Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2024) 32: 166 – 182

doi:10.55730/1300-0632.4061

Motion magnification-inspired feature manipulation for deepfake detection

Aydamir MIRZAYEV, Hamdi DİBEKLİOĞLU*
Department of Computer Engineering, Faculty of Engineering, Bilkent University, Ankara, Turkiye

Received: 12.09.2023 • **Accepted/Published Online:** 25.01.2024 • **Final Version:** 07.02.2024

Abstract: Recent advances in deep learning, increased availability of large-scale datasets, and improvement of accelerated graphics processing units facilitated creation of an unprecedented amount of synthetically generated media content with impressive visual quality. Although such technology is used predominantly for entertainment, there is widespread practice of using deepfake technology for malevolent ends. This potential for malicious use necessitates the creation of detection methods capable of reliably distinguishing manipulated video content. In this work we aim to create a learning-based detection method for synthetically generated videos. To this end, we attempt to detect spatiotemporal inconsistencies by leveraging a learning-based magnification-inspired feature manipulation unit. Although there is existing literature on the use of motion magnification as a preprocessing step for deepfake detection, in our work, we aim to utilize learning-based magnification elements to develop an end-to-end deepfake detection model. In this research, we investigate different variations of feature manipulation networks, both with spatially constant and spatially varying amplification. To clarify, although the proposed model draws from existing literature on motion magnification, we do not perform motion magnification in our experiments but instead use the underlying architecture of such networks for feature enhancement. Our objective with this work is to take a step towards applying learnable motion manipulation in improving the target accuracy of a task at hand.

Key words: Deep learning, deepfake detection, motion magnification, computer vision, video classification, spatio-temporal analysis

1. Introduction

Reenactment technologies have been around for at least a decade now [1–3], with earlier approaches relying on 3D geometry and signal analysis to synthesize the target representation. The modern term deepfake, often used to refer to synthesized image and voice media, is frequently associated with the popularized, publicly available image synthesis method DeepFakes[1]. In the scope of this paper, we will refer to the technology itself as deepfakes while using capitalized (DeepFakes) when referring to the name of the method in question. Initially, face replacement technology has been primarily used to generate entertainment content. In recent years, however, deepfakes have been increasingly often utilized for malicious purposes such as financial scams, voter manipulation, damaging the image of politicians, celebrities, and regular citizens. The earliest attempts at detecting artificially synthesized media are primarily inspired by antispoofing technologies [4–6] that utilize diverse set of methods to prevent security breaches in biometric identification systems. More recently, development of deep learning technologies and availability of large-scale datasets facilitated advancement of

*Correspondence: dibeklioglu@cs.bilkent.edu.tr
[1]FaceSwap. GitHub [online]. Website https://github.com/deepfakes/faceswap [accessed 10 September 2023]

learning-based detection techniques [7–9]. These approaches can be categorized into two groups: spatial and spatiotemporal discriminators. Image-based (spatial) deep learning approaches usually rely on use of off-the-shelf convolutional neural networks (CNN), architectures to approximate authenticity of each frame and then employ a voting scheme to calculate the cumulative probability of the entire video sequence being real. The second family of approaches aims to capture temporal features along with spatial information extracted with CNN network. To this end, such methods often utilize a combination of CNN and long short-term memory (LSTM) networks to capture spatiotemporal information relevant to classification [8, 10]. Although several other preprocessing routines have been employed for the task of deepfake detection, one particular temporal processing framework has been employed more frequently than others. Motion magnification has seen a fair bit of use in aiding detection of deepfake recordings as a preprocessing step with the goal of revealing subtler temporal inconsistencies in the video. General thesis among such approaches is that motion magnification will help to magnify temporal inconsistencies within consecutive video frames.

