Application of Neural Network Algorithms For Detecting Real and Fake Photos

Lorena Benavides Riano – PhD Student, Engineering Education
John Alexander Yepes – Master Student Data Analytics
Mississippi State University

Keywords

Real, Fake, Deepfake Detection, Neural Networks, Hybrid Models, StyleGAN, and Transfer Learning.

Abstract

This study compares deep learning models and hybrid deep learning machine learning models for deepfake detection using the StyleGAN-generated dataset. A total of six models were compared. Two convolutional neural networks (CNNs), Xception and DenseNet121, were trained first to classify the images as real or fake and to serve as feature extractors by removing the fully connected layer. The extracted deep features serve as inputs for the support vector machine (SVM) and Bernoulli Naïve Bayes (NB) models, creating four hybrid models from the CNN architectures. In the CNNs, strategies such as hyperparameter tuning through grid search, model checkpointing, early stopping, and reducing the learning rate upon reaching a plateau were implemented to optimize the models' performance. The results indicate that DenseNet121 achieved the highest performance among all models, with an accuracy of 99.33% and an AUC of 0,997. Hybrid models, particularly DenseNet-SVM, also demonstrated strong and balanced classification performance while consuming fewer computational resources and training faster. Although Naïve Bayes showed the lowest performance among all models, its results were still competitive, indicating that simpler machine learning classifiers can benefit from feature extraction and optimization for computational efficiency. The current study suggests the potential application of hybrid architectures to achieve high accuracy in complex classification tasks while optimizing computational resources.

1. Introduction

Detecting real and fake photos has become one of the most significant security concerns nowadays. With the proliferation of social media, communications, images, machine learning, and deep learning algorithms, there is an increase in fake and false information in the world [1]. The development of these methodologies is giant, and it is very common to see artificial images on social media about artists, singers, politicians, or social issues (war, pandemic, natural disasters), among others. Due to privacy concerns, citing problems with data collection, lack of age verification, and a reported data breach, some governments, such as Italy, prohibited using ChatGPT (OpenAI) as a control measure in March 2023.

However, efforts have increased in the last decades because of rapid technological growth. With the development of Deep Learning (DL) science, many techniques have been created to improve image forgery detection methods [2]. Most popular detection model base their analysis on classifying pixels into forged or pristine [3]. Others have created architectures that use a hierarchical-grained formulation to create models that learn to classify forgery attributes [4].

Some popular platforms can create images that look like real life or have a real appearance in seconds. These popular applications include Midjourney, DALL·E 2, Stable Diffusion, and DeepAi [5]. Among all the available developments, Generative adversarial networks (GANs) have drawn a lot of interest since they produce highly realistic images that can be used in advancements in many fields. Still, it represents a threat to recognizing identity fraud and security risks. As GANs image generators can learn quickly by being trained and improved, it is more difficult to identify whether they are real or fake. Identifying fake images of famous people can be easier, but what about non-famous people? So, how can we detect fakes from real photos? GANs are composed of two neural networks, a generator and a discriminator, in an adversarial process to learn and generate new data that resembles the training data. The generator network takes random noise as input and tries to produce data samples that are realistic and indistinguishable from the real data. On the other hand, the discriminator network tries to distinguish between the real and generated data, acting as a critic for the generator. Therefore, the main idea is that the generator is trying to fool the discriminator, and the discriminator is trying to identify fake data correctly [6].

One way to address this problem is to check the image in detail because sometimes the manipulated images show errors in the light and body parts (fingers, ears, light, among others). For end users, platforms, such as TinEye, help

us find the websites where the images have been published to ensure the context and realize if it is a fake image [7]. Another way is to check pixel by pixel to determine whether a pixel has been tampered with or not [8]. Since GAN-generated images are difficult to detect, in the current study, we aim to develop robust deep learning architectures and hybrid approaches that combined CNN based feature extraction with traditional machine learning classifiers such as Support Vector Machine (SVM) and Naive Bayes capable of classifying real or artificially generated images generated by StyleGAN, a type of GAN that excels at generating high-quality, realistic images, particularly human faces, by using a style-based generator that allows for intuitive control over various image attributes [9]. The study aims to compare these models in terms of performance metrics (precision, recall, F1-score, AUC), and computational efficiency, particularly in resource-constrained environments. The selected dataset includes human faces with diverse features, regardless of race, color, or age.

The structure of this study includes five sections. First, a general introduction to the importance of the research. Section 2 contains related works about data science techniques in detecting real vs fake human faces. Section 3 presents an explanation of the proposed methodology and dataset. Ultimately, Section 4 focuses on explaining the preprocessing steps implemented on the DL and ML models. It also talks about the tuning and improvements performed and the results obtained, highlighting the best model for this case study based on the model's evaluation. Likewise, it has a discussion and practical implications section. Finally, Section 5 concludes the study findings and suggestions for future work.

