A CNN Approach for Deepfake Detection

Poornima Bhatt
Department of Information and Technology
Indira Gandhi Delhi Technical Univesity for Women
Delhi, India
bhattpayal030@gmail.com

***Abstract*—In this 21st century Deepfake has become a prevalent practice. Image Forgery is the practice of altering, manipulating, and creating images. It includes deepfake, splicing, enhancement, copy-move forgery and unfortunately, it is often misused for spreading false information and intentionally deceive. Rapid technological advancement have made fake image generation real quick and easy, making it difficult for humans themselves to distinguish. The GANs have made it challenging to discriminate between real and fake images. This has serious implications for media security, legal risks, and many other significant issues. To resolve this problem Deep learning is being used vastly. In recent decades, there has been a significant increase in the development of image forgery detection technology. This paper proposes a simple and effective approach for Deepfake detection. This approach uses CNN(convolutional neural network) with good accuracy to distinguish between real and fake images.**

Keywords—Deepfake, CNN, GANs

I. INTRODUCTION

In this 21st century with advancing technology, Cybercrimes are increasing drastically, and deepfakes are one of them. Nowadays, generating fake images is easier than capturing real ones, and Generative Adversarial Networks (GANs) play a significant role in this process. Image Forgery, it involves altering and manipulating real images to produce realistic fake ones making it hard for the human brain to differentiate, and GANs are an appropriate architecture for generating them. GANs consist of two neural networks: the generator manipulates the original dataset to create fake images, while the discriminator distinguishes between real and fake images to evaluate the model's effectiveness. These fake images have various uses, such as enhancing images. This technology is very useful in space missions, satellite image detection, and creating realistic scenarios for training and control. It can be used to generate images from basic data given by the users, aiding in visualizing their ideas better. However, its misuse is more prevalent, leading to potential negative consequences. It's being used to create false information about individuals, fake illegal content, and more, resulting in security risks, trust issues, rumors, public confusion, and privacy violations. Therefore, it is very important to manage and regulate its usage to prevent these negative impacts.

Deep learning, a very hot topic of the decade is an approach to deal with this malpractice. Deep learning consists of neural networks that train on data to learn and mimic the human brain. Neural networks, not the ones in our brain, but their replica. In terms of deep learning neural networks are the network of interconnected nodes(neurons) that tries to work as human brain, neurons are the basic units that stores, receive and transfer data. CNN is a type of neural network that focuses on processing the grid data such as images and videos, particularly effective in image classification, detection, and segmentation. It consists of multiple layers of neural networks such as the input layer, hidden layer and output layer. The hidden layer is complex consisting convolutional, pooling, activation and normalization layers, these layers process and learn from the data to give good accuracy. In convolutional layer kernel(filters) are applied to the input layer to collect information, recognize patterns, and to reduce data size, then after this data is normalized and pooling is done to get the most information out of the data. After these in the final all layers are connected to activation layers and then to the output layer. In short, this is what happens in CNN.

With the increase in the generation of realistic fake images, it is getting real hard for us humans to detect fake and real ones. This technology is getting misused and to stop and prevent that we need to develop other technology which will help us to differentiate between these two. There are many approaches to it, in the past, it was done manually and with a few basic methodologies like probability distribution for DCT coefficient double jpeg localization, Color filter array (CFA) analysis, illumination model and steganalysis feature classification. CFA analysis was done by checking and analyzing CFA patterns differences between real and fake images. When a JPEG image is altered, then some inconsistencies are created and traces are left in altered regions, and it can be detected by checking DCT coefficient. In the illuminating models technique the lighting and shadow of an image is checked precisely, altered parts of image can have different illumination then the original and this can be used to detect them. Steganalysis feature classification is similar to the above techniques, in this inconsistencies and patterns are analyzed to check for alteration. The above defined techniques were used initially but were not appropriate and had many limitations. These fail when the generated fake image does not have many inconsistencies, and with evolving deepfake technology fewer traces are being left and these evolved models are adapting the pattern recognition, illumination, etc. Hence these techniques are lacking behind the continuously evolving technology.

Figure 1 GANs generated fake images

Researchers are trying to find out some ways to tackle this problem and for this they are deeply diving into this topic, how deepfakes are generating, the whole process and trying to figure out where and how we can create a way to distinguish. Many researches have been done in the past years and deep learning out as an effective approach for tackling these GAN generated images. The purpose is to prevent misuses of image forgery. The objective of this paper is to find an approach for distinguishing fake and real images, and to detect deepfakes.

II. LITERATURE REVIEW

Image Forgery and Deepfake are trending topics for a few decades. There are a lot of researches going on it and a lot have been already done. To distinguish, a lot of experiments has been done using many local traditional approaches and many machine learning based algorithms specially deep learning based are being frequently used and the results are quite good. Traditionally, traces left after image forgery were used for detection but those were not efficient. In a previous study[11] an machine learning algorithm based on Expectation-Maximization was tested by employing to extract the Convolutional Traces (CT) for the fake detection task, in this research the authors traced the CT and distinguished between fake and real images. Another previous study to detect Deepfakes generated through the generative adversarial network (GANs) model via an algorithm called DeepVision was conducted and got 87.5% accuracy rate[12], in this research the authors used DeepVision to analyze a significant change in the pattern of blinking, which is a voluntary and spontaneous action that does not require conscious effort. In year 2022, a survey was conducted to check the efficiency of the deepfake detection methods by summarizing hundreds of article[13], this paper also specifies the limitations, results and challenges in deepfake detection. As CNNs are widely in use for this purpose , a research was conducted using CNN based on diverse Gabor filters for fake detection[14] but in 2023, a better approach towards image forgery detection was published[15], using D-CNN achieving better accuracy. In recent studies GANs are being used for detection of GAN generated images, one of these research conducted in this year use a a novel generative adversarial ensemble learning method to train multiple discriminative and generative networks based on the adversarial learning to improve the discrimination ability rather than image generation one, but the limitation is accuracy, the accuracy is not better. In medical field, image forgery detection is very much important, a recent study on it was published using YOLO based algorithm for detection of fake medical images[17] getting 93% accuracy. In general the res-net50 and dense-net121 also gives accuracy around 93-95%. Our proposed model uses CNN to detect image forgery and give accuracy about 95% over 100 epochs and around 97% after 200 epochs.

