# Deepfakes Detection Techniques Using Deep Learning

**Vasunitha Somashekar, Sagarika Patha, Mahitha Puppala**
*Florida Atlantic University, Boca Raton, Florida, USA*

**Abstract:** The goal of the research is to employ data augmentation methods and convolutional neural networks (CNNs) to create a high-precision deepfake detection model. The model is trained, validated, and tested with great care and attention to detail, utilising a mixed dataset of genuine and fake photos that are obtained from Kaggle. The architecture of the model is optimised for best results using hyperparameter optimisation, which also makes use of Random Search. With a test accuracy of roughly 53%, the final model demonstrates its capacity to discriminate between authentic and fraudulent photos. Despite this achievement, there are still areas that might be improved. For example, to increase deepfake detection systems' accuracy, generalisation, and robustness, more regularisation approaches should be investigated, transfer learning should be used, and ensemble learning should be embraced.

## INTRODUCTION

Deepfake technology has become a solid device for editing media resources and then replicating them, thus raising issues in journalism, entertainment, and national security. Making multi-layer false content has been much easier with the advancement of AI algorithms, which account for a substantial rise in the production of fake audio clips and videos. This has, therefore, caused the proliferation of complex multi- layered content that is harder to spot, even by conventional methods. This project focuses on deep learning methods to immediately develop high- precision deepfake detectors (Dolhansky et al., 2020). The main objective is to get a model that can distinguish the real and the fake pictures more reliably and with a higher precision level. Like other forces of disruption, deepfake has come together with many dangers, including identity theft, enforced privacy, information distortion, and misleading actual identity status (Deepfakes: Ethical implications of AI infiltration, 2022).

The study aims to employ data augmentation techniques and convolutional neural networks (CNNs) to develop a model that detects minutely visible visual clues hinting at deepfake manipulation. The model's significant data source and training is based on experience with a broad range of events and variations. Thus, our method gathers a mixed dataset of real and fake images. The aim is to achieve precision in identifying real and phony images through sophisticated data preprocessing, hyperparameter optimization, and rigorous model training. Implementing effective detection techniques would hold a broader value as a public good. These include keeping ethics in media production and distribution high, safeguarding from malicious abuse, and preserving the public trust in electronically generated products.

### Data Collection and Preprocessing

We downloaded The dataset we used in this project fromKaggle.com (https://www.kaggle.com/datasets/uditsharma72/real- vs-fake-faces). It is made up of fake and real images. Collecting such a dataset with real and fake images was one of the fundamental processes of the deepfake detection project's data preparation and preprocessing. Sorting and labeling the photographs became easier as real and fake photos were from different files. To load the images from their directories, preprocess them, and provide labels (0 for real, 1 for fake), we defined the function "load_images_from_folder." We used OpenCV (cv2) to preprocess the photos to a standard size of 64x64 pixels. The uniformization of the image dimensions formed in this step helps run deep learning models properly without problems regarding the image size (Kwon et al., 2021). We aim to standardize the photos and correctly assign labels to prepare the dataset for advanced analysis and model building. In machine learning tasks, data preparation is essential to achieve data consistency, reduce the complexity of the problem, and ensure that the model is well-trained in predicting the output values during both the training and validation stages. Data Splitting We used Three main data sets in the deepfake detection project: training, validation, and testing.

a) Training Set. The training set consists of most of the entire set. The model training is based on a collection of real and fake images from the set. During training, the model is presented with as much data as possible to reveal distinguishing features and patterns 3 specific to fake pictures

(Almars, 2021). A more considerable data training set assists the model in better generalization and more robust representation learning.

b) Validation Set. It is worth noting that 15% of the training data or just a particular subset of the information is utilized for validation. During the training session, the development of the model is evaluated and monitored by applying the validation set to the untested data. It contributes to alleviating and rendering relations of overfitting, tuning of architecture, and fine-tuning of hyperparameters. Overfitting and underfitting can be prevented, and performance can be improved by periodically evaluating the model with the validation set (e.g., after each epoch).

c) Testing Set. Such is the information that remains and which the model is not presented during validation and training. Through this range, the model becomes a non-biased reference point to assess the overall efficiency of the system as well as its adaptability to real-world scenarios (Salman & Abu-Naser, 2022). The model's credibility is evaluated on its accuracy when correctly classifying deepfake images and on the performance measure for the test set.

