Celeb-DF Deep Fake detection with ResNets50V2 Implementation Report

Avi Varma - 957552

December 27, 2020

Contents

1 Proposed Method

The implementation for this project will be split into the following sections – dataset generation; dataset pre-processing; ML model implementation. Throughout these sections, I will be describing and explaining my approach to tackling the proposed problem within the paper: "DeepFake Face Image Detection based on Improved VGG Convolutional Neural Network"[1]. I used their proposed method as a benchmark to follow. Hence, I had chosen the same dataset, and filter kernel, with a different DNNs to challenge their findings.

1.1 Celeb-DF Dataset V2

The revised Celeb-DF dataset consists of 590 real videos and 5, 639 DeepFake MP4 videos. The real videos are chosen from publicly available YouTube videos, corresponding to interviews of 59 celebrities with diverse distribution in their genders, ages, and ethnic groups[2]. These are split into two directories - Celeb-real and YouTube-real. The Deep fake videos are stored within Celeb-synthesis directory. With these videos, a partial label set is provided where the files are labelled 'real' with '1' and fake with '0'.

1.2 Converting MP4 Videos to JPEG

To make the Celeb-DF dataset compatible with ResNet-50V2, the video's frames must be extracted to transform the video dataset into an image dataset. To perform this conversion OpenCV's 'VideoCapture' function is utilized which can load and read videos frame by frame.

The function 'extractFrame' takes the source path and destination path as parameters. The function itself loads the video at the source path and extracts the first frame from the video (cv2.read). This frame is saved (cv2.imwirte) at the destination path with the source file name. Through this function, the entire Celeb-DF dataset is cloned, and a new dataset of frames is generated.

1.3 Extract Faces from Frames

The images extracted from the videos consist of a large variety of background noise. This includes news media logos to backgrounds of TV sets. To prevent the AI from extracting features from these regions, all frames extracted must be cropped so the resulting image only contains the celebrity's face. The classifier "haarcascade_frontalface_alt.xml" of OpenCV for face location and capture is used to extract and prune the dataset of ineligible faces as used within the reference paper. From this, a new dataset is generated which consists of these faces and retain their original file names.

The function face_detect(PathIn, PathOut), takes the source path and destination path as parameters. The source image is converted to greyscale and parsed into the "haarcascade_frontalface_alt.xml" classifier with a minimum image size of 128x128 pixels, scale factor of 1.1 and minimum neighbours set to 5. The new image is then saved to the destination file path. Hence the image dataset generated is now replaced with these cropped images set whilst preserving the original dataset structure.

1.4 SRM Filter

"SRM filter in image forensics to extract local noise feature map from RGB image and takes the local noise distribution data of the image as the network input, and then uses the noise feature to provide the basis for image processing for authenticity classification."[1] To test its effectiveness in assisting ResNet50 classifying deepfake images, a clone dataset is created where images from the Celeb-DF-Image-Face-Crop dataset is parsed through the SRM filter and placed into Celeb-DF-Image-Face-Crop-SRM dataset. Again, the original Celeb-DF dataset's structure is preserved. The SRM kernel is stored within a NumPy array and applied to each image using OpenCV's 'cv2.filter2D' function.

1.5 Labelling the datasets

The celeb-DF dataset provides a small set of labels named 'test_labels.txt'. The format from these labels is preserved when generating the full label set. Since the directory structure for both Celeb-DF-Image-Face-Crop and Celeb-DF-Image-Face-Crop-SRM datasets are the same as the original dataset's, the labelling format from the original dataset will be compatible with the two image datasets.

The function 'generate_labels(directory_path_in, filename, subfolder, value)' takes the dataset's path, output file name, the subfolder within the dataset's parent directory and label value as parameters. With this, the function navigates to the specified subfolder within the dataset and stores a list of all files within the sub-directory. This list is iterated over and appended into the textfile with its corresponding value with the following format: ¡value¿ "file_path".

1.6 Dataset Pre-processing

First the label set is loaded, and the dataset is split into validation – train sets (15% - 85%) using sklearn.model_selection. Using tf.data.Dataset.from_tensor_slices() the image filenames and labels are split into training and validation sets.