First proposed by Liu et al. [11], motion magnification aims to magnify transitional or color fluctuations in the video sequence. Earliest approaches rely on use of complex Lagrangian signal analysis to isolate relevant motions of interest [11], while later approaches benefit from more efficient Eulerian signal estimation [12]. Inspired by recent advancements in deep learning technology, Oh et al. [13], propose an encoder-decoder network, to learn the magnification of small object movements on a synthetically generated dataset. In reviewed literature, most magnification-based deepfake detection approaches utilize one of the aforementioned magnification frameworks as a preprocessing step [14–17]. However, all of these magnification approaches, and many similar ones are designed and optimized for a limited set of controlled motions and there is no viable study proving that such approaches have the potential to generalize well to the recordings outside their domain. Moreover, even magnification of selected band of motion frequencies, such as frequencies corresponding to eye or mouth movement, does not guarantee optimal prediction outcome as many motion artefacts are likely to exist at much lower frequency domains. Lastly, existing magnification frameworks magnify selected motions of interest with a predetermined magnification constant. However, in the context of manipulation detection, it might be of interest to suppress certain motions instead of magnifying them. These limitations necessitate the development of a domain-specific magnification framework with the potential to generalize to the task at hand. However, because mechanisms guiding learnable motion magnification are still a topic of active research, and are not yet fully understood, we believe that a controlled and incremental approach is required in understanding the use of motion magnification in relevant classification tasks.

Our proposed network uses smoothly varying enhancing-dampening mask ensuring selection and suppression of spatial regions irrelevant to the final prediction while enhancing relevant ones. In this research, we take the first step towards understanding and evaluating the performance of task-focused motion magnification inspired network in the context of deepfake detection. For this reason, we propose an end-to-end learnable and deepfake-focused feature enhancement framework and verify that it yields consistent improvements over the baseline network. Our main contributions in this work are as follows:

1) We devise and test a learning-based, plug-in magnification-inspired model for the task of deepfake detection.

2) We improve upon the existing feature manipulation unit by adding learnable spatial variability across the image segments via use of spatially varying learned multiplication filter instead of a single predefined magnification scalar.

3) We enable free value range, and enforce spatial selectivity and dampening, by restricting the mean of multiplication matrix values to unity.

Unlike previous approaches utilizing motion magnification for the task of deepfake detection, we devise a plug-in learnable feature manipulation unit, inspired by motion magnification, for the purpose of deepfake detection.

This paper is structured as follows: In Section 2, we offer a concise survey of the existing literature on deepfake detection and motion magnification. Section 3, we present the proposed model in detail. In Section 4, we describe the experimental setup and report our findings. Finally, in Section 5, we provide a comprehensive summary of our contributions and key findings.

2. Related work

Over the past few years, a significant amount of progress has been made in the field of deepfake detection, and a diverse set of solutions have been proposed to tackle the problem. These approaches vary in modality, architecture, and types of features exploited. Given the visio-temporal nature of the problem, a significant amount of work in deepfake detection literature focuses on the use of spatiotemporal information to reveal transitional inconsistencies in the video sequence. To enhance such features some of the solutions focus on the use of magnification methods as a preprocessing step. In our work, we aim to implement a learning-based magnification-inspired feature manipulation architecture to explicitly learn and enhance representations valid for deepfake detection. To this end, we provide a brief overview of relevant benchmarks and frameworks. In the first section of this literature review, we conduct a summary of existing works on deepfake generation and detection. In the latter section, we describe existing motion magnification frameworks and their application in video forensics.

2.1. Generation and detection methods

Current fake media generation methods can be grouped either by the method of generation, or by types of manipulation applied. Earlier methods rely on 3D geometry of the face and tracked landmarks to warp the target face onto the source under different pose variations [18, 19]. More recent swapping approaches rely on learnable representations to synthesize artificial face using neural networks. Korshunova et al. [20] utilize convolutional neural networks to transform target image to the source after initial alignment, then realign the synthesized face back into the original image. Recent learning based approaches increasingly utilize generative networks to project the source face representation onto the target image. Bansal et al. [21] use spatiotemporally constrained generative network to create context aware face videos where surrounding imagery is also altered to increase the credibility of resulting video, while Zhu et al. [22] utilize hierarchical feature pyramids and feature refinement to achieve one shot face-swaps of high resolution.