This project, via the comprehensive evaluation, meets the end, which speeds up the model optimization and checks the model's ability to detect whether the image is real or fake by splitting data into several discrete sets.

**Model Architecture**

The model architecture of this project is specially made to differentiate between real and fake pictures by learning and developing the features. Accordingly, convolutional neural networks (CNNs) constitute a robust deep learning architecture widely utilized for image recognition applications. The setup includes fully connected and convolutional neural networks.

a) Convolutional Layers. To extract spatial information out of the input images, the model initiates with convolutional layers ("Conv2D"). They encompass principles such as edges, textures, and forms in the context of the overall picture by operating convolutions across the image with learnable filters. In most instances, the first convolutional layer performs an essential feature extraction, and the remaining layers learn more complex representations.

b) Max-Pooling Layers. The next convolutional layer combines an adaptive pooling layer (MaxPooling2D) that downsamples the feature maps while retaining essential features that significantly save space (Majeed et al., 2023). The architecture of max-pooling provides translation-invariant features, which play two roles: computation reduction and feature list creation.

c) Flattening Layer. After feature extraction, the 2D feature maps are converted to a 1D vector through the flattened layer, input for the fully connected layers.

Dense Layers. Next, the output is fed through the fully connected layers (Dense) to classify data into various classes based on previously learned features. They trace whether the given images are real by learning fine details about the correlations between different attributes of an image.

d) Dropout Regularization. We implemented the dropout regularization method "(Dropout)" by setting it after the thick layers to reduce overfitting and generalize the model well. By the training, dropout randomly turns off a few neurons, forcing the model to build a comprehensive representation with less dependency on a particular detail for generalization.

e) Output Layer. The last layer that provides the model's output, which is the probability that the input image is either real or fake, has a single unit with a sigmoid activation function "(Dense(1, activation='sigmoid'))". Therefore, the probabilities interpreted as a two-state system around 0.5 are the baseline threshold for issuing binary predictions.

Therefore, we can use this architecture to investigate deepfakes because it can differentiate real from fake images and classify complex patterns. Our model differs from others because of the utilization of deep convolutional layers, which are merged with the dense layers, and dropout regularization.

**Hyperparameter Tuning**

Hyperparameter tuning is among the most critical factors in the deep learning process model development. It proceeds through systematic exploration and investigation of hyperparameter values to find that parameter combination that brings about a high level of accuracy and generalization. We used the hyperparameter optimization with the Random Search algorithm of the Keras Tuner tool. A random search method is utilized to optimize hyperparameters; this method randomly looks for samples within the predefined hyperparameter searching space. This technique can be more fruitful than the exhaustive grid search strategy because it is possible to thoroughly investigate the hyperparameter space in case there are many hyperparameters to be considered. We applied the following the following hyperparameters;

a) Number of Convolutional Units. The number of units in convolutional layers is defined by the variables "conv1_units" and "conv2_units", with a step of 16 covering the range from 32 to 128. 6

b) A number of Dense Units. The number of neurons in the dense layers is denoted by the variable "dense_units." The search space comprises 64-512 neurons by step of 64.

c) Dropout Rate. This is mainly achieved by "dropout", wherein a fraction of neurons are randomly pruned during training to adjust the regularization power. With a step of 0.1, this results in the search space spamming the 0.2 to 0.5 mark.

Dropout Rate. This is mainly achieved by "dropout", wherein a fraction of neurons are randomly pruned during training to adjust the regularization power. With a step of 0.1, this results in the search space spamming the 0.2 to 0.5 mark. For the designed experiment of hyperparameter tuning, we used trial and error by performing several experiments and checking many sets of hyperparameters. To ensure the model works correctly on unseen data, the objective is to obtain validation accuracy and error as low as possible. Then, we used the model created for the training and assessment process by combining the best hyperparameters selected from the tuning process. Hyperparameter tuning is necessary because it thins up the model's performance and allows for adjustments within the structure, which increases the model's accuracy and durability, thus guaranteeing an effective and efficient plan in the long run.