2. Literature Review

The generation of deepfakes refers to facial manipulation created using various methods, including Generative Adversarial Networks (GANs), Face2Face, and FaceSwap. Manipulations can be classified into four categories: entire face synthesis, attribute manipulation, expression swap, and identity swap. Entire face synthesis creates fake faces that don't exist in reality. Attribute manipulation focuses on editing specific attributes such as skin color or hair color. Expression swap substitutes the source image's facial expression with the target image's. Ultimately, the identity swap works similarly to the expression swap, but in this case, the whole face is replaced [10]. Since artificial intelligence has grown significantly in recent years, this study will focus on detecting images generated by GANs (Entire face synthesis) as the outcomes are highly realistic and extremely difficult to detect for human eyes. This literature review explores studies focused on creating and refining architectures that can achieve high accuracy.

A significant advancement in convolutional neural network (CNN) to identify deepfakes was the development of the Xception architecture, which incorporates depthwise separable convolutions, allowing for more efficient parameter usage and improved performance [11]. Saxena et al [12] implemented an XceptionNet model previously trained, validated, and tested by a Machine Learning process to apply this convolutional neural network (CNN) to classify deepfake images, obtaining an AUC score of 0.97. The process performed by A. R. Khan, M. S. Hossain, and M. A. Rahman [13] implemented a different ML approach with MobileNetv2 architecture and CNN by classifying 60K images (50K for training and 10K for testing) and obtained an accuracy of 95.48% in detecting false face images. EfficientNetV2S model [14] was used between authentic and counterfeit photos in order to social protection. The information was split into 80% (training) and 20% (testing), and the results showed an accuracy of 0.99.

Rossler and colleagues proposed a benchmark for facial manipulation by introducing the FaceForensic++ dataset. The authors proposed forgery detection as a binary classification problem in a dataset of 1,000 videos. The study used a domain-specific forgery detection pipeline for facial manipulations to crop the face region fed into a learned classification network and evaluate five architectures, which include CNN-based networks inspired by InceptionNet and XceptionNet. The best performance was achieved with a combination of domain-specific information and an XceptionNet classifier, which achieved an accuracy of 95.73% for high-quality and 81% for low-quality videos [15].

Hsu et al. [16] developed a deep-learning model to classify real and fake face images generated by GANs using contrastive loss in a two-step learning process that integrates a Siamese network with the DenseNet, to achieve the discriminative common fake feature (CFF) based on pairwise learning strategy and classifier learning. The dataset employed contains 20,299 face images of various celebrities with facial attributes such as hair color, expressions, and glasses. The results achieved an accuracy of 90.9%. However, the new generator can encounter problems when the new generator's outcomes significantly differ from those employed during the training phase.

A study conducted in public datasets such as FFHQ, 100k-Faces, DFFD, and Casia-WebFace implemented A deep learning technique used by Fisherface to apply a reduction of the dimension in the face space using Local Binary Pattern Histogram (FF-LBPH) and the DBN classifier. All the images were resized and noise-removed using a

Kalman filter. The model attained 98.82% and 97.82% in the CASIA-WebFace and DFFD datasets, respectively [17]. Chen et al. [18] address the problem of compressed images with unknown compression factors. Two-branch Convolutional Networks with Similarity and Classifier TCNSC were proposed for binary classification and similarity learning. The model was trained by robust feature learning on the FaceForensics++ dataset. The result showed that the model performs well on the FF++ dataset for low-quality scenes.

A study developed a hybrid of VGG16 and a convolutional neural network CNN to detect deepfake media through a novel deepfake predictor (DFP) applying transfer learning techniques. The dataset had 2,041 images containing real and attribute-manipulated pictures. The proposed architecture merged layers from VGG16 and the CNN and achieved 95% precision and 94% accuracy for deepfake detection [19]. Some other hybrid approaches for deepfake detection include combining deep learning and machine learning techniques. The CNN architecture is used for feature extraction, and the second is used for machine learning classifiers (SVM, KNN, DT). Masood et al. [20] applied CNN and used an SVM classifier to detect fake videos on the DFDC database released by Facebook, where 70% of the data was used for training and 30% for testing. In total, ten CNN models were applied, and the DenseNet-169 achieved the highest accuracy at 98%, while VGG-16 achieved a lower accuracy of 89%.

3. Methodology

This section describes the proposed models to tackle the challenge of detecting fake vs real faces. Figure 1. shows the different methodologies applied in our study and the different stages involved. The proposed analysis consists of three steps. i) Face preprocessing, ii) CNN models and Feature extraction, iii) Classification Models. The main idea is that the images are preprocessed to serve as inputs for CNN models that are assessed in two ways: fully connected and in a hybrid model where CNN models serve as feature extractors for later classification using machine learning techniques.

