Enhancing Generalization of CNN-Based Systems in the Detection of Facial Spoofing Attacks

1st Christy Davis Maliyekkal
Dept. of Computing
National College of Ireland
Dublin, Ireland
x22151648@student.ncirl.ie

2nd Linda Susan Raju
Dept. of Computing
National College of Ireland
Dublin, Ireland
x22134409@student.ncirl.ie

3rd Shajo Varghese Eramplackal
Dept. of Computing
National College of Ireland
Dublin, Ireland
x22115943@student.ncirl.ie

***Abstract*—The project addresses the challenge of facial anti-spoofing, focusing on developing robust models to differentiate between genuine and spoofed facial images. A dataset, LCC FASD, is introduced, encompassing diverse spoofing scenarios captured from various devices. The methodology involves data collection, splitting, and preprocessing, followed by extensive model experimentation. Multiple CNN architectures are designed, encompassing various combinations of dropout and L2 regularization. Hyperparameter tuning and rigorous training strategies are employed to optimize model performance. Evaluation metrics, including accuracy and confusion matrices, are used to assess model effectiveness. The research contributes by offering insights into robust facial anti-spoofing techniques and providing a comprehensive analysis of model architectures and hyperparameters.**

***Index Terms*—Facial anti-spoofing, CNN, Generalisation, Dropout, Regularization**

I. Introduction

Facial spoofing, a notable concern in the realm of computer vision and biometric security, which refers to the harmful act of attempting to delude facial recognition system by fabricating facial information. As the facial recognition technique is widely used by the present era, it become highly prevalent in authentication, access and identity verification. The motive behind these spoofing attacks is to gain access to accounts, systems or any other facilities by destroying the vulnerabilities in the facial recognition techniques. The attacks might be in different forms which includes displaying 3D models, photos or videos of a genuine person is altered and impersonate them. These attacks can compromise the security and privacy, identity integrity and also financial losses. Figure 1 shows the different spoof attacks to be considered.

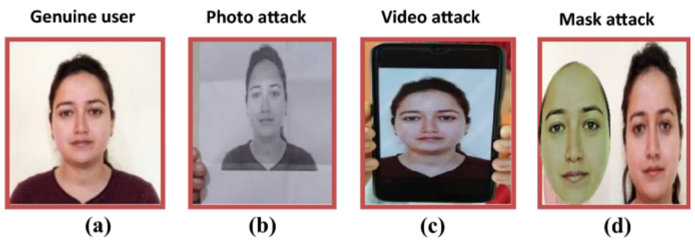

Fig. 1: Different spoof attacks

To tackle facial spoofing attacks, the researchers actively developing anti-spoofing techniques. These includes advanced algorithms, deep leaning models and Convolutional Neural Networks (CNNs). The objective is to differentiate between genuine and fake representations from static and dynamic images. The CNNs which is evolved as a powerful tool for the image classification approaches. but the challenges like overfitting and generalization capability limitations exists, especially in the area of facial spoofing. This study search through the complexities of improving the generalization of CNN models for detecting the facial spoofing attacks.

The motivation behind this research is that the critical requirement of the facial spoofing approaches. Nowadays, people depending more on the biometric techniques for identification and security purposes, the resulting of such systems to attacks calls for appropriate and robust solutions. The limitations within the current CNN based techniques, struggle with the overfitting on the training data and variations, which depicts the urgency of various novel strategies to enhance the generalization.

The paper aims to provide an extensive enhancement of generalization capabilities of CNN model on the facial spoofing attack detection. The primary research question which guiding the research is: To what extent can the generalization of CNN models be enhanced in the detection of facial spoofing attacks?

The particular objectives within the study includes firstly, assessment of the regularization techniques which investigates a range of regularization techniques, such as dropout and regularization, to reduce the overfitting and improve the generalization of the CNN models. Secondly, the Hyperparametric Tuning which scientifically explore the different hyperparameters and their impacts, filtering the sizes and the dense units to generalize across facial spoofing approaches on the capability of CNN models. Thirdly, comparing the efficiency of different regularization techniques and hyperparameter settings, both quantitatively and qualitatively to enhance the CNN models' generalization abilities for spoofing detection. Finally, the real-world applicability of the process by evaluating the practical implementation of enhanced generalization.

By considering the above research objectives, this study attempts to contribute to the development of facial spoofing detection by improving the strategies that can modify the generalization capabilities of the CNN models and can prevent the facial spoofing attacks. By handling the different challenges of overfitting and the limited capability of generalization, our study focuses on a reliable and robust facial spoofing detection techniques with an extensive real-world pertinence.