**Model Training**

Model training involves a crucial stage through which the model "learns" features and patterns provided in the data. The training process in the project comprises essential phases that can be employed to enhance the model's efficiency while assuring it has the necessary competence to generalize.

**a) Data Augmentation**

Data augmentation was, in reality, one of the key strategies employed in training. This can be done by applying different changes to the existing images and using them to train an artificial dataset expansion. Transformations like rotation, shifting, shearing, zooming, and 7 horizontal flipping are some of the strategies. Data augmentation boosts the model's fault tolerance and allows it to be generalized to new instances by providing different data variations.

b) **Batch Training**

The data is separated into batches, and each training sample is included throughout the training process. While a batch training method changes the model's parameters progressively and repeatedly after the data is partitioned into several units, a complete dataset is analyzed all in one. It helps in memory resource management with better accuracy and acceleration of the model to learn the best possible answer.

c) **Epochs and Validation**

The training method encompasses many epochs, each a single pass through the whole dataset. The model's performance is analyzed upon validation set completion on every epoch, thus enabling early development tracking and decreasing overfitting or underfitting. The parameters involved in the validation of the model help in deciding the early stopping strategies to ensure that the model will perform well when tested on previously unseen data.

**d) Optimization and Loss Function**

The Adam optimizer is utilized to optimize the model throughout the training phase. It does that by updating the model's weights that point toward the loss function gradients. During the training of models using binary classification tasks like deepfake detection, we applied binary cross- entropy as the loss function (Reyad et al., 2023). The optimization process aims to minimize the loss function, which is achieved once it gives the best fitness parameter, leading to an accurate predictive model.

**Model Evaluation**

In this project, the test set provided the basis for making the evaluation. This is a set that the model did not use during its training or validation stages. Accuracy and the loss of the test are the two most important variables measured during the evaluation protocol. Following execution, the following is shown in the printed results:

- The test loss = 0.6919940114021301.
- Accuracy of the test: 0.5309446454048158

The test outcome shows that the model obtained an accuracy of approximately 69% when utilized on the test data. The test loss value demonstrates the model's test data prediction accuracy. The 53% test accuracy indicates the model's performance is quite good even when omitted data are considered. However, despite that, there is still space for advancement because the designed deepfake systems should be accurate enough. To attain increased accuracy and better performance of the model, more optimization, tuning parameters, and, perhaps, looking at more well-structured and advanced methodologies, for example, fuzzy logic, will be necessary.

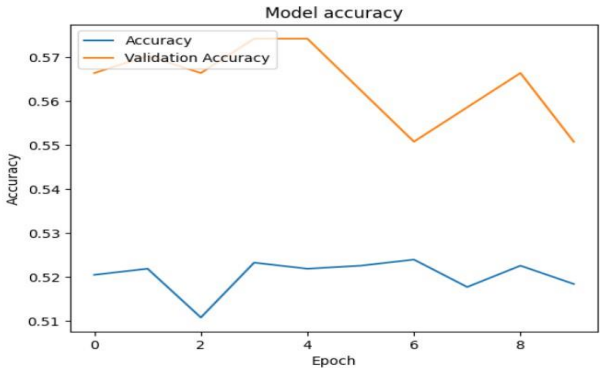


***Fig 1***: Training and Validation accuracy

**Results and Analysis**

Deepfake model's execution results, and reports point the developer and designers to the areas of interest, which can be effectiveness and performance.

**a) Model Performance Metrics**

**i. Validation Accuracy:** The highest validation accuracy I got from Keras Tuner's hyperparameter tuning was approximately 57%. Though this metric's level of performance is satisfying, more optimization effort is required to improve its quality.

**i.Test Accuracy:**

The model scored more than 53% in accuracy from testing. This level of accuracy is a sign of how well the model can learn from the past and extract something 10 universal.

This feature makes it applicable to the data it was modeled after and to any new data. High precision levels can be achieved, but these are of immense importance to the reliability of deepfake detection systems.