- GAN ARCHITECHTURE

GANs are the generative adversarial network, a very powerful tool, was discovered by Ian Goodfellow in 2014[1]. These are kind of neural networks made up of two separate deep neural networks, generator and discriminator. These two work opposite to each other, one generates and other discriminate. These GANs are used to create realistic images, 3D models or images, video and many more. All the problem of security risks, privacy violation, rumors, etc are due to deepfakes and deepfakes are getting generated by GANs. The generator component of GAN is a deep neural network which trains on a dataset and creates similar data(images) to the original data that is indistinguishable. Discriminator is specifically for distinguish correctly between generator's generated data(fake data) and the original data provided. It takes both data as input and gives probability as output, how real the fake data seems. Generator and discriminator both compete with each other.

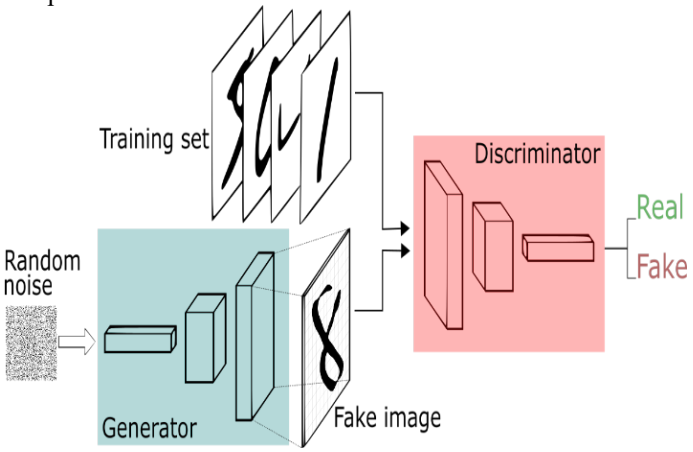


Figure 2 GAN Structure

Generators create replicas of training image and learn from them to create realistic fake images and the discriminator guides it to increase accuracy by giving output by judging the generated image as fake or real. The generator tries to maximize the probability of making the discriminator mistake. Generator's architecture involves many layers including, a input layer, deconvolution layers which increase the spatial dimensions and decrease the feature maps of the input noise vector, normalization layer, activation layer and output layer which generated image vectors. Usually batch normalization and ReLU are used for normalization and activation layer. For output layers in generator generally tanh activation function is used. Architecture of discriminator is also similar but here convolutional layers are used instead of deconvolutional to reduce the spatial dimension and increase feature maps of the input. The activation function used in output layer of discriminator is sigmoid function. The output of generator is the input of the discriminator. Discriminator takes input from both the training set and the generator, evaluates losses and gives probability as the output, if output probability is close to 1 then the input generated image is considered as real image and if it is close to 0 then it's considered fake[2],[3].

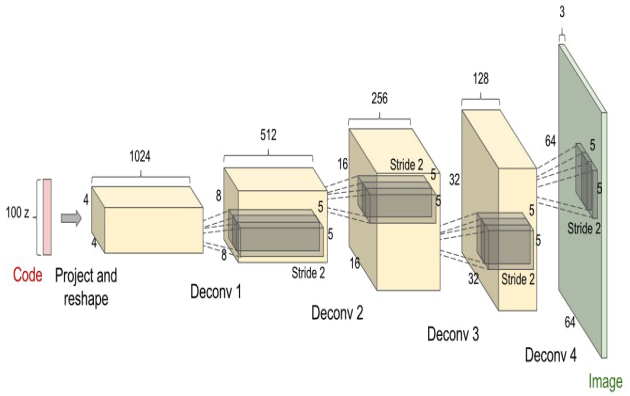


Figure 3 Generator architecture example

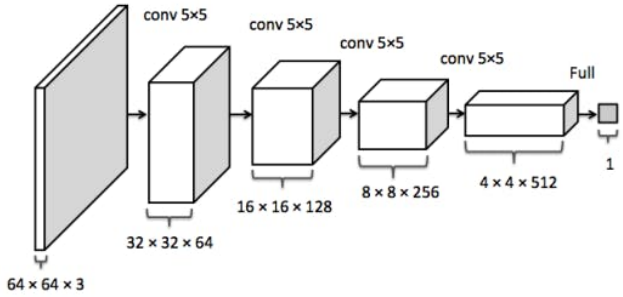


Figure 4 Discriminator architecture example

There are various GANs architectures available, a few popular ones are Vanilla GAN, DCGAN, CGAN, StyleGAN, WGAN, CycleGAN, etc.

