**ABSTRACT**

With the rapid development of face forgery technology, deepfake videos have attracted widespread attention in digital media. Perpetrators heavily utilize these videos to spread disinformation and make misleading statements. Most existing methods for deepfake detection mainly focus on texture features, which are likely to be impacted by external fluctuations, such as illumination and noise. Besides, detection methods based on facial landmarks are more robust against external variables but lack sufficient detail. Thus, how to effectively mine distinctive features in the spatial, temporal, and frequency domains and fuse them with facial landmarks for forgery video detection is still an open question. To this end, we propose a Landmark Enhanced Multimodal Graph Neural Network (LEM-GNN) based on multiple modalities’ information and geometric features of facial landmarks. Specifically, at the frame level, we have designed a fusion mechanism to mine a joint representation of the spatial and frequency domain elements while introducing geometric facial features to enhance the robustness of the model. At the video level, we first regard each frame in a video as a node in a graph and encode temporal information into the edges of the graph. Then, by applying the message passing mechanism of the graph neural network (GNN), the multimodal feature will be effectively combined to obtain a comprehensive representation of the video forgery. Extensive experiments show that our method consistently outperforms the state-of-the-art (SOTA) on widely-used benchmarks.

**Keywords** Deepfake Detection · Multimodal Fusion · Graph Neural Network · Computer Vision · Deep Learning

1 Introduction

Facial manipulation technology makes it simple for amateurs to produce excellent synthesized videos. Moreover, users can access convenient face-swapping tools, for instance, through applications like DeepFaceLab [1] and Zao [2]. In addition, the development of Artificial Intelligence (AI), especially deep learning technology, has further advanced the maturation of video forgery.
Manipulation schemes based on deep learning are typically called “deepfake,” and videos synthesized by deepfake are highly realistic, being extremely hard to identify with the naked eye. Unfortunately, such deepfake videos are often abused to create fake content that infringes individual privacy, spreads misleading information, and undermines the trust in digital media, having a significantly destructive impact on society [3].

As a result, developing a reliable and effective facial manipulation detection algorithm is vitally essential. In recent years, detecting such realistic videos has become a popular research area. Existing forensic methods can be roughly categorized into traditional and deep learning algorithms. The core mechanism of traditional detection algorithms is based on a video’s specific attributes, such as eye blinking, inconsistency of head poses, depth of field change [4, 5, 6, 7, 8, 9], or even biological features [10, 11, 12]. These traditional methods have the advantage of being highly interpretable and less prone to over-fitting. However, these methods heavily require expert digital image processing knowledge and rely on extensive feature engineering.

Compared to traditional schemes, detection methods based on deep learning have gained more attention in recent years, and most of them outperform traditional algorithms in experiments. Mainstream deep learning methods detect the forgery video by utilizing appearance features, namely spatial domain features [13, 14, 15], temporal domain features [16, 17], and frequency domain features [18, 19]. However, many of these detection methods fail to distinguish real scene videos. There are several possible reasons for that. First, a growing variety of facial manipulation methods have also emerged due to the rapid advancement of deep learning techniques. These AI-based forgery methods produce features that differ from one another [20]. As a result, when fake videos use various types of manipulation, the detection model has difficulty generalizing. Second, different models typically leverage different information from the forgery video for detection. For example, some models may pay more attention to spatial features such as colors and textures, some may judge the fake video through the temporal inconsistency of adjacent frames, and some may notice the changes in the frequency domain before and after manipulation. However, most current schemes do not utilize all the meaningful information above. Furthermore, although appearance features can capture subtle facial information, it has a high risk of being affected by external changes such as light, mask, and noise.

Another deep learning-based approach to solving this problem is to utilize facial landmarks to detect forgery. Landmarks represent the facial key points and can record the movements of facial organs. [21] proposes a reliable and effective method called LRNet for deepfake video detection by leveraging precise geometric features of landmarks. Because landmarks only sketch out the shapes and contours of the face, most landmark-based models are lighter-weighted. However, a synthetic image’s subtle changes in color and light cannot be represented with accuracy by a landmark. Therefore, combining the landmark geometric feature with appearance information from multiple modalities can help forgery detectors produce better results.

In addition, making effective use of the video’s temporal information is essential for getting desired experimental results on the detection of deepfake videos. Most popular research captures the spatial information of each frame and then uses RNN to extract the temporal information. Also, some other works like [22] use temporal transformer architecture to capture the temporal inconsistency between frames. Nowadays, transformer [23] has already appeared to be the standard in academic circles [24]. However, [25] denotes that transformer is a unique variety of GNN. Transformer can be viewed as a fully connected GNN without edge features. Because complete connectivity frequently requires much computational power, we choose GNN in this paper over transformer. The details of the GNN we use are shown in sec.3.2.

To this end, this paper proposes a novel deep learning framework that joints spatial-temporal-frequency three domains’ appearance information and landmark geometric features to mine a comprehensive representation for face forgery video detection. Besides, we apply Graph Attention Network (GAT) [26] and leverage its edge to model temporal forgery information between sequential frames. Extensive experiments show that our model can achieve SOTA on many benchmarks of deepfake detection.

Overall, the major contributions in this paper are summarized as follows:

- We are the first work to joint landmark geometric feature with multiple modalities’ appearance information for deepfake detection.

- We apply GNN and leverage its edge feature for better temporal forgery information modeling.

- We verify that our method can achieve SOTA results at the video-level with relatively small training data by conducting extensive experiments.
2 Related Work

2.1 Deepfake Manipulation Technologies

Traditional facial manipulation techniques manipulate the video by copying, looping, and compressing frames. However, most of these traditional methods can only tamper with a single image, and this approach entails significant time and labor costs. With the development of deep learning, Generative Adversarial Network (GAN) [27], Convolutional Neural Networks (CNN) [28], Recurrent Neural Networks (RNN) [29], Variational Auto-Encoder (VAE) [30], and other deep learning techniques have further advanced the maturity of facial manipulation technology.

The face forgery techniques based on DNNs, namely deepfake, can be found as early as 2017. The portmanteau word “deepfake” combines the words “deep learning” and “fake” and primarily relates to content generated by an artificial neural network. In general, deepfake can be categorized into two different techniques: face swapping and face reenactment.

FaceSwap [31] is the first face-swapping technology, which employs CNN to learn the appearance of the targets’ identity from their photographs. It uses classical graphics techniques, first acquiring several points of the face and then modeling the face using a traditional generic 3D model of the face. Deepfakes [32] is another representative algorithm in face forgery technology, and the main structure of the method is an auto-encoder. This auto-encoder structure significantly lowers the technical threshold for face swapping.

Face reenactment can be more difficult to detect than face swapping because it is highly realistic and the visual artifacts are hard to locate. Face2Face [33] and NeuralTexture [34] are the most typical face reenactment methods, which can achieve a more smooth expression manipulation than face-swapping approaches. Face2Face uses pairs of original and target faces as input and utilizes key points of the five senses as representations to portray and drive the generation of different expressions. NeuralTexture uses the rendered image of the 3D model of the face as a driver to migrate the expressions by exchanging the 3D expression parameters of the original face and the target face and incorporates temporal consistency techniques to ensure the consistency of the synthesized video.

2.2 Deepfake Detection Technologies

When deep learning algorithms were not popular, most of the mainstream algorithms were based on traditional digital image processing methods. These traditional methods are mainly based on the natural characteristic of the video, such as lighting conditions [6, 7, 8], depth of field [9], and the details on head poses [5] or biological signals like heart rate [11].