Rapid increase in the quantity and quality of face-swap generation methods, along with increasing concerns over possible public and private impact of such technologies, had sparked an increase in the number of frameworks attempting to detect such content. Early detection approaches rely on use of image classification methods for classification of individual video frames and then extend the final prediction to the entire sequence, either through use of probability accumulation, or through additional connected layers and classifiers [23, 24]. Although such approaches display competitive performance on many of the benchmark datasets, they do not incorporate temporal information within consecutive video frames, thus evading a breadth of exploitable information for classification. In contrast to frame-based approaches, Wu et al. [25] utilize constrained

convolutional filters to force the network to focus on low-level convolutional features and then fuse the obtained feature map with learned steganalysis features. Features are then fed into a recurred neural network for temporal consistency analysis. Chintha et al. [26] incorporate features extracted from optical flow and edges, in addition to standard image features, and make use of bidirectional-LSTM to devise a multifeature CNN-LSTM network. In a somewhat similar approach, Nassif et al. [27] propose to focus on inter-frame inconsistencies by estimating optical flow in the videos and then feeding the resulting map into a CNN network.

2.2. Motion magnification and forensics

Motion magnification techniques are, most commonly, divided into two families: Lagrangian and Eulerian motion magnification. With the Lagrangian magnification method, the pixels with correlated motion field are grouped together, and then the motion of each pixel is estimated and magnified [11]. The major drawback of Lagrangian motion magnification is the heavy computational cost associated with tracking and estimating pixel motions. The Eulerian motion magnification method circumvents this problem by using spatial decomposition and temporal filtering, thus eliminating the need for pixel-wise motion estimation [12]. Building further upon this approach, in their work on learning-based motion magnification, Oh et al. [13] propose using machine-learned features instead of using hand-crafted decomposition filters. They train and verify their network on a synthetically generated motion magnification dataset, proving the feasibility of the learned motion magnification. Our work borrows from the idea of learnable motion magnification, proposed by [13], and extends upon it by incorporating a modified magnification inspired processing layer into the CNN backbone of the network and reaiming loss function to optimize for the final prediction performance.

Earliest applications of motion magnification in the field of video forensics are with video spoofing detection [28–30], where Eulerian motion magnification is used as a preprocessing step to enhance the subtle motion cues in the video. Several others utilize motion magnification as a preprocessing step to enhance the biometric cues. In their work, Fernandes et al. [16] use video magnification to obtain heart-beat rhythms from the color variation in the face of the subject, and use the estimated signal to make the final prediction about the origin of the video. In a similar approach, Qi et al. [15] display the generalizability of the approach to a large-scale dataset. Several others postulate that the use of off-the-shelf Euler magnification as a preprocessing step has the potential to magnify motion and color variations in the video stream which then can be fed into a combination of CNN and LSTM networks [14, 31]. We present the summary of detection methods utilizing the concept of motion magnification in Table 1. A common feature among the aforementioned methods is to use readily implemented motion magnification features as a preprocessing step for the framework.

Our work differs from the above-mentioned methods in that it embeds a learnable magnification-inspired manipulator unit in between CNN layers of the network. We believe this approach is superior to the aforementioned models for a few reasons. First, commonly utilized off-the-shelf magnification algorithms are optimized to preserve the texture representation while the shape representation in the frames is transformed to magnified proportions [13]. However, visual observation of many synthesized videos reveals significant observable texture artifacts which, if magnified, can improve the prediction accuracy of the method in question. The relevance of texture perturbation is supported by several other existing works in the deepfake detection literature [33, 34, 45]. Preservation of texture information in magnified videos is not necessarily desirable when the objective is to detect artifacts caused by video tampering. Second, the two magnification methods [12, 13] most commonly used as a preprocessing step by detection approaches are designed and optimized for a very limited, and controlled set of small motions and color variations and are not necessarily optimized to reveal and magnify small

