Multimedia Tools and Applications
https://doi.org/10.1007/s11042-025-20877-0

A feature selection-aided deep learning based deepfake video detection method

Sk Mohiuddin[1] · Ayush Roy[2] · Saptarshi Pani[2] · Samir Malakar[3] · Ram Sarkar[4]

Received: 18 September 2024 / Revised: 27 March 2025 / Accepted: 21 April 2025

Abstract
The rapid advancements in deepfake technology are imposing significant challenges in detecting manipulated media contents. In this work, we have introduced a deepfake detection method that utilizes three pre-trained convolutional neural network models, namely DenseNet-121, EfficientNet-B0, and Xception to extract deep learning features from preprocessed extracted video frames. Next, these features are stacked to generate diverse feature representations, and thus end up with high-dimensional features. After that, a feature selection strategy is employed on the combined features to eliminate redundant and less effective features. The selected features are then fed into the K-nearest neighbors classifier to decide whether a questioned video is a fake video or not. Two widely used deepfake video datasets, namely FaceForensics++ and Celeb-DF (V2) are used to evaluate the performance of the proposed technique following two experimental setups: intra- and inter-datasets. The results show that the model achieves area under curve scores of 98.50% (85.70%) and 97.70% (79.00%) in the intra-dataset (inter-dataset) setups while trained on the FaceForensics++ and Celeb-DF (V2) datasets, respectively. The performance of the proposed model is comparable to that of state-of-the-art methods. Source code of the proposed method can be found at: here.

Keywords Deepfake detection · Feature selection · Stacking · KNN · Deep learning · Feature ranking

✉ Sk Mohiuddin
myselfmohiuddin@gmail.com

Ayush Roy
aroy80321@gmail.com

Saptarshi Pani
pani.saptarshi@gmail.com

Samir Malakar
s.malakar@uit.no

Ram Sarkar
raamsarkar@gmail.com

1 Department of Computer Science, Asutosh College, Kolkata 700026, India

2 Department of Electrical Engineering, Jadavpur University, Kolkata 700032, India

3 Department of Computer Science, UiT The Arctic University of Norway, Tromsø 9019, Norway

4 Department of Computer Science and Engineering, Jadavpur University, Kolkata 700032, India

Published online: 30 April 2025

1 Introduction

Deepfake detection refers to the process of identifying and determining whether a media content is authentic or generated with the help of generative artificial intelligence (AI)-based technology like generative adversarial networks (GANs), variational autoencoder (VAE), and diffusion model. The technology to create such malicious contents is known as deepfake technology. It can generate highly realistic and deceptive visual contents, which are hard to distinguish from their genuine counterparts with naked eye. Generative AI techniques are trained on voluminous real media content like images, videos, speech, and texts to learn the underlying patterns, features, and characteristics. Once trained, such techniques can be used to generate new contents, which have potential misuse for malicious purposes, such as spreading misinformation, creating fake news, or defaming individuals.

The recent advancements in deepfake technology have made the deepfake detection a challenging task. Moreover, its evolving nature (e.g., the emergence of more sophisticated generative models like GANs and diffusion models) is throwing new challenges to the research community in every passing day [1, 2]. Therefore, deepfake detection has become an important research and development area to counteract the negative implications of this technology. Before we dive more into it, it is important to mention that not all deepfake applications are malicious or harmful. Deepfake technologies also have some positive applications, such as in the entertainment industry [3], creative arts [4, 5], advertisement industry [3], and synthetic data creation [6]. Historically, the term "deepfake" was introduced for malicious media content like image and videos [6–8] and later this term has been adapted to other media contents like texts, speech, audio-video, etc. [9–12]. However, in this work, we have used the term "deepfake" only to mean AI generated malicious image and video contents.

Researchers are continuously exploring new algorithms, models, and approaches to improve detection capabilities for being ahead of evolving deepfake techniques [7, 13, 14]. Out of different approaches, stacking of multiple deep learning features [15–18] is a recent trend in deepfake detection. In this approach, diversified convolutional neural network (CNN) models are used to extract features first and then these extracted features are combined to form final features. Such approaches led to enhanced performances because each network can specialize in extracting different features or capturing specific patterns from the input data [19]. To be more specific, each of the stacked CNNs can learn and represent different levels of abstraction, resulting in more comprehensive feature representations.

However, the process of deep learning features' stacking returns high dimensional features that in turn increases the computational cost during classification tasks, specifically model training. Moreover, the CNN models used for feature extraction have some common characteristics and thus, are entitled to generate noisy, redundant, and irrelevant features when combined [20, 21]. To get rid of this limitation, a feature selection [20, 22–24] strategy should be employed on the top of the combined features. By eliminating redundant or irrelevant components and focusing on the crucial elements of the feature vector, the model can operate more efficiently, leading to faster processing and reduced ambiguity. Research finds several kinds of feature selection strategies, and filter-based methods [21, 22, 25] are one of those having less computation cost. In addition to that feature selection helps mitigate overfitting issues by reducing the dimensionality of the feature space. By selecting a subset of the most important features, the model becomes less prone to capturing noise or irrelevant information, resulting in better generalization and reduced overfitting.

By comprehensively examining deepfake technology, its applications, societal impact, and research challenges, this paper aims to design a deepfake detection method using a computationally less expensive process by extracting features from multiple diverse CNNs and then

stacking them. A novel feature selection strategy is then used to avoid over-fitting, reduce computation cost, and better generalization purposes. After the feature selection process, pass them into the K-nearest neighbors (K-NN) classifier to classify real and fake videos.

1.1 Key contributions

In a nutshell, the key contributions of the present work are as follows:

- The method fuses features from three pre-trained CNNs to diverse set of features for deepfake detection.
- A feature selection strategy is used to reduce the feature dimension by eliminating redundant and non-complimentary features.
- Instead of traditional deep learning classifiers, the proposed approach utilizes a KNN classifier, thus reducing computational resources.
- An extensive evaluation on two standard datasets using intra-dataset and inter-dataset setups, shows notable performances and generalizability compared to many state-of-the-art methods.

1.2 Organization of the article

The sections of this article are organized as follows. Section 2 presents an overview of prior research. Our proposed method is then elaborated in Section 3, while Section 4 is dedicated to presenting and discussing the obtained results. Finally, the paper is concluded in Section 5.

2 Related work

As a result of the advancements in technology and the development of methods specifically designed to counteract detection, the process of identifying and detecting deepfakes becomes increasingly complex and challenging. These advancements in deepfake generation and manipulation techniques aim to make the fabricated content appear more authentic and realistic, thereby evading traditional detection methods. By utilizing auto-encoders trained on real face images, Neves et al. [26] successfully remove traces of GAN fingerprints from synthetic images, making it harder to distinguish them from genuine content. Similarly, Liu et al. [27] tried to eliminate any noticeable signs of manipulation, making the deepfake blend seamlessly with authentic imagery. This pursuit of enhancing the authenticity of fabricated images further complicates the detection process. The increased intricacy in deepfake detection arises from the fact that these advanced techniques are specifically engineered to bypass existing detection mechanisms. As a result, traditional approaches that rely on identifying telltale signs of manipulation may no longer be as effective.

Face-swapping techniques, such as deepfake, modify the facial region to blend seamlessly with the surrounding environment while leaving the surroundings unaffected. In pursuit of detecting face swapping, Nirkin et al. [28] utilized disparities between these two regions, namely the faces and their context. They measured inconsistencies at the feature level of the source image and established a threshold to differentiate between genuine and fake faces in relation to their surroundings, as explored by Zhao et al. [29]. On the other hand, Nguyen and Derakhshani [30] demonstrated the effectiveness of matching the eyebrow area as a countermeasure against deepfake manipulation. The constant evolution and the emergence

of new techniques and algorithms for creating highly realistic and sophisticated deepfakes underscore the importance of ensuring robust generalization capability in deepfake detection systems. To address this challenge, Wu et al. [31] proposed a CNN model based on domain-invariant representation learning. In another work, Yan et al. [32] introduced a method that identifies common features added by the GAN networks to enhance the generalization ability of the detection system. Moreover, to focus on certain areas of the face, attention-based approaches [33, 34] have also been employed to assess the authenticity of facial manipulations. By incorporating attention-based approaches, the detection process is aided in reducing the search space for identifying manipulated artifacts. These approaches allow the model to focus on specific regions or features of the face that are more likely to exhibit signs of manipulation.

The previous approaches aim to identify spatial artifacts or extract consistent temporal patterns to detect deepfake videos that might fail to capture the temporal inconsistencies. As a result, a few researchers [35, 36] tried to capture temporal inconsistencies by designing spato-temporal techniques. For example, Pang et al. [35] proposed a method that addresses spatial-temporal inconsistencies in deepfake videos by dividing the input video into multiple groups and capturing short-term and long-term spatial-temporal inconsistencies in facial motion. Zhao et al. [37] improved upon the limitations of spatial-temporal models using transformers in their research, enabling the capture of both spatial artifacts and temporal inconsistencies for robust deepfake detection. Yang et al. [36] employed a spatiotemporal attention module to learn attention features of different facial regions through graph classification, aiming to learn local relationships by eliminating redundant information. While existing models performed well when the manipulation artifacts are known, they struggle when encountering unknown artifacts generated by different models.