With the further development of facial manipulation technology, traditional detection methods have difficulty detecting video forgeries. Due to the strong capability of feature extraction in the spatial domain, CNN has become the mainstream detection method.

Mo et al. [13] first leveraged CNN to identify forged images generated by GAN and achieved better results than the previous traditional methods. To further address the spatial inconsistencies created by the deepfake techniques in the forgery process, [35, 36, 37] aim to locate these visual artifacts and identify fake images based on said artifacts (such as color inconsistency, blending boundary, and blur artifacts). To address the problem that low-level image noise features tend to degrade with video compression and high-level semantic features are difficult to distinguish, Afchar et al. [14] proposed MesoNet combined with an Inception module to extract middle-level features for deepfake video detection.

Except for CNN, some current studies [38, 39] start to apply Vision Transformer (ViT) [40] as the feature extractor. As ViT has made a big splash in the vision field in recent years, more and more fields are looking at ViT. [38] employ multi-scale ViT to mine RGB features at different scales in forgery images. [41] combine CNN and ViT by using CNN to extract local features and ViT to extract global features. [42] add a distillation token based on standard ViT architecture. The teacher network trains the additional token to improve the performance of deepfake detection.

However, these techniques disregard the destroying level details of fake images, such as re-compression artifacts, which are reflected in the frequency domain.

After generative models such as GAN emerged, the differences between fake and real images in the spatial domain shrank gradually. Also, image quality is a necessity for many spatial-based methods, and various transformations, such as compression, can significantly affect the image quality [43]. As a result, an increasing number of academics are looking into frequency-domain-based detection methods for forgery detection [18, 43, 44, 45].

Since deepfake generation technologies will inevitably lead to changes in the frequency domain, many studies leverage these weaknesses of deepfake to design detection algorithms. Ricard et al. [46] show that the deepfake technologies
rely on convolution-based upsampling methods to generate images and videos, and most upsampling methods will cause a mismatch in the spectral distribution between the fake and real. Because of this, they have developed a frequency-based scheme, which outperforms many mainstream spatial-based methods. Other relevant research also analyzes the differences between forgery and real-world images in the frequency domain. For example, there are differences in high-frequency Fourier decay between the real and fake, and the noise spectrogram of images is also very different. The deep learning-based methods in the frequency domain also have been well explored. Stuchi et al. use filters to extract information in different ranges, followed by a fully connected layer to obtain the output. Qian et al. design a set of learnable filters to adaptively mine frequency forgery clues using frequency-aware image decomposition.

However, both spatial and frequency-based methods can only model a single image, and these methods ignore the inconsistency of faked videos in the temporal dimension.

Since the deep forgery video is generated frame by frame, it inevitably results in differences between successive frames. These subtle changes in appearance (such as noise, lighting, and motion) often lead to temporal incoherence. Detection methods based on temporal information leverage this incoherence to identify forgery videos.

There are two types of temporal-domain-based methods. One mainstream method is temporal-domain feature extraction based on the optical flow method. Amerini et al. use PWC-Net and Lucas-Kanade (LK) algorithms to convert the original RGB frames of the forgery video into optical flow vectors, capturing the difference in optical flow vectors formed frame-by-frame around the face to achieve the detection of forgery videos. Another mainstream pipeline of temporal detection methods is based on the CNN-RNN structure. Considering the variation in facial features over time, Sabir et al. adopted a CNN-RNN pipeline approach for deepfake detection, using CNN to extract frame-level features of the video and feeding the extracted feature vectors into the RNN module to learn the temporal incoherence between frame sets.

Whether the models are based on spatial, temporal, or frequency domains, they are sensitive to perturbations by external factors. At the same time, these methods are very dependent on large amounts of computational resources and are difficult to deploy. Therefore, gradually some scholars have started to focus on landmark-based detection methods. Landmarks are derived from a set of 68 selected facial key points, as shown in Fig.1. However, since a synthetic face region is placed to the source image to make Deepfake videos, inconsistent facial landmarks are always hard to eliminate. Based on this, utilizes the spatial relationships of landmark information and develops a head pose-based detector to distinguish between real and fake videos. To further consider landmark information in the temporal domain, proposes a temporal rotation angle and a new strategy for selecting facial landmarks. However, none of these methods apply deep neural networks but construct features manually by traditional methods. To this end, applies deep neural networks and shows they are effective in implicitly capturing the relationship between different landmarks in both the spatial and temporal domains.

However, these landmark-based detection efforts are still at the stage of using only landmark information and do not consider the fusion of information from different modalities, e.g., pixel-level RGB information with landmark information. Besides, most spatial- and frequency-based models ignore temporal inconsistencies in deepfake videos, and most temporal-based methods cannot model single-frame images adequately. However, intuitively, fusing the features of these different modalities can fully leverage all the information in a deepfake video.
In recent years, many researchers pay attention to studying how to combine multi-domains features for forgery detection [52, 38, 53]. Aayushi et al. [52] propose a cross-stitched network with two parallel branches that contain spatial- and frequency-domains information. The cross-stitch module is inserted between the two branches to share representations from other domains. Another similar approach is proposed by Wang et al. [54], which adopts a query-key-value (QKV) mechanism in self-attention [23] to integrate spatial- and frequency-domains features. However, they do not consider temporal-domain features, which have been widely proven crucial for deepfake detection. Besides, [53] considers spatial-temporal-frequency domains together but does not consider how to integrate them effectively.

3 Proposed Method

Our proposed method within the scope of this paper can be roughly categorized into the frame-level and video-level features extraction (Sec. 3.1, Sec. 3.2), and loss function (Sec. 3.3). The overall architecture of our framework can be seen in Fig. 2.

3.1 Frame Level Features Extraction

In this section, we will discuss the extraction of the spatial-domain features [3.1.1], frequency-domain features [3.1.2], the spatial-frequency fusion [3.1.3], landmark features [3.1.4], and multimodal feature fusion [3.1.5] respectively.

3.1.1 Spatial Features Extraction

Given the pre-processed fragment $X \in \mathbb{R}^{C \times H \times W}$, where $C$, $H$, and $W$ denote 3, 320, and 320 respectively. We utilize Xception [55] for spatial-domain features extraction since it is enabled to achieve relatively good results for many deepfake databases [56, 57].
In the original Xception setting, the shape of the output feature map at the final block is $R^{2048 \times 7 \times 7}$. However, [19] has shown that shallow and local texture information is more crucial than high-level semantic information for face forgery detection. So we redesign the configuration of the architecture and our improved Xception architecture with multi-scale fusion can be seen in Fig.3, where the feature maps of block 2 and block 5 will be fused in the block 6 to extract the multi-scale features.

Specifically, compared with the original Xception, we use the feature map $f_{\text{block4}}$ as the final output of the Xception: $X_{\text{xcep}} \rightarrow f_{\text{block4}}$. The output shape of $f_{\text{block4}}$ is $R^{1024 \times 10 \times 10}$, which is half as small as the original output shape of Xception. Also, the $f_{\text{block4}}$ will be then passed through a upsample module: $f_{\text{block4}} \xrightarrow{\text{upsample}} f_{\text{up}}$. After the upsample operation, the shape of the feature map $f_{\text{up}}$ is $R^{256 \times 40 \times 40}$, the same of that of $f_{\text{block2}}$.

Then the two feature maps $f_{\text{up}}, f_{\text{block2}}$ will be then fused and passed through a convolution layer with the kernel size of 1 to obtain the multi-scale features:

$$X_s = \text{Conv}_{1 \times 1}(f_{\text{up}} \parallel f_{\text{block2}}),$$

(1)

where $X_s \in R^{512 \times 40 \times 40}$, whose channel size (512) is 4 times smaller than that of the original Xception (2048). Also, in this paper, $\parallel$ donates the concatenate operation.

