Comparative Study of Custom CNN and ResNet18 for DeepFake Image Classification

Beria Nyangake

I. Introduction

The development of generative models like Stable Diffusion, DALL·E, [1] and GANs (Generative Adversarial Networks) [2], have advanced so quickly that it is now possible to create synthetic images that are almost identical to genuine ones. [3]. Significant questions concerning digital forensics, false information, and media trust have been brought up by this developing reality. [4]. Originally used to describe edited videos, the term "DeepFake" today refers to a wider variety of AI-generated visual information that may deceive both the human eye and even simple algorithms. [5]. Therefore, it is essential to develop reliable computational techniques to differentiate between actual and artificial intelligence-generated images. [6].

Convolutional neural networks (CNNs) were the main tool used in early DeepFake detection research because of the way they could effectively capture local texture and pattern variations. [7] Real-time detection techniques and effective model designs have been the subject of numerous investigations. For example, the paper "Faster Than Lies" [8] highlights computational economy without sacrificing accuracy by introducing a binary neural network tuned for real-time DeepFake detection. To identify images produced by deep diffusion models, another noteworthy method [9] looks at intrinsic dimensionality, providing fresh perspectives on the minute statistical distinctions between artificial and natural images.

The objective of this project is to create an original convolutional neural network (CNN) for DeepFake detection and assess how well it performs in comparison to the popular ResNet18 architecture. The benchmark for the experiment is the CIFAKE dataset, which consists of both actual and AI-generated images. Although transformer-based models, such as the Vision Transformer (ViT), have produced state-of-the-art outcomes across a range of fields, their high processing requirements render them impractical for real-time or lightweight applications. [10] making CNN a better fit for the position at hand.

The objectives of the project include:

- Designing and training a custom CNN tailored for DeepFake detection.
- Benchmarking the custom CNN against established models like ResNet18
- Analysing model performance to identify which architectures are most effective for this task.
- Developing a simple AI-powered application that allows users to upload images and receive a real/fake prediction.

II. Dataset

Overview

The classification model uses a 32x32 pixel colour image from the CIFAKE dataset as its input. There are two classes for each image: FAKE and REAL. The model's output is a binary label of 0 and 1 that indicates if the image is real or artificial intelligence (AI) created. CIFAKE is openly accessible on Kaggle and was first presented by Bird & Lotfi [11]. While FAKE images are created using Stable Diffusion v1.4, an effective generative model that can create photorealistic synthetic content, the REAL class is obtained from the CIFAR-10 dataset [12]. The 120,000 photos that make up the dataset CIFAKE are split equally between two classes. The dataset split is summed up in the following table:

Split	Total Images	REAL Class	FAKE Class
Train	100000	50000	50000
Validation	20000	10000	10000
Test	20000	10000	10000
Total	140000	70000	70000

Table 1

Training/Validation/Test Samples

For model training and evaluation, the dataset is split into a training set (83.3%) and a test set (16.7). The training set consists of 100,000 images, with 50,000 real images and 50,000 fake images. The testing set contains 20,000 images, split equally between the real and fake classes (10,000 per class).

The dataset does not explicitly contain a validation set, but the training set will be split into training and validation subsets during the model training process. This will help in tuning model parameters and avoiding overfitting.

Preprocessing and Augmentation

The CIFAKE dataset required minimal preprocessing due to its consistent structure and standardized 32x32 image format. However, additional steps were applied to optimize model performance and compatibility, especially for deeper architectures like ResNet18 and the custom CNN, therefore, all images were resized to 128x128 pixels for training and testing. The training set underwent resizing and augmentation, including random horizontal flips, slight rotations, and colour jittering, to increase data diversity and improve the model's robustness to changes in orientation and lighting conditions. Images were then converted to tensors and normalized to ensure consistent input scaling for stable training. For the testing set, only resizing and normalization were applied to maintain consistency while preserving evaluation integrity. (Figure 1) is a visual example representing samples from both classes, highlighting the diversity and complexity of the classification task. Label 1 being REAL, label 0, FAKE.

