Behind The Mask: A Deepfake Video Detection using a Convolutional Neural Network Model

Robbie Christian Emmanuel Espaldon
College of Computing and Information Technology
National University – Manila, Philippines
Manila, Philippines
espaldonre@students.national-u.edu.ph

John Angelo Basilio
College of Computing and Information Technology
National University – Manila, Philippines
Manila, Philippines
basilioja1@students.national-u.edu.ph

Dhan Micheal Tamparong
College of Computing and Information Technology
National University – Manila, Philippines
Manila, Philippines
tamparongdl@students.national-u.edu.ph

Abstract*–The rapid spread of deepfake videos threatens digital trust, privacy, and security by enabling highly realistic but deceptive content. This study provides a simplified end-to-end system intended for non-tech users. The Multi-Task Cascaded Convolutional Network (MTCNN) identifies video frames, resizes them to 224x224 pixels, normalizes them, and augments them to reduce overfitting. A custom CNN in PyTorch—comprising two convolutional blocks with max-pooling, a 128-unit fully connected bottleneck with dropout, and a two-unit output layer—was trained using the Deep Fake Detection (DFD) on the Kaggle dataset using an 80-20 split and tested through five-fold cross-validation. Deployed through a Streamlit interface, the system classifies frames and aggregates predictions via majority voting. Experimental results show 77.2% validation accuracy, class accuracies of 84.8% for fake videos and 64.3% for real videos, a weighted F1-score of 0.76, and an AUC of 0.83. These results verify the model's real-time detection capability and underscore opportunities for improving efficiency on authentic content.

Keywords—Deepfake, kaggle, PyTorch, Multi-Task Cascaded Convolutional Network, data augmentation, real-time inference, Streamlit interface, majority voting

I. Introduction

The rapid development of modern technology has led to the alarming increase of "deepfake" videos on the internet. The user-friendly platforms that enable the emergence of deepfakes have significantly reduced entry barriers, allowing anybody to create convincing counterfeit videos for illicit purposes, including dating scams, celebrity impersonations, political manipulation, and financial fraud. Deepfakes are realistic synthetic videos in which a person's likeness is digitally altered using sophisticated generative models, posing an important danger to global digital trust and security.

In the Philippines, this issue has gained national consideration: in February 2024, the Department of Information and Communications Technology (DICT) cautioned of an impending increase in deepfake audio and video content as the May 2025 midterm elections approach, urging Filipinos to critically evaluate online media prior to sharing or responding to it.[1] Manipulated videos, which can misinform, defame, or sway public opinion, are increasingly directed at platforms such as social media, video-sharing sites, and news publications.[2] Non-technical people are particularly vulnerable, as even little facial or lip-sync artifacts often go unnoticed. Recent surveys indicate that the quantity of deepfake videos circulating online doubled between 2022 and 2024, with malicious uses ranging from political disinformation to personal defamation.[3]

This study aims to create a deepfake detection system utilizing a Convolutional Neural Network (CNN) trained on the Deep Fake Detection (DFD) dataset sourced from Kaggle.[4] Unlike image-level approaches, the method used performs a video-level frame analysis in which each video frame goes through a trained model classifier, aggregating the frame-wise predictions to determine the authenticity of the video. The researchers' model acquires the ability to differentiate between authentic and forged content in realistic scenarios, as practical application necessitates real-time performance. This study underscores efficiency through lightweight CNN architectures and optimization methods that facilitate rapid inference on live video streams. The researchers implement the technology on Streamlit to demonstrate its usage. This interface implements the model in real time by uploading a video, enabling researchers to evaluate detection accuracy and latency in a practical environment.

The significance of this study extends beyond academic evaluation. By providing an accessible, real-time detection service, the deepfake detection system empowers content moderators, journalists, and ordinary users to verify the authenticity of video content before sharing or acting upon it. Digital platforms can embed the system as a pre-upload examination, and law enforcement agencies can use it to flag suspicious evidence. Moreover, the ongoing study enhances media literacy and fortifies defenses against the escalating deepfake threat.