Generalization has prime importance in the facial spoofing detection. According to the context of CNN models' generalization mentions about the capability to perform on new data or unseen data instead of just on the trained data. Identifying the spoofing methods accurately is important for a successful detection of a model, but the better adaptation of novel attack strategies, environmental changes and evolving attacks also need to be analysed and have the same level of importance in facial spoofing detection to prevent the spoofing attacks.

As mentioned earlier, facial spoofing attacks may be in the form of printed photos, which means the attackers use high-quality printed photos and they may use the glossy paper and colour to simulate the skin tone and features. The other one used by them is the digital displays which includes smartphones, digital screen or tablets displaying images and videos of legal users and attempt to manipulate. 3D mask, which is an advanced attack which is able to create life to users by the same manipulation. The deepfake is another advanced level of fake which includes artificial intelligence and machine learning techniques to develop realistic videos or images of the users and may be imposed by another person's body or face. These are the serious issues that need to be handled, otherwise it will highly affect negatively on the privacy and security of every individual.

Because of the above-mentioned problems, the facial spoofing detection is highly required on the current world. By all these insecurities happening in the present world, the paper aims to provide an enhancement on the generalization capabilities to tackle the facial spoofing attacks.

II. Related Works

[1] and [2] explore the utilization of dropout techniques for improving neural network generalization. While [1] introduces dropout within an extreme learning machine framework for image classification, [2] categorizes and evaluates dropout regularization methods across various neural network architectures. While [1] emphasizes practical implementation and performance, [2] seeks to establish a theoretical foundation and comprehensive overview of dropout methods.

[3] and [4] delve into the incorporation of dropout for regularization in different contexts. While [3] introduces Spectral Dropout for enhancing deep neural network generalization, [4] focuses on the application of dropout to CNNs for fault diagnosis. Both papers acknowledge the potential of dropout in enhancing model robustness, but their specific focus areas differ. These papers led to approaches in applying dropout for generalization on facial spoofing detection.

[5] and [6] address the application of convolutional neural networks (CNNs) for classification tasks. [5] explores face spoof detection using CNNs and highlights the architecture's capabilities against fraudulent attacks. On the other hand, [6] introduces a two-stage training process for enhancing CNN generalization, utilizing implicit regularization. Insights from both papers helped in approach to enhancing CNN-based facial spoof detection through regularization techniques.

[7] and [8] address the challenges of overfitting and poor generalization. [7] investigates input loss landscape analysis and regularization as strategies to enhance model robustness. In contrast, [8] focuses on data augmentation, dropout, and batch normalization for improving CNN generalization under limited sample sizes. [9] propose cut-out regularization for deepfake detection. These papers provided a comprehensive strategy for addressing overfitting challenges in the project.

[10] and [11] revolve around regularization techniques in neural networks. While [10] comprehensively surveys dropout methods' evolution, applications, and foundations, [11] critically evaluates the impact of various regularization techniques on convolutional neural networks (CNNs). The strengths of [10] lie in its broad overview and clarity, while [11] provides specific insights into regularization for CNNs. By combining findings from both papers, a well-rounded understanding of regularization techniques applicable can be developed.

[12] and [13] discuss the concept of regularization in different contexts. While [12] provides a comprehensive survey of regularization strategies, [13] narrows its focus to the improvement of face anti-spoofing detection using CNN classifiers. Insights from these help selecting appropriate regularization strategies for improving the generalization of the face anti-spoofing detection model.

[14] and [15] touch on multi-scale convolutional neural networks (CNNs) and dropout. [14] discusses fault diagnosis using multi-scale CNNs with dropout regularization, while [15] introduces a dataset fusion algorithm for anomaly detection. By combining insights from both papers, exploring how multi-scale CNNs and dropout can enhance the generalization of the face anti-spoofing detection model, particularly when dealing with periodic time series data can be done. Similarly, [16] explores regularization techniques for generalization, it showcases the benefits of dropout in enhancing model reusability.

[17] combines face liveness detection and CNN classification for anti-spoofing. Similarly [18] evaluates CNN architectures for anti-spoofing. Both papers contribute directly to the topic of facial spoofing detection by proposing innovative techniques and evaluating their effectiveness. By combining the approaches of these papers, we could create a comprehensive model that combines the strengths of face liveness detection and CNN classification, enhancing generalization in detecting various spoofing attacks.

[19] introduces a two-stream approach to fuse frequency features, enhancing generalization and [20] proposes dynamic regularization for CNNs to improve generalization. Both papers emphasize improving generalization through novel techniques. Integrating the frequency fusion approach with dynamic regularization could lead to a model that not only captures intricate features but also adapts its regularization based on training dynamics, ultimately enhancing the generalization

in facial spoofing detection.