Figure 1: Samples from both Real & Fake classes

III. Methodology

A. Model

Two different models were experimented with in this study to compare their performance:

a. ResNet-18 (pre-trained)

A deep residual convolutional neural network architecture, ResNet-18, was pretrained using the extensive ImageNet dataset, which has more than 14 million annotated images in 1,000 different classifications. [13]. Its demonstrated performance in picture classification tasks and its appropriateness for transfer learning on smaller, domain-specific datasets such as CIFake led to its selection for this work. and ability to leverage transfer learning for small datasets. (Figure 2) presents the ResNet-18 architecture

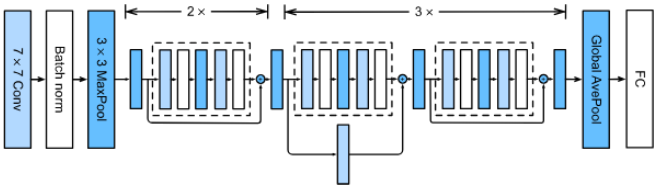

Figure 2: ResNet-18 architecture

The 18 layers that make up the ResNet-18 architecture include residual blocks and convolutional layers. [14] Utilizing residual blocks with skip connections, which enables the network to learn residual functions, is the fundamental innovation in ResNet models.

For this study, a new linear layer that produced two classes matching to the binary classification task (REAL vs. FAKE) was used in place of the original final fully connected (FC) layer, which was intended for 1,000 classes. Only the FC layer and the final residual block (referred to as layer 4) were unfrozen and trained using the CIFake dataset. To avoid overfitting on the smaller dataset, this approach adapts higher-level features to the domain while maintaining the useful pretrained features in previous layers. Training Procedure
5-fold cross-validation (k = 5) was used for fine-tuning to ensure the results' generalisability and robustness. The Stochastic Gradient Descent (SGD) optimizer was used for enhancing the network with: Learning Rate η = 0.01

Cross-entropy loss function [15] defined as:

$$\text{Loss} = -\sum_{i=1}^{\text{output size}} y_i \cdot \log \hat{y}_i$$

Figure 3 Cross entropy loss

(Figure 3) Shows the categorical cross-entropy loss used in multi-class classification, calculating the negative sum of the true labels (yi) multiplied by the logarithm of the predicted probabilities (y hat i) for each class (i). The loss measures how well the predicted probabilities match the true labels, penalizing incorrect predictions more and helping the model improve by increasing the probability of the correct class

The ResNet18 model was trained for 5 epochs per fold using a batch size of 16. Performance metrics were evaluated on both the training and validation sets after each epoch to monitor learning progress and prevent overfitting.

b. Custom CNN

A lightweight CNN with fewer parameters and quicker training was created as a train-from-scratch substitute. This model examines if ResNet-18 can be outperformed by a more straightforward architecture.
Concerning the architecture (Figure 4), ReLU activations, batch normalization, max pooling, and convolutional layers make up the CNN's three convolutional blocks, which are used to gradually extract and down sample feature maps.

```
Sequential(
  (0): Conv2d(3, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (1): ReLU()
  (2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (4): Conv2d(32, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (5): ReLU()
  (6): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (7): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (8): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (9): ReLU()
  (10): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (11): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (12): Flatten(start_dim=1, end_dim=-1)
  (13): Linear(in_features=32768, out_features=512, bias=True)
  (14): ReLU()
  (15): Linear(in_features=512, out_features=256, bias=True)
  (16): ReLU()
  (17): Linear(in_features=256, out_features=2, bias=True)
)
```

Figure 4: Custom CNN architecture

Training:
Optimizer: Adam with learning rate η=0.001
Epochs: 10, Batch size: 16
k=5k Cross-Validation
The cross-entropy loss function was used similarly as with ResNet-18. Table 2 shows a side-by-side comparison.

HyperParameter of ResNet18 vs Custom CNN

Parameter	ResNet-18	Custom CNN
Learning rate	0.01 (SGD)	0.001 (Adam)
Batch size	16	16
Epochs	5	10
Optimizer	SGD	Adam
Cross-validation	5-fold	5-fold

Table 2

Cross-Validation Procedure
To ensure robustness and avoid overfitting, both models were evaluated using 5-fold cross-validation. The dataset was split into 5 equal folds maintaining class balance. For each fold:

- The model was trained on 4 folds
- Validated on the remaining fold

Performance metrics accuracy & loss were recorded
The results from all folds were averaged for final performance estimation in model performance.

IV. Experiment

A. Evaluation Metrics

The models were evaluated using the following metrics, Accuracy measuring the overall correctness of the model, Precision, accessing the accuracy of positive predictions, Recall, exploring the model's ability to find all positive samples and F1 score the harmonic mean of precision and recall, the formulas represented in (Figure 5).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}},$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}},$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}},$$

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

.