### 3.1.2 Frequency Features Extraction

Since the compression of the image will heavily affect the spatial domain information but the information in the frequency domain can remain, we fuse the features from both spatial and frequency domains to learn a more robust representation for detection. To transform the input $X$ from the spatial domain to the frequency domain, the Discrete Cosine Transform (DCT) [58] is adopted in this paper. DCT is one of the most common transformations in image compression techniques. Also, [52] has indicated that DCT is more effective than other transformations such as Fast Fourier Transform (FFT) to interpret the features in the frequency domain. The formula for 2D DCT is:

$$\mathcal{D} (u, v) = C (u) C (v) \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f (i, j) \cos \left[ \frac{(i + 0.5) \pi N}{N} u \right] \cos \left[ \frac{(j + 0.5) \pi N}{N} v \right],$$

(2)

$$c (u) = \begin{cases} \frac{1}{\sqrt{N}}, & \text{if } u = 0. \\ \frac{1}{\sqrt{2}}, & \text{else } u > 0. \end{cases}$$
After DCT, high-frequency information is reflected in the lower right corner of the image, while low-frequency information is reflected in the upper left corner. To remove the redundant information in the frequency domain and reduce noise interference, we use a binary mask \( M \in \mathbb{R}^{320 \times 320} \) with direction:

\[
M : M_{ij} = \begin{cases} 
1, & \text{if } \text{low} < i + j < \text{up}, \\
0, & \text{else} 
\end{cases},
\]  

(3)

where \( \text{low} \) stands for lower cutoff frequency, and \( \text{up} \) stands for upper cutoff frequency.

Previous researches\cite{18, 38} have shown that the spectrum of face images is most efficiently extracted with filters in three different bands: low, medium, and high. To prevent degradation of the model, we use a residual structure to add the original frequency information. We add a corresponding learnable mask to each band \( M_s \in \mathbb{R}^{320 \times 320} \) to improve the mask’s capacity to extract information:

\[
y_f = \mathcal{D}(x) \odot \begin{pmatrix} M_{low} + \sigma(M_{s}^l) \\
M_{mid} + \sigma(M_{s}^m) \\
M_{high} + \sigma(M_{s}^h) \\
M_{all} + \sigma(M_{s}^a) \end{pmatrix},
\]  

(4)

where \( y_f \) denotes features after frequency domain transformation, \( \mathcal{D} \) represents DCT, \( \odot \) denotes the element wise dot-product, \( M_{all} \) is an all-pass mask \( \{M_{all} \mid M_{ij} = 1\} \) and \( \sigma \) denotes sigmoid function. We then invert the frequency domain features back to the spatial domain and then utilize the multi-scale Xception to extract high-dimensional information.

To be noted, since the shape of \( y_f \) is \( \mathbb{R}^{12 \times 320 \times 320} \), whose channel size (12) is four times that of \( X \) (3), we also apply a convolution layer with a kernel size of 1 to adjust the channel size from 12 to 3. Besides, the Xception for the frequency-domain feature is the same architecture as the spatial domain, but the inter-parameters are not shared:

\[
Y_f = \text{Xception}(\text{Conv}_{1 \times 1} (\mathcal{D}^{-1}(y_f))).
\]  

(5)

### 3.1.3 Spatial-Frequency-Fusion Block

Given the spatial domain features \( X_s \in \mathbb{R}^{512 \times 40 \times 40} \) and the frequency domain features \( Y_f \in \mathbb{R}^{512 \times 40 \times 40} \), the simplest way to fuse the modalities of two different domains is to concatenate or sum the corresponding elements. Although this linear operation is simple, it often fails to achieve good results. Therefore, inspired by \cite{38}, we apply the query-key-value (QKV) mechanism \cite{25} in Spatial-Frequency-Fusion (SFF) block to fuse the two domain features. Due to the noise in frequency information, such as the human face’s hair and eyelashes, which may be concentrated in the high-frequency region, we view the frequency domain module as an auxiliary modality and the spatial domain as the main modality. Specifically, after a \( 1 \times 1 \) convolution operation: \( \text{Conv}(X_s^i) \), the spatial-domain features can be embedded as query (Q), and the frequency domain features can be embedded as key (K) and value (V) by the same method. Then, the unified representation \( A_i \) for spatial and frequency domain can be obtained by the QKV attention formula:

\[
A_i = \text{SoftMax} \left( \frac{QK^T}{\sqrt{dk}} \right) V + X_s^i,
\]  

(6)

where \( A_i \in \mathbb{R}^{512 \times 40 \times 40} \) integrates the forgery information for both spatial and frequency domain, \( dk \) is equal to \( 512 \times 40 \times 40 \), and \( \text{Conv} \) denotes the \( 1 \times 1 \) convolution operation.

### 3.1.4 Landmark Geometric Feature Extraction

following \cite{21}, by utilizing Dlib face detector \cite{59}, we can obtain 68 landmarks for a face. Each landmark point at \( t \) moment can be represented by \( l_t^i = [x_t^i, y_t^i] \), where \( i \) is from 1 to 68 and represents different point of the face. And \( x_t^i \) and \( y_t^i \) can be formulated by \( x_t^i = [x^1, x^2, ..., x^{68}] \) and \( y_t^i = [y^1, y^2, ..., y^{68}] \) respectively.

### 3.1.5 Multimodal Feature Fusion

The multimodal fusion approach we propose involves learning the joint representations from multimodal inputs at the frame level, specifically the landmark modality \( l_t \) and spatial-frequency domain modality \( A_i \). The landmark modality
provides the shape, contour, and position of face important points, whereas the spatial-frequency-domain modality combines feature maps in the spatial and frequency domain.

To fuse the geometric information of the landmark with the feature maps in the spatial-frequency domain while keeping the landmark information as complete as possible, we concatenate the spatial-frequency modality $A_t$ with the landmark modality $l_t$. We then use a learnable weight matrix $M_t$ to extract the important information from the combined modality.

$$F_t = (A_t \parallel l_t) \circ M_t,$$

where $F_t \in \mathbb{R}^{512 \times 40 \times 40}$ will be then pass through a max pooling layer to obtain the final combined representation of multimodal inputs at the frame level.

### 3.2 Video Level Features Extraction

The above theory is only about operations at the frame level, so the above process does not work when considering temporal domain information. In this section, we will first give our strategy for random sampling Sec. 3.2.1 and then give details about GNN we use in this paper in later subsections.

#### 3.2.1 Video Sampling Strategy

Because of the expensive cost of sampling too many frames for training (e.g. 270 frames), [22] has indicated that 32 may be the trade-off between performance and computing cost. So, in our paper, we only select the 32 frames of each video for training, validating and testing. Furthermore, to obtain more diverse training data, we shuffle all the frames in the video and then sort the first 8 frames, thus ensuring that we sample 8 randomly spaced frames in sequence. By repeating the above procedure 4 times, we can use 32 frames in one video for training.

#### 3.2.2 Transformation from Frame-Level to Video-Level

To transform the input $X$ from frame-level to video-level, an intuitive idea is to calculate the mean value for the input frames sequence of a video. For instance, given the unified features sequence $\{x_1, x_2, ..., x_n\}$, the mean value of $n$ frames (the video-level representation) should be $\frac{1}{n} \sum_{i=1}^{n} x_i$. However, this approach cannot capture the whole video well due to several limitations: (i). It assumes that each frame has the same weight for deepfake video detection (ii). It cannot learn the interactive information of different frames’ unified features. Another mainstream method for video-level feature extraction is to utilize the CNN-RNN pipeline. However, RNN-based models always suffer from the problem of long term dependencies [60] and parallel computing [23]. Furthermore, because we use a random sampling strategy (see Sec. 3.2.1), the intervals between the frames we sampled are not equal. However, RNN will view the frames to be equally spaced, which is not in line with reality.

