Comparative Analysis of Deep Learning Models for Deep Fake Detection in Human Face Images

Bestin K Benny.[b, c, e], Arjun Vasavan.[b, c], E N Aadhila Nazeer.[b, c], Hena Maria Biju.[b, c], Jervin Abraham Kuriakose.[b, c], Geetha S.[a, c, d], Sulaja Sanal.[a, c]
[a]Department of Computer Engineering, College of Engineering Chengannur
[b]Department of Electronics Engineering, College of Engineering Chengannur
[c]APJ Abdul Kalam Technological University, Kerala, India
[d]geetha@ceconline.edu, [e]chn21ae002@ceconline.edu

***Abstract*—The advancements in generative adversarial networks (GANs) and deep learning have enabled the creation of highly realistic manipulated multimedia content, commonly referred to as DeepFakes. While DeepFakes offer transformative applications in media and entertainment, they also pose significant risks, including misinformation and identity theft. This paper presents a DeepFake detection framework that leverages ResNet50, MobileNetV2, and a custom sequential deep convolutional neural network (CNN) model to classify human face images as either real or fake. The system is trained and tested on a dataset containing 700 GAN-generated DeepFake images and 700 real images, sourced from Kaggle. Detection results are deployed using TensorFlow Serving and integrated into a website with FastAPI connecting the backend and user interface. Experimental evaluations demonstrate the proposed system's high accuracy and robustness. Furthermore, the challenges associated with developing generalized detection models are discussed, emphasizing the need for reliable and scalable tools to mitigate the risks posed by DeepFake technology.**

***Index Terms*—DeepFake detection, real or fake classification, GAN-generated datasets, TensorFlow Serving, ResNet50, MobileNetV2, FastAPI.**

I. Introduction

Recent developments in generative adversarial networks (GANs) and deep learning have completely changed the way audiovisual information is produced and altered. Detecting and mitigating the risks associated with DeepFakes has become an urgent research area. Traditional detection methods rely on the identification of artifacts, inconsistencies, or irregularities in manipulated content. However, as DeepFake generation methods improve in sophistication, the need for robust and generalized detection systems becomes increasingly critical.

In order to categorize photos of human faces into two groups, this research presents a DeepFake detection framework: real or fake. The framework leverages state-of-the-art deep learning models, including ResNet50, MobileNetV2, and a sequential deep convolutional neural network architecture, to identify manipulation in face images with high accuracy. The detection framework is trained and evaluated using a balanced dataset comprising 700 GAN-generated DeepFake images and 700 real images sourced from Kaggle.

To ensure scalability and accessibility, the detection results are served using TensorFlow Serving, and a user-friendly website is developed to connect the backend and frontend using FastAPI. The proposed system demonstrates high performance in detecting DeepFakes and provides a scalable solution for real-time applications.

This is how the rest of the paper is structured. Section II examines relevant DeepFake detection research. The suggested methodology is described in Section III, along with the model design, deployment tactics, and dataset preparation. Section IV explains the experimental design and findings, emphasizing the system's efficacy. Section VI wraps up the study, while Section V discusses issues, constraints, and potential paths forward.

II. Literature Review

DeepFake technology, leveraging generative adversarial networks (GANs) to create realistic synthetic media, has raised concerns about image manipulation. Detecting DeepFakes has become a critical research area, with many methods proposed to address the challenges posed by this technology.

Early detection methods focused on pixel-level inconsistencies, but proved ineffective against GANs, which can generate nearly indistinguishable images from real ones. This led to the use of deep learning models for detection [1]. Goodfellow et al. [3] introduced GANs, where the generator and discriminator networks compete to create and detect realistic images, revolutionizing image manipulation but also complicating detection.

A significant advancement in DeepFake detection came with convolutional neural networks (CNNs), which proved effective for image classification. Baldi [4] emphasized the role of autoencoders and unsupervised learning for feature extraction, enabling detection of fine-grained discrepancies in manipulated images. Large-scale models like ImageNet [7] also helped advance detection methods. Tolosana et al. [5] highlighted the need for both pixel- and non-pixel-based approaches to detect subtle artifacts in manipulated faces, with Zeiler and Fergus [8] further visualizing CNNs to identify abnormal patterns in synthetic images. The introduction of deeper CNN architectures like VGG [9] allowed for better detection of subtle facial manipulations.