Figure 5

B. Quantitative Results

Model	Train Loss	Train Accuracy	Validation Loss	Validation Accuracy
ResNet18	0.072731	0.973787	0.071200	0.973800
Custom CNN	0.145691	0.943225	0.148945	0.944950

Table 3

Metric	ResNet18	Custom CNN
Accuracy	0.977300	0.950500
Precision	0.982800	0.951900
Recall	0.971600	0.948900
F1 Score	0.977200	0.950400

Table 4

C. Custom CNN results Analysis

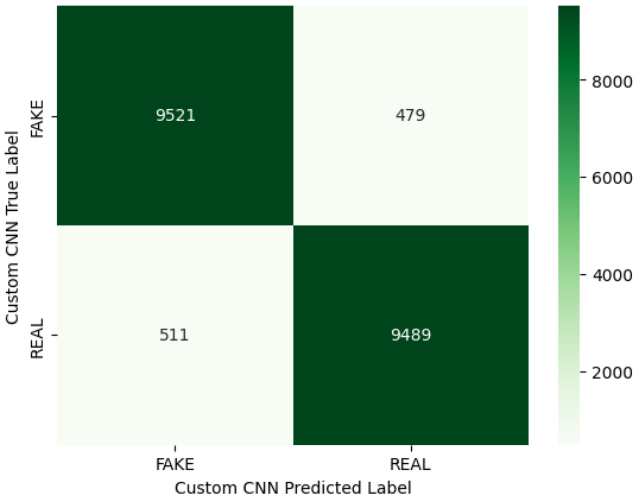

Figure 6: Custom CNN Confusion Matrix Performance

The confusion matrix (Figure 6) indicates that the model maintains a balanced classification performance between the "FAKE" and "REAL" classes. While there are slightly more false positives and false negatives compared to true predictions, the overall misclassification rates remain low, demonstrating effective discrimination by the model. This is supported by the high validation accuracy of approximately 94% and test accuracy of 95%, which suggest strong generalization to unseen data.

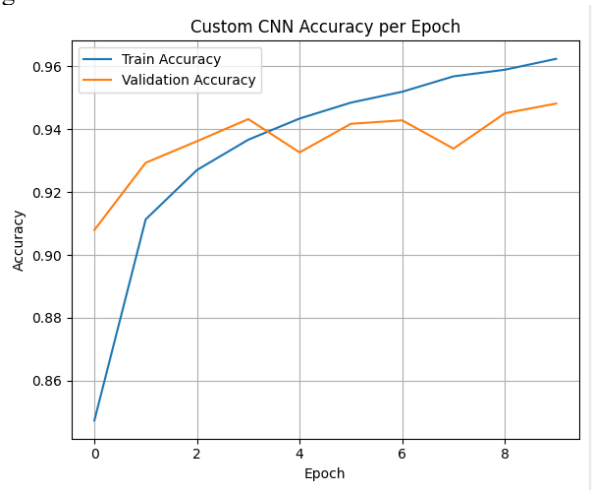

Figure 7

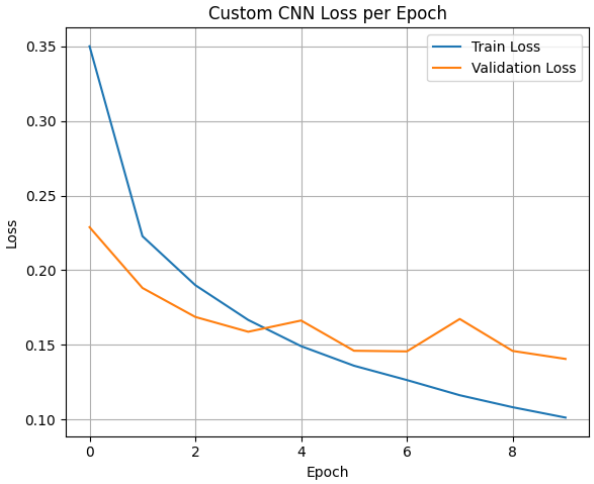

Figure 8: Custom CNN Loss Per Epoch

The analysis of the epoch-wise performance reveals that both the training and validation metrics converge effectively. Specifically, the validation accuracy (Figure 7) approaches a stable state, indicating a plateau in performance improvement. The corresponding validation loss (Figure 8) also stabilises, and the minimal divergence between training and validation curves suggests that the model generalised well to unseen data and did not show significant signs of overfitting within the 10-epoch training duration.