Based on this, we apply GNN with attention to learning the contribution for each frame and the interactive information between adjacent frames. Also, we model the feature differences and temporal intervals between frames by edges in the graph structure. After the operation of GNN, the video-level representation can be obtained by aggregating the features of each node and regarding the graph-level representation as the output. $\{x_1, x_2, ..., x_n\} \xrightarrow{\text{GNN}} \{x'_1, x'_2, ..., x'_n\} \xrightarrow{\text{Pool}} \text{Output}$. Next we will discuss how to do the above operation with GNN.

#### 3.2.3 Preliminaries of Graph Neural Network

Specifically, different frames in the same video can be represented by a graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a node-set and $\mathcal{E}$ is an edge set. Following the definitions of the previous GNNs [61], the features of a node $v$ are represented by $x_v$ and the features of an edge $(u, v)$ are represented by $e_{uv}$. Taking node features, edge features and the graph structure as inputs, a GNN learns the representation vectors of the nodes and the entire graph, where the representation vector of a node $v$ is denoted by $h_v$, and the representation vector of the entire graph is denoted by $h_G$. A GNN iteratively updates a node’s representation vector by messages passing mechanism [62] to aggregate the messages from the node’s neighbors. Given a node $v$, its representation vector $h_v^{(k)}$ at the $k$-th iteration is formalized by:

$$a_v^{(k)} = AGGREGATE^{(k)}(\{(h_v^{(k-1)}, a_u^{(k-1)}, x_{uv} | u \in \mathcal{N}(v))\}),$$

$$h_v^{(k)} = COMBINE^{(k)}(h_v^{(k-1)}, a_v^{(k)}),$$

where $\mathcal{N}(v)$ is the set of neighbors of node $v$, $AGGREGATE^{(k)}$ is the aggregation function for aggregating messages from a node’s neighborhood, and $COMBINE^{(k)}$ is the update function for updating the node representation.
**READOUT** function is introduced to integrate the nodes’ representation vectors at the final iteration to gain the graph’s representation vector $h_G$, which is formalized as:

$$h_G = \text{READOUT}(h_v^K | v \in V),$$ (9)

where $K$ is the number of iterations. In most cases, **READOUT** is a permutation invariant pooling function, such as summation and maximization.

### 3.2.4 Framework of Graph Attention Network

In this paper, given the unified features sequence of each frame of a video $\{x_1, x_2, ..., x_n\}$, we regard them as $n$ different nodes and then apply Graph Attention Network (GAT) [25] to aggregate the messages from the node’s neighbors to obtain the interactive representation. The output of GAT will be then passed through several modules, followed by a residual module and a $3 \times 3$ convolutional layer to further extract the deeper semantic features for classification.

### 3.2.5 Attention Mechanism

Original GNNs cannot calculate the contribution of each node to the final result. GAT, on the other hand, is enabled to automatically specify different weights to different nodes in a neighborhood without requiring any expensive matrix operations.

In general, given a node $v_i$, its representation vector after updating $h_i'$ is formalized by:

$$h_i' = \alpha_{i,i} \Theta h_i + \sum_{j \in \mathcal{N}(i)} \alpha_{i,j} \Theta h_j,$$ (10)

where $\Theta$ is a learnable weights matrix. $h_i \in \mathbb{R}^{512}$ and $h_j \in \mathbb{R}^{512}$ are its and its neighbors’ representation vector before updating. $\alpha_{i,j}$ is the attention coefficients (score) for different nodes, which is one of the most core parameters in GAT.

For the graph that has multi-dimensional edge features $e_{i,j}$, the attention coefficients $\alpha_{i,j}$ are computed as:

$$\alpha_{i,j} = \frac{\exp \left( \text{ReLU} \left( a^T \left[ \Theta h_i \parallel \Theta h_j \parallel \Theta_e e_{i,j} \right] \right) \right)}{\sum_{k \in \mathcal{N}(i) \cup \{i\}} \exp \left( \text{ReLU} \left( a^T \left[ \Theta h_i \parallel \Theta h_k \parallel \Theta_e e_{i,k} \right] \right) \right)},$$ (11)

where $e_{i,j}$ represents the edge features between node $i$ and node $j$, and we will discuss it in detail at next section.

### 3.2.6 Temporal Feature Extraction

Since each node of the graph in this paper is essentially a different frame of a video, then the connection between two nodes, the edge, can be seen as a correlation between two frames. Here, we encode two important information into the graph as edge features, namely, the feature difference between two frames and the time interval. Specifically, after constructing facial feature vectors with landmarks, the edge feature $e_{i,j} \in \mathbb{R}^{512}$ for node $i$ and node $j$ can be expressed by:

$$e_{i,j} = \frac{l_{t_i} - l_{t_j}}{i - j},$$ (12)

where $l_{t_i} - l_{t_j}$ represents the first-order difference between the i-th and j-th frames, and $i - j$ donates the time interval between two frames. This equation quantitatively describes the difference in temporal domain between any two frames of facial landmarks.

Finally, node features $h_v$ initialized as $x_v$ are updated by GATConv layer-by-layer, and the output is denoted as $h_v^{(K)}$. The graph-level representation $h_G$ is obtained via the graph mean pooling and the graph max pooling over the node and edge representations of all nodes. Then, the two graph-level features will be concatenated to obtain the final output:

$$Y_G = [\text{MeanPool}_G(h_G) \parallel \text{MaxPool}_G(h_G)].$$ (13)
3.2.7 Classification

To extract the deeper semantic information, we first apply several $3 \times 3$ convolutional layers for $Y_G$. Then, we normalize the output and pass through a Linear layer. Last, the final predicted label by our model can be calculated by SoftMax function:

$$\hat{y} = \text{SoftMax}(\text{LayerNorm}(\text{Linear}(\text{Conv}3 \times 3(Y_G)))) .$$  \hspace{1cm} (14)

3.3 Loss Function and Parameter optimization

The predictions of the model output are defined as $L$. For a set of videos, the prediction result is also a sequence. Because the image deepfake detection is an image classification task, the cross-entropy loss function is always effective:

$$l(x, y) = L = \{l_1, \ldots , l_N\}^T,$$

$$\mathcal{L}_c(x, y) = -\log\left(\frac{\exp(x[y])}{\sum_j \exp(x[j])}\right) = -x[y] + \log \left(\sum_j \exp(x[j])\right),$$  \hspace{1cm} (15)

where $y$ is set to 1 if the face image has been manipulated, otherwise it is set to 0, and $j$ is the number of classes which is equal to 2.

The parameters of the network are updated via back-propagation and the overall training procedure for LEM-GNN is provided in Algorithm 1.