II. Review of Related Literature

In order for the researchers to understand how they can approach the creation of their flagship Convolutional Neural Network model for deepfake video detection, they would need to understand what studies are out there that may be doing the same things or can steer them in the right direction with proper foundational building of what they need to know about deepfake detection model building.

The use of Convolutional Neural Network Model

When looking into the idea of creating a deepfake detection model, the researchers initially looked into Support Vector Machine (SVM) and Convolutional Neural Network (CNN), but to figure out which one would perform better, they looked into the Literature Review by Vardhan et al.[8], dsIn their review, they compared 18+ deep fake detection studies published from 2018–2024. They found that CNN-based models can perform at 77-98% accuracy levels. It is also robust against compression and can capture spatial artifacts better when compared to other models. The problem however, is that it can be prone to overfitting. One recommendation they had is to have a larger dataset for training to combat overfitting, and also using a multi-modal approach for better context understanding. Although this paper was not able to utilize said recommendation, it will explore another method of adding better context to the models' training method.

In order to solidify the initial chosen model choice—Convolutional Neural Network (CNN)---The researchers looked into similar studies that explored this route of creating deepfake detection models. One study conducted by Afchar et al.[9] explored the use of a shallow 4-layer CNN that they named "Meso-4." Their Meso-4 model performed 98.4% accuracy in deepfake detection. This was achieved by focusing on the mesoscopic facial artifacts. In order to create an improved model, the researchers kept the 4-layer model design of Meso-4 and added configurations to have a better training methodology: Scaling the input resolution to 224x224 (vs. MesoNet's 256×256) for compatibility with modern datasets and adding a colorjitter method to better simulate real-world lighting variations.

Data Augmentation Strategies

According to AWS, "Data augmentation is the process of artificially generating new data from existing data, primarily to train new machine learning (ML) models"[10]. In order to train the researcher's model in a manner that avoids overfitting, they looked into different papers to compare what has been done. A study conducted by Kolagati et. al.[11] utilized a similar data preprocessing method to the one the researchers of this paper used. Kolagati et. al. resized their frames to 224x224x3 for uniform input dimensions, but because the former used a hybrid model setup—Multilayer Perceptron (MLP) – Convolutional Neural Network (CNN)—they had differing processing steps. Although not utilized in this paper, it is worth mentioning for future implementations: "Facial Landmark Detection is a computer vision task that involves detecting and localizing specific points or landmarks on a face, such as the eyes, nose, mouth, and chin" PapersWithCode[12]. It was utilized in Kolagati et. al. 's paper for better contextual understanding for their hybrid model, this meant that they focused more into spatial inconsistencies rather than general textures, this produces a more accurate model but with its own unique set of caveats.

Spatial Artifact Detection

While exploring the possible ways the researchers can train their models, they came across a study conducted by Singh D. et. al.[13], wherein they created an enhanced version of the typical CNN only model for deep fake video detection. In the study, they also utilized a Long Short-Term Memory (LSTM) Model to boost the performance of the overall system of deep fake detection. The reason for doing this is to focus on how each model shines in training. A study conducted by Chris. H. et. al. said: "Spatial analysis using CNNs is critical for detecting deepfake artifacts, as they excel at identifying inconsistencies in individual frames."[14]. This means that combining an already solidly trained and tuned CNN model trained on spatial analysis methods will have significant benefits when paired with LSTM model trained to detect temporal features. Although the researchers did not implement the LSTM in their model, it still solidifies the reason as to why this research paper focused on training its model using spatial analysis. With that said, future improvements can bring out the best in the model, hence Singh's study is worth looking into for future betterment of training deep fake detection models. By incorporating temporal features training into the mix, it makes the model more robust to analysis as it gets better training.

III. Methodology

In this section, the researchers will describe the dataset, deep learning algorithm, tools, and techniques used in this study. Developing a deepfake detection model involves different steps to ensure the data fed into the model is reliable enough to be used to train it for prediction.