- CNN ARCHITECHTURE

CNN is a type of neural network primarily used for analyzing visual data such as images and videos. It is most useful in tasks like image classification, object detection and face recognition. CNN contains multiple layers of neural networks. Its key components are convolutional layers, pooling layers and flattening layers. These are the most important components of cnn which makes them unique. In convolutional layer convolution operation is applied on the input vector/matrix(image or video in form of 0 or 1), there are kernels(feature detectors) these are matrices containing 1s and 0s these feature detectors are applied on the input and a feature map is generated as an output. These feature maps are the reduced form of the input and contains major relevant informations, patterns, etc. There are lots of feature maps generated. Multiple convolutional layers are deployed to get better results and maximum informations out of the input data. Convolutional layer also includes activation, ReLU activation function to increase non-linearity in data(images). Then after convolutional layer pooling is done, pooling is a technique to reduce input's spatial dimensions and increase computational efficiency by preserving the most relevant informations, mainly used pooling techniques are max pooling and average pooling. In max pooling, maximum is considered and in average pooling average is considered. Afterwards flattening is done, as the name suggests it is used to flatten our pooled feature map to a single column map. After this the model is connected to the activation, hidden and output layers

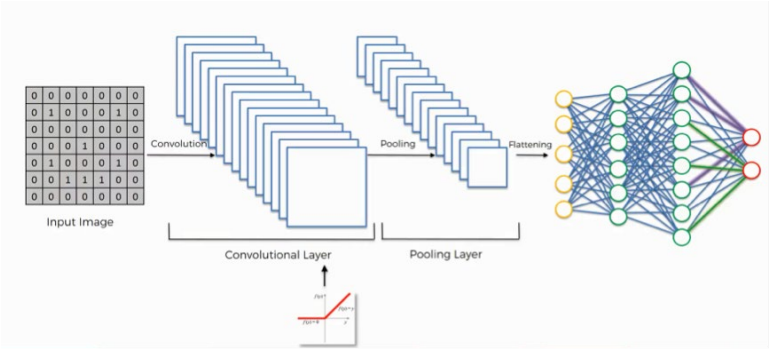


Figure 5 CNN Structure

III. METHODOLOGY

In this paper, an CNN Approach for deepfake detection is proposed. CNN is very effective in object detection and classification, thus used to detect deepfakes. In this detection model, proper steps are followed to clean, train and test data. The dataset used here is "140k Real and Fake Faces" version 2[4]. The dataset contains 70k real faces from Flickr dataset collected by Nvidia and 70k fake faces sampled from the 1 Million FAKE faces (generated by StyleGAN) provided by Bojan. The dataset has 3 directories test, train and valid. Each directory have 2 internal directories real and fake consisting real and fake images separately. Test and valid directories contain 10k real and 10k fake images, and train contains 50k each in real and fake image directories. This dataset have a large number of fake and real images, which will make the training process effective. Following are the steps:

- Importing Data
- Data Preprocessing
- Building Model
- Model Compilation
- Training
- Testing And Evaluating
- IMPORTING DATA

The dataset used is obtained from Kaggle.com using the Kaggle API. First the "kaggle.json" file was downloaded which contains necessary credentials and imported, then the dataset "140k Real and Fake Faces" was downloaded as a zip folder and then unzipped next. The dataset was organized in train, test and valid directories.

- DATA PREPROCESSING

The next is to preprocess the data imported. The required libraries were imported such as pandas, matplotlib, numpy, tensorflow and keras. Then ImageDataGenerator a class of keras library under tensorflow was deployed to preprocess images in the dataset.. The training set images were rescaled-all the pixel value were divided by 255 to get into a range of [0,1], sheared-slightly slanted to avoid geometric distortion, zoomed and horizontally flipped to generate new images to improve the model's training and robustness. The test set images were only rescaled by a factor of 1/255. Then the preprocessed images were loaded using the "flow_from_directory" function from the "ImageDataGenerator" class, resized to 64*64 pixels and in a batch of 32 means 32 images will be used to train at a time. Binary class mode was used to classify as real or fake.

- MODEL ARCHITECTURE

After preprocessing the data, CNN model is built. The CNN architecture includes multiple layers. The following are the designed layers[5]:

1. Convolutional layers: Convolution is a mathematical term, basically it's a operation which operates on two functions to produce a third function(f*g).

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$$

Figure 6 Convolution formula[5]

Graphically it describes how the shape of one function is modified by other.
So basically a convolutional layer work exactly same as above, it reduces the size of the input metric of the pixels of the input images. There are 3 main components in convolution operation- input image, feature detector and feature map. Input image is in the form of 0s and 1s matrix, can be a 2d(greyscale image) or 3d(RGB image) matrix. Feature detector is also a matrix of 0s and 1s of given shape, generally 3. A feature detector is also known as a "kernel" or a "filter". A feature detector is placed over the beginning of the input image from the top right corner as shown in Figure 7, then the number of cells in which the pixel value is same are counted. Then the number of counted cells are inserted in the feature map from the topleft corner. We then move to the next cell to the right of the input image within the boundaries, this is a unit stride, again counted the number of cells having same values and inserted in feature map next cell to the right. After a row gets completed move onto another row means one cell downward and repeat the process until the whole matrix completed. Similarly we created multiple feature maps with multiple kernels as shown in Figure 8.