3.1. Preprocessing

The dataset is initially divided into training (60%), validation (20%), and testing (20%) sets. Then, the images are resized according to the models' requirements (299 x 299 x 3 pixels for XceptionNet, and 256 x 256 x 3 pixels for Densenet121). Standardization techniques such as scaling or normalization are applied to refine the inputs for the models, emphasizing facial features by eliminating background noise. The models were trained on the training set and then assessed on the validation set, which was categorized into fake or real images. Once training is complete, the generalization to unseen images is evaluated in the test set.

3.2. CNN models and Feature Extraction.

During this phase, pre-trained models XceptionNet and Densenet121 are used separately, and for the hybrid model, instead of fully training the CNN, they are used as feature extractors, and later, classification is performed using machine learning techniques. XceptionNet and Densenet121 were chosen due to their capacity to identify subtle deepfake anomalies using depthwise separable convolutions, being lightweight, and low computational cost. Additionally, since these models have been trained on big datasets such as ImageNet, they are more reliable for learning complex patterns.

- **XceptionNet:** is a deep NN that is especially beneficial for image recognition and learns complex features and patterns from input data. XceptionNet has separable convolutions that limit the number of model parameters, managing the overfitting issue. This model has an outstanding performance, which has a lower computational cost than other learning models due to the utilization of separable convolutions that reduce the number of training parameters [11]. The architecture of this NN is compounded by Depthwise Convolution, and it is obtained as follows [21]:

$$\hat{G}_{k,p,m} = \sum \hat{K}_{i,j,m} x F_{k+i-1,p+j-1,m} \quad (1)$$

  Where $\hat{G}$ describes the alternatives of the feature maps output produced by F as the input feature map and, $\hat{K}$ defines the depthwise convolution kernel.

- **Densenet121**: Dense Net refers to a densely connected convolutional network where each layer connects to every other layer. It has 120 convolutions and four average pooling layers. Instead of summing feature

maps, it concatenates them, reducing the number of parameters [22]. The network is organized into Dense Blocks (with constant feature map sizes but varying filters) and Transition Layers (which halve the number of channels between blocks). Mathematically, a layer in Dense Net can be represented as follows:

$$X_e = H_1([X_0, X_1, \dots, X_{e-1}]) \quad (2)$$

Where X_e is the output of layer l, H_1 are composite functions consisting of batch normalization, rectified linear unit (ReLU) activation, and convolutional operations, and $[X_0, X_1, \dots, X_{e-1}]$ denotes the concatenation of feature maps from all previous layers up to l [23].

3.3. Classification Models:

In this final stage, features extracted from the XceptionNet and Densenet121 are connected with the machine learning classifier: Support Vector Machine (SVM) and Naïve Bayes (NB). During the training phase, the model learns relevant features (like edges, shapes, and textures) from CNN architectures, which can be challenging for machine learning feature engineering, and then the classifiers take the extracted features and place them into the binary category (Fake or Real). SVM classifier is beneficial due to its capacity to generate hyperplanes to find optimal decision boundaries in high-dimensional feature space. Additionally, the SVM classifier is robust for overfitting, which makes it reliable for this study.

- **Support Vector Machine (SVM):** It is a linear classifier that distinguishes classes by generating multiple hyperplanes to separate the dataset. It determines the optimal by using the closest support vectors (data points) from each class. SVM includes a hyperparameter, C, which balances maximizing the margin and minimizing classification errors. Additionally, the Kernel hyperparameter allows SVM to operate in a higher-dimensional space, enabling it to handle non-linear decision boundaries [24].

  In this case, SVM is used as a classifier for the binary class (real or fake) since it can handle high dimensionality and overfitted data. The training data contains N feature vectors organized as: $(x^{(i)}, y^{(i)})$, i=1,.N, where $y^{(i)} \in \{1, -1\}$ shows the two classes [20]. SVM constructs a hyperplane that linearly divides the two classes as follows:

$$w^T.x^{(i)} + \beta \geq 1 \; if \; y^{(i)} = +1 \quad (3)$$

$$w^T.x^{(i)} + \beta < 1 \; if \; y^{(i)} = -1 \quad (4)$$

  Here, w represents the weight vector, and β denotes the bias. The goal is to maximize the margin between the two support vectors by minimizing the norm $||w||$. This optimization can be formulated as a quadratic programming problem.

$$\min || \, w \, ||, \text{such that } y^{(i)}(w^T.x^{(i)} + \beta) \geq 1 \quad (5)$$

  The binary classes real and fake can be established by using the discriminant function f(x) = sign (wT. x(i) + β) as follows:

$$\begin{cases} real, f(x^{(i)}) = +1, \\ fake, f(x^{(i)}) = -1 \end{cases} \quad (6)$$