For the proposed facial spoofing detection, these papers collectively offer valuable insights into improving neural network generalization. The application of dropout techniques in [1], [2], [3] and [4] can enhance the robustness of CNN models against adversarial attacks, a crucial aspect in facial spoofing detection. Papers 10 and 15, which deal with CNN architectures and implicit regularization, can guide the design of your detection model.

Furthermore, [5] and [6] approaches to combating overfitting and improving generalization are directly relevant to the challenges posed by limited data in facial spoofing detection. Incorporating data augmentation, dropout, and regularization techniques, as discussed in these papers, can help mitigate overfitting and enhance the model's ability to generalize.

Finally, [10], [11] and [12] can provide a broader context and understanding of regularization techniques and their implications. In summary, leveraging the insights from these papers significantly enhanced the robustness and performance of the facial spoofing detection model, addressing challenges related to data scarcity, overfitting, and adversarial attacks.

III. Methodology

In this section, a detailed overview of the methodology used in this project with the key approaches and techniques is highlighted. Here, we apply the Knowledge Discovery in Databases (KDD) methodology to develop an advanced anti-spoofing algorithm for facial recognition systems. The methodology outlines the gradual process followed to develop and evaluate CNN models for the facial anti-spoofing detection. Following the KDD framework, we look into the data collection, model architecture design, hyperparameter tuning, training procedures, evaluation metrics, and analysis, offering a clear understanding of how we approached this problem.

A. Data Collection

The data used for this project is obtained from the "LCC FASD" (LCC Facial Anti-Spoofing Database), a dataset specifically designed for training and evaluating facial anti-spoofing algorithms. The database was created to keep a set of core principles aimed at ensuring the authenticity, diversity, and real-world applicability of the collected data.

The "LCC FASD" dataset was established in direct comparison to previous databases such as NUAA and Idiap, which are commonly referenced in anti-spoofing research. The database was constructed without requiring users to mimic facial expressions, thereby promoting a frictionless user experience. Furthermore, the dataset encompasses a wide array of image sources, including screens and devices like smartphones, reflecting the diverse landscape of capturing devices available in the market.

The dataset comprises of three distinct subsets: training, development, and evaluation. In total, the "LCC FASD" dataset contains 1942 genuine images and 16885 spoofing images, which were collected from sources such as YouTube, Amazon Mechanical Turk, and Yandex Toloka. The training subset consists of 118 identities with 1223 genuine images and 7076 spoofing images. Similarly, the development subset includes 25 identities with 405 genuine images and 2543 spoofing images. Lastly, the evaluation subset involves 100 identities, encompassing 314 genuine images and 7266 spoofing images. This division ensures a balanced and robust representation of both genuine and spoofing instances across different data subsets. Another characteristic of the "LCC FASD" dataset is the varying size of images, with absolute dimensions ranging from 150 to 1350 pixels. This resizing approach preserves the original properties of the images and contributes to a more diverse and challenging dataset for training and testing facial anti-spoofing models [21].

B. Data Splitting and Preparation

The dataset already has separate genuine and spoofing images folders. To mitigate any potential biases, these images were randomly shuffled. The dataset was then split with ratios, 80% for training, 10% for validation, and the remaining 10% for testing. In the dataset, distinct folders were created for each subset—'train,' 'validation,' and 'test.' Within these folders, the images corresponding to each class (genuine or spoofing) were further organized. This structuring ensured a clear separation of data subsets and facilitated efficient data loading during model training and evaluation.

To enhance the model's generalization capability and mitigate overfitting, data augmentation techniques were applied during the training phase. The *ImageDataGenerator* from TensorFlow's Keras API was used for this purpose and two data generators were defined: *train_datagen* and *test_datagen*. The *train_datagen* is configured to apply data augmentation techniques like rotation, width and height shifts, and horizontal flipping. These augmentations create variations of the training images, making the model more robust to different patterns and features. The *test_datagen* is used to normalize the pixel values of the test images, similar to the training images, to ensure consistency.

There are also three data flow generators: *train_generator*, *validation_generator*, and *test_generator*. The *flow_from_directory* method loads data from specified directories and generates batches of data. It takes several parameters:

- *target_size*: Specifies the dimensions to which the images should be resized.
- *batch_size*: Determines the number of samples in each batch.
- *class_mode*: Specifies the type of labels. In this case, 'binary' indicates binary classification.

To ensure that the order of images is maintained during testing for accurate evaluation and comparison, the *shuffle* parameter in the *test_generator* is set to *False*.