The current validation and training set only contains file names, not actual images. Therefore, a function is defined which can load the images from the file path and perform the necessary pre-processing. The function '_parse_fn(img, label)' reads the image and decodes it using the TensorFlow library. The decoded image is re-sized to 128x128 resolution and returned with the provided label. Each split set is now shuffled and batched into sizes of 32 images.

1.7 Model for Transfer Learning

Transfer learning performs cross-domain learning by extracting useful information from data in a related domain and transferring them for being used in target tasks [3]. Through this method, training is sped up since only the top layer is re-trained that determines the classes to which an image belongs to.

As a control test, MobileNetsV2 [4] are modelled. The model has the fastest training times allowing quick model tuning. Hence, all models use the imagenets weights, max pooling and relu activation function. The pre-trained model's weights are frozen for each layer. To compare against the Proposed VGG16 methods, the 'tf.keras.applications.VGG16' is trained with imagenets weights, max pooling and relu activation and classes=2.

The main base model for this project is Keras ResNet50V2. This is an improved residual unit which is 1000- layer deep network that can be easily trained and achieve improved accuracy [5]. Within implementation, the 'tf.keras.applications.ResNet50V2' is used with imagenets weights, max pooling and relu activation function. The pre-trained model's weights are frozen for each layer.

A new classification head is added to the base model with output dimensionality set to one and activation function set to sigmoid. From this a sequential model is created where the input is first parsed through the three base models, the outputs are then parsed through the prediction layer which classifies the provided inputs. For all models, adam optimizer with a learning rate of 0.0001 is used and binary crossentropy loss function. SGD optimizer was not used like in the reference paper because the adam optimizer can utilize the momentum concept from "SGD with momentum" and adaptive learning rate from "Ada delta". Hence, the optimizer is computationally efficient and appropriate for problems with noisy gradients. Within the training function, both the validation data and the training data are set to repeating.

1.8 Evaluation Techniques

Once the respective models are trained, first the accuracy and loss are evaluated of the model with matplotlib.pyplot. With these graphs, the models training performance can be evaluated which can show if there needs to be tuning on the model before further evaluating the performance of the predictor.

The predictor returns the labels of the data passed as an argument based upon the trained data obtained from the model. This is parsed into a confusion matrix to visualize the predicted label's accuracy. Through this individual classification's performance can be evaluated and help compare between the three different models.

The final evaluation step is to calculate the ROC curve and AUC score. The ROC shows the performance of the classification models trained at all classification thresholds. Through this, the model's behaviour can be compared against the other models and their true accuracy can be visualized and evaluated with the AUC. AUC stands for "Area under the ROC Curve" which is more reliable than the accuracy score because it measures how well predictions are ranked, rather than their absolute values.

2 Results & Discussion

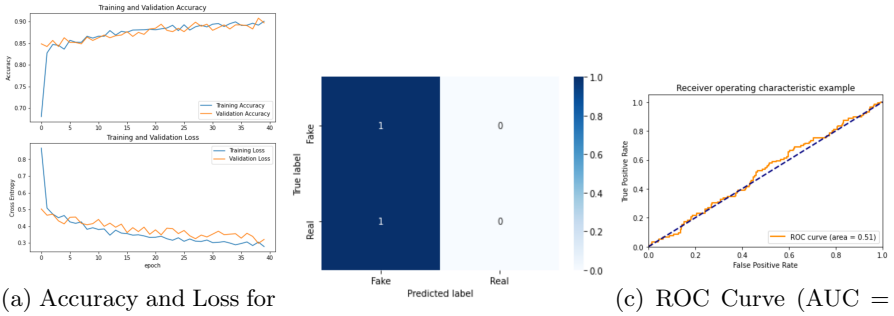
Table 1: AUC performance of different methods on Celeb-DF [1]

Methods	**Average AUC performance**
Average of Several Methods	48.7%
Two-Stream	55.7%
VGG16	56.4%
SRM filter +VGG16	73.2%
NA-VGG	85.7%

In this section will break down the individual models' performance; explain the reasoning behind the results and if there is room for improvement, how the better results could have been achieved. Also, the reference paper provides experimental results which of the average AUC score of each detection method on Celeb-DF. Therefore, the minimum AUC accuracy to beat for each model will be 48.7% and target accuracy of 73.2%.