**Algorithm 1:** Training Procedure for LEM-GNN

| Input: face frames training set $\{x_1, x_2, ..., x_n\}$ |
| Output: Trained model parameters $\alpha = \{w_1, w_2, b_1, b_2, \ldots\}$ |  
1 Randomly initialize $\alpha$;  
2 for each batch from training set do  
3 | Obtain the multi-scale spatial feature $X_s$ using Eq.1;  
4 | Obtain a frequency set $D(x)$ by DCT using Eq.2;  
5 | for filter in filters set do  
6 | | Calculate and fuse features in different frequency bands to get $y_f$ using Eq.3-4;  
7 | end  
8 | Obtain the frequency-domain feature $Y_f$ using Eq.5; Utilize attention mechanism to fuse and obtain the spatial-frequency-domain feature $A_i$ by Eq.6;  
9 | Build a graph for each video and calculate the attention weights matrix $\alpha_{i,j}$ between two nodes $i, j$ using Eq.11;  
10 | Calculate the edge feature $e_{i,j}$ between two nodes $i, j$ using Eq.12;  
11 | Obtain the fused representation $h'_i$ after updating using Eq.10;  
12 | Calculate the prediction $\hat{y}$ using Eq.14;  
13 | Calculate the prediction loss $\mathcal{L}_c$ using Eq.15;  
14 | Update parameters $\alpha$ according to the gradient $\mathcal{L}_c$;  
15 end  
16 Return $\alpha$;  

4 Experiments

In this section, we firstly declare the experiment settings and the training details. Then we will perform two kinds of experiments: (i) train on one specific manipulation type and compare our model with baseline methods (ii) train on four manipulation types together and compare our model with previous detection methods.
Table 1: Comparison our LEM-GNN with other baseline methods on FF++ dataset by using different metric.

| Metric          | AUC (Higher is better) | ACC (%) (Higher is better) | EER (Lower is better) |
|-----------------|------------------------|----------------------------|-----------------------|
| Dataset         | FF++ High Quality (c23)| FF++ Low Quality (c40)     |                       |
| HeadPose        | 0.978                  | 0.989                      | 0.987                 |
| FDFFClassifier  | 0.981                  | 0.990                      | 0.988                 |
| Xception        | 0.993                  | 0.995                      | 0.977                 |
| MesoNet         | 0.989                  | 0.990                      | 0.976                 |
| Meso-Incep      | 0.984                  | 0.994                      | 0.979                 |
| CapsuleNet      | 0.987                  | 0.994                      | 0.979                 |
| FiNet           | 0.998                  | 0.999                      | 0.987                 |

4.1 Experiment Setting

4.1.1 Datasets

During the research process of deepfake detection, several challenging datasets have been released. In this paper, we mainly adopt the FaceForensics++ (FF++) dataset because it is one of the most typical deepfake datasets and has been widely adopted in the deepfake field. FF++ contains 1000 original videos and each video has three versions, namely the original version (raw), slightly-compressed version (c23), and heavily-compressed version (c40). We conduct all experiments for low (c40) and high (c23) compression levels like many other works do [19, 63, 64]. The dataset also contains four manipulated methods, including face-swapping and face-reenactment manipulation technology.

It is worth noting that the number of negative training samples in FF++ is 4 times larger than that of the positive. The unbalance problem will inevitably cause the classifier to be more inclined toward the negative class. Thus, to balance the number of positive and negative samples, we sample 16 times for positive training samples and only 4 for the negative in this work.

We also select Celeb-DF [57] dataset as one of the test datasets when doing generalization experiments. This dataset is a newly proposed dataset with high visual quality, which contains 5639 fake videos and 540 real videos. Celeb-DF also provides a benchmark that facilitates our evaluation. In our generalization experiment, following [65], we train our model on FF++, and evaluate it on Celeb-DF (unseen data).

4.1.2 Data Pre-processing

In pre-processing step, following [67], we apply MTCNN (Multi-Task Convolutional Neural Network) [68] to carry out face detection. It uses the idea of cascading to increase the accuracy of information acquisition, and is convenient to train and deploy [68]. The original video is sampled by OpenCV [69], and the video frame sequence \([frame_1, frame_2, ..., frame_n]\) is obtained every 2 frames. We adopt the library of Albumentations [70] to do data augmentation since it is a popular data augmentation tool and is very compatible with PyTorch [71]. In this paper, the following augmentations are considered: (i) BC: Brightness and Contrast changes (ii) HSV: Hue, Saturation and Value changes (iii) GB: Gaussian blur (iv) JPEG: JPEG compression with a random quality factor between 50 and 99. The RGB images of the extracted face regions are resized to \(3 \times 320 \times 320\) to obtain the face sequence \([x_1, x_2, ..., x_n]\).
4.1.3 Parameters and Training Details

We split the dataset after pre-processing into three sets, namely training set, validation set, and test set. According to [56], we adopt a 720:140:140 dataset split, that is, 720 videos for training, 140 for evaluating, and 140 for testing. For all databases, we applied normalization with mean = (0.485, 0.456, 0.406) and standard deviation = (0.229, 0.224, 0.225) in the ImageNet Large Scale Visual Recognition Challenge [23]. In training procedure, for Xception backbone, we load the model weights that has already pre-trained in ImageNet. The hidden size of LSTM in our model is 256, and the number of layers is 3. The GAT layer number is also 3. We adopt mean graph pooling as a READOUT function over the node and edge representations of all nodes. The optimizer we adopt is AdamW [73]. The initial learning rate we set is 0.0002. The weight decay of the optimizer is 0.003. The batch size in our training procedure is 4. The model finishes training for up to 30 epochs to convergence.

4.1.4 Evaluation Metrics

We apply the Accuracy score (ACC), Area Under the RoC Curve (AUC) and Equal error rate (EER) as our evaluation metrics, which are commonly used in the field of deepfake detection [74, 75, 65, 76, 77].

• AUC = \( \sum_{i=1}^{n} (AUC_i) \), where \( n \) represents the total number of selected frames in a video. Besides, \( AUC_i \) means the AUC value for i-th frame.

• ACC = \( \frac{TP+TN}{TP+TN+FP+FN} \), where \( TP, TN, FP \) and \( FN \) are true positive, true negative, false positive, and false negative, respectively.

• EER: the value when the false acceptance rate (FAR) is equal to the false rejection rate (FRR), where \( FAR = \frac{FP}{TP+TN} \) and \( FRR = \frac{FN}{TP+FN} \).

4.2 Comparison with baseline methods

Overall, we have the best testing results than ten baselines for AUC, ACC, and EER, showing that our model can effectively learn distinctive forgery features and achieve excellent performance on four manipulated datasets with both low and high compression levels. Specifically, table 1 compares our model with eight baseline models on three metrics. For FF++ (c40), the average AUC of four manipulated datasets in our method ranges from 0.920 to 0.996 with an average value of 0.966, which is 2.2% higher than the best frequency-based baseline (F3Net) of 0.942, 5.0% higher than the best spatial domain baseline (CViT) of 0.916 and 11.1% higher than the temporal domain baseline (CNN-RNN).

4.3 Comparison with previous methods

In addition to the comparison with baselines, we also compare the model with previous methods on FF++ (c23) and FF++ (c40) benchmarks in recent years (see Table 2). Since most previous models are not open-source, we cite the experimental results directly from the original paper or cite the results from [78]. As can be seen in table 2 our proposed model is able to achieve better experimental results compared to many previous methods by using less training data.

| Methods          | Frames | AUC [%] | AUC [%] | AUC [%] | AUC [%] |
|------------------|--------|---------|---------|---------|---------|
| AUC              |        |         |         |         |         |
| FF++ (c23)       |        |         |         |         |         |
| FF++ (c40)       |        |         |         |         |         |
| Ours             |        |         |         |         |         |

Table 2: Experiment results and comparison with other methods.
4.4 Results

We demonstrate visualization experiments, ablation studies, and robustness experiments in this section. (i) For visualization experiments, we first visualize the embedding of the last layer in a high-dimensional vector space to see if our model can distinguish forgery videos. Then, we visualize the detection results of our model to visualize the response of our model to different regions of the input image. (ii) For the ablation study, we discuss the contribution of each module of our model to see which module contributes more or less. (iii) For robustness experiments, we verify the generalization ability of our model for unseen datasets and perturbations.