In summary, data generators and data flow generators efficiently load and preprocess data in a real-time manner, making them ideal for training, validation, and testing deep learning models. They handle the data augmentation, normalization, batching processes, and optimizing memory usage.

C. Model Architectures

The code defines and evaluates different CNN architectures to train models for image classification, which are the basic CNN, CNN with dropout, CNN with L2 regularization and CNN with Dropout and L2 Regularization models. Each type of model is built and evaluated using different hyperparameters.

1) Basic CNN Model: This model has a simple architecture with tunable hyperparameters. It consists of convolutional layers, max-pooling layers, dense layers, and an output layer. The *build_basic_cnn_model* function builds the basic CNN model with the specified hyperparameters. The model consists of:

- Convolutional layer with specified filters and kernel size.
- Max-pooling layer to downsample the data.
- Flattening layer to convert the 2D features into a 1D vector.
- Dense layers with specified units.
- Output layer with sigmoid activation.
- Model is compiled with Adam optimizer, binary cross-entropy loss, and accuracy metrics.

The code manually performs hyperparameter tuning by iterating through different learning rates, filters, kernel sizes, and dense units. Models are trained and evaluated using training, validation, and test data. The best hyperparameters that yield the highest test accuracy are saved. The best basic CNN model is then built using the best hyperparameters and trained further. The best model's performance is evaluated using test data. The test accuracy is calculated, and confusion matrix is printed.

2) CNN with Dropout: This model includes dropout layers, a technique to prevent overfitting. The *build_cnn_with_dropout* function builds the CNN model with dropout layers added after dense layers and different dropout rates are tried. Multiple models are trained using different dropout rates. Training, validation, and test data are used for each model. Each model's test accuracy is calculated, and confusion matrix is printed.

3) CNN with L2 Regularization: This model includes L2 regularization to prevent overfitting. The *build_cnn_with_regularization* function builds the CNN model with L2 regularization added to the convolutional layer with different L2 regularization values tried. Multiple models are trained using different L2 regularization values. Training, validation, and test data are used for each model with each model's test accuracy calculated, and confusion matrix printed.

4) CNN with Dropout and L2 Regularization: This model combines both dropout and L2 regularization techniques. The *build_cnn_with_dropout_and_regularization* function builds the CNN model with both dropout and L2 regularization. The model is trained using training and validation data. The trained model's test accuracy is calculated, and confusion matrix is printed.

The code thus showcases different variations of CNN architectures with different techniques to improve generalization and prevent overfitting, demonstrating how they perform on the given dataset for image classification.

D. Hyperparameter Tuning

1) Basic CNN Model Hyperparameter Tuning: In the basic CNN model the following hyperparameters are tuned:

- Learning Rate (*learning_rate*): The code tests two learning rates, 0.001 and 0.0001, to determine which one leads to better model performance.
- Number of Filters in Convolutional Layer (*conv1_filters*): The number of filters in the initial convolutional layer is set to 32, a common choice for basic models.
- Kernel Sizes in Convolutional Layer (*conv1_kernel*): Two kernel sizes are tested, 3 and 5, to understand which one captures relevant patterns.
- Dense Units (*dense_units*): Two choices for the number of units in the dense layer, 128 and 256, are evaluated.

The code uses nested loops to go through combinations of these hyperparameters and builds a model with each combination. Each model is then trained on the training data and evaluated on the validation data. The model with the best performance (highest validation accuracy) is chosen as the best basic CNN model.

2) CNN with Dropout Hyperparameter Tuning: Similar to the basic CNN model, the dropout model involves hyperparameter tuning:

- Dropout Rate (*dropout_rate*): Three dropout rates, 0.1, 0.3, and 0.5, are tested.
- Best Hyperparameters: The best hyperparameters from the basic CNN model (*learning_rate*, *conv1_filter*, *conv1_kernel*, *dense_units*) are used as a starting point.

The code creates and trains models with different dropout rates using the best hyperparameters from the basic CNN model. Each model is trained and evaluated, and the model with the highest validation accuracy is chosen. The goal is to find the dropout rate that contributes to better generalization.

3) CNN with L2 Regularization Hyperparameter Tuning: For the L2 regularization model, the hyperparameters to tune are:

- L2 Regularization Strength (*l2_regularization*): Three values, 0.01, 0.001, and 0.0001, are tested.
- Best Hyperparameters: Similar to previous cases, the best hyperparameters from the basic CNN model are used as a starting point.

The code creates models with different L2 regularization strengths using the best hyperparameters from the basic CNN model. Models are trained and evaluated, and the model with the highest validation accuracy is selected. The objective is to determine the optimal L2 regularization strength.