To counter such problems, Wang et al. [38] addressed the issue by analyzing the mouth region and considering both global (entire face) and local (mouth region) regions to enhance the generalization ability. Transformers, commonly used in natural language processing, also exhibit promising results in improving the generalization ability of deepfake detection, as demonstrated by Wang et al. [39] who incorporate transformer models to capture essential image features at both local and global levels. Similarly, Heo et al. [40] utilized an efficient vision transformer model for deepfake detection, extracting both local and global features. In another work, Wang et al. [41] designed a two-branch structural network that integrates a cascaded multi-self-attention mechanism in parallel with EfficientNet-B4, while Reis and Ribeiro [42] proposed a threshold classifier using similarity scores from a deep CNN trained for facial recognition. By comparing faces from questioned videos to references, it classify the videos as authentic or fake based on the highest similarity score. Tian et al. [43] introduced a channel attention module that combines local and global contexts to address potential semantic inconsistencies in local artifacts and global features, aiming to overcome over-fitting problems.

The majority of research efforts have primarily focused on two areas: the development of CNN architectures, and the reduction of synthesis artifacts in fake images and videos. However, there is potential for improvement by directing attention towards feature fusion and reduction techniques that enable models to train more efficiently without sacrificing performance. For example, Essa [17] proposed a feature fusion-based deepfake detection method using three Vision Transformers: DaViT, iFormer, and GPViT. An MLP-Mixer integrated their strengths, enhancing detection by combining local-global contexts, frequency spectrum broadening, and high-resolution feature retention. In another work, Zhang et al. [44] focused on improving the cross-database performance by leveraging image noise to reveal

forgery traces. Their dual feature fusion and local enhancement attention modules enhanced feature representation, boosting discrimination ability. Wang et al. [18] tackled performance degradation in compressed images. Their method improved forgery detection by utilizing spatial-frequency fusion and multi-knowledge distillation with attention-based guidance for enhanced classification. Zhu et al. [45] detected abnormality by focusing on the inconsistency of illumination between video frames. Mohiuddin et al. [46] pursued this avenue by examining feature-level analysis, combining both handcrafted and deep features for deepfake detection. In their study, they employed a novel feature reduction mechanism to select near-optimal features from the deep feature set, thereby enhancing the overall detection performance. Overall, these studies highlight the ongoing efforts to detect and combat deepfakes, acknowledging the need for innovative approaches that can adapt to the ever-evolving landscape of deepfake generation and manipulation techniques.

3 Present work

The proposed deep fake detection method is empowered with a feature selection technique employed on top of stacked deep learning features as shown in Fig. 1. Initially, three pretrained CNN models DenseNet-121 [47], EfficientNet-B0 [48], and Xception [49] trained on the ImageNet dataset [50] are fine-tuned on present data, and utilized to extract features from the cropped face images extracted from the videos. These extracted features from CNN models are then stacked to have a single feature vector. These stacked features are subsequently fed into a novel feature selection algorithm to mask redundant and less important features while retaining the relevant ones i.e., to obtain a near-optimal feature subset. The next part, called inclusion-exclusion based feature selection approach, uses the KNN classifier to rank the candidate solutions, which is also used as the classifier in our classification process. Different sub-parts of the proposed deepfakes detection technique are described in the following subsections.

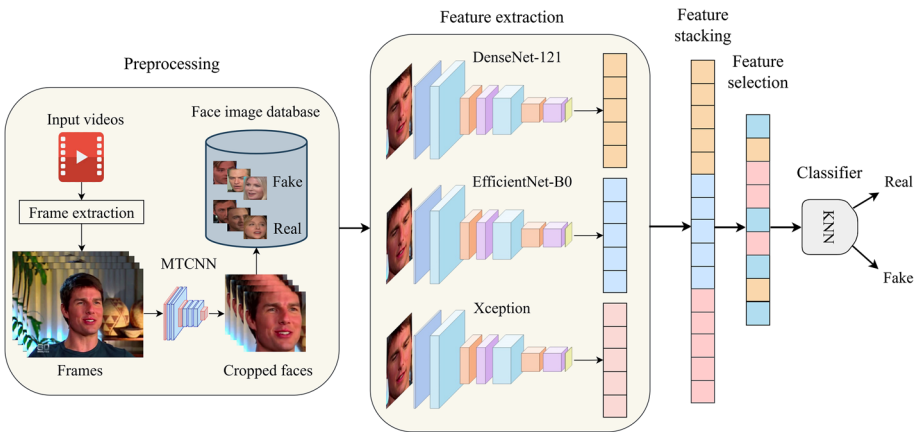


Fig. 1 Illustration of the overall workflow of the proposed method. The features are extracted using DenseNet-121, EfficientNet-B0, and Xception trained on the ImageNet dataset. These features are stacked together and passed on to the proposed feature selection strategy. Finally, selected optimal features are fed to the KNN classifier for predicting whether a questioned video is fake or real

3.1 Video processing

Here, deepfakes videos are identified by generating cropped face images from frames in a video. A multi-step process that utilizes Multi-task Cascaded Convolutional Networks (MTCNN) is used for this purpose. Initially, the videos undergo preprocessing to extract frames at regular intervals. Next, MTCNN is employed to perform face detection, facial landmark localization, and bounding box regression. By isolating the face region, we reduce the impact of backgrounds and irrelevant content, enabling a more focused analysis. This initial step allows us to determine the position and size of the faces within the frames. Subsequently, face scaling is applied to standardize the face images, thereby enhancing the accuracy of feature extraction. Finally, the faces are cropped based on the bounding box information obtained from MTCNN. In this particular case, all face images are scaled to a resolution of 299 $\times$ 299 pixels. The authenticity of a questioned video is evaluated by examining the retrieved face images from the video. Figure 2 depicts the process have followed to generate cropped faces for our experiment.

3.2 Feature extraction using deep learners

We have utilized three different CNN models, namely DenseNet-121 [47], EfficientNet-BO [48], and Xception [49]. These models are fine-tuned on the present datasets before using as deep-learning feature extractors. To obtain the flattened feature maps, we applied Global Average Pooling (GAP) to the top of the last layer's feature maps generated from CNN models. For the DenseNet-121 model, the dimension of the last layer's feature map is $7 \times 7 \times 1024$. This dimension reduces to 1024 after applying GAP. Similarly, for the EfficientNet-BO model, the dimension of the feature map is $7 \times 7 \times 1280$, resulting in 1280 deep learning features. Lastly, the Xception model produces feature maps with dimensions of $10 \times 10 \times 2048$, which are flattened to a 2048-dimensional feature vector. By applying global average pooling, we obtain a compact representation of the feature maps, facilitating subsequent feature combination and analysis for our deep fake detection methodology [51]. By leveraging the strengths of these diverse CNN models and combining their feature representations, we aim to capture better features that can potentially improve the overall performance.

3.2.1 DenseNet-121

Huang et al. [47] proposed DenseNets, a densely connected CNN architecture, which has demonstrated superior performance compared to traditional CNNs and residual networks in various applications, including deep fake detection [52, 53]. Unlike traditional CNNs,

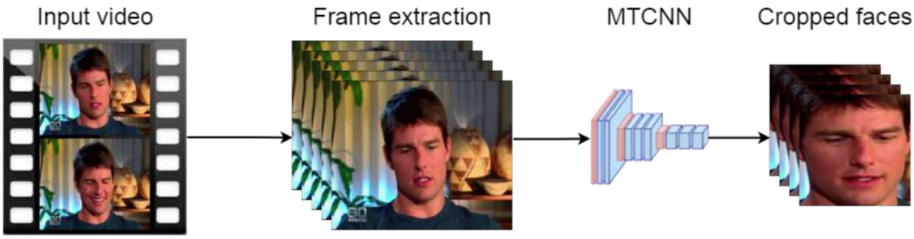


Fig. 2 Flow diagram of cropped face image generation for our experiments

DenseNets introduce dense connections between layers, where each layer receives inputs from all preceding layers within the network. This dense connectivity results in a rich information flow and addresses the vanishing gradient problem, leading to improved learning. The DenseNet-121 architecture consists of multiple convolutional layers, average pooling layers, and a fully connected layer. Each layer receives inputs from all previous layers, promoting extensive feature reuse and knowledge sharing among layers.

The composite function within each layer, denoted as H_n, comprises batch normalization, rectified linear unit (ReLU) activation, and convolution operations. Dense blocks are utilized to maintain constant feature map dimensions while adjusting the number of filters, and transition layers are introduced to reduce the number of channels and manage feature map dimensions. Additionally, a growth rate parameter, known as K_n, controls the number of added features as the network progresses through each dense layer. To enhance computational efficiency, a bottleneck layer with a 1×1 convolution can be incorporated before each 3×3 convolution layer in DenseNet-121. This bottleneck layer reduces the number of parameters while improving efficiency. A visual representation of the DenseNet-121 architecture can be found in Fig. 3.