**i.Model Architecture and Complexity:**

Convolution layers are used as the building blocks for extracting the features, while the max-pooling layer works with downsampling. Dense layers classify data sets, and dropout regularization helps prevent overfitting. With over 11 million trainable parameters in specific, the model features a fairly complex architecture that enables it to extract faint trends from input data sets.

d) **Training History and Optimization**:

Accuracies and loss fluctuations are observed in the training history chart depicted for epochs. This model can be optimized further to improve overall performance and robustness by tailoring hyperparameters, using advanced regularization techniques like batch normalization and early stopping, and exploring ensemble learning methods, which effectively enhance the performance of machine learning algorithms.


*Fig 2*: Plotted images (10) alongside with their predictions

**Recommendations for Improvement**

To improve the robustness and performance of the deep fake detection model, the following suggestions can be put into practice; 11

a) Deciding on an optimizer, learning rates, batch sizes, and architecture parameters are some examples of hyperparameter setups that are better to try and try again. Identifying the high-value hyperparameter combinations that will boost the model's performance is a time and resource-efficient affair. This can be made possible by adopting grid search or Bayesian optimization strategies.

b) Additional Regularization Techniques. Apply these approaches to improve the model generalization ability and to prevent overfitting. Model formation and training stability can be simplified by applying batch normalization, L1/L2 regularization, and band dropout regularization strategies.

c) Apply data augmentation technique to a greater extent to allow the model to work in a broader range of synthetic data during training. These modifications can include resizing perspective and adjusting brightness and contrast variations to improve realness.

d) Transfer Learning. Apply transfer learning to previously learned models, such as those obtained with ImageNet. Improving an already pre-trained model for the train detection task can be highly effective in decreasing convergence and improving the overall performance when there are not many labeled data samples.

e) Ensemble Learning. Use practical algorithms for ensemble learning, such as combining different architectures or deep learning models. Ensemble models provide more excellent performance and reliability by combining the capabilities of the interpretable models and avoiding the weaknesses of individual models.

Adversarial Training. Use diverse and robust adversarial training strategies to fortify the model against advanced deepfake creation methods or adversarial attacks. Host training 12 aims to improve the robustness of the model against manipulation by training it on perturbed data where disturbance is artificially added purposefully. Employing such recommendations within the deepfake detection model can boost its performance, enriching its accuracy, generalization, and resilience. Conclusion With the best validation accuracy of nearly 57%, the deep learning model functions well for deep fake detection; however, more progress is required in this field to get higher accuracy and reliability. Changing the model architecture, tuning the hyperparameters, and implementing new regularization techniques and ensemble methods should be considered. Deepfake detection is quite a challenging but meaningful field that needs continued experiments to counter deepfake technology development properly. The accuracy and generalization abilities of the model can be improved through optimization and a thorough exploration of the latest techniques; in turn, these advanced detection systems are more dependable.

**REFERENCES:**


Almars, A. M. (2021). Deepfakes detection techniques using deep learning: a survey. Journal of Computer and Communications, 9(05), 20-35.

Dolhansky, B., Bitton, J., Pflaum, B., Lu, J., Howes, R., Wang, M., & Ferrer, C. C. (2020). The deepfake detection challenge (dfdc) dataset—arXiv preprint arXiv:2006.07397.

Dong, S., Wang, J., Liang, J., Fan, H., & Ji, R. (2022, October). I am explaining deepfake detection by analyzing image matching. In European Conference on Computer Vision (pp. 18-35). Cham: Springer Nature Switzerland.

Kwon, P., You, J., Nam, G., Park, S., & Chae, G. (2021). Kodf: A large-scale Korean deepfake detection dataset. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 10744–10753).

Liao, L., Li, H., Shang, W., & Ma, L. (2022). An empirical study of the impact of hyperparameter tuning and model optimization on the performance properties of deep neural networks. ACM Transactions on Software Engineering and Methodology (TOSEM), 31(3), 1-40

Majeed, F., Shafique, U., Safran, M., Alfarhood, S., & Ashraf, I. (2023). Detection of drowsiness among drivers using novel deep convolutional neural network model. Sensors, 23(21), 8741.

Reyad, M., Sarhan, A. M., & Arafa, M. (2023). A modified Adam algorithm for deep neural network optimization. Neural Computing and Applications, 35(23), 17095-17112

Salman, F. M., & Abu-Naser, S. S. (2022). Classification of real and fake human faces using deep learning