With the development of large datasets, such as the DeepFake Detection Challenge (DFDC) [14] and Celeb-DF [15], training models to differentiate real and fake images became more feasible. Jiang et al. [16] introduced DeeperForensics-1.0, offering a diverse set of samples for more robust training and evaluation.

Beyond CNNs, approaches like Capsule Networks [20] have been explored for their ability to capture spatial hierarchies in images, improving face forgery detection. However, adversarial attacks still pose a challenge, as shown by Gandhi and Jain [23], who demonstrated that perturbations could deceive detectors. Methods like OC-FakeDect [26], using one-class variational autoencoders, aim to enhance resilience against adversarial modifications. Despite progress, challenges remain, including generalization to unseen manipulations and vulnerability to adversarial attacks. This paper addresses these issues by:

- **Enhanced Generalization Across Datasets:** Using a balanced dataset of GAN-generated and real images from diverse sources to ensure consistent model performance.
- **Real-Time Application:** Integrating detection models with FastAPI and TensorFlow Serving for real-time predictions, overcoming delayed detection systems.
- **Optimized Deep Learning Frameworks:** Leveraging pre-trained deep CNNs like ResNet50 and MobileNetV2 for efficient feature extraction, addressing scalability and reliability concerns.

This approach bridges the gap between theoretical advancements in DeepFake detection and practical, deployable solutions for identifying manipulated face images.

III. Methodology

The detection of GAN-generated images involves a systematic approach encompassing data preparation, model development, and deployment. The workflow for this task integrates several critical components, as illustrated in Fig. 1. The key steps include data cleaning and preprocessing, model building using deep learning architectures, and deployment through efficient APIs.

This study focused on developing a robust pipeline for detecting GAN-generated images using curated datasets of real and synthetic content. TensorFlow was employed for model development and training, while FastAPI facilitated the deployment of the detection system. Three deep learning models—custom Convolutional Neural Networks (CNN), ResNet50, and MobileNetV2—were utilized for feature extraction and image classification. To enhance model performance, preprocessing steps, including resizing, normalization, and augmentation, were applied to the dataset. The trained models were integrated into a web-based system using TensorFlow Serving, enabling efficient and real-time detection of GAN-generated images. This approach ensures a seamless workflow from data preparation to deployment, delivering accurate and scalable results.

A. Datasets and Preprocessing

A balanced dataset of 700 real images and 700 GAN-generated images was used to train the model, sourced from Kaggle. The images, initially 1024 × 1024 pixels, were resized to 225 × 225 pixels with RGB channels and normalized to [0, 1]. The dataset was split into 80% for training and 20% for validation. To enhance model generalization, data augmentation techniques such as random flips and 20% rotations were applied. The dataset included real images from authentic sources and GAN-generated images created using StyleGAN, ensuring high visual realism and clear distinctions from real images.
Real images from the accessed from kaggle at the following link: https://www.kaggle.com/c/deepfake-detection-challenge/discussion/122786.
GAN generated images from the accessed from kaggle at the following link: https://www.kaggle.com/datasets/tunguz/1-million-fake-faces.

B. Proposed Workflow

The overall workflow for detecting GAN-generated images is presented in Fig. 1. The pipeline consists of four main components: Data Cleaning and Preprocessing, Model Building, TensorFlow Serving, and FastAPI Integration. Each component is outlined below.

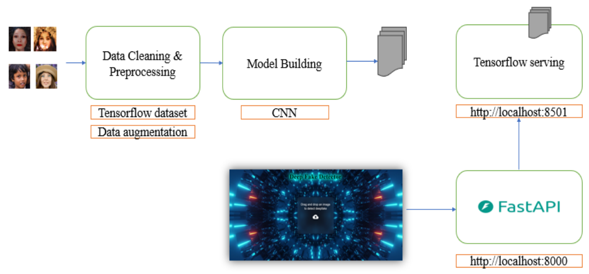

Fig. 1. Proposed Workflow for GAN-Generated Image Detection

1) **Data Cleaning and Preprocessing:** The dataset undergoes preprocessing, including resizing images, normalizing pixel values, and applying augmentation techniques to enhance the model's learning ability and increase data diversity.
2) **Model Building:** Three deep learning models are built and trained:
   - **Custom CNN:** A baseline custom CNN model with convolutional, max-pooling, and fully connected layers.
   - **ResNet50:** A fine-tuned pre-trained ResNet50 model known for its feature extraction capabilities.
   - **MobileNetV2:** A lightweight, pre-trained MobileNetV2 model, ideal for resource-constrained devices.