4.4.1 Visualization of the embedding

In deep learning, the information learned by neural networks appears to the naked eye as a countless number of numbers and can be difficult for us to understand. Therefore, visualizing a low-dimensional dense vector representation (Embedding) can help us better determine whether the neural networks have learned the distinguished knowledge from the training data.

Specifically, in our experiments, we first use MesoNet [14] and LEM-GNN to do forward propagation in the FF++ (c40) testing set and then obtain the feature map of the last layer before passing through the fully connected layer. We
demonstrate the training procedure of t-SNE from iteration 100 to iteration 1000. (from left to right). The visualized result is in Fig.5. In this figure, we can see that our model can learn a more distinctive representation than MesoNet, and it can well isolate the real and fake videos in the vector space.

4.4.2 Visualization of the Detection Results

To see the ability of our model to locate forgery regions, we visualize the detection results of the model (see Fig.4). In our experiments, we apply Grad-CAM [88] to visualize the regions of the response of our model. From Fig.4, we can see that our model does learn the more meaningful features of the face and finds most of the forgery regions.

4.4.3 Ablation Study

In this subsection, we will verify the contribution of each part of our model to the results separately. Generally, there are five modules in our proposed model. They are Frequency Module (FM), Spatial-Frequency Fusion module (SFF), Landmark Module (LM), Graph Attention Module (GAM), and Temporal Module (TM). Results in Table.3 demonstrate that all five parts contribute positively to the final prediction, and removing any of them would result in an unexpected drop in the final prediction. Also, we find that there is no significant difference in the contribution of each module to the results.

In addition to the above five main components, we also explore the ablation of the relevant settings of GAT module. In table.4, we explore the effectiveness of layers of GNN. The results show that constantly adding layers of GNN does not bring a continuous improvement in the performance of the model.

4.4.4 Robustness for cross-datasets

To verify the robustness of our model, we train the model on the FF++ (c23) dataset and test it on the unseen dataset (Celeb-DF). The results from Table.5 confirm the generalization ability of our model for cross-datasets detection outweighs many other methods. It is worth noting that although the diversity of training data plays an important role in the robustness of the model, we only sample 32 frames per video when training the FF++ dataset. Our training data are relatively small, but our model is still enabled to achieve the comparable performance, indicating that our model is indeed able to learn more general features.

4.4.5 Robustness for unseen perturbations

To confirm that our model is more robust to these noises than other approaches, we add various perturbations to the original images separately to test the generalization ability of the model. Specifically, we compress the original image, add Gaussian noise and blur, change the contrast, and drop the pixel values of some regions randomly, respectively. Fig.6 compares our model with other methods based on the spatial (Xception), frequency (F3Net), and temporal domain (CNN-RNN). From this Figure, we can find that the spatial-domain approach is the most sensitive to unseen perturbations and the frequency-domain method is the opposite. Since the information of temporal, spatial, and frequency domains are all regarded into our model, it has the best performance compared to other models when exposed to unknown perturbations.
Table 5: Cross-dataset experiment results and comparison with other methods. Most of the results are directly cited from [19, 52].

| Methods                  | FF++   | Celeb-DF |
|--------------------------|--------|----------|
| Two-stream [89]          | 0.701  | 0.538    |
| Celeb-DF [52]            | 0.647  | 0.548    |
| Meso4 [14]               | 0.830  | 0.548    |
| HeadPose [5]             | 0.473  | 0.546    |
| FWA-DF [37]              | 0.801  | 0.569    |
| VA-MLP [70]              | 0.664  | 0.550    |
| Xception-raw [56]        | 0.997  | 0.482    |
| Xception-c23 [56]        | 0.997  | 0.653    |
| Xception-c64 [56]        | 0.996  | 0.655    |
| Multi-task [50]          | 0.763  | 0.543    |
| Capsule [1]              | 0.966  | 0.577    |
| DSP-FWA [37]             | 0.930  | 0.646    |
| Face-XRay [81]           | 0.991  | 0.742    |
| F3Net [18]               | 0.981  | 0.652    |
| Two-Branch [63]          | 0.932  | 0.734    |
| Efficient-B6 [66]        | 0.997  | 0.643    |
| SPSL [19]                | 0.969  | 0.769    |
| MD-CSND [67]             | 0.995  | 0.688    |
| Ours                     | 0.997  | 0.738    |

Figure 6: Robustness against unknown disturbances. The term "average" refers to the average of all corruptions for each severity level. The idea of this figure mainly comes from [90, 22].

5 Conclusion and Future works

Because of the development of facial manipulation techniques, face forgery detection has gotten a lot of attention in digital media forensics. Most current methods for deepfake detection suffer from several limitations. Most of the methods based on spatial, frequency, or temporal domains are sensitive to external perturbations such as illumination. Moreover, while many studies explore the spatial, frequency, and temporal domains separately, few works have considered how to effectively fuse the features of these different modalities. In addition, although landmark-based methods are relatively robust, landmark neglects much of the detailed information of a forgery image.

Therefore, in this paper, we propose a multimodal fusion framework that combines the spatial, frequency, temporal, and landmark features simultaneously. Moreover, to better model temporal features, we propose a random sampling strategy and introduce GAT to explicitly model temporal inconsistencies between different frames. Extensive experiments have shown that these features from different domains can all contribute positively to the detection performance. Also, our
model can achieve the SOTA and performance better than many previous methods by using relatively small training data. Furthermore, we conduct visualization experiments, robustness experiments, and ablation studies to explore the contribution of each module of our framework.

However, there are still some limitations. First, How to fuse the features of different modalities more effectively is still a problem to be solved. Second, one of our motivations is to allow the model to learn more robust and discriminative features. However, only combining information from different modalities may not be sufficient for generalizability issues. For example, our results on cross-datasets detection are still not very satisfactory. So we think a more desirable way is to add some prior knowledge to the model to assist the learning process of the model, which is our next step.

Acknowledgments

This work was supported by the Technical Research Program of Ministry of Public Security [2020JSYJC25]; Open Project of Key Laboratory of Forensic Science of Ministry of Justice [KF202014]; Innovative Talents Support Program of Liaoning Province[LNCX202007] and the Young scientific and technological talents breeding project [JYT2020130].