4) *CNN with Dropout and L2 Regularization Hyperparameter Tuning:*

In this final architecture with both dropout and L2 regularization combined, and the hyperparameters to tune include:

- Dropout Rate (*dropout_rate*): The same dropout rates as before, 0.1, 0.3, and 0.5.
- L2 Regularization Strength (*l2_regularization*): The same L2 regularization values as before.
- Best Hyperparameters: The best hyperparameters from the basic CNN model.

Similar to previous steps, models with different combinations of dropout rates and L2 regularization strengths are trained and evaluated. The model with the highest validation accuracy is chosen. The goal is to identify the combination of dropout and L2 regularization that enhances model performance.
In all these steps, the code systematically explores a range of hyperparameter values for each model architecture, evaluating their impact on model accuracy using the validation data. This process allows to optimize the model's performance by selecting the best-performing hyperparameters for each architecture.

E. *Training Procedure*

The training procedure for the all the models are same as mentioned below, it involves the following steps:

- Number of Epochs: Each model is trained for 10 epochs, as specified by *epochs=10*.
- Batch Size: The batch size for training data is set to 32, defined by *batch_size=32*.
- Optimization Algorithm: The Adam optimizer is used with a learning rate determined during hyperparameter tuning.
- Loss Function: The binary cross-entropy loss (*loss='binary_crossentropy'*) is used since this is a binary classification problem.
- Evaluation Metrics: The accuracy metric is used to evaluate model performance during training (*metrics=['accuracy']*).

The training process remains consistent across all model architectures, with 10 epochs, a batch size of 32, and Adam optimization. The loss function and evaluation metric are also consistent, which allows for direct comparison of model performance.

IV. Results

In this section, we present the results and our analysis of the four CNN models which were designed to detect and mitigate a diverse range of spoofing attacks. The goal of this section is to explore different techniques that improve the generalization of CNN systems. The performance of every model is evaluated based on test accuracy , validation accuracy, loss and the confusion matrix .Critical metrics like precision recall and F-1 score are also calculated from confusion matrix to provide a more sophisticated assessment of every model's performance. In this context of confusion matrix, true positives means the number of images that were positive (genuine) and were correctly classified as genuine. False negatives means instances where genuine samples were mislabelled as as spoofed attacks. These instances were missed by the model. False positives means the number of instances where the image was a spoofed image but were incorrectly classified as positive (genuine). True negative means the number of instances where the images belonged to negative class (spoofed) and were correctly classified as negative (spoofed).

A. *Basic CNN Model*

The basic CNN model was trained on different combinations of learning rate, convolution filters, convolution filters and dense units. The best parameters obtained were as follows:

```
learning_rate: 0.0001,
conv1_filters: 32,
conv1_kernel: 5,
dense_units: 128
```

The basic CNN model was then trained on these hyperparameters, namely, learning rate of 0.0001, 32 filters in the first convolutional layer with a kernel size of 5, and 128 units in the dense layer. After training for 10 epochs, the model achieved a test accuracy of 0.899, validation accuracy of 0.884 and a loss of 0.2647. Table 1 shows the results of this model and Figure 2 shows the architecture of the basic CNN model with the best hyperparameters.

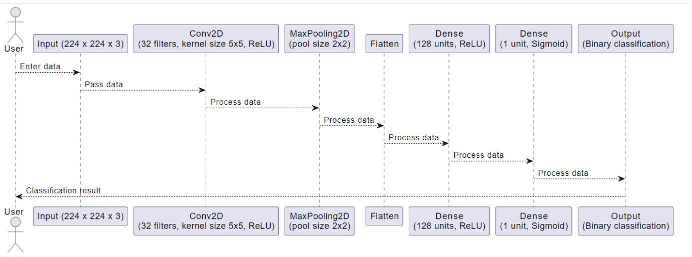

Fig. 2: Architecture of the basic CNN model with the best hyperparameters

Table 1: Basic CNN Model Results

Precision	0.898
Recall	0.994
F-1 Score	0.943

B. *CNN model with dropout*

A CNN model was built with the best parameters achieved above and was experimented with three different values of dropout rates .The findings are as follows:

1) Dropout rate -0.1: The CNN model with dropout rate 0.1 achieved a test accuracy of 0.917, validation accuracy of 0.907 and a loss of 0.22. Table 2 shows the results of this model

Table 2: CNN model with dropout rate -0.1

Precision	0.935
Recall	0.970
F-1 Score	0.952

2) Dropout rate -0.3: The CNN model with dropout rate 0.3 achieved a test accuracy of 0.894, validation accuracy of 0.886 and a loss of 0.275. Table 3 shows the results of this model.