3.2.2 EfficientNet-B0

EfficientNet, introduced by Tan and Le [48], is a family of CNN architectures known for their state-of-the-art performance, computational efficiency, and compact model size. It incorporates compound scaling, which simultaneously scales the depth, width, and resolution of the network for an optimal trade-off between efficiency and representational power. EfficientNet introduces the Mobile Inverted Residual Bottleneck (MBConv) block, initially proposed in MobileNetV2 [54], as a crucial building block. The MBConv block combines depth-wise separable convolutions with inverted residual connections, making it a key component of EfficientNet. The squeeze-and-excitation (SE) operation in the EfficientNet calibrates the channel-wise feature responses. It learns a channel-wise attention vector through a fully connected layer and scales the input feature maps adaptively. The success of the EfficientNet lies in its utilization of compound scaling, the incorporation of the MBConv block, and the application of the SE operation. These techniques enable EfficientNet to achieve remarkable performance, computational efficiency, and model compactness, establishing it as a popular choice for various computer vision tasks [48, 55]. EfficientNet-BO architecture, depicted in Fig. 4, provides a block diagram overview of the network structure.

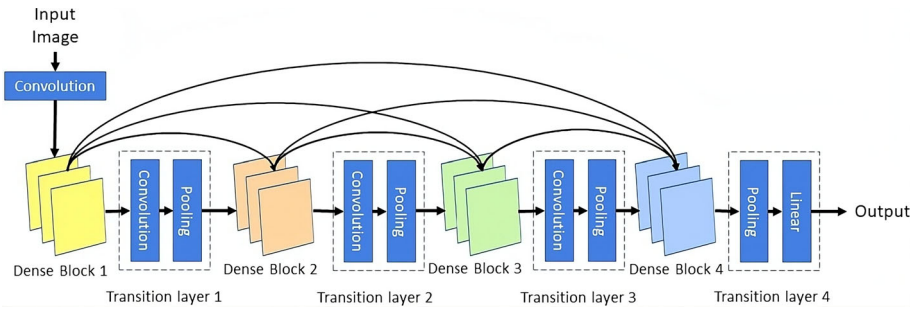


Fig. 3 Block diagram of the DenseNet-121 architecture

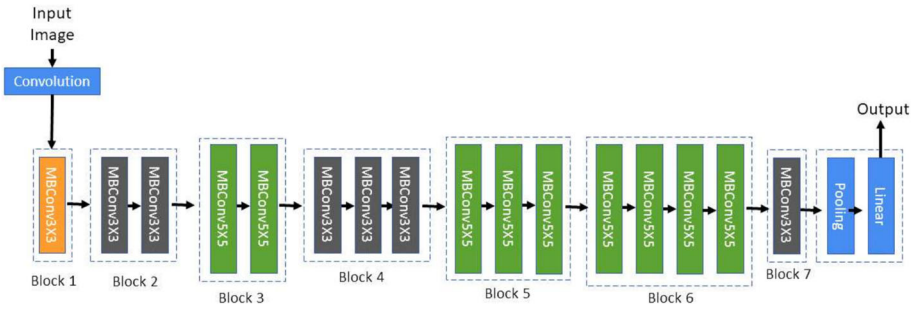


Fig. 4 Block diagram of the EfficientNet-BO architecture

3.2.3 Xception

Xception is a CNN architecture known for its efficient and effective feature extraction using depth-wise separable convolutions and residual connections. It aims to improve the efficiency and effectiveness of convolutional neural networks [49]. By employing depth-wise separable convolutions, Xception reduces computational complexity while preserving expressive power. The depth-wise convolution applies separate filters to each input channel, followed by a point-wise convolution that mixes and transforms the output channels. This architectural choice enables efficient computation by reducing the number of parameters. Additionally, Xception incorporates residual connections, similar to other residual-based architectures, to facilitate gradient flow and capture residual information. These connections aid in the learning of complex features and promote better training of deep networks demonstrated in the case of deep fakes [56, 57]. A visual representation of the Xception architecture is shown in Fig. 5, providing a block diagram overview of the network's structure.

3.3 Feature stacking

The flattened features extracted from the CNN models, with dimensions of 1024, 1280, and 2048 for DenseNet-121, EfficientNet-B0, and XceptionNet respectively, are stacked to form a 1D feature vector consisting of 4352 elements. The resulting features are then fed into the feature selection algorithm, which selectively identifies and retains the most informative and relevant features while masking the less significant ones. The feature selection algorithm aims to enhance the discriminative power of the features by focusing on the most discriminating features and filtering out any noisy or redundant ones [59, 60].

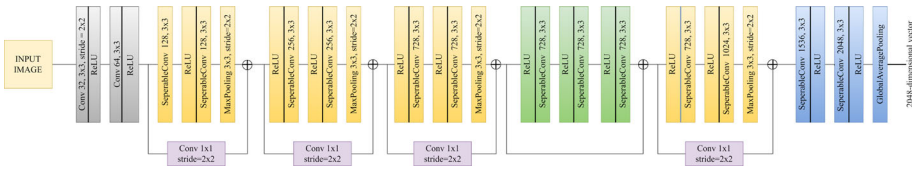


Fig. 5 Block diagram of the Xception architecture [58]

3.4 Feature selection

A novel feature selection technique is developed here. In the first part, features are ranked by considering feature relevance scores returned from three filter methods: ReliefF, MI, and mRMR. Next, a certain percentage of top-ranked features are fed to the next part, a novel feature selection technique, to optimize it further. The second part is a kind of single solution-based method [52] or local search employed in the work [61] on the top of GA that randomly includes some non-included features into the candidate solution and excludes a few from the previously selected ones. In a nutshell, the newly proposed feature selection component tries to enhance the selected features from the filter technique with a low computational cost and also verifies the effectiveness of selected features from the filter counterpart. An illustration of the feature selection process is shown in Fig. 6.

3.4.1 ReliefF

The ReliefF algorithm is a feature selection method that assesses the importance of features by considering the weights assigned to them based on their relevance and redundancy with the target variable. It is particularly useful for handling noisy and overlapping features. ReliefF measures the quality of features by estimating how well they distinguish between instances of the same and different classes. It works by iteratively updating feature weights based on the nearest neighbors of each instance, taking into account both the target variable and other features.

We initialize the ReliefF vector to zeros. We find the k nearest neighbors for each instance in the dataset based on a distance metric. Then, for each feature denoted as X_i, we compute the difference between the feature value of the current instance (x_i) and the corresponding feature values of the nearest hit instance (x_i^h) and the nearest miss instance (x_i^m). The difference is added or subtracted based on whether the instances belong to the same or different classes. By accumulating these differences, ReliefF updates the scores. Finally, ReliefF scores are

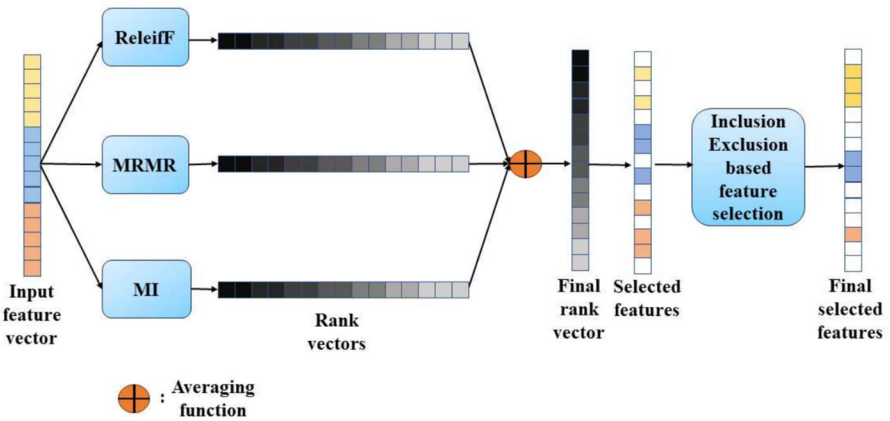


Fig. 6 Overview of the feature selection process. The feature vector obtained by stacking the features of the deep learners is fed to the three filter-based feature selectors: ReliefF, mRMR, and MI. The ranking scores obtained from these three selectors are averaged, producing a final ranking vector. According to the final ranking vector, the inclusion-exclusion-based feature selection (see Algorithm 4) is performed to get the final optimal features

normalized to ensure they fall within a specific range, providing a measure of importance for each feature. The entire process is illustrated in Algorithm 1.

Algorithm 1 ReliefF Feature Selection Algorithm.

Input: Data $X_{m\times n}$, and target variables $Y_{m\times 1}$ with y_i is related to feature vector X_i, $i = 1, 2, \ldots, m$. Here, m and n represent the number of samples in a dataset and the feature dimension respectively.
Output: Array $ReliefF^{scores}_{1\times n}$ where $ReliefF^{scores}_{j}$, $(j = 1, 2, \ldots, n)$ represents importance of i^{th} feature i.e., $X_{(1:m)j}$.