2.1 MobileNets Deepfake Detection

Two identical Mobilenets models are trained with two different datasets (Celeb-DF-Image-Face-Crop & Celeb-DF-Image-Face-Crop-SRM). With the Celeb-DF-Image-Face-Crop dataset, MobileNets begin with a strong initial training accuracy jump from 67% to 84%. From here the model's learning rate eases off and gradually reaches a training accuracy of 90%. The validation accuracy the same as training accuracy, therefore, the model is not overfitting. The same trend is true for the model's loss, initially 0.87, then sharp decrease to 0.5 within one step, gradually decreases to 0.27 with the validation loss only roughly 0.4 above the training loss value. The confusion matrix on the other hand displays a worse accuracy. The model predicts all images as fake, hence, the mobilenets model is unable to deduce if an image is real and to take precautions is labelling all images as fake. This hypothesis is visible in action form the ROC curve. The ROC is slightly above the no predictive line with AUC = 51%, hence, the model is unable to accurately predict the true labels.



(a) Accuracy and Loss for training and validation

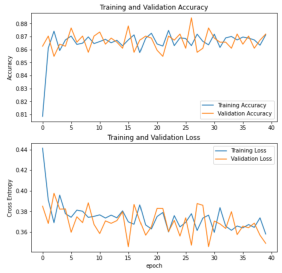
(b) Confusion Matrix

(c) ROC Curve (AUC = 51%)

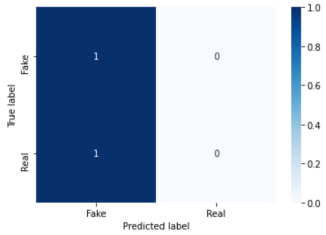
Figure 1: MobileNetsV2 metrics with "Celeb-DF-Image-Face-Crop"

When the Celeb-DF-Image-Face-Crop-SRM dataset is inputted into MobileNets, the initial accuracy is 80% which then gradually increases to 87% with validation accuracy fluctuating by ± 1%. The loss value for the model's trend is like the accuracy trend. Initially, 0.44 and from there gradually decreases to o.36 with the validation loss ± 0.01. For the confusion matrix, the same trend is visible as without the SRM filter. The model still predicts all images as fake, also, the ROC curve suggests that the image augmentation reduced the classifier's confidence as the ROC line is much closer to the no predictive line with finally, the same AUC score of 51%.

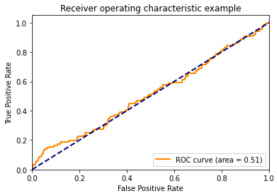
Therefore, the results clearly show that during the training process MobileNets can gain a good accuracy without underfitting nor overfitting. However, the model is still unable to effectively classify with confidence against



(a) Accuracy and Loss for training and validation



(b) Confusion Matrix



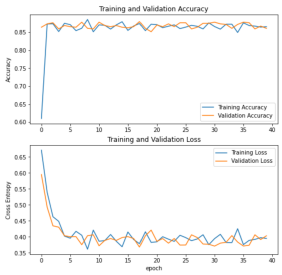
(c) ROC Curve (AUC = 51%)

Figure 2: MobileNetsV2 metrics with "Celeb-DF-Image-Face-Crop-SRM"

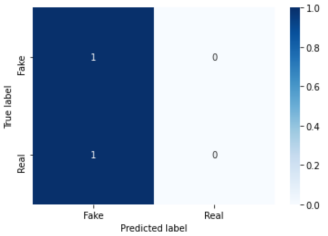
the deepfake images, which is shown by the AUC score. However, from transfer learning, MobileNets is still able to perform better than the average AUC metric given with the dataset.

2.2 VGG16 Deepfake Detection

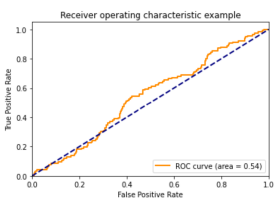
With the Celeb-DF-Image-Face-Crop dataset, VGG16's accuracy initially begins with a poor training accuracy of 36% and validation accuracy of 86%. The accuracy increases to 87% in the next step, from there the accuracy stagnates and stabilizes at 86% with the validation accuracy. The loss, on the other hand, the training loss starts at 0.7 and validation loss of 0.6. This rapidly decreases down to 0.4 where both training and validation stagnates at 0.4. Hence, VGG16 is not underfitting nor overfitting just like the Mobilenets. Like MobileNets, the VGG16's confusion matrix classifies all images as fake. The ROC, however, shows a much better AUC at 54%. Therefore, VGG16 can classify with much more confidence compared to the MobileNets and are only 2% under the reference AUC value provided.