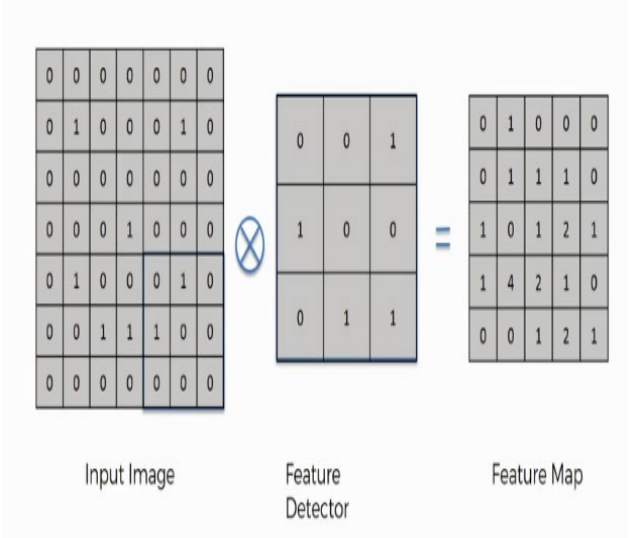


Figure 7

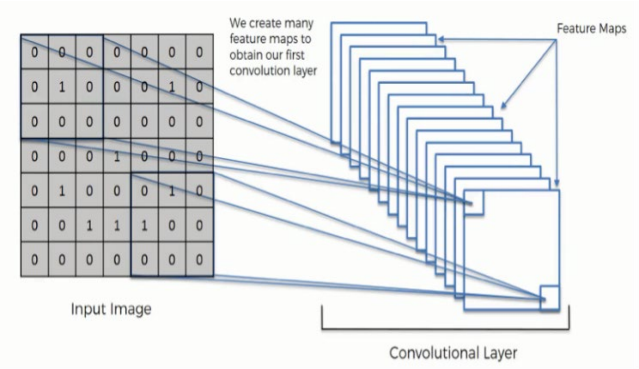


Figure 8

Convolution is done to reduce data size and maximize the information that could be gained from the data, its like sifting rice from paddy. Here, its shown with a greyscale(2d) image but similar happens with an RGB image, only difference is that for a single image there are 3 matrices thus, the number of feature maps get increased in numbers and dimensionality. The convolutional layer also include normalization and applying Rectified Linear Unit(ReLu) function to the feature maps to increase non linearity in the data. In the proposed model batch normalization is used frequently in convolutional layers to normalize

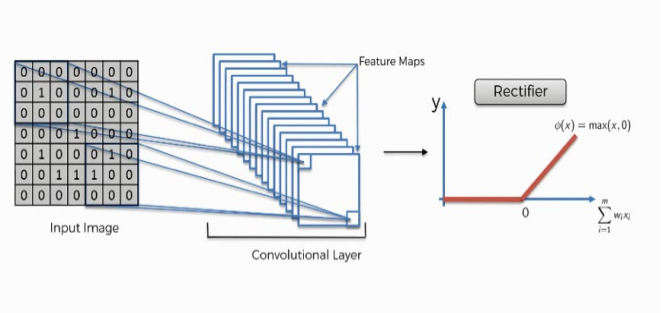


Figure 9

2. Pooling layer: It is done to reduce spatial dimensions of the feature map, computational load and overfitting. Spatial Variance, this property makes the network capable of detecting the object in the image without being confused by the differences in the image's textures, the distances from where they are shot, their angles, or otherwise. Pooling basically handles this spatial variance.
   there are several types of pooling techniques such as max pooling, average(mean) pooling and sum pooling. In this model, max pooling and average pooling are used. The process of pooling is similar to convolution but here we do not take any kernel, for example, a 2*2 box is placed on the top left corner in the boundaries of input image matrix and then the maximum in max pooling and average in the average pooling of the four cells is stored in the top left corner cell of pooled feature matrix, after this move two cells right, that is stride=2, when the row is finished move a cell downwards and repeat from the start of the new row. Repeat the process until the matrix is finished. The reason we extract maximum or average is to discard unwanted features and to collect the most important features from the

image, suppose there are different images of dog from different angles so to identify as the same dog max pooling is important, after max pooling the same features or the most common features will be extracted from all those images, thus providing spatial variance capability. It prevents overfitting. So the pooling layer is made up of pooled feature maps. To get a better idea check out the number game [6].

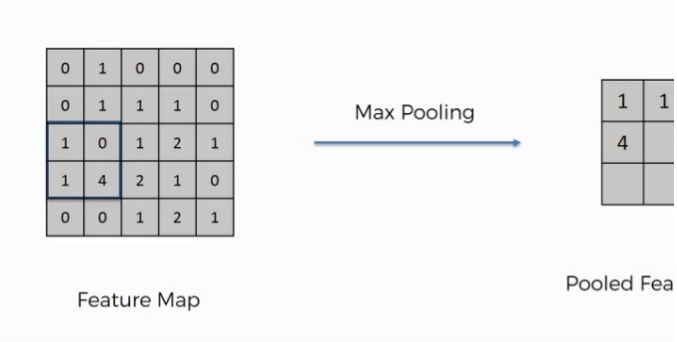


Figure 10

3. Flattening: After the pooling layer, concentrated matrices are extracted and then the matrices need to be flattened. As the name suggests, "flattening" means to flatten the pooled feature matrices i.e. reducing the spatial dimensions and converting them to a column as shown in Figure 11. This step is required because the Dense input layer of the ANN fully connected layers after flattening part of the CNN model expects its input to be a 1d vector figure 11. The flattened matrices will be the input layer of the CNN model.