Initialize: $ReliefF^{scores}_{ij} \leftarrow 0$, $\forall i, j$
for each instance X_i with class label y_i in the dataset **do**
  Find the k nearest neighbors of X_i (say, X'_i) based on a distance metric
  Initialize: NearestHit $\leftarrow \phi$, NearestMiss $\leftarrow \phi$
  for each neighbor $\eta \in X'_i$ **do**
    if $y_i = y_\eta$ **then** ▷ Same class
      NearestHit $\leftarrow$ NearestHit $\cup \{\eta\}$
    else ▷ Different class
      NearestMiss $\leftarrow$ NearestMiss $\cup \{\eta\}$
    end if
  end for
  for each feature $x_j \in X_i$ **do**
    Initialize: Difference $\leftarrow 0$
    if NearestHit $\neq \phi$ **then**
      for each member $\zeta_k \in NearestHit$ **do**
        Difference $\leftarrow$ Difference + $|x_j - \zeta_{kj}|$
      end for
    end if
    if NearestMiss $\neq \phi$ **then**
      for each member $\zeta_k \in NearestMiss$ **do**
        Difference $\leftarrow$ Difference - $|x_j - \zeta_{kj}|$
      end for
    end if
    $ReliefF^{scores}_{j} \leftarrow ReliefF^{scores}_{j}$ + Difference
  end for
end for
Normalize $ReliefF^{scores}$
Return $ReliefF^{scores}$

3.4.2 Mutual information

The MI feature selection algorithm quantifies the statistical dependency between features (represented by X_{ij}) and the corresponding target variable (represented by y_i) by calculating the MI score. It measures the shared information between the two random variables based on their joint and marginal probabilities. By evaluating the information gained or MI with the target variable, the algorithm determines the relevance of features for classification tasks.

The algorithm iterates through each feature in the dataset and calculates the joint and marginal probabilities. Specifically, for each feature X_{ij}, the algorithm calculates the joint probability distribution $P(X_{ij}, X_{ik}, y_i)$, where X_{ik} represents other features in the dataset, and y_i represents the target variable. Additionally, it computes the individual probabilities $P(X_{ij}, y_i)$ and $P(X_{ik}, y_i)$. Using the calculated probabilities, the algorithm computes the MI score for each feature X_{ij}. The MI^{scores} obtained from the algorithm represents the importance of the j-th feature, $X_{(1:m)j}$. Features with higher MI values exhibit stronger

associations and are considered more relevant for classification tasks. The MI feature ranking scheme, described in Algorithm 2, provides a systematic approach to evaluate and rank the relevance of features based on their MI^{scores}.

Algorithm 2 MI Feature Selection Algorithm.

Input: Data $X_{m\times n}$, and target variables $Y_{m\times 1}$ with y_i related to feature vector $X_i, i = 1, 2, \ldots, m$. Here, m and n represent the number of samples in the dataset and the feature dimension, respectively.

Output: Array $MI^{scores}_{1\times n}$ where MI^{scores}_{ij} $(j = 1, 2, \ldots, n)$ represents the importance of the j-th feature i.e., $X_{(1:m)j}$.

Initialize: $MI^{scores}_{ij} \leftarrow 0, \forall i, j$
for each feature X_{ij} in the dataset **do**
  for each feature X_{ik} in the dataset **do**
    Initialize: JointProb $\leftarrow 0$
    for each class label y_i in the dataset **do**
      JointProb $\leftarrow$ JointProb + $P(X_{ij}, X_{ik}, y_i) \cdot \log_2\left(\frac{P(X_{ij}, X_{ik}, y_i)}{P(X_{ij}, y_i)\cdot P(X_{ik}, y_i)}\right)$
    end for
    $MI^{scores}_{ij} \leftarrow MI^{scores}_{ij} + \frac{1}{m} \cdot$ JointProb
  end for
end for
Normalize MI^{scores}
Return MI^{scores}

3.4.3 Minimum redundancy maximum relevance

The mRMR algorithm aims to select an optimal subset of features by maximizing their relevance to the target variable while minimizing redundancy among the selected features. It combines both relevance and redundancy measures to identify informative and non-redundant features, thereby enhancing the interpretability and performance of machine learning models. The relevance criterion is assessed using mutual information (MI) scores, represented by the MI^{scores}, while the redundancy criterion is evaluated using conditional MI scores.

The algorithm starts by initializing an empty set and then iterates through each feature in the dataset. For each feature X_{ij}, the algorithm computes its redundancy with respect to the previously selected features. It calculates the sum of conditional MI scores (RedundancySum) between X_{ij} and the selected features. Additionally, the algorithm evaluates the relevance of X_{ij} to the target variable by computing the MI score $MI(X_{ij}; Y)$. To quantify the trade-off between relevance and redundancy, the mRMR score $mRMR(X_{ij})$ is computed as the difference between the relevance and the average redundancy among the selected features. This measure captures the importance of X_{ij} based on its ability to provide relevant information while minimizing redundancy with other selected features. The algorithm keeps track of the feature with the highest $mRMR^{scores}$ at each iteration, ensuring that the selected features strike a balance between maximum relevance and minimum redundancy. The algorithm continues this process until a stopping criterion is met, such as selecting a fixed number of features or reaching a predefined mRMR score threshold. The resulting selected features form the optimal subset, which exhibits a balance between relevance and redundancy. By leveraging the mRMR criterion, the algorithm effectively identifies a compact and informative feature subset for subsequent machine learning tasks. Algorithm 3 provides a step-by-step description of the mRMR feature selection process, incorporating the calculation of relevance and redundancy measures using the MI^{scores}.

Algorithm 3 mRMR Feature Selection Algorithm.

Input: Data $X_{m\times n}$, and target variables $Y_{m\times 1}$ with y_i related to feature vector $X_i, i = 1, 2, \ldots, m$. Here, m and n represent the number of samples in the dataset and the feature dimension, respectively.
Output: Array $mRMR_{1\times n}^{scores}$ where $mRMR_j^{scores}$ $(j = 1, 2, \ldots, n)$ represents the importance of the j-th feature i.e., $X_{(1:m)j}$.

Initialize: $mRMR_{ij}^{scores} \leftarrow 0, \forall i, j$
for each feature X_{ij} in the dataset **do**
  Initialize: Redundancy $\leftarrow 0$
  for each feature X_k in the selected feature set **do**
    Redundancy $\leftarrow$ Redundancy + $MI(X_{ij}; X_{ik})$
  end for
  Initialize: MaxRelevance $\leftarrow 0$
  for each class label y_i in the dataset **do**
    MaxRelevance $\leftarrow$ max(MaxRelevance, $MI(X_{ij}; y_i)$)
  end for
  $mRMR_{ij}^{scores} \leftarrow MaxRelevance - \frac{1}{|S|} \cdot$ Redundancy
end for
Normalize $mRMR^{scores}$
Return $mRMR^{scores}$

3.4.4 Final feature ranking

The scores obtained from the three feature selectors, ReliefF, MI, and mRMR are combined to generate an overall score for each feature. The feature score $score(X_i)$ is computed as the average of the scores obtained from the individual filter methods and represented in (1).

$$score(X_i) = \frac{ReliefF^{scores} + mRMR^{scores} + MI^{scores}}{3} \tag{1}$$

In (1), $ReliefF(X_i)$, $MI(X_i)$, and $mRMR(X_i)$ represent the scores assigned by ReliefF, MI, and mRMR filter methods to feature X_i respectively (see Algorithms 1, 2, 3). Using these final feature relevance scores, the features are ranked.

3.4.5 Inclusion-exclusion based feature selection

The traditional way of selecting features using filter methods is based on feature relevance score of top-ranked features. But such selection of top-ranked features is made via experimentation [62]. No validation except experimental results is there in such a selection process. Whether the selected features contribute to the classification process is not ensured. Even, always selecting the best might not lead to better selection if there is some redundant information. Thus in this work, we have used a new feature selection method, designed following the single solution-based feature selection strategies [46], that improves the quality (in terms of some evaluation metrics like classification accuracy) of the selected features iteratively by including some features from the features discarded during the filter process and excluding some features selected earlier. In each iteration, the importance of selected features is calculated and, if better then substitute the solution from which it has been generated.

First, we select the top $\tau\%$ of original features based on their rank, considering it as the initial solution. This solution is then iteratively updated using the inclusion-exclusion based feature selection strategy. During each iteration, $\alpha\%$ of the selected features are randomly discarded, while $\beta\%$ of the non-selected features are included for comparison with the previously generated feature subset. The choice of τ, $\alpha\%$, and $\beta\%$ is made experimentally. The

comparison between the newly generated feature subset and the previous feature subset is performed based on the fitness function (see (2)).

$$fitness(X) = wt * acc + (1 - wt) * (1 - \frac{N_{selected}}{n}) \tag{2}$$

The fitness function in our case, as shown in (2), balances the trade-off between the accuracy (acc) of the features selected and the length of the selected feature subset $N_{selected}$) and the trade-off is controlled by weight wt. In our case, we choose $wt = 0.9$. acc is calculated on the validation dataset using KNN classifier. The inclusion-exclusion-based feature selection technique is outlined in Algorithm 4.