A. Data Collection

This study utilizes a publicly available dataset from Kaggle called the "Deep Fake Detection (DFD) Entire Original Dataset"[4]. FaceForensics and FaceForensics++ initially collected the dataset, and Sanika Tiwarekar later uploaded it to Kaggle. The dataset consists of 3000+ MP4 files, which are grouped in folders that have separated manipulated and original sequences or videos for easier classification for the user.

B. Data Pre-Processing

The researchers conducted fundamental data pre-processing to enhance the training data for the model before the training procedure. This process includes frame extraction (using MTCNN for detecting faces only) for converting video data into CNN input, scaling to standardize input dimensions, normalization to stabilize training by centering data around zero, data augmentation for

generalization and to mitigate overfitting, and an 80-20% train-test split.

C. Data Augmentation

On the data augmentation aspect, the researchers tweaked and aspect of it that can help with giving the model better training, "ColorJitter (brightness = 0.2, contrast = 0.2, saturation = 0.2, hue = 0.02)" was used to make it so that the model can still perform even if the lighting of the data is not the best, by tweaking this part, they made the model XX% better.

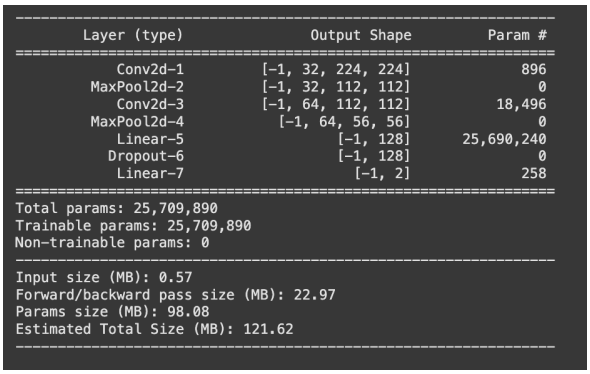
D. Experimental Setup

The researchers conducted this study using Python as the main language for development, incorporating libraries for easier implementation. The libraries used include OpenCV for video frame extraction and basic image processing, PyTorch and TorchVision for training the CNN model and evaluating its performance, and Matplotlib/Seaborn for visualizing the training process and results. The data underwent pre-processing before training to ensure context retainability from the raw data and for deeper understanding on the model side. The dataset was subsequently divided into an 80-20% train-test split to reflect the real data volume. The model's performance was then evaluated using the validation dataset with Torch's function. The studies were carried out on a local system equipped with an M3 Pro chip processor (12 CPU cores and 18 GPU cores) and 18 GB of memory. This setup ensured that the researchers had enough computational resources to train and assess the model in this study effectively. The researchers also used a five-fold cross-validation and the proper evaluation metrics for regression. The model's hyperparameters were set to the default settings in Scikit-learn. This approach allows us to assess the model's performance without requiring considerable hyperparameter adjustment, establishing a baseline for future optimization. These utilities are available in the following specific versions:

E. Model

The researchers utilized a Convolutional Neural Network (CNN) model trained using the PyTorch framework. CNN models have proven to be effective in this type of use case—albeit, with some caveats. One of these caveats is the tendency of CNN models to overfit, which varies depending on the dataset. To counter this, the researchers did some things to help mitigate some of the overfitting issues; one example is randomizing the rotation, adding blur, and looking at the spatial values of the data itself. Such analysis is done for better generalization of the dataset and to be more robust against data that are closely similar. To address data with suboptimal lighting conditions, the researchers used a ColorJitter phase to improve the illumination prior to inputting the data, thus enhancing the model's contextual comprehension.