(a) Accuracy and Loss for training and validation



(b) Confusion Matrix

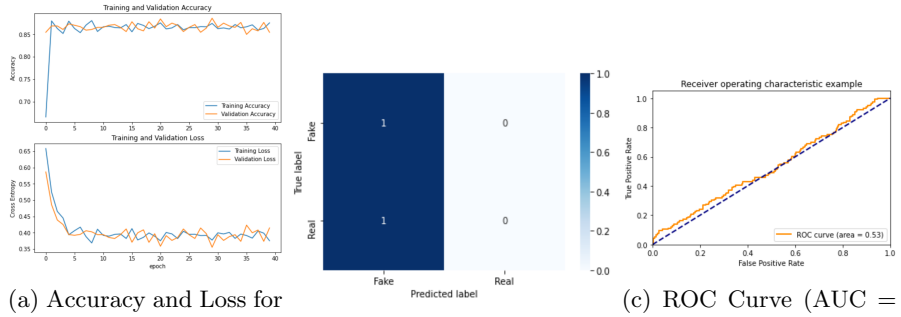


(c) ROC Curve (AUC = 54%)

Figure 3: VGG16 metrics with "Celeb-DF-Image-Face-Crop"

When the Celeb-DF-Image-Face-Crop-SRM dataset is inputted, on the other hand, the initial accuracy is lower at 15% which sharply increases to 50% to 80%. From here the VGG's accuracy stagnates at 87%. The same correlation is true for the validation accuracy, hence, there is no overfitting present within

the training. VGG16 starts with a higher loss value of 0.86 and validation loss of 0.72. The loss exponentially reduces to 0.39 with the curve stabilising around epoch 10. The confusion matrix again is the same where all images are predicted to be fake. Surprisingly, the ROC curve shows a much worse performance compared to VGG16 without SRM filter with an AUC score of 53%. This is just above the average AUC performance of 48%, and significantly worse than the VGG16 + SRM AUC score of 76%.



(a) Accuracy and Loss for training and validation

(b) Confusion Matrix

(c) ROC Curve (AUC = 53%)

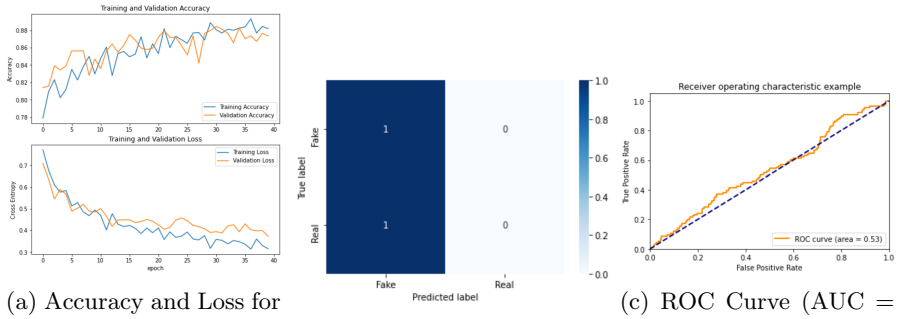
Figure 4: VGG16 metrics with "Celeb-DF-Image-Face-Crop-SRM"

Therefore, VGG16 performs well with the base Celeb-DF-Image-Face-Crop dataset where the AUC score was only 2% below the reference VGG16 Average AUC performance of 56%. However, when the same model is provided with the SRM dataset, the UAC score on average decreases down to 53%, in some cases, it was even 48%. This result is possible due to the difference in pre-processing between the reference paper. The paper proposes no repeats within training. To increase the dataset's size, for each video ten frames are extracted which are also augmented by flipping the images.

2.3 ResNet-50v2 Deepfake Detection

The main base model, ResNet-50, the Celeb-DF-Image-Face-Crop dataset begins with a training accuracy of 78% and validation accuracy of 81%. The training accuracy increases by up to 88% with a final validation accuracy of 87%. As visible from the loss graph, there is a slight amount of overfitting visible compared to the previous training graphs. However, the difference of 1%, hence negligible. The confusion matrix like all previous models, predicts all images as fake. The ROC curve shows majority of the curve is above the false positive section with an overall AUC score of 53%. Hence, the ResNet is only 1% worse compared to the VGG16 and 3% worse from the reference VGG16.