- **Naïve Bayes (NB):** This classifier is based on the Bayes theorem, similar to linear models. It calculates the probability of a given set of images for the binary class and selects the output with the highest likelihood. NB is computationally efficient for large datasets and assumes independence of features. Three kinds of NB classifiers are implemented: Gaussian, Bernoulli, and Multinomial. The Gaussian variant is suitable for continuous data, while the Bernoulli model is designed for binary data. The Multinomial classifier, on the

other hand, is used for count-based data, where each feature represents the frequency of an occurrence [25]. The Bayes formula is described as follows:

$$P(C|X) = \frac{P(c)P(X|C)}{P(X)}$$

$$P(C|x_{1,}x_{2,\dots}x_n) = \frac{P(c)P(x_{1,}x_{2,\dots}x_n,|C)}{P(x_{1,}x_{2,\dots}x_n)} \quad (7)$$

Where: $X = (x_1,\ x_2, x_3, \dots, x_n)$ are the attributes, C: class, $P(C|X)$ the probability of event C given X has occurred, $P(X|C)$ the probability of event X given C has occurred, $P(C)$ probability of event C, and $P(X)$ probability of event X.

The following graphic summarizes the proposed CNN model's sequential stages and hybrid approach.

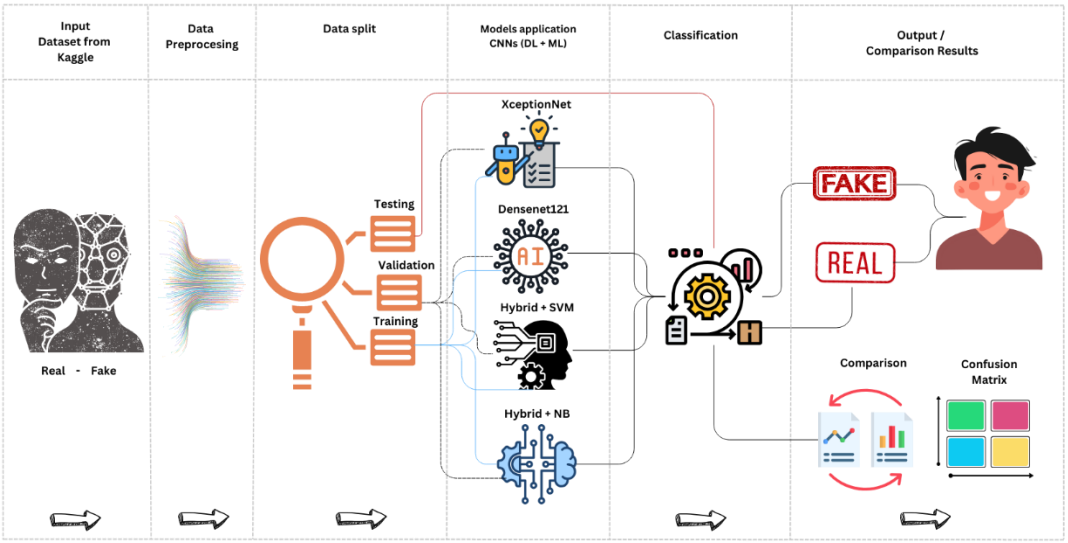

Figure 1: Flow methodology diagram

3.4. Data Description

As for the numbers, the dataset consists of 140,000 images with 70,000 Real faces from the Flickr dataset, as well as 70,000 fake faces sampled and generated by StyleGAN [26]. As the proportion of real and fake images is equal, we ensure that the architecture represents the two classes well and improves the dataset. Figure 2 shows sample images of the two classes in the dataset.

Real **Fake**

Figure 2: Example of images in the selected dataset [26].

4. Results and Discussion

4.1. Preprocessing Details

The full dataset was mounted on Google Drive to access stored data and save results. Then, the upload of the files and the preprocessing process, Xception and DenseNet models, were performed in Google Colab+ with a GPU (NVIDIA A100-SXM4-40GB), and the hybrid models were performed on a CPU. In this step, it was crucial to check the resources available to ensure appropriate computing capability.

The main libraries for CNN models were Keras and TensorFlow, while for hybrid models, the numpy and sklearn packages were used. It also implemented ImageDataGenerator, resizing and normalization techniques applied to images to obtain pixel values between 0 and 1 with a batch size of 64 and 10 epochs for training.

4.2. Tuning Xception and DenseNet architectures

Initially, a grid search was conducted on the full dataset, but with three epochs to efficiently identify suitable hyperparameters for the CNN architectures. Hyperparameters such as Learning Rate, Dropout, and Dense layer Units were varied to determine the best combinations to find the optimal performance. Once the best hyperparameter combination was identified from the previous search, the selected configuration was used to train the full dataset with at least 10 epochs.