The fundamental CNN architecture consists of two convolutional blocks, each containing a 3x3 Conv2d layer with 32 and 64 filters, respectively, succeeded by 2x2 max-pooling. These components facilitate the extraction of low to mid-level visual characteristics while systematically diminishing spatial dimensions from 224x224 to 56x56. The convolutional layers have fewer than 20,000 total parameters yet generate the essential feature maps for classification. Prior to flattening, the network traverses a fully connected layer comprising 128 units (with dropouts) before a final 2-unit output layer produces the real-vs-fake logits. The thick "bottleneck" layer constitutes more than 99% of the model's 25.7 million parameters, significantly contributing to its expressive capacity while simultaneously increasing its tendency to overfit. Through the integration of comprehensive data augmentation techniques (including random rotations and ColorJitter), alongside dropout and careful hyperparameter optimization, the researchers attain resilient generalization across varied lighting and compression scenarios, all while preserving a manageable model size (≈ 98 MB of weights) appropriate for real-time inference in a Stream-lit based web application.



```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [-1, 32, 224, 224]             896
         MaxPool2d-2         [-1, 32, 112, 112]               0
            Conv2d-3         [-1, 64, 112, 112]          18,496
         MaxPool2d-4           [-1, 64, 56, 56]               0
            Linear-5                 [-1, 128]      25,690,240
           Dropout-6                 [-1, 128]               0
            Linear-7                   [-1, 2]             258
================================================================
Total params: 25,709,890
Trainable params: 25,709,890
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.57
Forward/backward pass size (MB): 22.97
Params size (MB): 98.08
Estimated Total Size (MB): 121.62
----------------------------------------------------------------
```

Figure 1. Summary of the Layers used in the Model

F. Evaluation Metrics

The performance of the proposed DeepFake detection model that was rigorously evaluated using a suite of classification metrics and visual analytics. Given the binary nature of the task (REAL vs. FAKE), the following evaluation criteria were defined as:

Accuracy: The researchers defined this as the proportion of correctly classified instances among the total number of predictions, accuracy provides a general measure of model effectiveness across both classes.

Cross-Entropy Loss: The researchers defined this as the objective function during training, cross-entropy loss quantifies the divergence between the predicted probability distribution and the actual labels. Monitoring the loss during both training and validation phases allows for early detection of overfitting or under-fitting.

Training and Validation Curves: Epoch-wise accuracy and loss values were visualized to assess model convergence behavior. These metrics were derived from either CSV or JSON training logs, which were parsed and rendered within the interface to facilitate comparative evaluation across training runs.

Frame-Level Confidence Analysis: During inference, the model processes multiple uniformly sampled frames from

the input video. For each frame, the softmax probability corresponding to the predicted label was recorded. These confidence scores were visualized using bar charts, offering insight into the model's certainty across temporal segments.

Label Distribution Summary: A statistical summary of frame-level predictions was computed per video. Both bar charts and pie charts were utilized to present the proportion of frames classified as REAL or FAKE, providing an interpretable metric for intra-video classification consistency.

Majority Voting Decision Scheme: The final classification of each video was determined by aggregating frame-level predictions through majority voting. This ensemble-like approach increases robustness by mitigating errors in individual frame classifications, especially under varying lighting, occlusion, or compression conditions.

In addition, the system facilitated comparative evaluation of multiple trained models by allowing the user to select from a set of available .pth checkpoints. This modular approach supports empirical benchmarking under consistent evaluation protocols.

The researchers then integrated all evaluation metrics and visualizations into an interactive web-based interface developed using Streamlit, enabling both quantitative and qualitative assessment of the model's performance on real-world video samples.

G. Graphs

To further understand and have more context on the performance of a DeepFake detection model, the researchers made a series of visualizations that were generated using both training metrics and evaluation results on the validation set. These visual tools provide critical insights into the model's behavior, generalization ability, and classification performance.

Loss Curve: This is a dual-line plot that was generated to illustrate the trend of training and validation loss over epochs. A decreasing trend in both curves generally indicates good convergence, while any significant divergence between them may suggest overfitting.

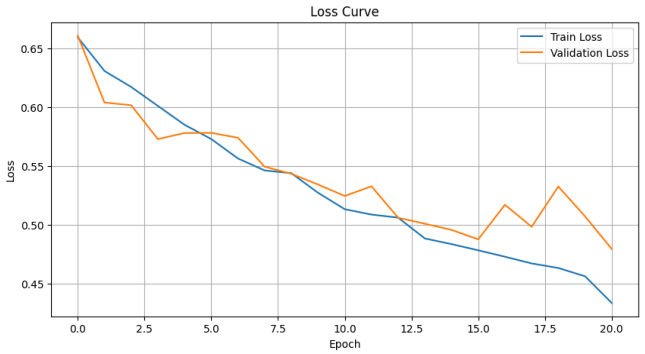


Figure 2. Loss Curve of the Model

Accuracy Curve: Epoch-wise training and validation accuracy were plotted to monitor the model's learning progress. This visualization helps assess how well the model generalizes to unseen data as training progresses.

