet and vision transformers for video deepfake detection.

Dolhansky, B., Bitton, J., Pflaum, B., Lu, J., Howes, R., Wang, M., & Ferrer, C. C. (2020). The deepfake detection challenge (dfdc) dataset.

Hu, T., Qi, H., Huang, Q., & Lu, Y. (2019). See better before looking closer: Weakly supervised data augmentation network for fine-grained visual classification.

Jacob Solawetz (2020). Occlusion techniques in computer vision. https://blog.roboflow.com/occlusion-computer-vision/#enter-occlusion-techniques.

James, H. & Ian, P. (2020). https://github.com/jphdotam/DFDC.

Selim, S. (2020). dfdc_deepfake_challenge. https://github.com/selimsef/dfdc_deepfake_challenge.

Shao, J., Shi, H., Yin, Z., Fang, Z., Yin, G., Chen, S., Ning, N & Liu,Y.. (2020). RobustForensics. https://github.com/Siyu-C/RobustForensics.

Zhou, W., Cui, H. & Zhao, H. (2020). kaggle-dfdc. https://github.com/cuihaoleo/kaggle-dfdc.