Table 3: CNN model with dropout rate -0.3

Precision	0.935
Recall	0.970
F-1 Score	0.952

3) Dropout rate -0.5: The CNN model with dropout rate 0.5 achieved a test accuracy of 0.872, validation accuracy of 0.874 and a loss of 0.294. Table 4 shows the results of this model.

Table 4: CNN model with dropout rate -0.5

Precision	0.871
Recall	0.999
F-1 Score	0.930

From the analysis, the model with a dropout rate of 0.1 exhibited the best balance between precision and recall, had the best test accuracy and minimal loss among others making it the most optimal choice.

C. CNN model with regularisation

A CNN model was built with the best parameters achieved in the first step and was experimented with three different values of l2 regularisation.The findings are as follows:

1) L2 Regularization Value: 0.01: The CNN model with regularisation value of 0.01 achieved a test accuracy of 0.90, validation accuracy of 0.898 and a loss of 0.287. Table 5 shows the results of this model.

Table 5: CNN model with Regularisation value 0.01

Precision	0.949
Recall	0.945
F-1 Score	0.947

2) L2 Regularization Value: 0.001: The CNN model with regularisation value of 0.001 achieved a test accuracy of 0.927, validation accuracy of 0.907 and a loss of 0.227. Table 6 shows the results of this model.

3) L2 Regularization Value: 0.0001: The CNN model with regularisation value of 0.0001 achieved a test accuracy of 0.932, validation accuracy of 0.924 and a loss of 0.215. Table 7 shows the results of this model.

Table 6: CNN model with Regularisation value 0.001

Precision	0.93
Recall	0.99
F-1 Score	0.95

Table 7: CNN model with Regularisation value 0.0001

Precision	0.946
Recall	0.976
F-1 Score	0.961

From the analysis, the model with an L2 regularization value of 0.001 exhibited the best precision and recall trade-off while an L2 regularization value of 0.0001 had the highest test accuracy.

D. CNN model with Dropout and regularisation

A CNN model was built with the best hyperparameters and with the optimum droput rate and regularisation values of 0.1 and 0.0001 respectively.The model achieved a test accuracy of 0.932, validation accuracy of 0.918 and loss of 0.2. Table 8 shows the results of this model.

Table 8: CNN model with Dropout rate 0.1 and Regularisation value 0.0001

Precision	0.929
Recall	0.997
F-1 Score	0.962

V. Interpretation

- The CNN model coupled with dropout and regularisation achieved the highest test accuracy of 93.27% and an optimum validation accuracy of 91.8% at the 10th epoch. Figure 3 shows the accuracy of the training and validation.
- The model also achieved a minimum loss of 0.2%. Figure 4 shows the loss in the training and validation.
- Furthermore, to evaluate the performance of the model, we have analysed its ROC curve. Plotting the ROC curve helps us to understands the models ability to discriminate between spoof and real classes across different thresholds. Figure 5 shows the ROC curve
- The model obtained a ROC-curve value of 0.95, which indicates model's ability in correctly classifying spoof class while limiting the number of false alarms effectively.
- Among other individual models, CNN model with regularization with a L2 regularization value of 0.0001 also performed efficiently.
- The basic CNN model had a respectable test accuracy of 89.9% but also reported instances of misclassification evident by the confusion matrix.