Algorithm 4 Inclusion-Exclusion based Feature Selection.

Input: Features $X_{1\times n}$, and ranks (say, $R_{1\times n}$) obtained using (1). X_i is related to rank R_i, $i = 1, 2, \ldots, n$.
Output: Selected features (say, $X_{selected}$).

Extract top-τ% of features based on ranks and store in X_{temp}.
Set $X_{selected} \leftarrow X_{temp}$
Calculate fitness score of $X_{selected}$ using (2) and store in $Fitness_{selected}$
for $iteration = 1, 2, \ldots$ **do**
    Set $X_{temp} \leftarrow X_{selected}$
    Generate a new solution using inclusion and exclusion strategy.
    Exclude $[1, \alpha]$% of total features from X_{temp} randomly and include top $[1, \beta]$% of total features from $X - X_{temp}$ to form new X_{temp}. The exact values of inclusion and exclusion are decided randomly.
    Calculate fitness score of X_{temp} using (2) and store in $Fitness_{temp}$
    if $Fitness_{temp} > Fitness_{selected}$ **then**
        Set $X_{selected} \leftarrow X_{temp}$
        Set $Fitness_{selected} \leftarrow Fitness_{temp}$
    end if
end for
Return $X_{selected}$

4 Results and discussion

In this section, we first discuss the performance of the proposed deepfake detection method and then a comprehensive analysis of the obtained results. We carefully examine how these methods fare when subjected to various challenges and perturbations, assessing their ability to maintain effectiveness in adverse conditions. Additionally, we extend our investigation to encompass not only the performance within the same dataset (intra-dataset setup) but also their performance when confronted with different datasets (inter-dataset setup). We have considered Area Under Curve (AUC) score and test accuracy as performance measure metrics.

4.1 Dataset description

In order to evaluate the performance of our proposed deep fake detection methodology, we have conducted experiments on two widely used public benchmark datasets: Celeb-DF (V2) [63], abbreviated as "CeDF", and FaceForensics++ [64], abbreviated as "FF++" now onward. We are first describing the datasets and then the data preparation, and data splitting for experiments.

CeDF dataset The dataset consists of 5,639 high-quality videos featuring different celebrities. These videos were generated from 590 original videos collected from YouTube, with various filters applied to create variations in age, gender, and background. The dataset provides a total of 518 videos, including 178 real videos and 340 fake videos, which are suitable for evaluating the performance of new deep fake detection methodologies.

FF++ dataset The dataset comprises four categories of mixed videos: Deepfakes, Face2Face, FaceSwap, and NeuralTextures. Each category contains 1,000 videos generated from unique source videos. For our experiments, we focused on the deep fake category, considering it as the fake video class, while the original category videos were regarded as real videos. The FF++ dataset is available in different compression rates, and we utilized the c23 version.

Data preparation for experiments To facilitate experiments, we have divided both datasets into train, validation, and test sets. To extract frames from the train video sets, we have selected equidistant frames in the time domain for the training sets. This is so done to include maximum variations in the training image dataset. For the test and validation sets, we have considered only the first I-frame from each video. This means the authentication of a video is decided by the first I-frame in a video. During the generation of cropped face images using the MTCNN algorithm, we have extracted only one cropped face image having the highest confidence score. Table 1 shows the detailed distribution of video and image levels used in our research.

4.2 Experimental protocol

We have employed two main evaluation setups to assess the performance of our model on the two datasets:

Intra-Dataset Evaluation In this setup, the model has been trained and tested on the images collected from the same dataset. In this case, the artifacts left by the deepfaking methods remain the same for both train and test samples. More specifically, the model learns the artifacts during training that are present in the test samples. We represent the evaluation protocols as FF++_FF++, and CeDF_CeDF when trained and tested the model on the FF++ dataset, and CeDF dataset respectively. This evaluation setup allows one to analyze the model's performance (i.e., its capability to learn and discriminate between real and fake videos) within a single dataset i.e., when deepfake visual contents are generated using same GAN architecture.

Inter-Dataset Evaluation In this setup, the model has been trained on one dataset and tested on the others. It is noteworthy to mention that inter-dataset deepfake detection is more

Table 1 The class distribution of the datasets used here

Dataset	#Video						#Image					
	Train		Val		Test		Train		Val		Test	
	Re	Fa	Re	Fa	Re	Fa	Re	Fa	Re	Fa	Re	Fa
Celeb-DF	612	4399	100	900	178	340	1130	8022	100	900	178	340
FF++	700	700	200	200	100	100	2930	2946	200	200	100	100

Labels "Re" and "Fa" correspond to Real and Fake, respectively

challenging, as deep learning models often learn dataset-specific patterns, reducing their effectiveness on unseen data with different characteristics. Moreover, variations in manipulation techniques, GAN architectures, and post-processing methods create inconsistencies in the artifacts detected by the model. As a result, this experiment is helps to understand how the model will perform in real scenario. In this experimental setup, we have trained the model on the FF++ dataset and tested it on the CeDF dataset (referred to as FF++_CeDF). Similarly, we have also trained the model on the CeDF dataset and tested it on the FF++ dataset (referred to as CeDF_FF++). This evaluation setup allows us to examine the model's generalization and robustness across different datasets, representing different modalities and characteristics.

4.3 Parameters used

We have stacked deep learning features extracted from three different CNN models: DenseNet-121, EfficientNet-B0, and Xception. When training and fine-tuning these CNN models, we set the batch size to 32, the number of epochs to 100, and the steps per epoch to 40, and used the Adam optimizer. The loss curves during training on the validation set are shown in Fig. 7. The graph offers valuable insights into the progression of the model's training, illustrating how the model adjusts and enhances its predictions. After obtaining the deep learning features from the three CNN models, we have merged them into a single combined feature vector. Subsequently, our proposed methodology is applied to this combined feature vector. During each iteration of the algorithm, we have computed the accuracy using the KNN classifier, with a specified value of $n\_neighbors$ set to 5.

We have made a number of experiments to decide the parameters: τ (i.e., percentage of top-ranked features from the filter method used in the second part, called incluion-exclusion based

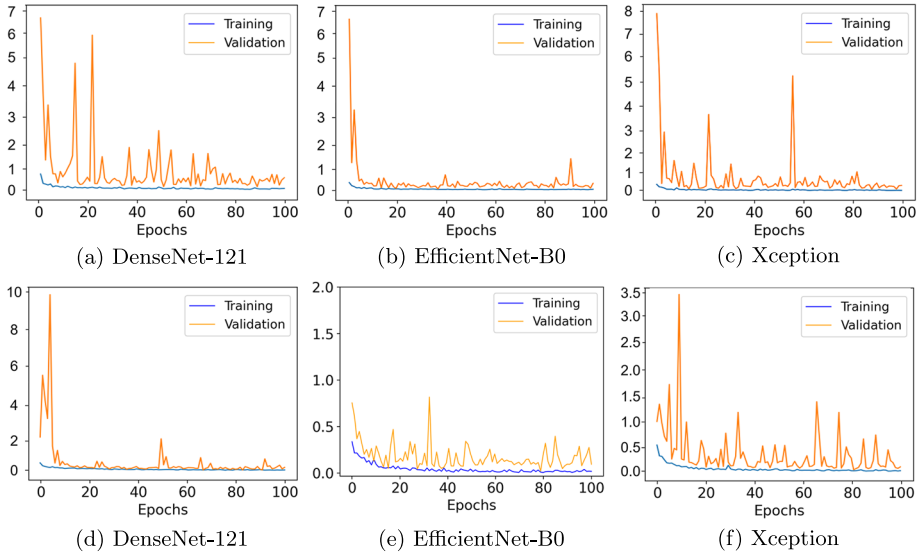


Fig. 7 The loss curve for FF++ (top row) and CeDF (bottom row) is depicted from DenseNet-121, EfficientNet-B0, and Xception

feature selection), α (i.e., the upper limit of the percentage of features from candidate solution is excluded in each iteration), and β (i.e., percentage of features from outside candidate solution is included in each iteration) before going into further experiments (see Algorithm 4 for more details). It is to be noted here that the percentages are considered with respect to the original feature-length and the actual values of α, and β is selected randomly in the range $[1, \alpha]$ and $[1, \beta]$ respectively. We have varied values of τ from 30 to 70 with step size 10 while the values of α, and β from 5 to 10 with step size 5. All the experiments have been conducted on the FF++ dataset following the intra-dataset evaluation scheme. All the experimental performances in this case are evaluated on the validation set and the performance scores are recorded in Table 2. This table illustrates that the best objective score is achieved when $\tau = 30$, $\alpha = 10$, and $\beta = 10$. Therefore, we have continued with these parameter values for the rest of the experiments.

To check the impact of the randomness of the proposed feature selection algorithm, we have executed the second part (i.e., the inclusion-exclusion based method) of our feature selection algorithm for 1000 iterations on all the experimental setups (see Section 4.2). The experimental outcomes are presented in Fig. 8. The fitness scores are calculated using validation datasets. As depicted in this figure, the fitness score reaches a saturation point after the 250^{th} iteration except in the case $CeDF\_FF++$ that also saturated after 850 iterations. These results indicate that the near-optimal features are obtained and thus almost no improvement is expected by increasing the iteration number.