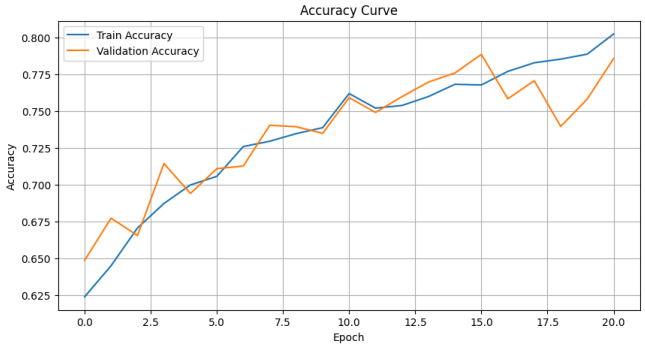


Figure 3. Accuracy Curve of the Model

Confusion Matrix: A normalized confusion matrix was used to present the distribution of correct and incorrect predictions across the two classes: REAL and FAKE. This matrix highlights the model's sensitivity to both types of samples and allows for quick identification of any class imbalance issues or misclassification patterns.

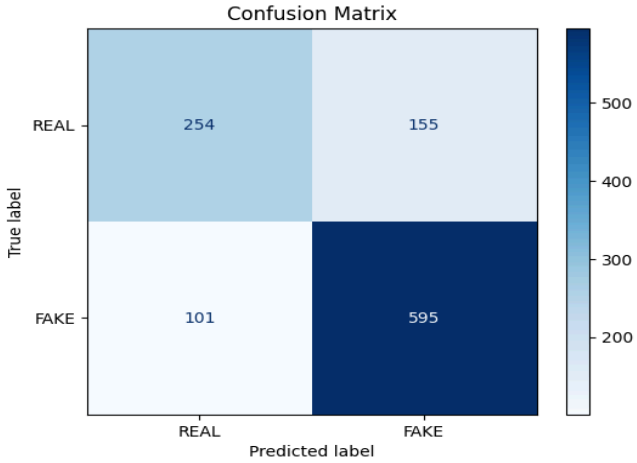


Figure 4. Confusion Matrix of the Model

Classification Report: Quantitative metrics including Precision, Recall, and F1-Score for both REAL and FAKE classes were computed. These measures help interpret the model's robustness and its handling of imbalanced or difficult samples. The classification report was displayed as tabular output in the interface.

```
Classification Report:

              precision    recall  f1-score   support

        REAL       0.72      0.62      0.66       409
        FAKE       0.79      0.85      0.82       696

    accuracy                           0.77      1105
   macro avg       0.75      0.74      0.74      1105
weighted avg       0.76      0.77      0.76      1105
```

Figure 5. Classification Report of the Model

Receiver Operating Characteristic (ROC) Curve: In this study, the ROC curve was used not only as a standard classification evaluation tool but also as a strategic measure to assess the DeepFake detection model's reliability in distinguishing between authentic and manipulated content.

The ROC curve plots the true positive rate (TPR) against the false positive rate (FPR) across various classification thresholds, allowing the researchers to observe how the model performs beyond a single decision boundary. The Area Under the Curve (AUC) was also reported, providing a scalar value summarizing the model's ability to discriminate between classes across thresholds. A higher AUC indicates stronger classification performance.