   Each model is evaluated for its performance in classifying GAN-generated and real images.
3) **TensorFlow Serving:** The best-performing model is deployed using TensorFlow Serving, providing a scalable solution for integrating the model into production environments with REST or gRPC APIs.
4) **FastAPI Integration:** A web interface is developed using FastAPI, enabling real-time predictions. The trained

model is connected to the FastAPI application, allowing users to upload images and receive classification results instantly.

C. Model Architectures

This section describes the architectures utilized for detecting GAN-generated images, focusing on their design and advantages in this context. Three models are used: a custom Convolutional Neural Network (CNN), ResNet50, and MobileNetV2. These architectures were chosen for their complementary strengths in feature extraction, computational efficiency, and classification performance.

1) Custom Convolutional Neural Network (CNN): The custom CNN architecture is designed to provide a lightweight solution for binary classification of GAN-generated and real images, offering a baseline for performance evaluation. It is tailored to detect spatial features in images, which are critical for distinguishing subtle GAN artifacts.

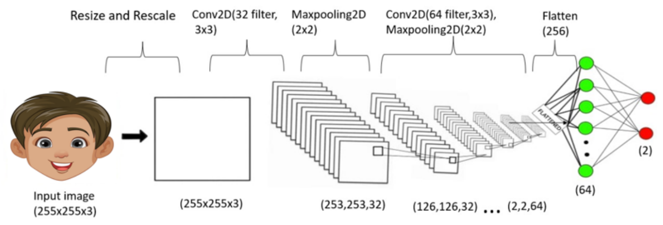

Fig. 2. Custom CNN Architecture.

The network comprises six convolutional layers with ReLU activation to extract low- and high-level spatial features, MaxPooling layers after each convolutional layer to reduce spatial dimensions and prevent overfitting, a flatten layer to transform spatial features into a one-dimensional vector, and dense layers for classification with a softmax activation for binary output. The custom CNN is simple, computationally efficient, and provides a solid baseline for comparison with more complex architectures.

2) ResNet50: ResNet50 is employed for its robust residual learning capabilities, making it well-suited for training deep networks without degradation in performance. Residual blocks in ResNet50 enable the model to learn deep representations, capturing intricate features that differentiate real and GAN-generated images. To provide gradient stability during training,

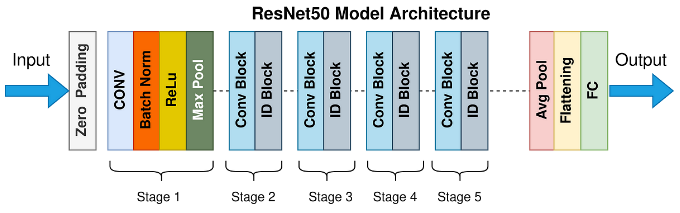

Fig. 3. ResNet50 Architecture. Adapted from [27].

the architecture incorporates a 50-layer deep residual network with identity connections. It is fine-tuned by adding thick layers with sigmoid activation for binary classification and a global average pooling layer after being pre-trained on the ImageNet dataset for feature-rich initialization. ResNet50 excels in extracting complex features, particularly useful for detecting high-level GAN artifacts, and mitigates the risk of vanishing gradients.

3) MobileNetV2: MobileNetV2 is selected for its computational efficiency and suitability for resource-constrained environments. Its lightweight architecture effectively balances computational cost and performance, making it ideal for deployment on devices with limited resources. The architecture employs depthwise separable convolutions to reduce computational complexity without compromising performance. An inverted residual structure with linear bottlenecks ensures efficient feature extraction. MobileNetV2 (Fig. 4) is initialized with ImageNet pre-trained weights and fine-tuned for binary classification using dense layers with a sigmoid activation. This architecture achieves efficient feature extraction, enabling it to process high-resolution images while maintaining classification accuracy.

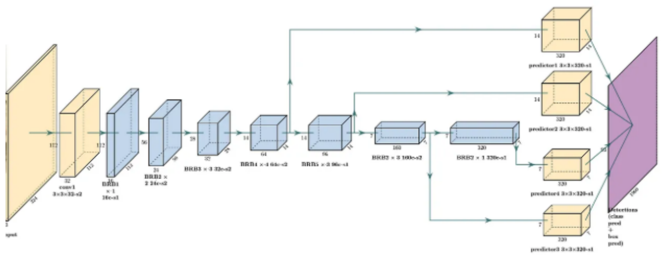

Fig. 4. MobileNetV2 Architecture. Adapted from [28].