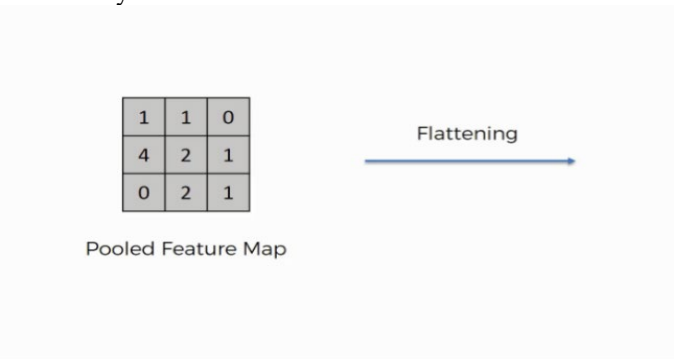


Figure 11 the flattened pooled feature map is the input for the next fully connected layer

4. Full Connection: the input layer contains the vector of data that was created in the flattening step. The features that we distilled throughout the previous steps are encoded in this vector. At this point, they are already sufficient for a fair degree of accuracy in recognizing classes(real or fake). We now want to take it to the next level in terms of complexity and precision. Thus, after the Flattening layer, an ANN consisting of an input layer, hidden layers and an output layer is deployed. The flattened feature map is the input for the ANN, here it is not different than the CNN, ANN is the part of CNN after the flattening layer here. After flattening the CNN is fully connected with external layers i.e. ANN as shown in Figure 12. As we work to optimize the network, the information keeps flowing back and forth over and over until the network reaches the desired state, and that's how the model will be trained.

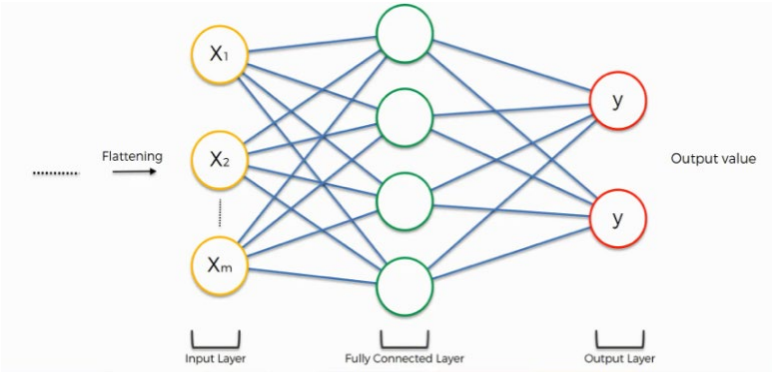


Figure 12

- MODEL COMPILATION AND TRAINING
  After the fully connected layer, the model need to be compiled. Model compilation is the step for training the model, in the above steps we prepared the data, architectured the model and now it's the turn for training and putting all the above steps together. This step includes specifying optimizers, loss function, loss weights, metrics and weighted metrics. Optimizers determine how the weights will be updated during training to minimize the loss. It is an important argument to train an efficient model. Common optimizer algorithms are SGD (Stochastic Gradient Descent), Adam, etc. The loss function measures the difference between prediction and actual target. Some loss functions are mean squared error, mean absolute error, binary crossentropy, Sparse categorical crossentropy, etc. Metrics are to evaluate the accuracy, score, etc., basically metrics are used to evaluate the model performance, these monitors and display how well the model's doing. Here in this proposed model "Stochastic Gradient Descent" is used as optimizer as it deals better with large dataset, and the "Sparse Categorical Crosssentropy" for the loss function.
  SGD and sparse categorical crossentropy with dropout and normalization works better for diverse inputs and is better for detection tasks. After compiling, model need to be finally trained, by giving model_name.fit() command with required arguments.
- PROPOSED MODEL

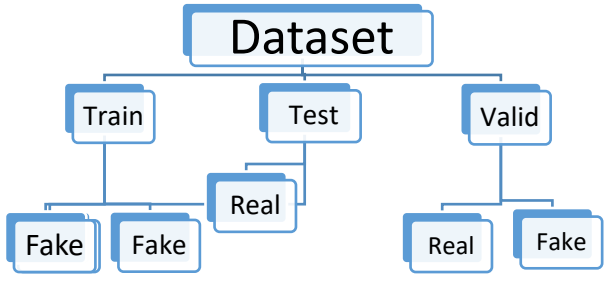


Figure 13 dataset used: "140k Real and Fake faces"