Different callback approaches were employed during the training process to find the optimal model for the deepfake classification process and identify potential overfitting. The ModelCheckpoint callback periodically saved the model based on its lowest validation loss and weights in a file that was used later for feature extraction. The EarlyStopping callback monitored the validation loss and stopped the model when this metric was not improving after a defined number of epochs (patience=5), preventing unnecessary training beyond optimal convergence. Finally, the ReduceLROnPlateau callback adjusts the learning rate when this has stopped improving. This strategy involves monitoring the validation loss and reducing the learning rate by a 0.2 factor when no improvement is observed over three epochs, also referred to as patience. The combination of these callbacks aimed to facilitate further learning and mitigate the risk of overfitting by allowing the preservation of the model at an optimal point during training.

Once the DL models were finished, the features and labels were extracted. This new data needed to be scaled, and the dimensionality adjusted by applying PCA = 0.95 to preserve 95% of the variance and a standard scaler to help the ML model have similar scales, handle any possible outliers, and generalize better. Similarly to the DL model, a grid search was implemented in the SVM model to find the best hyperparameter and pick it for the best optimal performance.

There were some differences in the models' analysis, and they are shown in the following table:

Table 1: Details of the architectures

Feature/Step	***Xception***	***DenseNet121***
Architecture Type	Depthwise separable convolutions	Dense connectivity (connected outputs)
Pre-trained Weights	ImageNet	ImageNet
Input Image Size	299 x 299	256 x 256
Preprocessing	Xception preprocessing	Standard Normalization (1/255)
Batch Size	32	64
Gridsearch Epochs	3	3
Epochs for Training	10	10

Checkpointing	Weights based on validation loss	Weights based on validation loss
EarlyStopping	Patience=5	Patience=5
Learning Rate	Patience=3, Factor=0.2	Patience=3, Factor=0.2
Custom Layers	Dense(256), Dropout(0.2), BatchNormalization	Dense(448), Dropout(0.3), Global Avg. Pooling
Optimizer	Adam(lr=1e-5 tunned)	Adam(lr=1e-5 tunned)
Classification Output Layer	Dense(1) with sigmoid	Dense(1) with sigmoid
Fine-tuning/Transfer Learning	Yes (initially layers frozen)	Yes (initially layers frozen)
Performance Evaluation	Accuracy, ROC-AUC, Confusion Matrix	Accuracy, ROC-AUC, Confusion Matrix

4.3. Models' evaluations

Table 2 shows the accuracy and metrics of the evaluated models. The DenseNet model obtained the highest accuracy, with an accuracy of 99.33%, very low loss (0.0190), and strong classification scores. Although the Xception model did not perform as strongly as DenseNet, its results demonstrated consistent and balanced performance across all metrics.

On the other hand, the hybrid Xceptionet-SVM model improved over its base DL model, achieving 97.43% accuracy, and DenseNet121-SVM accuracy results were slightly lower, with 99.03%. Lastly, both Naive Bayes hybrids demonstrated solid performance with an accuracy of 97.05% and 96.46% for the hybrid of DenseNet and Xception, respectively. These results showed that Naive Bayes can maintain the performance of CNN models while reducing model complexity. Overall, the results suggest that SVM and NB can benefit from feature extraction by CNN models because the extracted deep features from the deep learning models without the classification head (base output layer) helped to generate a rich feature set for classical machine learning classifiers.

Table 2: Models' Metrics

Model	Accuracy	Loss	Precision	Recall	F1-Score	AUC
Xception	0.9722	0.0756	0.9723	0.9721	0.9721	0.9721
Xception -SVM	0.9743	--	0.9743	0.9743	0.9743	0.9963
Xception NB-Bernoulli	0.9646	--	0.9647	0.9647	0.9646	0.9925
DenseNet121	0.9933	0.0190	0.9875	0.9873	0.9873	0.997
DenseNet-SVM	0.9903	--	0.9904	0.9904	0.9903	0.9969
DenseNet NB-Bernoulli	0.9705	--	0.9721	0.9705	0.9704	0.9815

A milestone in this analysis result is the confusion matrix. The following image shows the confusion matrices for the six classification models. Overall, they all performed very well in classification and misclassification. Starting from the DL models, DensetNet correctly classifies 9.974 Fake and 9,772 Real faces, with a low misclassification (26 Fake, 228 Real). The Xception Model also performed well, correctly classifying 9.802 Fake and 9,641 Real faces, but with slightly higher errors for Real samples (359 misclassifications)

The results of the hybrid models confirm the improvement achieved. The best model is DenseNet-SVM, with an outstanding performance, correctly classifying 9,898 Fake and 9.909 Real faces, and a low misclassification (102 Fake, 91 Real). Although the Hybrid DenseNet-Bernoulli NB excels remarkably in identifying the Real class (9,997 correct, only three errors), it significantly underperforms in classifying Fake samples, misclassifying 588 instances. Meanwhile, the Xception-based models exhibit higher overall errors. In conclusion, the Hybrid DenseNet-SVM emerges as the optimal model due to its high accuracy and balanced predictive performance across both classes.