D. Reset18 results Analysis

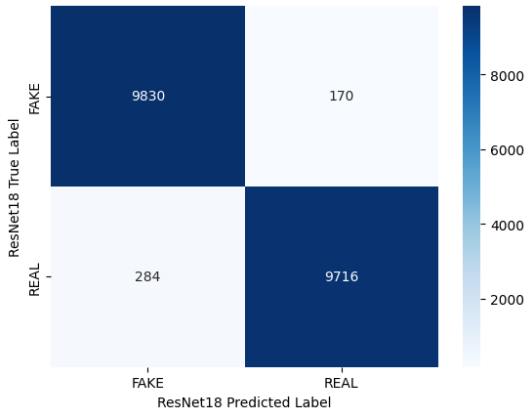

Figure 9: ResNet18 Confusion Matrix Performance

ResNet18 demonstrates superior performance compared to the Custom CNN, achieving higher accuracy and fewer misclassifications, as reflected in its confusion matrix (Figure 9). The model shows stronger confidence in correctly predicting both "FAKE" and "REAL" classes, which corresponds with its higher validation accuracy (98%) and test accuracy (98%). This performance advantage stems from ResNet18's deeper architecture and residual connections, which enable more effective feature extraction and better handling of complex patterns in the data.

Across all evaluation metrics, (Table 4), accuracy, precision, recall, and F1 score, ResNet18 consistently outperforms the Custom CNN, indicating its enhanced ability to generalise beyond the training set. (Figure 10).

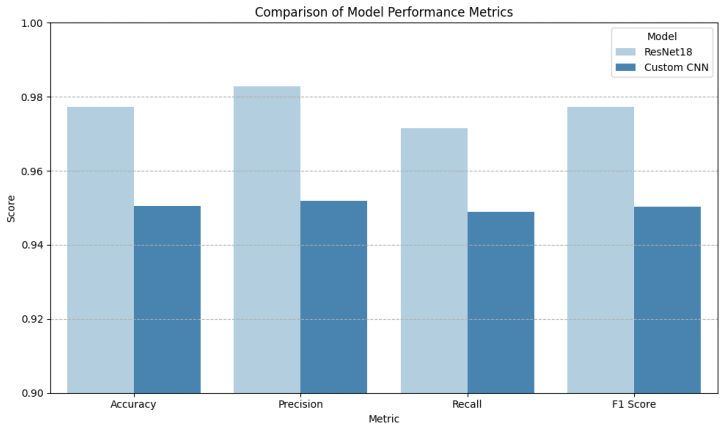

Figure 10: Model Matrics Performance Comparison

E. Qualitative Results

Qualitative analysis based on confusion matrices and example images reveals that both models perform well in distinguishing between "REAL" and "FAKE" images, with ResNet18 showing superior accuracy and fewer misclassifications. ResNet18's deeper architecture and pre-trained feature extraction enable it to generalize better, especially in images with subtle or complex visual cues. Correct classifications from both models align with clear, distinctive features, while misclassifications often involve ambiguous textures, lighting, or highly realistic fakes. These errors highlight the limits of each model's feature representation, with the custom CNN struggling more in fine-grained cases.

Figure 11: Correctly Classified CNN Examples:

F. Comparisons

Both models effectively classify "REAL" and "FAKE" images, but ResNet18 consistently outperforms the custom CNN across all metrics, achieving 98% accuracy on both validation and test sets. This superior performance is due to its deeper architecture and pre-training on ImageNet, enabling better feature extraction and generalisation. While the custom CNN performs well (94–95% accuracy), it struggles more with complex patterns. Confusion matrices and visual inspection confirm ResNet18's robustness, particularly on ambiguous images, highlighting the clear benefits of transfer learning.

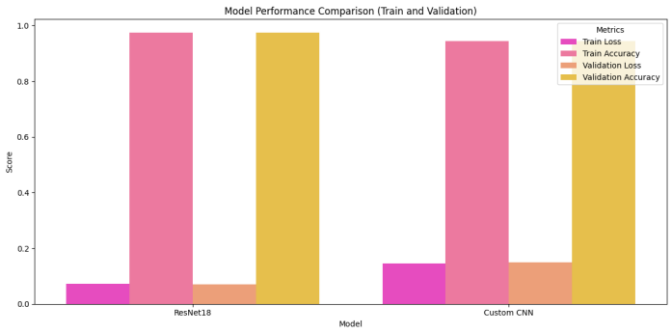

FIGURE 12

REFERENCES

Bibliography

[1] S. Bengesi, H. El-Sayed, M. K. Sarker, Y. Houkpati, J. Irungu, and T. Oladunni, "Advancements in Generative AI: A Comprehensive Review of GANs, GPT, Autoencoders, Diffusion Model, and Transformers," IEEE Access, vol. 12, pp. 69812–69837, 2024, doi: https://doi.org/10.1109/access.2024.3397775.