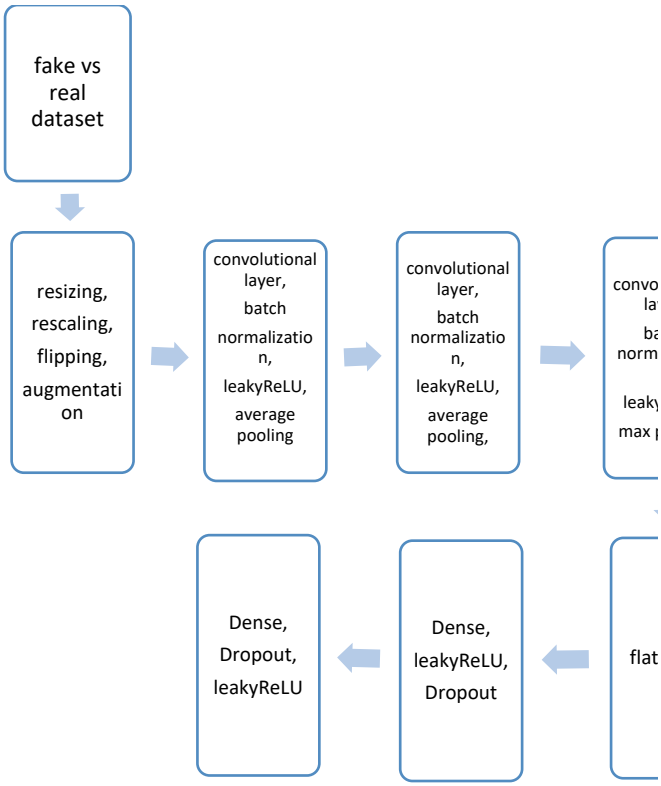


Figure 14proposed model architecture

IV. Results

In this section results of the proposed model are discussed and the terminologies required are explained. There are different parameters to evaluate a model on such as accuracy, f1score, recall, precision and confusion matrix

Confusion matrix: Confusion matrix is an essential tool for checking a model's evaluation. It contains TP, FP, TN, FN. In a binary classification model having two classes, it's of 2*2, but for more classes let's say n classes it's of n*n.

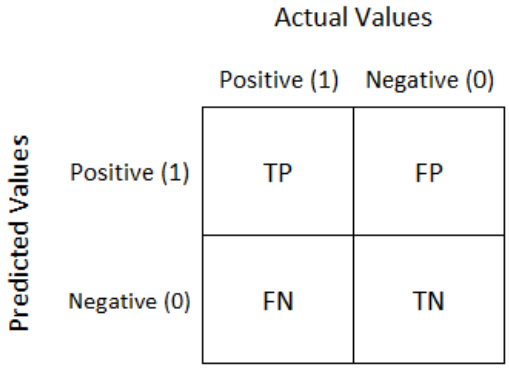


*Figure 15 A 2*2 confusion matrix*

Accuracy: It means the percentage of correct predictions made by the model out of all the predictions.

Accuracy= (TP+TN)/(TP+FP+FN+TN)

Precision is the ratio of the TP (true positive) predictions to the total number of positive predictions by the model.

Precision= TP/(TP+FP)

Recall is the ratio of true positive predictions to the total number of actual positives.

Recall= TP/(TP+FN)

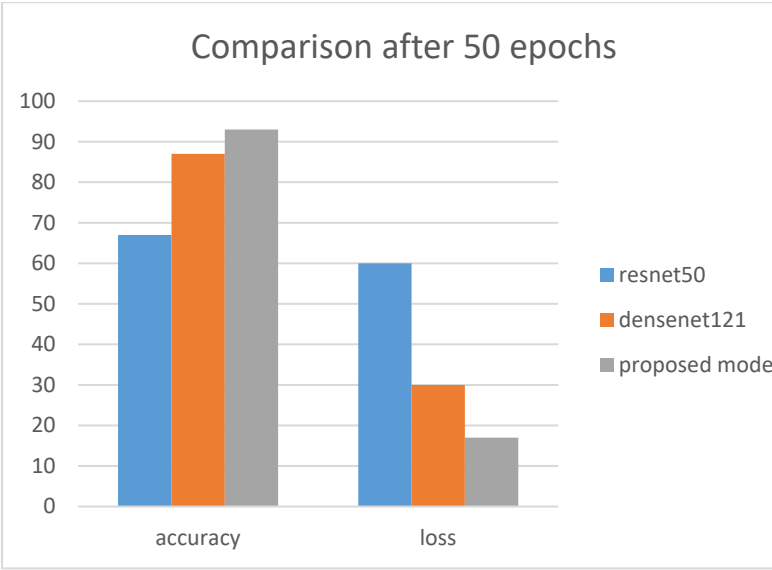
F1 score is the harmonic mean of precision and recall, providing a balance between the two, especially in cases where precision and recall are in tension.

F1 score = 2/((1/precision)+(1/recall))

= (2*recall*precision)/(precision+recall)

This research aims to detect deepfake images with high accuracy and robustness. As the GAN which is responsible for deepfake generation has the mechanism to improve performance on the basis of the output given by the discriminator so it becomes very difficult to identify fake images hence, the research is conducted. In this research, CNN models with different architectures were tested on the dataset and based on the accuracy, the model is proposed.

The proposed model was compared with other deepfake detection methods. DenseNet121 and ResNet50 were trained on the same dataset with same preprocessing of data over same epochs and as a result the proposed model performed the best.



The proposed model in is an deep learning CNN model with multiple layers described in the above sections. As CNN models are known for their simplicity and high accuracy in deepfake detection, our model also have good accuracy and not overfitted. The model is evaluated on various parameter and graphs are plotted as shown in Figure 16. The model is trained under 100 epochs on google colaboratory with T4 GPU and give an accuracy of 0.9512.

Table 1 epochs accuracy and loss

epochs	accuracy	loss	val_accuracy	val_loss
20	0.9146	0.2094	0.9157	0.2134
30	0.9238	0.1915	0.9244	0.1905
60	0.9406	0.1529	0.9344	0.1647
100	0.9531	0.1225	0.9524	0.1273
150	0.9617	0.0984	0.9524	0.1273

As shown in Table 1 with increasing epoch the loss is getting reduced and accuracy is increasing. The models efficiency and effectiveness improved steadily throughout the training process. Over after 100 epochs accuracy reached upto 0.9512 and it took approximately 4 hours to train this much. The F1 score, precision and recall are given below in the table 2.