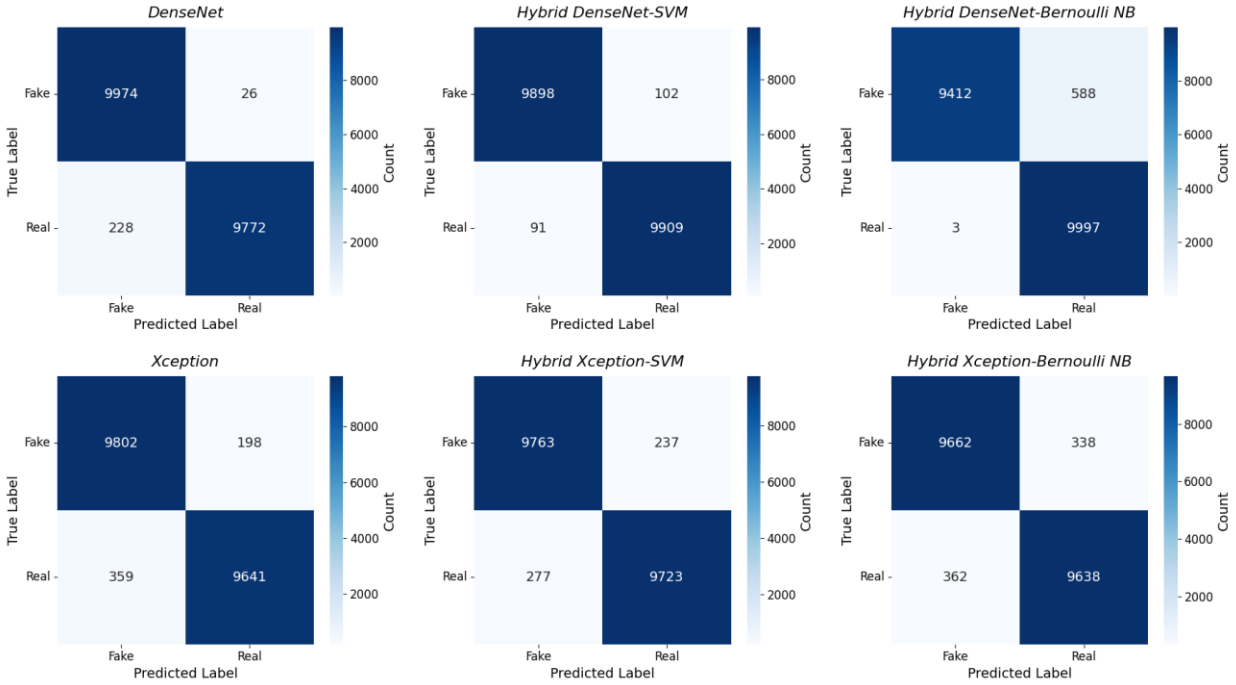

Figure 3: The confusion Matrix of DenseNet, Xception, and their hybrid models (SVM and Bayes).

4.4. Discussion

The integration of deep learning models with traditional machine learning classifiers differentiates feature extraction from the classification task. In this case, Xception and DenseNet121 are first used to extract high-level features from face images and then enter into SVM and Naive Bayes classifiers for the final predictions. CNNs models extract higher-level representations of images, reducing the need for manual feature engineering. However, their classification layers used fully connected layers contain an activation function, sigmoid in this case, which predicts the probability from 0 to 1 for each classification label [27]. The use of a fully connected layer may not always be optimal for generalization or computational efficiency. Hybrid models replace the final classification layer, potentially reduce model complexity, and leverage decision boundaries in order to address these shortcomings. In this study, it was observed that CNN-SVM models showed competitive performance and, in some cases, outperformed their baseline CNN on several metrics. Xception-SVM slightly outperformed the Xception model in all reported metrics, especially an improvement was observed in AUC from 97.21% to 99.63%. Dense-Net SVM model maintained a high-performance classification similar to the base model. In contrast, the Naive Bayes hybrids, while slightly lower in performance, still performed well for the classification task. Xception-NB reached 96.46% accuracy and 99.25% AUC, while DenseNet NB had 97.05% accuracy and 98.15% AUC.

These results showed that even simpler machine learning classifiers can maintain strong results when the best model is selected from CNN architectures. Naive Bayes classifiers tend to be faster in training and have slightly lower generalization performance than linear classifiers such as SVM. The advantages of selecting BernoulliNB in this case are that this distribution assumes binary data and that every feature of each class is not zero.

Additionally, tuning the models is simpler as it only has a single parameter that controls model complexity (alpha) and works well in large datasets [25].