Table 2 Ablation studies concerning hyperparameters of the proposed feature selection algorithm used for deepfake detection

Weight parameter			Performance score (in %)		
τ	α	β	Validation accuracy	Feature length	Objective score
30	05	05	97.75	480	95.12
30	05	10	98.25	553	94.94
30	10	05	98.50	350	96.47
30	10	10	98.75	294	97.01
40	05	05	96.50	1026	90.53
40	05	10	96.75	1197	89.56
40	10	05	94.50	1186	88.00
40	10	10	97.00	1169	89.77
50	05	05	96.00	1501	86.79
50	05	10	95.75	1686	85.34
50	10	05	96.00	1528	86.57
50	10	10	96.00	1513	86.70
60	05	05	95.75	1763	84.78
60	05	10	95.75	1784	84.62
60	10	05	95.00	1801	84.15
60	10	10	95.50	1790	84.60
70	05	05	95.50	1816	84.39
70	05	10	95.75	1827	84.34
70	10	05	95.50	1788	84.61
70	10	10	95.75	1786	84.65

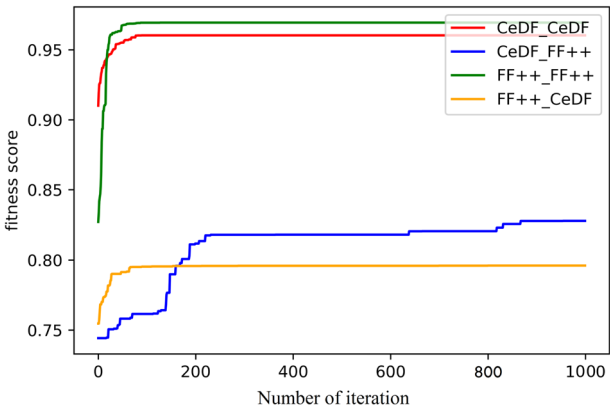


Fig. 8 Fitness score variations vs the number of epochs. This graph illustrated the saturation of the model for all four experimental setups

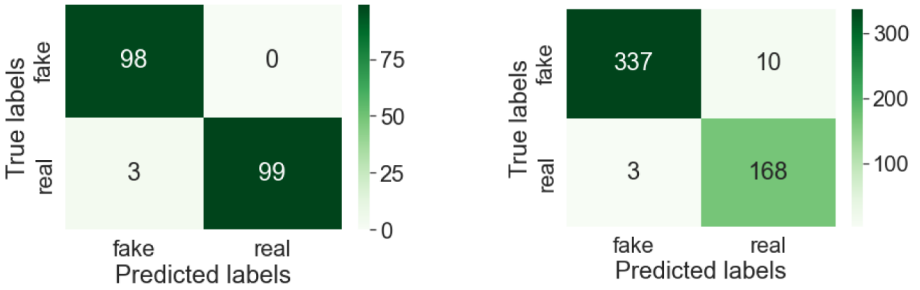
4.4 Results using intra-dataset experimental setup

In this subsection, we provide a comprehensive evaluation of the proposed method's performance using an experimental setup within both datasets. This particular setup allows us to assess how well the entire architecture and its components perform when the test artifacts are already known during the model training phase. We assess three different models: DenseNet121, EfficientNetB0, and Xception, both individually and in combinations of their components. Furthermore, we evaluate the performance of the proposed method, which involves applying a feature selection process to the combined features based on our proposed algorithm. The test accuracy and AUC scores for both datasets are summarized in Table 3. It is observed that the performance of the model utilizing combined features shows some improvement compared to the individual components model. This feature selection approach enhances the end performance 1.00% for FF++ and 1.74% in the case of the CeDF dataset while reducing the feature dimension by 93.19% and 92.08% for FF++ and CeDF datasets respectively. The performances of the proposed method through the confusion matrix on both the intra-dataset experimental setup are shown in Fig. 9. From the results, we observe that only three real videos are mistakenly classified as fake when the proposed method is trained and tested on FF++ dataset, while the misclassification number is 13 for the CeDF dataset.

Table 3 Performances of individual and combined models on FF++ and Celeb-DF intra-datasets

Model	FF++			CeDF		
	# Features	Acc	AUC Score	# Features	Acc	AUC Score
DenseNet-121	1024	95.50	96.06	1024	92.47	90.78
EfficientNet-B0	1280	96.50	96.48	1280	92.86	91.48
Xception	2048	97.50	97.89	2048	95.17	93.11
Stacked feature	4352	97.50	98.21	4352	95.75	97.00
Proposed	294	98.50	98.50	342	97.49	97.70

Here "Acc" refers to the classification accuracy



(a) FF++_FF++ (b) CeDF_CeDF

Fig. 9 Confusion matrices obtained using the intra-dataset experimental setup

4.5 Results using inter-dataset experimental setup

We have evaluated the performance of the proposed method using the inter-dataset experimental setup. The objective is to assess the generalization capability of the proposed model by training on one dataset and testing on another dataset. We follow the experimentation protocols outlined in Section 4.2. The experimental results are provided in Table 4. When we train the model on the CeDF dataset and test it on the FF++ dataset (i.e., CeDF_FF++ experimental setup), we achieve an accuracy of 85.50% and an AUC of 85.70%. In this case, we have observed 3.0% increased test accuracy with 92.08% feature reduction compared to the original stacked features. On the other hand, training on the FF++ dataset and testing on the CeDF dataset (i.e., FF++_CeDF experimental setup) yields an accuracy of 71.82% and an AUC score of 79.00% using our proposed model. Here also we can see a 1.16% increment in test accuracy with 93.08% reduced features compared to the original stacked features. The confusion matrices obtained by evaluating the proposed model using the inter-dataset experimental setup are shown in Fig. 10. It is worth mentioning that the number of misclassified videos is 145 for the CeDF test dataset and 29 for the FF++ test dataset.

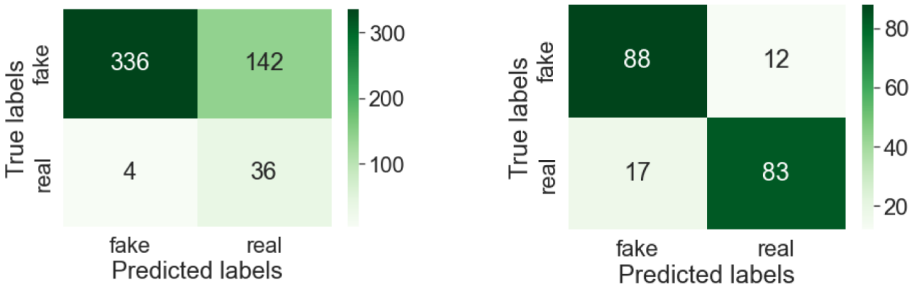
4.6 Comparison with state-of-the-art methods

To ensure a fair and meaningful comparison, we have conducted evaluations of the methods in accordance with our established experimental protocols (given in Subsection 4.2), which were described earlier. This approach allows us to assess and compare different methods

Table 4 Performances of individual and combined models on FF++ and Celeb-DF inter-datasets

Model	FF++_CeDF			CeDF_FF++		
	# Features	Acc	AUC score	# Features	Acc	AUC score
DenseNet-121	1024	71.04	66.00	1024	65.00	71.00
EfficientNet-B0	1280	66.22	62.00	1240	79.00	87.00
Xception	2048	66.83	52.00	2048	69.00	73.00
Stacked feature	4352	70.66	70.00	4352	81.50	87.00
Proposed	359	71.82	79.00	252	85.50	85.70

Here "Acc" refers to the classification accuracy



(a) FF++_CeDF (b) CeDF_FF++

Fig. 10 Confusion matrices obtained using the inter-dataset experimental setup

based on important factors such as their detection performance, robustness and generalization ability. We present the performance results of the deepfake detection methods on our experimental setup, specifically on the two image dataset. The corresponding performance metrics are test accuracy and AUC score. These results, reported in Table 5, show the performance against intra-dataset experiments, and Table 6 exhibits against inter-dataset

Table 5 Comparison with state-of-the-art methods based on intra-dataset experiments

Experiment	Method	# Features	Test accuracy	AUC score
CeDF	Li et al. [63]	2048	95.37	98.88
	Afchar et al. [8]	1024	65.64	65.33
	Afchar et al. [8]	1024	65.83	64.80
	Qian et al. [66]	4096	87.06	81.48
	Guo et al [67]	300	68.33	78.04
	Mohiuddin et al. [46]	270	97.30	99.35
	Zhu et al. [45]	–	96.92	99.61
	Roy et al. [68]	512	95.17	93.25
	Khalid et al. [65]	–	96.05	95.00
	Proposed	342	97.49	97.70
FF++	Li et al. [63]	2048	96.00	98.34
	Afchar et al. [8]	1024	67.50	66.81
	Afchar et al. [8]	1024	65.00	66.93
	Qian et al. [66]	4096	95.50	96.52
	Guo et al. [67]	300	78.50	87.62
	Mohiuddin et al. [46]	297	98.00	99.16
	Zhao et al. [69]	–	96.00	98.97
	Pellicer et al. [70]	–	95.10	–
	Roy et al. [68]	512	97.50	97.50
	Khalid et al. [65]	–	98.20	97.00
	Proposed	294	98.50	98.50