Table 2 classification report after 100 epochs

	Precision	F1 score	Recall	support
Real	0.95	0.95	0.95	10000
Fake	0.95	0.96	0.96	10000
macro avg	0.95	0.95	0.95	20000
weighted avg	0.95	0.95	0.95	20000

The classification matrix as shown in Table 2 gives precision to be 0.95, f1 score as 0.95 and recall as 0.95. These scores are excellent and better than the existing models. The more the scores are close to 1 better the model is.

The confusion matrix gives an overall overview of how much correct and wrong the model is predicting. It is the most important tool for evaluating a neural network. It's diagonal gives the measure of correct predictions (i.e. TP, TN).

Figure 16 shows the confusion matrix of the proposed model after 100 epochs. The TP(True Positive) are the number of images which were predicted as real and were real in actual, TN(True Negative) are the images which were predicted as real but were fake in actual. FP are the real images but predicted as fake and FN are the images which were fake in real and were predicted as fake. In the matrix the diagonal elements are in the darkest shade of blue hence it shows that more than 8000 images were correctly predicted in out test dataset of 20000 images. In the matrix it is shown that how many images are predicted accurately(add the diagonal elements), 19070 and how many are incorrectly predicted, 930 here.

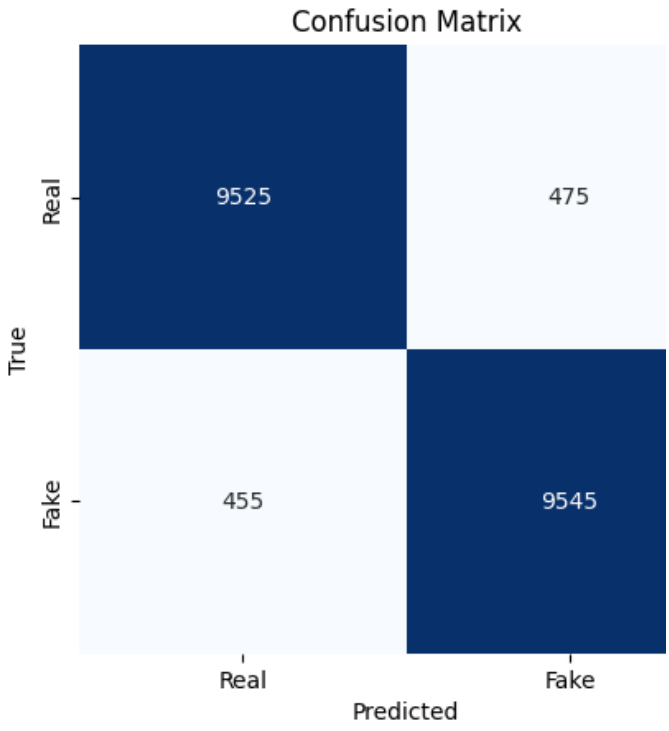


Figure 16 Confusion matrix

ROC, Receiver operating characteristic curve is used to evaluate binary classification models. It is a great tool to check how good the binary classification is working in a model. It is a graphical representation that plots TPR(Total positive rate) and FPR(False positive rate) over varied thresholds. TPR or recall is the ratio of TP to the sum of TP and FN, i.e., the proportion of actual positives identified by the model. FPR is the ratio of FP to the sum of FP and TN, i.e., the proportion of actual negatives identified by the model. AUC, area under the ROC curve is a value that summarizes the performance of the model. AUC value closer to 1 is considered excellent. The AUC value is 0.99, it denotes that the model is performing excellently with very high accuracy.

Table 3

FPR	FNR	TPR	TNR	Log loss
0.0475	0.0455	0.9545	0.9525	0.1213

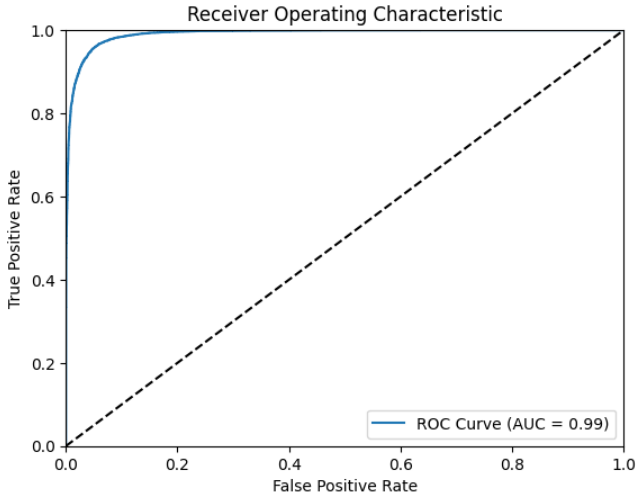


Figure 17 ROC curve

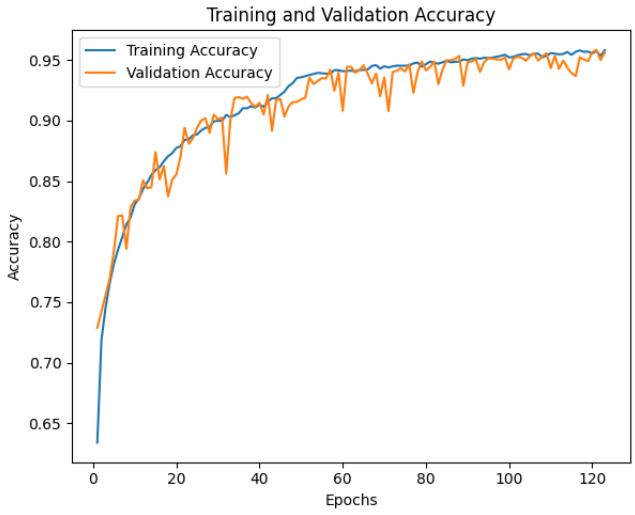


Figure 18(a)