D. Implementation Details

1) Software Requirements:: This section outlines the tools, libraries, and hardware or software environment utilized for training and deploying the models. The implementation was carried out using Python 3.11 for coding and experimentation, with TensorFlow 2.14 serving as the primary deep learning framework for building, training, and evaluating the models. FastAPI, a modern web framework, was employed for API development, while ReactJS was used to design the user interface for interacting with the models. Development was carried out primarily on Visual Studio Code (VS Code), which served as the integrated development environment (IDE).

2) System Requirements:: The experiments and deployment were performed on a system powered by an AMD Ryzen 5 5600H processor with Radeon Graphics, operating at 3.30 GHz. The system had 16 GB of RAM (15.3 GB usable) and was equipped with a 64-bit Windows 11 operating system. This hardware and software setup ensured an efficient environment for model training, testing, and deployment.

E. Evaluation Metrics

To evaluate the performance of custom deep convolutional neural network (CNN) models, the following metrics were employed, specifically tailored for binary classification tasks:

- **Accuracy:**Evaluates the model's overall accuracy :
$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$
- **Precision:** Evaluates the ratio of correct positive predictions:
$$\text{Precision} = \frac{TP}{TP + FP}$$
- **Recall:** Measures the model's ability to identify true positives:
$$\text{Recall} = \frac{TP}{TP + FN}$$
- **F1-Score:** The harmonic mean of precision and recall:
$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$
- **AUC:** Area under the ROC curve, indicating the model's discriminative ability:
$$\text{AUC} = \int_0^1 \text{TPR}(x)\, dx$$

F. Results and Discussion

The performance of the three deep learning architectures, namely custom Deep CNN, MobileNet, and ResNet, for deepfake detection was evaluated using multiple metrics such as accuracy, precision, recall, F1-score, and AUC. The models were trained and validated on a dataset, and their performance was analyzed to assess their ability to detect deepfakes effectively.

1) Model Performance:

1) **Training Dataset Performance:** Upon evaluation on the training dataset, the models exhibited varying levels of performance (Table 1).
MobileNet performed the best, with an accuracy of 93.25%, precision of 92.89%, recall of 93.67%, F1-score of 93.28%, and AUC of 98.37%, demonstrating its robustness in detecting deep fakes. The custom deep CNN, while achieving perfect recall (1.00), struggled with low precision (50.54%) and an F1-score of 67.14%, reflecting a high false positive rate. ResNet performed similarly poorly, with an accuracy of 50.36%, recall of 73.43%, F1-score of 59.66%, and AUC of 50.58%, indicating significant limitations in learning and class discrimination.
2) **Validation Dataset Performance:** On the validation dataset, the models exhibited the following performance (Table 2). MobileNet demonstrated strong generalization on unseen data, achieving an accuracy of 74.00%, precision of 76.09%, recall of 70.00%, F1-score of 72.92%, and AUC of 84.94%, despite a slight drop in recall. The custom deep CNN maintained perfect recall (1.00) but had low precision (54.69%) and AUC (0.4243), indicating challenges in distinguishing true positives from false positives. ResNet continued to underperform, with an accuracy of 50.00%, precision of 50.00%, recall of 70.00%, F1-score of 58.33%, and AUC of 54.56%, highlighting its limited ability to generalize and accurately classify the validation data.

TABLE I
COMPARISON OF PERFORMANCE METRICS FOR CUSTOM DEEP CNN, MOBILENET, AND RESNET ON THE TRAINING DATASET.

Metrics	**Training Dataset**		
	Custom Deep CNN	**MobileNet**	**ResNet**
Accuracy	50.54%	93.25%	50.36%
Precision	50.54%	92.89%	50.24%
Recall	100%	93.67%	73.43%
F1-Score	67.14%	93.28%	59.66%
AUC	53.59%	98.37%	50.58%

TABLE II
COMPARISON OF PERFORMANCE METRICS FOR CUSTOM DEEP CNN, MOBILENET, AND RESNET ON THE VALIDATION DATASET.