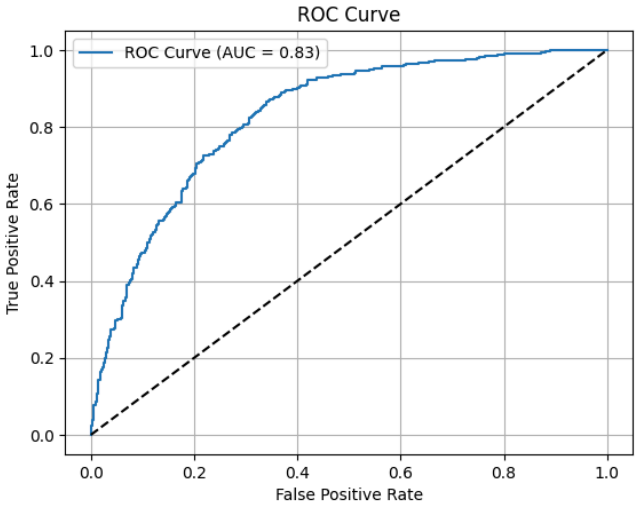


Figure 6. ROC of the Model

These visualizations, especially the training and validation's accuracy and loss, were integrated into the Streamlit interface to facilitate immediate feedback during experimentation. They also enable visual comparison across multiple .pth model checkpoints, which the researchers used to identify the most performant model based on both quantitative metrics and qualitative trends.

IV. Results and Discussion

In this section, the researchers share their findings and discuss their significance. The important results described below are addressed to provide a thorough comprehension of the research findings.

A. Key Findings

The key findings of this study are summarized as follows: Using the dataset by Katkar (2021) that was then customized and preprocessed by the researchers to be smaller and easier to train, the researchers used a binary classification approach with a custom convolutional neural network (CNN), the researchers successfully trained and evaluated a deepfake detection model on a real-world dataset containing both authentic and manipulated videos. The model was trained for 21 epochs using early stopping and achieved a final validation accuracy of 77.19%. When evaluating the model, the researchers observed that the model is showing stronger performance on detecting fake videos, achieving an accuracy of 84.77% (590/696) compared to 64.30% (263/409) on real videos. The researchers have also observed a class imbalance in performance that was further reflected in the classification report, where it shows that fake samples achieved a higher F1-score of 0.82, while real samples had an F1-score of 0.66. The weighted average F1-score across both classes was 0.76, indicating reasonably balanced overall predictive performance. The training process showed continuous improvement, with training accuracy increasing from 62.39% to 80.24% and validation accuracy from 64.86% to 78.85% before early stopping was triggered. The researchers have observed that the model achieved its best validation performance at Epoch 16, where the validation loss was minimized to 0.4875. Moving on to the confusion matrix, the researchers have observed that the model made 155 false negative predictions (real classified as fake) and 101 false positives (fake classified as real), suggesting that there's a higher confidence in identifying fake content while there might be instances that the model will occasionally misclassify real videos. On the other hand, the researchers have also observed that the Receiver Operating Characteristic (ROC) curve showed a consistent upward trend in the true positive rate, with an Area Under the Curve (AUC) of 0.83. This is a sign saying that the model trained has a strong ability to distinguish between real and fake video content most of the time. The key findings of this study demonstrate that the custom CNN model by the researchers, even without pre-trained weights or transfer learning, can effectively detect deepfake content with considerable accuracy. However, the observed disparity in class-specific performance suggests that future work may focus on class balancing techniques or alternative architectures to improve real video classification.

B. Advantages and Limitations

The custom CNN model developed in this study effectively detected deepfake content, achieving a validation accuracy of 77.19% and an AUC of 0.83 without the use of pre-trained weights or transfer learning. This highlights the potential of lightweight, custom architectures for real-world binary classification tasks. The key advantage of the study was the downsizing and preprocessing of the original Katkar (2021) dataset. The researchers decided to do this to allow faster training and lower computational cost. Additionally, Multi-task Cascaded Convolutional Networks or MTCNN was used for face extraction from video frames, enabling the model to focus on relevant facial features. While the researchers have observed that the MTCNN generally performed well in extracting faces per frame, the researchers noted occasional inaccuracies in face localization, especially in cases of low lighting, occlusion, or angled poses resulting into it having to fail to detect faces accurately, introducing noise in the dataset. Despite the strong performance on fake videos, the researchers also observed that the model showed weaker accuracy on real videos, indicating that there's a class imbalance and suggesting a need for improved representation or balancing techniques. Even though the custom model's CNN architecture is efficient, the researchers have also observed that there's instances that the model's ability to capture subtle manipulations in deepfake videos is limited and sometimes misclassify real videos. Overall, the model that was trained and evaluated by the researchers provides a solid and efficient baseline for deepfake detection, though future work should address class performance disparity and explore architectural enhancements for improved generalization.