Figure 18(b)

V. Conclusion

In this research, a deep learning model is introduced to tackle the rising problem of deepfakes, an threat to cybersecurity, in an era of emerging technology of image forgery and increasingly sophisticated picture alteration techniques. In order to develop a reliable and precise method for distinguishing real images from fake images , the study made use of the CNN architecture. On a large dataset of 100000 images, the suggested model performed excellently, obtaining high precision, recall, F1-score, accuracy, and ROC-AUC score. These findings highlight the model's efficiency and effectiveness in the critical task of detecting forged images. this can be used in many fields, such as cybersecurity, identity verification, and social media content control to solve the issues ethically. In these domains, the ability to discriminate between real and altered faces is crucial, and our model provides a potent tool for tackling this problem. Future research in the field of fake face detection should focus on a few crucial areas to further improve the capabilities of deep learning models specially CNNs. For example, new deep learning architectures should be investigated, and optimization techniques should be investigated to increase the model's precision, recall, and overall accuracy. Future work could certainly explore cross database evaluations to further validate the generalizability of the proposed model across different datasets and scenarios.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio ``Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672_2680.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] Marco Del Pra, "Generative Adversarial Networks," Medium. [Online]. Available: https://medium.com/@marcodelpra/generative-adversarial-networks-dba10e1b4424.

[3] Chanchana Sornsoontorn, "How do GANs intuitively work?," Hackernoon. [Online]. Available: https://hackernoon.com/how-do-gans-intuitively-work-2dda07f24/7a1.

[4] Xhlulu,"140k Real and Fake Faces," Dataset, 2019. [Online]. Available: https://www.kaggle.com/datasets/xhlulu/140k-real-and-fake-faces.

[5] Superdatascience team**, "**The Ultimate Guide to Convolutional Neural Networks(CNN)," SuperDataScience, Aug 28, 2018. [Online]. Available: https://www.superdatascience.com/blogs/the-ultimate-guide-to-convolutional-neural-networks-cnn.

[6] Adam Harley, "3D convolutional network visualization", adamharley.com, [Online]. Available: https://adamharley.com/nn_vis/cnn/3d.html.

[7] A. W. Harley, "An Interactive Node-Link Visualization of Convolutional Neural Networks," in ISVC, pages 867-877, 2015

[8] S. Safwat, A. Mahmoud, I. Eldesouky Fattouh, and F. Ali, "Hybrid deep learning model based on GAN and ResNet for detecting fake faces," IEEE Access, vol. 12, pp. 3416910, June 2024, doi: 10.1109/ACCESS.2024.3416910.

[9] Q. Zhou, Z. Zhou, Z. Bao, W. Niu, and Y. Liu, "IIN-FFD: Intra-Inter network for face forgery detection," Tsinghua Science and Technology, vol. 29, no. 6, pp. 1839–1850, Dec. 2024, doi: 10.26599/TST.2024.9010022.

[10] A. Malik, M. Kuribayashi, S. M. Abdullahi, and A. N. Khan, "DeepFake detection for human face images and videos: A survey," IEEE Access, vol. 10, pp. 123456-123469, Feb. 2022, doi: 10.1109/ACCESS.2022.3151186

[11] L. Guarnera, O. Giudice, and S. Battiato, "Fighting Deepfake by Exposing the Convolutional Traces on Images," IEEE Transactions on Information Forensics and Security, vol. 17, no. 3, pp. 789-802, Mar. 2022.

[12] T. Jung, S. Kim, and K. Kim, "DeepVision: Deepfakes Detection Using Human Eye Blinking Pattern," IEEE Access, vol. PP, pp. 1-1, Apr. 2020, doi: 10.1109/ACCESS.2020.2988660..

[13] A. H. Khalifa, N. A. Zaher, A. S. Abdallah and M. W. Fakhr, "Convolutional Neural Network Based on Diverse Gabor Filters for Deepfake Recognition," in *IEEE Access*, vol. 10, pp. 22678-22686, 2022, doi: 10.1109/ACCESS.2022.3152029.

[14] Y. Patel, S. Tanwar, P. Bhattacharya, R. Gupta, T. Alsuwian, I. E. Davidson, and T. F. Mazibuko, "An improved dense CNN architecture for deepfake image detection," *IEEE Access*, vol. XX, pp. 1-10, Mar. 2023, doi: 10.1109/ACCESS.2023.3251417.

[15] J. Zhou, X. Zhao, Q. Xu, P. Zhang, and Z. Zhou, "MDCF-Net: Multi-Scale Dual-Branch Network for Compressed Face Forgery Detection," IEEE Access, vol. PP, no. XX, pp. XX-XX, Apr. 2024, doi: 10.1109/ACCESS.2024.3390217.

[16] M. Karaköse, H. Yetiş, and M. Çeçen, "A New Approach for Effective Medical Deepfake Detection in Medical Images," IEEE Access, vol. PP, no. XX, pp. XX-XX, Apr. 2024, doi: 10.1109/ACCESS.2024.3386644.