Metrics	**Validation Dataset**		
	Custom Deep CNN	**MobileNet**	**ResNet**
Accuracy	54.69%	74.00%	50.00%
Precision	54.69%	76.09%	50.00%
Recall	100%	70.00%	70.00%
F1-Score	70.71%	72.92%	58.33%
AUC	42.43%	84.94%	54.56%

3) **Model Comparison:** The results highlight that MobileNet consistently outperformed both custom Deep CNN and ResNet in terms of accuracy, precision, recall, F1-score, and AUC, both for the training and validation datasets. MobileNet's superior performance can be attributed to its efficient architecture, which allows it to learn robust features while maintaining good generalization capabilities. In contrast, custom Deep CNN showed high recall but lacked precision, and ResNet failed to achieve satisfactory performance in either dataset, suggesting it was unable to effectively capture the complexities of deepfake detection.

2) Validation and Training Loss and Accuracy: : The performance of each model across the training and validation phases is further visualized in the following figures:

- The custom Deep CNN training and validation accuracy and loss curves are depicted in Fig. 5. The graph indicates that the custom Deep CNN is overfit, as indicated by the notable discrepancy between the training and validation accuracy and loss curves.
- The accuracy and loss of MobileNet training and validation are displayed in Fig. 6. Even if the loss continuously declines, the widening disparity between training and validation accuracy points to mild overfitting.
- The accuracy and loss of ResNet during training and validation are shown in Fig. 7. ResNet shows poor performance, with both training and validation accuracy stagnating at low levels, and the loss curve not converging effectively.

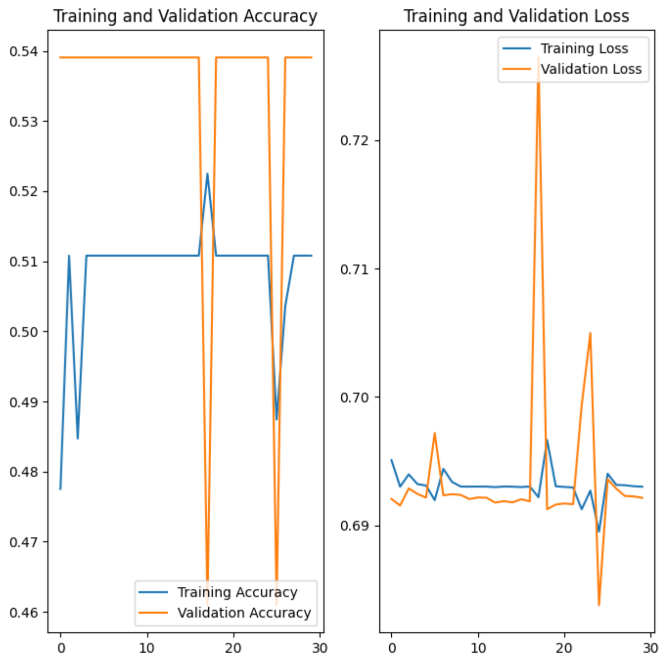

Fig. 5. Validation and Training Loss and Accuracy for the custom Deep CNN model.

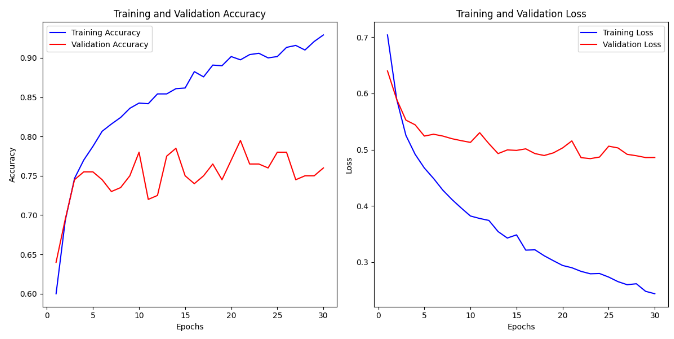

Fig. 6. Validation and Training Loss and Accuracy for the MobileNetV2 model.

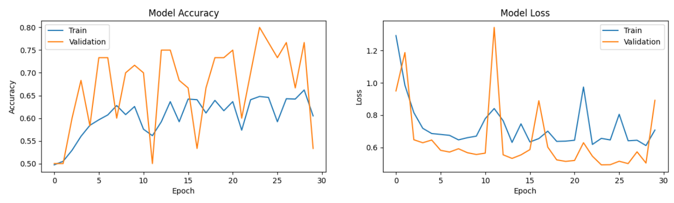

Fig. 7. Validation and Training Loss and Accuracy for the ResNet50 model.