With the Celeb-DF-Image-Face-Crop-SRM dataset, the initial training accuracy of 66%, there is a sharp improvement in the training accuracy to 86%. From here the accuracy stagnates and does not improve. The validation accuracy is initially 85% and does not change throughout the training process with minimal fluctuations. Similar is true for the loss, the training loss begins at 0.65 and drops down to 0.43 in the next step. The validation loss beings at 0.45 and both loss curves reduce to 0.39. The curves show that there is less overfitting compared to previous ResNet model. For the confusion matrix, all images are



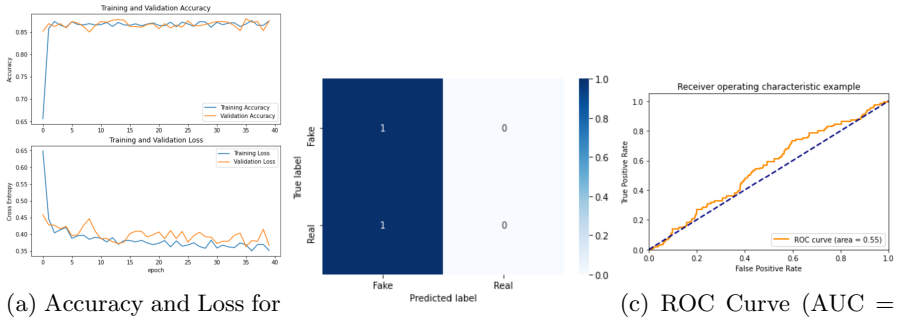
(a) Accuracy and Loss for training and validation

(b) Confusion Matrix

(c) ROC Curve (AUC = 53%)

Figure 5: ResNet metrics with "Celeb-DF-Image-Face-Crop"

predicted as fake. The ROC curve shows there being an improvement in the AUC score to 55%. This is the highest scoring model on average where VGG16 (with no SRM dataset) model had the second highest AUC score of 54%.



(a) Accuracy and Loss for training and validation

(b) Confusion Matrix

(c) ROC Curve (AUC = 55%)

Figure 6: ResNet metrics with "Celeb-DF-Image-Face-Crop-SRM"

Finally, from this experiment, it is evident that the ResNet-50v2 performs well compared to the reference AUC scores where it surpasses the average AUC performance set by the dataset at 48%. The ResNet classifier however is unable to beat the VGG-16 which was proposed to be the optimal classifier for the presented deepfake detection problem. When the VGG-16 was trained with transfer learning, the AUC was 2% higher, however 1% worse than the VGG-16 trained within the reference paper. Interestingly, the ResNet-50 was able to improve on its performance with the addition of the SRM filter where the AUC score improved from 53% to 55%. This improvement wasn't as significant as recorded by the reference paper where the proposed VGG-16's AUC improved from 56% to 73%. Surprisingly, the VGG-16 trained with transfer learning, consistently performed worse with the SRM filter. As proposed previously, this can be due to the difference in pre-processing. However, for the difficulty set by this high-quality dataset, which is known to have the lowest AUC performance, transfer learned, ResNets can beat the Average AUC performance, hence the trained models have a much higher probability in performing better against the weaker datasets.

Table 2: summary AUC performance of different methods

Avg AUC performance		
Methods	**Without SRM**	**With SRM**
MobileNetsV2	51%	51%
VGG16	54%	53%
ResNet50V2	53%	55%

References

[1] X. Chang, J. Wu, T. Yang, and G. Feng, "Deepfake face image detection based on improved vgg convolutional neural network," in *2020 39th Chinese Control Conference (CCC)*, July 2020, pp. 7252–7256.

[2] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-df: A new dataset for deepfake forensics," *CoRR*, vol. abs/1909.12962, 2019. [Online]. Available: http://arxiv.org/abs/1909.12962

[3] L. Shao, F. Zhu, and X. Li, "Transfer learning for visual categorization: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 5, pp. 1019–1034, 2015.

[4] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation," *CoRR*, vol. abs/1801.04381, 2018. [Online]. Available: http://arxiv.org/abs/1801.04381

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," *CoRR*, vol. abs/1603.05027, 2016. [Online]. Available: http://arxiv.org/abs/1603.05027