References

[1] iperov. Deepfacelab. https://github.com/iperov/DeepFaceLab 2019.
[2] CGTN America. Face-swapping app “zao” amazes and alarms with deepfake capabilities. https://www.youtube.com/watch?v=LNYY5r63Ac 2019.
[3] Avondale Kendja. The dangers of deepfakes. https://www.garbo.io/blog/deepfakes 2021.
[4] Yuezun Li, Ming-Ching Chang, and Siwei Lyu. In ictu oculi: Exposing ai created fake videos by detecting eye blinking. In 2018 IEEE International Workshop on Information Forensics and Security (WIFS), pages 1–7. IEEE, 2018.
[5] Xin Yang, Yuezun Li, and Siwei Lyu. Exposing deep fakes using inconsistent head poses. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 8261–8265. IEEE, 2019.
[6] Tiago Carvalho, Hany Farid, and Eric R Kee. Exposing photo manipulation from user-guided 3d lighting analysis. In Media Watermarking, Security, and Forensics 2015, volume 9409, page 940902. SPIE, 2015.
[7] Bo Peng, Wei Wang, Jing Dong, and Tieniu Tan. Optimized 3d lighting environment estimation for image forgery detection. IEEE Transactions on Information Forensics and Security, 12(2):479–494, 2016.
[8] Shruti Agarwal, Hany Farid, Ohad Fried, and Maneesh Agrawala. Detecting deep-fake videos from phoneme-viseme mismatches. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pages 660–661, 2020.
[9] Yonghyun Jeong, Jongwon Choi, Doyeon Kim, Sehyeon Park, Minki Hong, Changhyun Park, Seungjai Min, and Youngjune Gwon. Dofnet: Depth of field difference learning for detecting image forgery. In Proceedings of the Asian Conference on Computer Vision, 2020.
[10] Steven Fernandes, Sunny Raj, Eddy Ortiz, Iustina Vintila, Margaret Salter, Gordana Urosevic, and Sumit Jha. Predicting heart rate variations of deepfake videos using neural ode. In Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, pages 0–0, 2019.
[11] Umur Aybars Ciftci, Ilke Demir, and Lijun Yin. Fakercatcher: Detection of synthetic portrait videos using biological signals. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020.
[12] Hua Qi, Qing Guo, Felix Juefei-Xu, Xiaofei Xie, Lei Ma, Wei Feng, Yang Liu, and Jianjun Zhao. Deep rhythm: Exposing deepfakes with attentional visual heartbeat rhythms. In Proceedings of the 28th ACM International Conference on Multimedia, pages 4318–4327, 2020.
[13] Huaxiao Mo, Bolin Chen, and Weiqi Luo. Fake faces identification via convolutional neural network. In Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security, pages 43–47, 2018.
[14] Darius Afchar, Vincent Nozick, Junichi Yamagishi, and Isao Echizen. Mesonet: a compact facial video forgery detection network. In 2018 IEEE International Workshop on Information Forensics and Security (WIFS), pages 1–7. IEEE, 2018.
[15] Huy H Nguyen, Junichi Yamagishi, and Isao Echizen. Use of a capsule network to detect fake images and videos. arXiv preprint arXiv:1910.12467, 2019.
[16] David Güera and Edward J Delp. Deepfake video detection using recurrent neural networks. In 2018 15th IEEE international conference on advanced video and signal based surveillance (AVSS), pages 1–6. IEEE, 2018.

[17] Ekraam Sabir, Jiaxin Cheng, Ayush Jaiswal, Wael AbdAlmageed, Iacopo Masi, and Prem Natarajan. Recurrent convolutional strategies for face manipulation detection in videos. Interfaces (GUI), 3(1):80–87, 2019.

[18] Yuyang Qian, Guojun Yin, Lu Sheng, Zixuan Chen, and Jing Shao. Thinking in frequency: Face forgery detection by mining frequency-aware clues. In European Conference on Computer Vision, pages 86–103. Springer, 2020.

[19] Honggu Liu, Xiaodan Li, Wenbo Zhou, Yuefeng Chen, Yuan He, Hui Xue, Weiming Zhang, and Nenghai Yu. Spatial-phase shallow learning: rethinking face forgery detection in frequency domain. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 772–781, 2021.

[20] Brian Dolhansky, Russ Howes, Ben Pflaum, Nicole Baram, and Cristian Canton Ferrer. The deepfake detection challenge (dfdc) preview dataset. arXiv preprint arXiv:1910.08854, 2019.

[21] Zekun Sun, Yujie Han, Zeyu Hua, Na Ruan, and Weijia Jia. Improving the efficiency and robustness of deepfakes detection through precise geometric features. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 3609–3618, 2021.

[22] Yinglin Zheng, Jianmin Bao, Dong Chen, Ming Zeng, and Fang Wen. Exploring temporal coherence for more general video face forgery detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 15044–15054, 2021.

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.

[24] Anthony Gillioz, Jacky Casas, Elena Mugellini, and Omar Abou Khaled. Overview of the transformer-based models for nlp tasks. In 2020 15th Conference on Computer Science and Information Systems (FedCSIS), pages 179–183. IEEE, 2020.

[25] Chaitanya Joshi. Transformers are graph neural networks. The Gradient, 2020.

[26] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. arXiv preprint arXiv:1710.10903, 2017.

[27] Jie Gui, Zhenan Sun, Yonggang Wen, Dacheng Tao, and Jieping Ye. A review on generative adversarial networks: Algorithms, theory, and applications. arXiv preprint arXiv:2001.06937, 2020.

[28] Jianxin Wu. Introduction to convolutional neural networks. National Key Lab for Novel Software Technology. Nanjing University. China, 5(23):495, 2017.

[29] Larry R Medsker and LC Jain. Recurrent neural networks. Design and Applications, 5:64–67, 2001.

[30] Guangquan Lu, Xishun Zhao, Jian Yin, Weimei Yang, and Bo Li. Multi-task learning using variational auto-encoder for sentiment classification. Pattern Recognition Letters, 132:115–122, 2020.

[31] Iryna Korshunova, Wenzhe Shi, Joni Dambre, and Lucas Theis. Fast face-swap using convolutional neural networks. In Proceedings of the IEEE international conference on computer vision, pages 3677–3685, 2017.

[32] Mika Westerlund. The emergence of deepfake technology: A review. Technology Innovation Management Review, 9(11), 2019.

[33] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2387–2395, 2016.

[34] Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. Advances in neural information processing systems, 28, 2015.

[35] Peisong He, Haoliang Li, and Hongxia Wang. Detection of fake images via the ensemble of deep representations from multi color spaces. In 2019 IEEE International Conference on Image Processing (ICIP), pages 2299–2303. IEEE, 2019.

[36] Falko Matern, Christian Riess, and Marc Stamminger. Exploiting visual artifacts to expose deepfakes and face manipulations. In 2019 IEEE Winter Applications of Computer Vision Workshops (WACVW), pages 83–92. IEEE, 2019.

[37] Yuezun Li and Siwei Lyu. Exposing deepfake videos by detecting face warping artifacts. arXiv preprint arXiv:1811.00656, 2018.

[38] Junke Wang, Zuxuan Wu, Jingjing Chen, and Yu-Gang Jiang. M2tr: Multi-modal multi-scale transformers for deepfake detection. arXiv preprint arXiv:2104.09770, 2021.
[39] Yuting Xu, Gengyun Jia, Huaibo Huang, Junxian Duan, and Ran He. Visual-semantic transformer for face forgery detection. In 2021 IEEE International Joint Conference on Biometrics (IJCB), pages 1–7. IEEE, 2021.

[40] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929, 2020.

[41] Davide Coccomini, Nicola Messina, Claudio Gennaro, and Fabrizio Falchi. Combining efficientnet and vision transformers for video deepfake detection. arXiv preprint arXiv:2107.02612, 2021.

[42] Young-Jin Heo, Young-Ju Choi, Young-Woon Lee, and Byung-Gyu Kim. Deepfake detection scheme based on vision transformer and distillation. arXiv preprint arXiv:2104.01353, 2021.

[43] Xu Zhang, Svebor Karaman, and Shih-Fu Chang. Detecting and simulating artifacts in gan fake images. international workshop on information forensics and security, 2019.

[44] Tarik Dzanic, Karan Shah, and Freddie D. Witherden. Fourier spectrum discrepancies in deep network generated images. arXiv: Image and Video Processing, 2019.

[45] Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, and Alexei A. Efros. Cnn-generated images are surprisingly easy to spot... for now. computer vision and pattern recognition, 2020.

[46] Ricard Durall, Margret Keuper, and Janis Keuper. Watch your up-convolution: Cnn based generative deep neural networks are failing to reproduce spectral distributions. computer vision and pattern recognition, 2020.