Optimization strategies implemented in this training model process, such as tuning it through grid search and applying callback techniques with small epochs on the complete data, contributed to enriching and obtaining the results seen. Additionally, using EarlyStopping prevented unnecessary resource waste or misuse, unnecessary training epochs, and mitigated overfitting risk once the optimal performance was achieved. Saving the weights of the optimal set by ModelCheckpoint was another contribution to ensure reproducibility and facilitate a stable future extraction for the hybrid models. Likewise, the improvement obtained by applying the learning rate step enhanced the model convergence. When the validation loss was not improving, it helped the model learn even after some plateaus. Furthermore, applying an extra gridsearch in the ML model, scaling, dimension reduction by PCA, and testing on a small sample, reducing the resource waste, helped find the best way to make the hybrid model work. All these implementations demonstrated their effectiveness in the hybrid architecture, highlighting that careful training and strategic hyperparameter adjustment play critical roles in maximizing model generalization and efficiency.

4.5. Practical Implications

From the study perspective, one advantage of hybrid models lies in computational efficiency after feature extraction using a GPU; the final classification with SVM and NB was conducted on a CPU, obtaining accurate results in less time and with fewer computational resources. The rapid iteration allows for changing the classifiers without retraining the full CNN. As a result, the Hybrid models in image classification can be applied to a broad variety of applications from different fields, such as object recognition, including medical images. However, some shortcomings of these approaches are that CNN is frozen after feature extraction, and the ML classifiers are trained separately; the two models cannot learn and improve together like end-to-end deep learning models do. Additionally, they require more manual handling and data preprocessing steps compared to deep models. Tuning the hyperparameters, especially in SVM, might represent an additional challenge to achieve optimal performance compared to CNNs models, where optimization is more standardized. Finally, as each approach requires different libraries (TensorFlow and scikit-learn), it can introduce complexity to run the code on other Hardware accelerators.

5. Conclusions

The present study explored and compared six models for deepfake detection. Two CNNs models: Xception and DenseNet121, and four hybrid models that combined the baseline models with Support Vector Machine (SVM) and Naive Bayes (Bernoulli). The results demonstrated that CNNs models are powerful for classification tasks, with DenseNet121 reaching an accuracy of 99.33% and an AUC of 0.997, outperforming all other models. CNN-SVM hybrid models showed close performance with their baseline, with 99.03% and 97.43% for DenseNet and Xception, respectively. Although the Naive Bayes hybrid model did not outperform base models, it suggests that ML simpler classification models can be effective in detecting realistic StyleGAN-generated images, suggesting that hybrid approaches can be effective alternatives, particularly when computational resources are limited.

The advantages of hybrid approaches are that once the heavy feature extraction is conducted with CNN, simpler machine learning classifiers can be trained quickly on CPUs, reducing training time and increasing computational efficiency. However, some of the limitations of these models are more complex coding with careful tuning of hyperparameters and separate training stages that can make their implementation difficult. In conclusion, this study confirms the power of convolutional neural networks for GAN-generated image deepfake detection and shows that hybrid methods can be an efficient alternative when computational resources are limited.

6. References

[1] M. Kubanek, K. Bartłomiejczyk, and J. Bobulski, "Detection of artificial images and changes in real images using convolutional neural networks," presented at the Computational Intelligence in Security for Information Systems Conference, Springer, 2019, pp. 197–207.

[2] M. Zanardelli, F. Guerrini, R. Leonardi, and N. Adami, "Image forgery detection: a survey of recent deep-learning approaches," *Multimed. Tools Appl.*, vol. 82, no. 12, pp. 17521–17566, 2023.

[3] H. Wu, Y. Chen, and J. Zhou, "Rethinking Image Forgery Detection via Contrastive Learning and Unsupervised Clustering." 2023. [Online]. Available: https://arxiv.org/abs/2308.09307

[4] X. Guo, X. Liu, Z. Ren, S. Grosz, I. Masi, and X. Liu, "Hierarchical Fine-Grained Image Forgery Detection and Localization." 2023. [Online]. Available: https://arxiv.org/abs/2303.17111

[5] Y. Chen, N. Akhtar, N. A. H. Haldar, and A. Mian, "On quantifying and improving realism of images generated with diffusion," *ArXiv Prepr. ArXiv230914756*, 2023.

[6] 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, 2020.

[7] S. S. Thomas and R. K. Prakash, "Image De-Duplication by using Tin Eye Match Service Engine in Cloud Computing," *-Manag. J. Cloud Comput.*, vol. 8, no. 2, p. 5, 2021.

[8] X. Liu, Y. Liu, J. Chen, and X. Liu, "PSCC-Net: Progressive Spatio-Channel Correlation Network for Image Manipulation Detection and Localization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 11, pp. 7505–7517, 2022, doi: 10.1109/TCSVT.2022.3189545.