perturbations in deepfake videos. Lastly, spatial decompositions in filter-based magnification algorithms and encoding in learning-based magnification methods are optimized for the visually consistent reconstruction of the input video frames, which is not necessarily beneficial in the context of classification as they might compensate for innate temporal and spatial irregularities in the fake videos that help to detect them. To address these shortcomings of applying magnification as a brute preprocessing for deepfake detection, we devise a learnable motion enhancement unit inspired by the learnable magnification unit proposed by [13]. Because magnification technologies are very much in their prime, we make the case that a phased approach is required to study the effects of learning-based magnification for classification tasks, including deepfake detection. By embedding an end-to-end feature manipulation layer, we can directly learn feature enhancement that is most relevant to the prediction outcome. In addition, instead of using a predefined scalar value as a multiplication factor of the magnification, we utilize a learnable multiplication mask, which facilitates different levels of amplification on separate regions of the feature vector.

Table 1. Structural summary of the existing video forensics methods utilizing concept of motion magnification.

Author	Overall structure	Magnification utilization	Description
Fei et al. [14]	CNN, LSTM	Preprocessing	*Use [13] to magnify video, extract spatial features with InceptionV3 [40] network, and feed the features to LSTM network.*
Das et al. [31]	SSIM, CNN, LSTM, HR	Preprocessing	*Use [12] to magnify input video, then use frame similarity (SSIM), CNN-LSTM network, and heartbeat rhythms (HR) to make the final prediction.*
Qi et al. [15]	CNN, LSTM, Spatiotemporal attention	Preprocessing	*Use [12] to magnify video, separate frames into ROI, and feed them into temporal and spatial attention module, aggregate the results with per-frame classification.*
Fernandes et al. [16]	Neural-ODE	Feature Extraction	*Use [12] as one of the feature extraction methods to feed to Neural Ordinary Differential Equations (Neural-ODE) model.*
Bharadwaj et al. [28]	LBP, HOOF	Preprocessing	*Use [12] to magnify input video, then extract Local Binary Pattenrs (LBP) and Histogram of Oriented Optical Flow (HOOF) for SVM and LDA classifiers.*
Raja et al. [29]	Fourier, CPI	Adapted Preprocessing	*Use phase-adapted Eulerian magnification [12] on input video. Use magnified representation to compute and threshold cumulative phase information.*
Ge et al. [30]	CNN, LSTM, Temporal attention	Preprocessing	*Use [12] to magnify input videos, extract features using VGG-Face network, and feed it to LSTM unit enhanced by temporal attention.*
Mehra et al. [17]	CNN, Residual	Preprocessing	*Use [13] to magnify input videos. Use the difference of original and magnified frames for classification.*

3. Method

The overall architecture of our approach is summarized in Figure 1. Here, we utilize a combination of CNN and LSTM networks enhanced with the plug-in, end-to-end trainable magnification-inspired manipulator unit. The network is trained on fixed-length input sequences. Unlike existing applications, we are proposing to

learn a spatially aware multiplication array, to create spatially varying dynamic feature enhancement. In addition, by restricting the mean of the multiplication array to unity, we enforce a mixture of enhancement and dampening across the spatial domain to ensure that only relevant image sections are enhanced while features in irrelevant regions are dampened. To enforce gradual variation across the multiplication array, we apply Gaussian smoothing before multiplying the mask with the feature vector. During testing, we train the model, both with and without a manipulator unit to compare the performance of feature-manipulated and baseline variants.