C. Classifying Real and Fake Videos

To demonstrate the model's performance in practical use, the researchers deployed a Streamlit application capable of running inference on uploaded videos. For evaluation, one real and one fake video were tested using a single representative frame. The app outputs key metrics including an inference summary, frame-by-frame prediction confidence, label distribution, and a pie chart visualization. In the real video example, the model predicted REAL (66.13%), indicating moderate confidence in the classification. In the fake video example, the model predicted FAKE (82.12%), showing strong confidence in identifying manipulated content. These results confirm that the model can deliver clear and interpretable predictions per frame, with the final verdict determined by majority vote across the analyzed frames. The visual and statistical breakdown provides an accessible interface for verifying video authenticity.

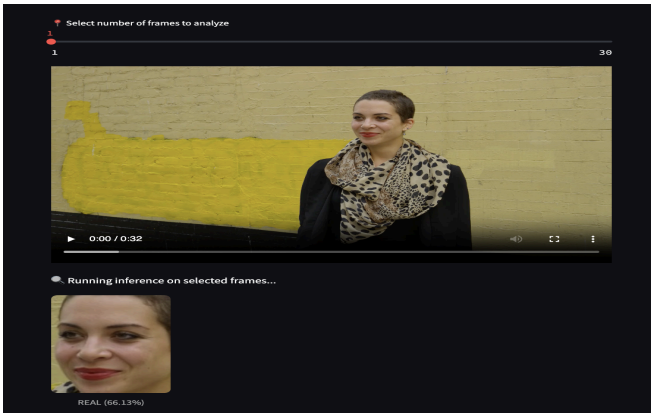


Figure 7. Real video of a woman about to hug someone

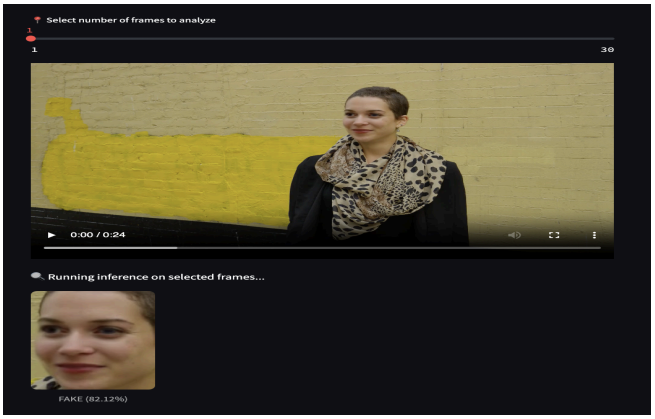


Figure 8. Deepfake video of a woman about to hug someone

V. Conclusion

The increasing sophistication of deepfake technologies poses a significant challenge to digital trust and security, requiring effective detection tools. This study introduces "Behind The Mask," a convolutional neural network framework that utilizes MTCNN for precise face localization, methodical preprocessing (resizing, normalization, and augmentation), and a comprehensive Convolutional Neural Network (CNN) architecture designed for real-time inference. Trained and validated on the Kaggle Deep Fake Detection (DFD) dataset, it achieved a validation accuracy of 77.2%, a weighted F1-score of 0.76, and an AUC of 0.83, so affirming its effectiveness despite the dataset's inherent constraints. Through the integration of data augmentation methods, the researchers maintained essential discriminative features while reducing overfitting, thereby building a solid foundation for dependable deepfake detection.