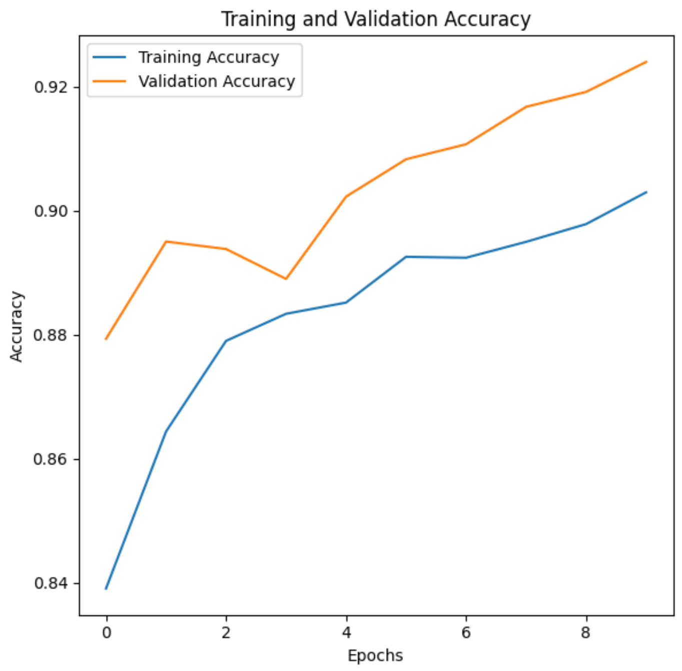

Fig. 3: Training and Validation Accuracy

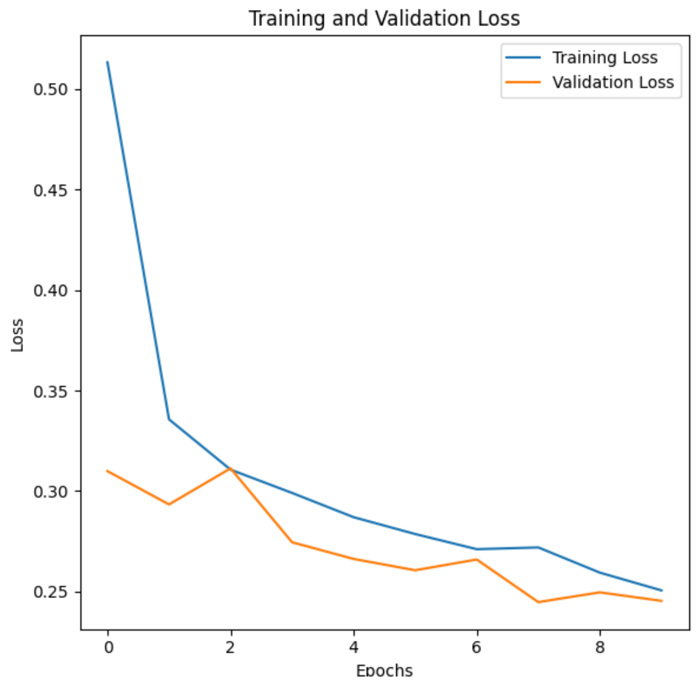

Fig. 4: Training and Validation Loss

- The CNN models with dropout rate showed different trend in accuracy when incorporated with different dropout rates. The model with a dropout rate of 0.1 performed the best compared to other dropout rates, advocating the importance of selecting the appropriate dropout rate.

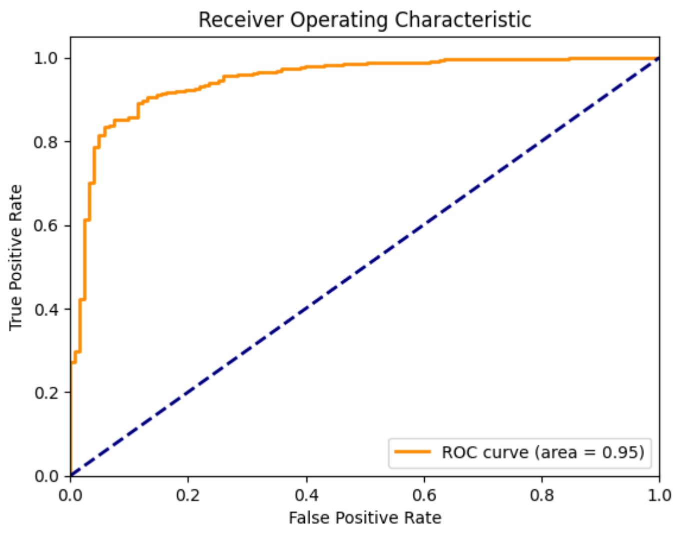

Fig. 5: ROC curve

VI. Conclusion and Future Work

In conclusion, the CNN model with Dropout and regularization was the best performing model among the four models. It achieved a balance between preventing overfitting and enhancing generalization, leading to highest accuracy and minimal loss among other models. The AUC-ROC value of our model supports its robustness and its ability to generalize well across diverse spoofing attacks. The results also emphasize the importance of hyperparameter tuning, selecting appropriate dropout rates and utilization of regularization techniques to improve CNN model performance in detecting diverse spoofing attacks and improving overall performance of the model, thus contributing to enhancing generalization of these models. While L2 regularization showed promising results, exploring other types of regularization techniques like L1 regularisation could provide insights into their effectiveness in enhancing generalization. Better initialization and extraction could be achieved by employing transfer learning techniques. Investigation could also be done on the transferability of developed models to other domains with different types of spoofing attacks. Testing the performance of the developed models in real time scenarios is also crucial to verify whether the model performs well in real world practical applications like security systems. The model can also be further trained on new and evolving spoofing attacks to ensure model can also handle new spoofing threats and is up to date.