Here, '-' indicates that the information is not available from the respective work

Table 6 Comparison with state-of-the-art methods based on inter-dataset experiments

Experiment	Method	# Features	Test accuracy	AUC score
CeDF_FF++	Li et al. [63]	2048	64.50	75.19
	Afchar et al. [8]	1024	50.50	51.51
	Qian et al. [66]	4096	54.50	54.04
	Ganguly et al. [34]	1024	65.00	63.80
	Mohiuddin et al. [16]	1024	60.00	59.93
	Mohiuddin et al. [46]	270	66.50	76.72
	Roy et al. [68]	512	69.00	68.94
	Khalid et al. [65]	–	74.71	–
	Proposed	252	85.50	85.70
FF++_CeDF	Li et al [63]	2048	58.06	55.60
	Afchar et al. [8]	1024	66.41	65.58
	Qian et al. [66]	4096	63.89	53.89
	Ganguly et al. [34]	1024	68.04	66.12
	Mohiuddin et al. [16]	1024	63.71	56.43
	Zhao et al. [69]	–	–	67.44
	Miao et al. [71]	–	–	66.12
	Roy et al. [68]	512	68.53	65.06
	Khalid et al. [65]	–	81.26	–
	Proposed	359	71.82	79.00

Here, '-' indicates that the information is not available from the respective work

experiments. By examining the results, it could be found that the method proposed by Khalid et al. [65] achieves higher test accuracy than ours in FF+_CeDF experimental setup. But test accuracy is not a right metric for comparison on CeDF dataset as it is a skewed dataset i.e., number of samples in both the classes are not same. From the performance scores in Tables 5, and 6, it is clear that our method surpass most of the state-of-the-art methods used here for comparison, particularly in the context of intra-dataset experimental setups. In other words, the proposed method demonstrates superior performance when tested within the same dataset (see Table 5). Furthermore, when considering inter-dataset experiments (see Table 6), which essentially evaluate the generalization ability of a model, once again the current method outperforms most of the other methods and achieves state-of-the-art results in the majority of the experiments. Overall, these findings underscore the impressive performance of the current method compared to other methods, both within and across different datasets, confirming its superiority in deepfake detection.

5 Conclusion

In this study, we introduced a deep learning-based method for deepfake video detection using three pre-trained CNN models, namely DenseNet-121, EfficientNet-B0, and Xception. By extracting and integrating deep features from these models, we obtained more robust and discriminative representations. Then a novel feature selection technique was used to reduce the dimension of combined features, optimizing computational efficiency and eliminating

feature redundancy. The use of a KNN classifier, rather than a conventional deep learning classifier, introduced a lightweight yet effective alternative for deepfake video classification. Comprehensive evaluations on standard datasets demonstrated the effectiveness and generalizability of our method in both intra-dataset and inter-dataset scenarios, surpassing several state-of-the-art methods in terms of classification accuracy and AUC score. Furthermore, our findings provide insights into the importance of diverse feature representations in detecting deepfake artifacts, paving the way for future research to explore more efficient ensemble-based feature extraction and classification techniques. While our method showed generalizability, the AUC drop in inter-dataset evaluation highlights challenges due to out of distribution problem. As deepfake techniques evolve, continuous improvements in detection strategies remain crucial. Future work could focus on enhancing adaptability to novel deepfake techniques and optimizing the trade-off between detection accuracy and computational efficiency.

Author Contributions Sk Mohiuddin: Conceptualization, Methodology, Formal analysis, and investigation, Writing original draft preparation, Writing review and editing, Resources. Ayush Roy: Conceptualization, Methodology, Formal analysis, and investigation, Writing original draft preparation, Writing review and editing, Resources. Saptarshi Pani: Conceptualization, Methodology, Formal analysis, and investigation, Writing original draft preparation, Writing review and editing, Resources. Samir Malakar: Conceptualization, Methodology, Formal analysis, and investigation, Writing original draft preparation, Writing review and editing, Resources, Supervision. Ram Sarkar: Conceptualization, Methodology, Formal analysis, and investigation, Writing original draft preparation, Writing review and editing, Resources, Supervision.

Funding Not applicable

Data Availability All the data used in this work are publicly available to the researchers through the works Li et al. [63], Rössler et al. [64].

Materials availability Not applicable

Code availability Not applicable

Declarations

Ethical Approval Not applicable

Conict of interest/Competing Interests All the authors declare that they have no conflict of interest or competing interest.

References

1. Babaei R, Cheng S, Duan R, Zhao S (2025) Generative artificial intelligence and the evolving challenge of deepfake detection: A systematic analysis. J Sens Actuator Netw 14(1):17
2. Naitali A, Ridouani M, Salahdine F, Kaabouch N (2023) Deepfake attacks: Generation, detection, datasets, challenges, and research directions. Computers 12(10):216
3. Ehtesham A, Kumar S, Singh A, Khoei TT (2025) Movie gen: Swot analysis of meta's generative ai foundation model for transforming media generation, advertising, and entertainment industries. In: 2025 IEEE 15th Annual Computing and Communication Workshop and Conference (CCWC), pp 00189–00195. IEEE
4. Karras T, Laine S, Aila T (2019) A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 4401–4410
5. Lavault A, Roebel A, Voiry M (2022) Stylewavegan: Style-based synthesis of drum sounds with extensive controls using generative adversarial networks. arXiv preprint. arXiv:2204.00907

6. Figueira A, Vaz B (2022) Survey on synthetic data generation, evaluation methods and gans. Mathematics 10(15):2733
7. Mohiuddin S, Malakar S, Kumar M, Sarkar R (2023) A comprehensive survey on state-of-the-art video forgery detection techniques. Multimed Tools Appl 82(22):33499–33539
8. Afchar D, Nozick V, Yamagishi J, Echizen I (2018) Mesonet: a compact facial video forgery detection network. In: 2018 IEEE International Workshop on Information Forensics and Security (WIFS), pp 1–7. IEEE
9. Almutairi Z, Elgibreen H (2022) A review of modern audio deepfake detection methods: challenges and future directions. Algorithms 15(5):155
10. Fagni T, Falchi F, Gambini M, Martella A, Tesconi M (2021) Tweepfake: About detecting deepfake tweets. PLoS ONE 16(5):0251415
11. Boutadjine A, Harrag F, Shaalan K (2025) Human vs. machine: A comparative study on the detection of ai-generated content. ACM Trans Asian Low-Res Lang Inf Process 24(2):1–26
12. Xie Y, Lu Y, Fu R, Wen Z, Wang Z, Tao J, Qi X, Wang X, Liu Y, Cheng H et al (2025) The codecfake dataset and countermeasures for the universally detection of deepfake audio. IEEE Trans Audio Speech Lang Process
13. Heidari A, Jafari Navimipour N, Dag H, Unal M (2024) Deepfake detection using deep learning methods: A systematic and comprehensive review. Wiley Interdiscipl Rev: Data Min Knowl Discov 14(2):1520
14. Kumar A, Singh D, Jain R, Jain DK, Gan C, Zhao X (2025) Advances in deepfake detection algorithms: Exploring fusion techniques in single and multi-modal approach. Inf Fus 102993
15. Rana MS, Sung AH (2020) Deepfakestack: A deep ensemble-based learning technique for deepfake detection. In: 2020 7th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2020 6th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom), pp 70–75. IEEE
16. Mohiuddin S, Ganguly S, Malakar S, Kaplun D, Sarkar R (2021) A feature fusion based deep learning model for deepfake video detection. In: International Conference on Mathematics and Its Applications in New Computer Systems, pp 197–206. Springer
17. Essa E (2024) Feature fusion vision transformers using mlp-mixer for enhanced deepfake detection. Neurocomputing 598:128128
18. Wang B, Wu X, Wang F, Zhang Y, Wei F, Song Z (2024) Spatial-frequency feature fusion based deepfake detection through knowledge distillation. Eng Appl Artif Intell 133:108341
19. Naskar G, Mohiuddin S, Malakar S, Cuevas E, Sarkar R (2024) Deepfake detection using deep feature stacking and meta-learning. Heliyon 10(4)
20. Das S, Chatterjee A, Dey S, Saha S, Malakar S (2022) Breast cancer detection from histology images using deep feature selection. In: Proceedings of International Conference on Frontiers in Computing and Systems: COMSYS 2021, pp 323–330. Springer
21. Sarkar A, Hossain SS, Sarkar R (2023) Human activity recognition from sensor data using spatial attention-aided cnn with genetic algorithm. Neural Comput Appl 35(7):5165–5191
22. Yan F, Xu J, Yun K (2019) Dynamically dimensioned search grey wolf optimizer based on positional interaction information. Complexity 2019:1–36
23. Malakar S, Banerjee N, Prasad DK (2025) Compact representation for memory-efficient storage of images using genetic algorithm-guided key pixel selection. Eng Appl Artif Intell 139:109540
24. Malakar S, Ghosh M, Bhowmik S, Sarkar R, Nasipuri M (2020) A ga based hierarchical feature selection approach for handwritten word recognition. Neural Comput Appl 32:2533–2552
25. Ahmed S, Ghosh KK, Garcia-Hernandez L, Abraham A, Sarkar R (2021) Improved coral reefs optimization with adaptive β-hill climbing for feature selection. Neural Comput Appl 33(12):6467–6486
26. Neves JC, Tolosana R, Vera-Rodriguez R, Lopes V, Proença H, Fierrez J (2020) Ganprintr: Improved fakes and evaluation of the state of the art in face manipulation detection. IEEE J Sel Topics Signal Process 14(5):1038–1048
27. Liu C, Chen H, Zhu T, Zhang J, Zhou W (2023) Making deepfakes more spurious: Evading deep face forgery detection via trace removal attack. IEEE Trans Depend Secure Comput
28. Nirkin Y, Wolf L, Keller Y, Hassner T (2021) Deepfake detection based on discrepancies between faces and their context. IEEE Trans Pattern Anal Mach Intell 44(10):6111–6121
29. Zhao T, Xu X, Xu M, Ding H, Xiong Y, Xia W (2021) Learning self-consistency for deepfake detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 15023–15033
30. Nguyen HM, Derakhshani R (2020) Eyebrow recognition for identifying deepfake videos. In: 2020 International Conference of the Biometrics Special Interest Group (BIOSIG), pp 1–5. IEEE
31. Wu Y, Wo Y, Li C, Han G (2023) Learning domain-invariant representation for generalizing face forgery detection. Computers & Security 130:103280