[2] S. Tripathi et al., "Recent advances and application of generative adversarial networks in drug discovery, development, and targeting," Artificial Intelligence in the Life Sciences, vol. 2, p. 100045, Dec. 2022, doi: https://doi.org/10.1016/j.ailsci.2022.100045.

[3] S. Nightingale and H. Farid, "AI-synthesized faces are indistinguishable from real faces and more trustworthy | Montreal AI Ethics Institute," Montreal AI Ethics Institute, Oct. 04, 2022. https://montrealethics.ai/ai-synthesized-faces-are-indistinguishable-from-real-faces-and-more-trustworthy/

[4] J. Askari, "Deepfakes and Synthetic Media: What are they and how are techUK members taking steps to tackle misinformation and fraud," www.techuk.org, Aug. 18, 2023. https://www.techuk.org/resource/synthetic-media-what-are-they-and-how-are-techuk-members-taking-steps-to-tackle-misinformation-and-fraud.html

[5] F. Abbas and Araz Taeihagh, "Unmasking deepfakes: A systematic review of deepfake detection and generation techniques using artificial intelligence," Expert systems with applications, vol. 252, pp. 124260–124260, May 2024, doi: https://doi.org/10.1016/j.eswa.2024.124260.

[6] R. Ramanaharan, D. B. Guruge, and J. I. Agbinya, "DeepFake video detection: Insights into model generalisation — A Systematic review," Data and Information Management, p. 100099, Mar. 2025, doi: https://doi.org/10.1016/j.dim.2025.100099.

[7] L. Romeo Luigi ·, F. Federico, D. Anxhelo, M. M. Raoul, and C. Luigi , "Faster Than Lies: Real-time Deepfake Detection using Binary Neural Networks," Paperswithcode.com, 2025. https://paperswithcode.com/paper/faster-than-lies-real-time-deepfake-detection (accessed May 16, 2025).

[8] P. Lorenz, R. Durall, and J. Keuper, "Papers with Code - Detecting Images Generated by Deep Diffusion Models using their Local Intrinsic Dimensionality," Paperswithcode.com, 2023. https://paperswithcode.com/paper/detecting-images-generated-by-deep-diffusion (accessed May 16, 2025).

[9] V. Vallabhaneni et al., "View of A Comprehensive Review of Deepfake Detection Pertaining to Images, Videos, Audio, and News using Deep Learning Techniques," Everant.org, 2025. https://everant.org/index.php/etj/article/view/1849/1352 (accessed May 16, 2025).

[10] O. Elharrouss et al., "ViTs as backbones: Leveraging vision transformers for feature extraction," Information Fusion, pp. 102951–102951, Jan. 2025, doi: https://doi.org/10.1016/j.inffus.2025.102951.

[11] J. J. Bird and A. Lotfi, "CIFAKE: Image Classification and Explainable Identification of AI-Generated Synthetic Images," IEEE Access, vol. 12, pp. 1–1, Jan. 2024, doi: https://doi.org/10.1109/access.2024.3356122.

[12] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," Apr. 2009. Available: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

[13] A. Gupta, P. Pawade, and R. Balakrishnan, "Deep Residual Network and Transfer Learning-based Person Re-Identification," Intelligent Systems with Applications, vol. 16, p. 200137, Nov. 2022, doi: https://doi.org/10.1016/j.iswa.2022.200137.

[14] D. Singh, "DLOA (Part-18)-ResNet CNN and Implementation," Medium, May 20, 2023. https://medium.com/@singhdewansh99/dlao-part-18-resnet-cnn-and-implementation-e3ce3d719807

[15] PyTorch Forums, "CrossEntropyLoss getting value > 1," PyTorch Forums, Sep. 11, 2023. https://discuss.pytorch.org/t/crossentropyloss-getting-value-1/188115 (accessed May 17, 2025).