The researchers are looking forward to future performance that can be improved by integrating more complex, multi-model inputs such as audio elements, temporal dynamics through 3D or recurrent layers, and metadata indicators to offer a comprehensive representation of deepfake artifacts. Expansion of the training corpus with recent deepfake variants and adversarial examples will improve generalizability, while the deployment of real-time streaming APIs and edge-optimized inference can enable live monitoring in practical applications. Furthermore, continuous feature engineering, along with strict validation methods, will be important for adapting the model to changing manipulation techniques and maintaining accuracy in real-world scenarios.

References

[1] D. M. P. Codis, "DICT warns of deepfake surge ahead of midterm polls," *SunStar Cebu*, Oct. 21, 2024. [Online]. Available: https://www.sunstar.com.ph/cebu/dict-warns-of-deepfake-surge-ahead-of-midterm-polls [Accessed: June 9, 2025].

[2] A. Jain et al., "Deepfakes and Digital Trust: A Survey," Proc. ACM Comput. Surv., vol. 55, no. 4, pp. 1–34, Jul. 2023.

[3] S. Patel and R. Gomez, "Quantifying the Proliferation of Deepfake Videos Online," IEEE Trans. Multimedia, vol. 26, pp. 1124–1136, Feb. 2024.

[4] S. N. Katkar, "Deep Fake Detection (DFD) Entire Original Dataset," Kaggle, 2021. [Online]. Available: https://www.kaggle.com/datasets/sanikatiwarekar/deep-fake-detection-dfd-entire-original-dataset [Accessed: May 5, 2025].

[5] M. A. L. Reyes, "The danger of deepfakes," *Philippine Star*, Mar. 23, 2025. [Online]. Available: https://www.philstar.com/business/2025/03/23/2430318/danger-deepfakes [Accessed: June 9, 2025].

[6] A. Miotti and A. Wasil, "Combatting deepfakes: Policies to address national security threats and rights violations," *arXiv*, Feb. 14, 2024. [Online]. Available: https://arxiv.org/pdf/2402.09581 [Accessed: June 9, 2025]

[7] M. S. Rana, B. Murali, and A. H. Sung, "Deepfake Detection Using Machine Learning Algorithms," in *Proc. 2021 10th Int. Congr. Adv. Appl. Informatics (IIAI-AAI)*, Niigata, Japan, 2021, pp. 458–463, doi: 10.1109/IIAI-AAI53430.2021.00079.

[8] H. Vardhan, N. Varshney, M. K. R, P. R, and L. N. R, "Deep fake video detection," Int. Res. J. Adv. Eng. Hub (IRJAEH), vol. 2, no. 4, pp. 830–835, Apr. 2024, doi: 10.47392/IRJAEH.2024.0117.

[9] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "MesoNet: A Compact Facial Video Forgery Detection Network," in IEEE Workshop on Information Forensics and Security (WIFS), Dec. 2018, pp. 1–7, doi: 10.1109/WIFS.2018.8630761.

[10] Amazon Web Services (AWS). Data augmentation in machine learning. [Online]. Available: https://aws.amazon.com/what-is/data-augmentation/ [Accessed: June 10, 2025].

[11] S. Kolagati, T. Priyadharshini, and V. M. A. Rajam, "Exposing deepfakes using a deep multilayer perceptron–convolutional neural

network model," Int. J. Inf. Manag. Data Insights, vol. 2, no. 1, p. 100054, May 2022, doi: 10.1016/j.jjimei.2021.100054.

[12] Papers With Code. Facial Landmark Detection. [Online]. Available: https://paperswithcode.com/task/facial-landmark-detection Accessed: 06/10/2025.

[13] D. Singh, P. Singh, and R. Bhandari, “Enhancing Deepfake Video Detection: A Hybrid CNN-LSTM Approach,” pp. 130–135, Jun. 2024, doi: 10.1109/tiacomp64125.2024.00031.

[14] G. Hannah Chris, K. Rachana, G. Naga Sujini, and N. Musrat Sultana, "Deep Fake Image and Video Detection Using CNN," *Int. J. Res. Publ. Rev.*, vol. 6, no. 4, pp. 17087–17090, Apr. 2025.