32. Yan Z, Zhang Y, Fan Y, Wu B (2023) Ucf: Uncovering common features for generalizable deepfake detection. arXiv preprint. arXiv:2304.13949
33. Chen B, Li T, Ding W (2022) Detecting deepfake videos based on spatiotemporal attention and convolutional lstm. Inf Sci 601:58–70
34. Ganguly S, Mohiuddin S, Malakar S, Cuevas E, Sarkar R (2022) Visual attention-based deepfake video forgery detection. Pattern Anal Appl 25(4):981–992
35. Pang G, Zhang B, Teng Z, Qi Z, Fan J (2023) Mre-net: Multi-rate excitation network for deepfake video detection. IEEE Trans Circ Syst Video Technol
36. Yang Z, Liang J, Xu Y, Zhang X-Y, He R (2023) Masked relation learning for deepfake detection. IEEE Trans Inf Forensics Secur 18:1696–1708
37. Zhao C, Wang C, Hu G, Chen H, Liu C, Tang J (2023) Istvt: interpretable spatial-temporal video transformer for deepfake detection. IEEE Trans Inf Forensics Secur 18:1335–1348
38. Wang H, Liu Z, Wang S (2023) Exploiting complementary dynamic incoherence for deepfake video detection. IEEE Trans Circ Syst Video Technol
39. Wang T, Cheng H, Chow KP, Nie L (2023) Deep convolutional pooling transformer for deepfake detection. ACM Trans Multimed Comput Commun Appl 19(6):1–20
40. Heo Y-J, Yeo W-H, Kim B-G (2023) Deepfake detection algorithm based on improved vision transformer. Appl Intell 53(7):7512–7527
41. Wang S, Zhu D, Chen J, Bi J, Wang W (2024) Deepfake face discrimination based on self-attention mechanism. Pattern Recogn Lett 183:92–97
42. Reis PMGI, Ribeiro RO (2024) A forensic evaluation method for deepfake detection using dcnn-based facial similarity scores. Forensic Sci Int 358:111747
43. Tian C, Luo Z, Shi G, Li S (2023) Frequency-aware attentional feature fusion for deepfake detection. In: ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp 1–5. IEEE
44. Zhang D, Chen J, Liao X, Li F, Chen J, Yang G (2024) Face forgery detection via multi-feature fusion and local enhancement. IEEE Trans Circ Syst Video Technol
45. Zhu C, Zhang B, Yin Q, Yin C, Lu W (2024) Deepfake detection via inter-frame inconsistency recomposition and enhancement. Pattern Recogn 147:110077
46. Mohiuddin S, Sheikh KH, Malakar S, Velásquez JD, Sarkar R (2023) A hierarchical feature selection strategy for deepfake video detection. Neural Comput Appl 35(13):9363–9380
47. Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 4700–4708
48. Tan M, Le Q (2019) Efficientnet: Rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning, pp 6105–6114. PMLR
49. Chollet F (2017) Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 1251–1258
50. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L (2009) Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp 248–255. Ieee
51. Gholamalinezhad H, Khosravi H (2020) Pooling methods in deep neural networks, a review. arXiv preprint. arXiv:2009.07485
52. Khalid F, Javed A, Irtaza A, Malik KM (2023) Deepfakes catcher: A novel fused truncated densenet model for deepfakes detection. In: Proceedings of International Conference on Information Technology and Applications: ICITA 2022, pp 239–250. Springer
53. Jian W, Zhou Y, Liu H (2020) Densely connected convolutional network optimized by genetic algorithm for fingerprint liveness detection. IEEE Access 9:2229–2243
54. Sandler M, Howard A, Zhu M, Zhmoginov A, Chen L-C (2018) Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 4510–4520
55. Pokroy AA, Egorov AD (2021) Efficientnets for deepfake detection: Comparison of pretrained models. In: 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus), pp 598–600. IEEE
56. Biswas A, Bhattacharya D, Kumar K (2021) Deepfake detection using 3d-xception net with discrete fourier transformation. J Inf Syst Telecommun 9(35):161–168
57. Sahib I, AlAsady TAA (2022) Deep fake image detection based on modified minimized xception net and densenet. In: 2022 5th International Conference on Engineering Technology and Its Applications (IICETA), pp 355–360. IEEE
58. Ganguly S, Ganguly A, Mohiuddin S, Malakar S, Sarkar R (2022) Vixnet: Vision transformer with xception network for deepfakes based video and image forgery detection. Expert Syst Appl 210:118423

59. Sarkar S, Ghosh M, Chatterjee A, Malakar S, Sarkar R (2019) An advanced particle swarm optimization based feature selection method for tri-script handwritten digit recognition. In: Computational Intelligence, Communications, and Business Analytics: Second International Conference, CICBA 2018, Kalyani, India, July 27–28, 2018, Revised Selected Papers, Part I 2, pp 82–94. Springer
60. Shaw SS, Ahmed S, Malakar S, Garcia-Hernandez L, Abraham A, Sarkar R (2021) Hybridization of ring theory-based evolutionary algorithm and particle swarm optimization to solve class imbalance problem. Compl Intell Syst 7(4):2069–2091
61. Ghosh M, Malakar S, Bhowmik S, Sarkar R, Nasipuri M (2017) Memetic algorithm based feature selection for handwritten city name recognition. In: Computational Intelligence, Communications, and Business Analytics: First International Conference, CICBA 2017, Kolkata, India, March 24–25, 2017, Revised Selected Papers, Part II, pp 599–613. Springer
62. Roy S, Bhattacharya A, Sarkar N, Malakar S, Sarkar R (2020) Offline hand-drawn circuit component recognition using texture and shape-based features. Multimed Tools Appl 79:31353–31373
63. Li Y, Yang X, Sun P, Qi H, Lyu S (2020) Celeb-df: A large-scale challenging dataset for deepfake forensics. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 3207–3216
64. Rössler A, Cozzolino D, Verdoliva L, Riess C, Thies J, Nießner M (2019) FaceForensics++: Learning to detect manipulated facial images. In: International Conference on Computer Vision (ICCV), pp 1–11
65. Khalid F, Javed A, Malik KM, Irtaza A (2024) Explanet: A descriptive framework for detecting deepfakes with interpretable prototypes. Behavior, and Identity Science, IEEE Transactions on Biometrics
66. Qian Y, Yin G, Sheng L, Chen Z, Shao J (2020) Thinking in frequency: Face forgery detection by mining frequency-aware clues. In: European Conference on Computer Vision, pp 86–103. Springer
67. Guo Z, Yang G, Chen J, Sun X (2021) Fake face detection via adaptive manipulation traces extraction network. Comput Vis Image Underst 204:103170
68. Roy A, Mohiuddin S, Minenko M, Kaplun D, Sarkar R (2025) Deepspace: Navigating the frontier of deepfake identification using attention-driven xception and a task-specific subspace. Proceedings Copyright 163:172
69. Zhao H, Zhou W, Chen D, Wei T, Zhang W, Yu N (2021) Multi-attentional deepfake detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 2185–2194
70. Pellicer AL, Li Y, Angelov P (2024) Pudd: Towards robust multi-modal prototype-based deepfake detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 3809–3817
71. Miao C, Chu Q, Li W, Li S, Tan Z, Zhuang W, Yu N (2021) Learning forgery region-aware and id-independent features for face manipulation detection. IEEE Trans Biomet Behav Ident Sci 4(1):71–84