[9] T. Karras, S. Laine, and T. Aila, "A Style-Based Generator Architecture for Generative Adversarial Networks." 2019. [Online]. Available: https://arxiv.org/abs/1812.04948

[10] Q. Jiang *et al.*, "Robust manipulated media localization and detection based on high frequency and texture features," *Discov. Comput.*, vol. 28, no. 1, p. 10, Feb. 2025, doi: 10.1007/s10791-025-09500-w.

[11] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions." 2017. [Online]. Available: https://arxiv.org/abs/1610.02357

[12] A. Saxena *et al.*, "Detecting Deepfakes: A Novel Framework Employing XceptionNet-Based Convolutional Neural Networks.," *Trait. Signal*, vol. 40, no. 3, 2023.

[13] O. Akram, A. Mohamed, H. Magdy, M. M. Abdellatif, and S. Abdelghafar, "Comparative Analysis of Custom CNN Architecture and MobileNet for Deepfake Image Detection," in *Proceedings of the 11th International Conference on Advanced Intelligent Systems and Informatics (AISI 2025)*, A. E. Hassanien, R. Y. Rizk, A. Darwish, M. T. R. Alshurideh, V. Snášel, and M. F. Tolba, Eds., Cham: Springer Nature Switzerland, 2025, pp. 58–68. doi: 10.1007/978-3-031-81308-5_6.

[14] V. Khullar, R. Ahuja, V. Solanki, and A. Chaudhary, "Real Fake Image Classification using Explainable EfficientNetV2S: A Comparative Analysis," in *2024 International Conference on Electrical Electronics and Computing Technologies (ICEECT)*, IEEE, 2024, pp. 1–5.

[15] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Niessner, "FaceForensics++: Learning to Detect Manipulated Facial Images," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea (South): IEEE, Oct. 2019, pp. 1–11. doi: 10.1109/ICCV.2019.00009.

[16] J. Jheelan and S. Pudaruth, "Using Deep Learning to Identify Deepfakes Created Using Generative Adversarial Networks," *Computers*, vol. 14, no. 2, Art. no. 2, Feb. 2025, doi: 10.3390/computers14020060.

[17] S. St *et al.*, "Deep learning model for deep fake face recognition and detection," *PeerJ Comput. Sci.*, vol. 8, p. e881, Feb. 2022, doi: 10.7717/peerj-cs.881.

[18] P. Chen, M. Xu, and X. Wang, "Detecting Compressed Deepfake Images Using Two-Branch Convolutional Networks with Similarity and Classifier," *Symmetry*, vol. 14, no. 12, Art. no. 12, Dec. 2022, doi: 10.3390/sym14122691.

[19] A. Raza, K. Munir, and M. Almutairi, “A Novel Deep Learning Approach for Deepfake Image Detection,” *Appl. Sci.*, vol. 12, no. 19, Art. no. 19, Jan. 2022, doi: 10.3390/app12199820.

[20] M. Masood, M. Nawaz, A. Javed, T. Nazir, A. Mehmood, and R. Mahum, “Classification of Deepfake Videos Using Pre-trained Convolutional Neural Networks,” in *2021 International Conference on Digital Futures and Transformative Technologies (ICoDT2)*, May 2021, pp. 1–6. doi: 10.1109/ICoDT252288.2021.9441519.

[21] X. Lu and Y. A. Firoozeh Abolhasani Zadeh, “Deep Learning-Based Classification for Melanoma Detection Using XceptionNet,” *J. Healthc. Eng.*, vol. 2022, no. 1, p. 2196096, 2022, doi: 10.1155/2022/2196096.

[22] “Densenet121-DNN-Based Hybrid Approach for Advertisement Classification and User Identification,” *Int. J. Intell. Eng. Syst.*, vol. 16, no. 3, pp. 162–174, Jun. 2023, doi: 10.22266/ijies2023.0630.13.

[23] M. Yousufi, R. Damaševičius, and R. Maskeliūnas, “Multimodal Fusion of EEG and Audio Spectrogram for Major Depressive Disorder Recognition Using Modified DenseNet121,” *Brain Sci.*, vol. 14, no. 10, Art. no. 10, Oct. 2024, doi: 10.3390/brainsci14101018.

[24] D. Isa, L. H. Lee, V. P. Kallimani, and R. RajKumar, “Text Document Preprocessing with the Bayes Formula for Classification Using the Support Vector Machine,” *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 9, pp. 1264–1272, Sep. 2008, doi: 10.1109/TKDE.2008.76.

[25] A. C. Muller and Guido, Sarah, “Introduction to Machine Learning with Python”.

[26] “140k Real and Fake Faces.” Accessed: Mar. 24, 2025. [Online]. Available: https://www.kaggle.com/datasets/xhlulu/140k-real-and-fake-faces

[27] “ML Practicum: Image Classification | Machine Learning,” Google for Developers. Accessed: May 05, 2025. [Online]. Available: https://developers.google.com/machine-learning/practica/image-classification/convolutional-neural-networks