[47] José Augusto Stuchi, Marcus de Assis Angeloni, Rodrigo de Freitas Pereira, Levy Boccato, Guilherme Folego, Paulo V. S. Prado, and Romis Attux. Improving image classification with frequency domain layers for feature extraction. In International Workshop on Machine Learning for Signal Processing, 2017.

[48] Irene Amerini, Leonardo Galteri, Roberto Caldelli, and Alberto Del Bimbo. Deepfake video detection through optical flow based cnn. In Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, pages 0–0, 2019.

[49] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8934–8943, 2018.

[50] Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework. International journal of computer vision, 56(3):221–255, 2004.

[51] Meng Li, Beibei Liu, Yongjian Hu, Liepiao Zhang, and Shiqi Wang. Deepfake detection using robust spatial and temporal features from facial landmarks. In 2021 IEEE International Workshop on Biometrics and Forensics (IWBF), pages 1–6. IEEE, 2021.

[52] Aayushi Agarwal, Akshay Agarwal, Sayan Sinha, Mayank Vatsa, and Richa Singh. Md-csdnetwork: Multi-domain cross stitched network for deepfake detection. arXiv: Computer Vision and Pattern Recognition, 2021.

[53] Yongjian Hu, Hongjie Zhao, Zeqiong Yu, Beibei Liu, and Xiangyu Yu. Exposing deepfake videos with spatial, frequency and multi-scale temporal artifacts. In International Workshop on Digital Watermarking, pages 47–57. Springer, 2021.

[54] Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, and Alexei A. Efros. Cnn-generated images are surprisingly easy to spot... for now. 2019.

[55] François Chollet. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1251–1258, 2017.

[56] Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 1–11, 2019.

[57] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, and Siwei Lyu. Celeb-df: A large-scale challenging dataset for deepfake forensics. In Proceedings of the IEEE/CVF International Conference on Computer Vision and Pattern Recognition, pages 3207–3216, 2020.

[58] Nasir Ahmed, T_ Natarajan, and Kamisetty R Rao. Discrete cosine transform. IEEE transactions on Computers, 100(1):90–93, 1974.

[59] Davis E King. Dlib-ml: A machine learning toolkit. The Journal of Machine Learning Research, 10:1755–1758, 2009.

[60] Jingyu Zhao, Feiqing Huang, Jia Lv, Yanjie Duan, Zhen Qin, Guodong Li, and Guangjian Tian. Do rnn and lstm have long memory? In International Conference on Machine Learning, pages 11365–11375. PMLR, 2020.
[61] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[62] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.

[63] Iacopo Masi, Aditya Killekar, Royston Marian Mascarenhas, Shenoy Pratik Gurudatt, and Wael AbdAlmageed. Two-branch recurrent network for isolating deepfakes in videos. In *European Conference on Computer Vision*, pages 667–684. Springer, 2020.

[64] Huy H Nguyen, Junichi Yamagishi, and Isao Echizen. Capsule-forensics: Using capsule networks to detect forged images and videos. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2307–2311. IEEE, 2019.

[65] Hanqing Zhao, Wenbo Zhou, Dongdong Chen, Tianyi Wei, Weiming Zhang, and Nenghai Yu. Multi-attentional deepfake detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2185–2194, 2021.

[66] Binh M. Le and Simon S. Woo. Exploring the asynchronous of the frequency spectra of gan-generated facial images. 2022.

[67] Akash Kumar, Arnab Bhavsar, and Rajesh Verma. Detecting deepfakes with metric learning. In *2020 8th international workshop on biometrics and forensics (IWBFB)*, pages 1–6. IEEE, 2020.

[68] Kaipeng Zhang, Zhanpeng Zhang, Zhiheng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.

[69] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. O’Reilly Media, Inc., 2008.

[70] Alexander Buslaev, Vladimir I Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A Kalinin. Albumentations: fast and flexible image augmentations. *Information*, 11(2):125, 2020.

[71] pytorch. pytorch. [https://github.com/pytorch/pytorch](https://github.com/pytorch/pytorch) 2019.

[72] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[73] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. 2018.

[74] Javier Hernandez-Ortega, Ruben Tolosana, Julian Fierrez, and Aythami Morales. Deepfakeson-phys: Deepfakes detection based on heart rate estimation. *arXiv preprint arXiv:2010.00400*, 2020.

[75] Ruben Tolosana, Ruben Vera-Rodriguez, Julian Fierrez, Aythami Morales, and Javier Ortega-Garcia. Deepfakes and beyond: A survey of face manipulation and fake detection. *Information Fusion*, 64:131–148, 2020.

[76] Pavel Korshunov and Sébastien Marcel. Vulnerability assessment and detection of deepfake videos. In *2019 International Conference on Biometrics (ICB)*, pages 1–6. IEEE, 2019.

[77] Hoang Mark Nguyen and Reza Derakhshani. Eyebrow recognition for identifying deepfake videos. In *2020 International Conference of the Biometrics Special Interest Group (BIOSIG)*, pages 1–5. IEEE, 2020.

[78] Zehao Chen and Hua Yang. Attentive semantic exploring for manipulated face detection. In *ICASSP 2021–2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1985–1989. IEEE, 2021.

[79] Jessica Fridrich and Jan Kodovsky. Rich models for steganalysis of digital images. *IEEE Transactions on information Forensics and Security*, 7(3):868–882, 2012.

[80] Davide Cozzolino, Giovanni Poggi, and Luisa Verdoliva. Recasting residual-based local descriptors as convolutional neural networks: an application to image forgery detection. In *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, pages 159–164, 2017.

[81] Lingzhi Li, Jianmin Bao, Ting Zhang, Hao Yang, Dong Chen, Fang Wen, and Baining Guo. Face x-ray for more general face forgery detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5001–5010, 2020.

[82] Belhassen Bayar and Matthew C Stumm. A deep learning approach to universal image manipulation detection using a new convolutional layer. In *Proceedings of the 4th ACM workshop on information hiding and multimedia security*, pages 5–10, 2016.
[83] Nicolas Rahmouni, Vincent Nozick, Junichi Yamagishi, and Isao Echizen. Distinguishing computer graphics from natural images using convolution neural networks. In 2017 IEEE Workshop on Information Forensics and Security (WIFS), pages 1–6. IEEE, 2017.

[84] Teddy Surya Gunawan, Siti Amalina Mohammad Hanafiah, Mira Kartiwi, Nanang Ismail, Nor Farahidah Za’bah, and Anis Nurashikin Nordin. Development of photo forensics algorithm by detecting photoshop manipulation using error level analysis. Indonesian Journal of Electrical Engineering and Computer Science, 7(1):131–137, 2017.

[85] Mo Chen, Vahid Sedighi, Mehdi Boroumand, and Jessica Fridrich. Jpeg-phase-aware convolutional neural network for steganalysis of jpeg images. In Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security, pages 75–84, 2017.

[86] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In International conference on machine learning, pages 6105–6114. PMLR, 2019.

[87] Aayushi Agarwal, Akshay Agarwal, Sayan Sinha, Mayank Vatsa, and Richa Singh. Md-csdnetwork: Multi-domain cross stitched network for deepfake detection. In 2021 16th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2021), pages 1–8. IEEE, 2021.

[88] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE international conference on computer vision, pages 618–626, 2017.

[89] Peng Zhou, Xintong Han, Vlad I Morariu, and Larry S Davis. Two-stream neural networks for tampered face detection. In 2017 IEEE conference on computer vision and pattern recognition workshops (CVPRW), pages 1831–1839. IEEE, 2017.

[90] Alexandros Haliassos, Konstantinos Vougioukas, Stavros Petridis, and Maja Pantic. Lips don’t lie: A generalisable and robust approach to face forgery detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 5039–5049, 2021.