G. Visualization of Results

The system's results are displayed through an interactive web UI developed with FastAPI, allowing users to upload images for deep fake detection. When an image is uploaded, the system processes it and displays the predicted label (real or GAN-generated) and the confidence score in real-time, making it simple to evaluate the model's performance (Fig. 8). The output visualization helps users understand how well the model differentiates between real and GAN-generated images.

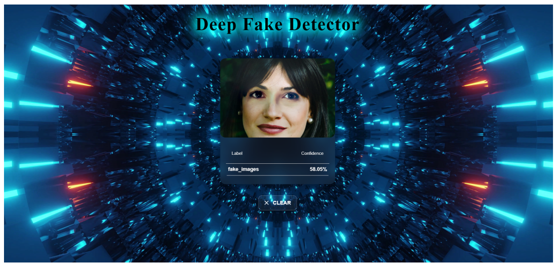

Fig. 8. Model classification output with predicted label and confidence score.

IV. Limitations

The Deep Fake Detector encounters challenges in model generalization, computational efficiency, and scalability. The Custom CNN model exhibits overfitting, resulting in high false positives, while ResNet50 shows limited accuracy (50%). The dataset size (1,400 images) restricts the model's ability to generalize across diverse DeepFake manipulations. The absence of a dedicated GPU increases training time and inference latency, limiting real-time applications. Additionally, high power consumption affects continuous operation, and FastAPI with TensorFlow Serving may experience scalability issues under heavy workloads. Addressing these challenges requires dataset augmentation, model optimization, multi-modal analysis, and scalable deployment strategies to enhance detection performance.

V. Discussion

The evaluation of the deep fake detection models, including the custom Convolutional Neural Network (CNN), MobileNetV2, and ResNet50, yielded varied performance across different architectures. The custom CNN model showed moderate results, indicating potential but also suggesting the need for further optimization. Although it was able to process the data, it failed to generalize well to the validation set, pointing to issues such as overfitting or insufficient model complexity.

MobileNetV2, a pre-trained model, demonstrated superior performance due to its lightweight and efficient architecture. It exhibited better generalization and higher accuracy than the custom CNN, highlighting its robustness in detecting deep fake images. Its high AUC further indicated the model's ability to differentiate between real and fake content effectively.

On the other hand, the ResNet50 model, although a powerful pre-trained architecture, exhibited limited performance in this case. The low accuracy and AUC suggested that additional training or fine-tuning was required to leverage the deep residual connections effectively. The performance discrepancy indicates that further experimentation, such as adjusting hyperparameters or adding additional layers, may be necessary to improve its results.

VI. Conclusion

In conclusion, MobileNetV2 demonstrated superior performance in deep fake detection, effectively balancing accuracy and computational efficiency. Its lightweight structure enabled robust detection capabilities, making it suitable for real-world deployment, particularly in environments with limited resources. The custom CNN model showed potential; however, its performance necessitates further refinement to enhance both accuracy and generalization. The ResNet50 model, while effective in feature extraction, requires additional fine-tuning to improve its detection accuracy for deep fake images.

In order to help the model prioritize and capture more discriminative information in the photos, future research in deep fake detection can concentrate on investigating sophisticated deep learning techniques, such as the incorporation of attention processes. Furthermore, by subjecting detection algorithms to increasingly intricate perturbations, adversarial training with Generative Adversarial Networks (GANs) may increase their robustness. In order to maximize feature extraction and better the handling of high-resolution datasets, these models could be further improved by hybrid architectures that combine convolutional neural networks (CNNs) with transformers or other sophisticated models. Furthermore, by offering a more comprehensive understanding of the material, multi-modal data—such as merging visual and auditory information—could enhance the detection of deep fakes. These advancements would contribute to creating more accurate, efficient, and reliable systems for detecting deep fake content in practical applications.

References

[1] A. Malik, M. Kuribayashi, S. M. Abdullahi, and A. N. Khan, "Deep-Fake Detection for Human Face Images and Videos: A Survey," in *IEEE Access*, vol. 10, pp. 18757-18775, 2022, doi: 10.1109/ACCESS.2022.3151186.