References

[1] G. S. Nandini, A. S. Kumar, and K. Chidananda, “Dropout technique for image classification based on extreme learning machine,” *Global Transitions Proceedings*, vol. 2, no. 1, pp. 111–116, 2021.

[2] I. Salehin and D.-K. Kang, “A review on dropout regularization approaches for deep neural networks within the scholarly domain,” *Electronics*, vol. 12, no. 14, p. 3106, 2023.

[3] S. H. Khan, M. Hayat, and F. Porikli, “Regularization of deep neural networks with spectral dropout,” *Neural Networks*, vol. 110, pp. 82–90, 2019.

[4] C. Tingting, X. Jianlin, and C. Huafeng, “Improved convolutional neural network fault diagnosis method based on dropout,” in *2020 7th International Forum on Electrical Engineering and Automation (IFEEA)*. IEEE, 2020, pp. 753–758.

[5] L. G. Muradkhanli, P. A. Namazli *et al.*, “Face spoof detection using convolutional neural network,” *Problems of Information Society*, pp. 40–46, 2023.

[6] Q. Zheng, M. Yang, J. Yang, Q. Zhang, and X. Zhang, “Improvement of generalization ability of deep cnn via implicit regularization in two-stage training process,” *IEEE Access*, vol. 6, pp. 15 844–15 869, 2018.

[7] L. Li and M. Spratling, “Understanding and combating robust overfitting via input loss landscape analysis and regularization,” *Pattern Recognition*, vol. 136, p. 109229, 2023.

[8] P. Thanapol, K. Lavangnananda, P. Bouvry, F. Pinel, and F. Leprévost, “Reducing overfitting and improving generalization in training convolutional neural network (cnn) under limited sample sizes in image recognition,” in *2020-5th International Conference on Information Technology (InCIT)*. IEEE, 2020, pp. 300–305.

[9] S. D. Gummadi and A. Ghosh, “Deep residual learning based discriminator for identifying deepfakes with cut-out regularization,” in *2022 IEEE World Conference on Applied Intelligence and Computing (AIC)*. IEEE, 2022, pp. 149–155.

[10] A. Labach, H. Salehinejad, and S. Valaee, “Survey of dropout methods for deep neural networks,” *arXiv preprint arXiv:1904.13310*, 2019.

[11] C. F. G. D. Santos and J. P. Papa, “Avoiding overfitting: A survey on regularization methods for convolutional neural networks,” *ACM Computing Surveys (CSUR)*, vol. 54, no. 10s, pp. 1–25, 2022.

[12] Y. Tian and Y. Zhang, “A comprehensive survey on regularization strategies in machine learning,” *Information Fusion*, vol. 80, pp. 146–166, 2022.

[13] S. R. Chavan, S. S. Sherekar, and V. M. Thakre, “Factors related to the improvement of face anti-spoofing detection techniques with cnn classifier,” in *2021 International Conference on Computational Intelligence and Computing Applications (ICCICA)*. IEEE, 2021, pp. 1–4.

[14] X. Liu, H. Tian, and Z. Dai, “Bearing fault diagnosis based on multi-scale convolution neural network and dropout,” in *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, vol. 1. IEEE, 2020, pp. 1401–1406.

[15] A. Elhalwagy and T. Kalganova, “A dataset fusion algorithm for generalised anomaly detection in homogeneous periodic time series datasets,” *arXiv preprint arXiv:2305.08197*, 2023.

[16] J. Farebrother, M. C. Machado, and M. Bowling, “Generalization and regularization in dqn,” *arXiv preprint arXiv:1810.00123*, 2018.

[17] R. B. Hadiprakoso, H. Setiawan *et al.*, “Face anti-spoofing using cnn classifier & face liveness detection,” in *2020 3rd International Conference on Information and Communications Technology (ICOIACT)*. IEEE, 2020, pp. 143–147.

[18] C. Nagpal and S. R. Dubey, “A performance evaluation of convolutional neural networks for face anti spoofing,” in *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2019, pp. 1–8.

[19] B. Chen, W. Yang, and S. Wang, “Face anti-spoofing by fusing high and low frequency features for advanced generalization capability,” in *2020 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*. IEEE, 2020, pp. 199–204.

[20] Y. Wang, Z.-P. Bian, J. Hou, and L.-P. Chau, “Convolutional neural networks with dynamic regularization,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 5, pp. 2299–2304, 2020.

[21] D. Timoshenko, K. Simonchik, V. Shutov, P. Zhelezneva, and V. Grishkin, “Large crowdcollected facial anti-spoofing dataset,” in *2019 Computer Science and Information Technologies (CSIT)*, 2019, pp. 123–126.