[2] H. Farid, "Image forgery detection," *IEEE Signal Process. Mag.*, vol. 26, no. 2, pp. 16-25, Mar. 2009.

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 19-27.

[4] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," in *Proc. ICML Workshop Unsupervised Transf. Learn.*, 2012, pp. 37-49.

[5] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, "Deepfakes and beyond: A survey of face manipulation and fake detection," *Inf. Fusion*, vol. 64, pp. 131-148, Dec. 2020.

[6] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, and T. Chen, "Recent advances in convolutional neural networks," *Pattern Recognit.*, vol. 77, pp. 354-377, May 2018.

[7] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, and A. C. Berg, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211-252, Dec. 2015.

[8] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, Cham, Switzerland: Springer, 2014, pp. 818-833.

[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, arXiv:1409.1556.

[10] M. Liu, Y. Ding, M. Xia, X. Liu, E. Ding, W. Zuo, and S. Wen, "STGAN: A unified selective transfer network for arbitrary image attribute editing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3673-3682.

[11] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Niessner, "Face2Face: Real-time face capture and reenactment of RGB videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2387-2395.

[12] J. Thies, M. Zollhöfer, and M. Niessner, "Deferred neural rendering: Image synthesis using neural textures," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 1-12, 2019.

[13] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1125-1134.

[14] B. Dolhansky, J. Bitton, B. P aum, J. Lu, R. Howes, M. Wang, and C. C. Ferrer, "The DeepFake detection challenge (DFDC) dataset," 2020, arXiv:2006.07397.

[15] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-DF: A large-scale challenging dataset for DeepFake forensics," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 3207-3216.

[16] L. Jiang, R. Li, W. Wu, C. Qian, and C. C. Loy, "DeeperForensics-1.0: A large-scale dataset for real-world face forgery detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2889-2898.

[17] A. Haliassos, K. Vougioukas, S. Petridis, and M. Pantic, "Lips Don't Lie: A generalisable and robust approach to face forgery detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 5039-5049.

[18] F. Lugstein, S. Baier, G. Bachinger, and A. Uhl, "PRNU-based deep fake detection," in *Proc. ACM Workshop Inf. Hiding Multimedia Secur.*, Jun. 2021, pp. 7-12.

[19] D. Guera and E. J. Delp, "Deepfake video detection using recurrent neural networks," in *Proc. 15th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Nov. 2018, pp. 1-6.

[20] H. H. Nguyen, J. Yamagishi, and I. Echizen, "Capsule-forensics: Using capsule networks to detect forged images and videos," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 2307-2311.

[21] H. Jeon, Y. Bang, J. Kim, and S. S. Woo, "T-GD: Transferable GAN-generated images detection framework," 2020, arXiv:2008.04115.

[22] C.-C. Hsu, Y.-X. Zhuang, and C.-Y. Lee, "Deep fake image detection based on pairwise learning," *Appl. Sci.*, vol. 10, no. 1, p. 370, Jan. 2020.

[23] A. Gandhi and S. Jain, "Adversarial perturbations fool deepfake detectors," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1-8.

[24] X. Wu, Z. Xie, Y. Gao, and Y. Xiao, "SSTNet: Detecting manipulated faces through spatial, steganalysis and temporal features," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 2952-2956.

[25] Z. Liu, X. Qi, and P. H. S. Torr, "Global texture enhancement for fake face detection in the wild," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8060-8069.

[26] H. Khalid and S. S. Woo, "OC-FakeDect: Classifying deepfakes using one-class variational autoencoder," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 656-657.

[27] R. Gomes, C. Kamrowski, J. Langlois, P. Rozario, I. Dircks, K. Grottodden, M. Martinez, W. Tee, K. Sargeant, C. LaFleur, and M. Haley, "A Comprehensive Review of Machine Learning Used to Combat COVID-19," Diagnostics, vol. 12, p. 1853, Jul. 2022, doi: 10.3390/diagnostics12081853.

[28] A. Das, "MobileNet: Revolutionizing Mobile and Edge Computing with Efficient Neural Networks," *Medium*, Oct. 4, 2023. [Online]. Available: https://medium.com/the-modern-scientist/mobilenet-revolutionizing-mobile-and-edge-computing-with-efficient-neural-networks-5a2cd01a4e47. [Accessed: Jan. 14, 2025].