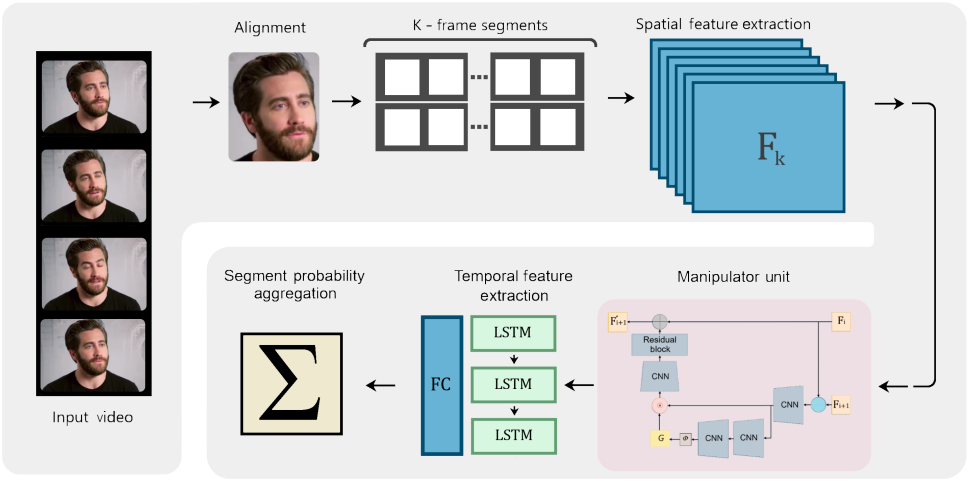

Figure 1. A top-down overview of the proposed deepfake detection framework utilizing a magnification-inspired manipulator unit. We first perform alignment of input video frames, then we segment the input video into nonoverlapping sections of length K. Next, we use multilayer CNN network to extract spatial features from the video, which are then fed into the magnification-inspired manipulator unit. Subsequently, we feed manipulated features to an LSTM-FC network, followed by the utilization of two distinct aggregation methods to accumulate the results.

3.1. Video preprocessing

As the first step, we track the face landmarks in input videos using OpenFace tracking library [35–37]. Landmarks are then smoothed with running median smoothing. We then rotate, resize, center, and crop the face so that the eyes of the subject are always at the center of the frame and the line connecting the center of the eyes is parallel to the horizontal axis. To approximate the roll of the face, first, the center of each eye is calculated by averaging six available eye landmarks. Then, the angle between the horizontal plane and the line connecting the centers is calculated as described in Equation 1.

$$\begin{aligned} \theta &= \arccos \frac{\mathbf{u} \cdot \tau}{||\mathbf{u}||\,||\tau||}, \text{ where } \tau = \overrightarrow{c_1 c_2}, \\ c_1 &= \frac{1}{|S_1|} \sum_{i \in S_1} p_i, \text{ and } c_2 = \frac{1}{|S_2|} \sum_{i \in S_2} p_i \end{aligned} \tag{1}$$

Here, in Equation 1, $\mathbf{u} = \langle 1, 0 \rangle$ is the horizontal unit vector, and S_1 and S_2 represent the subset of landmarks p_i corresponding to the left and right eye respectively. While τ represents the vector connecting the centers of the eyes. As the next step, the image is rotated around the center by an angle θ. Landmarks in

the image are rotated using a standard rotation matrix. After rotating the entire image, we resize the image so that the distance between the center of the eyes is equal to a quarter of the width of the desired preprocessing image shape. As the last step, we crop the image uniformly around the center of the line connecting the center of the eyes. Frames are then ordered into consecutive sequences of length K, and each sequence is treated as a separate sample during training. The reason for the choice of this windowing, and alignment process is in consideration of the ultimate task at hand. Standard cropping utilized by many other approaches [23, 38] introduces fluctuations between consecutive cropped frames, as bounding boxes shift considerably between images. Inputs with no cropping, on the other hand, require a larger network to learn more content-rich representations. By aligning the faces at the center of the image, and cropping a wider rectangle around it, we hope to achieve a compromise between the input size and spatiotemporal consistency.

3.2. Backbone model

It has been shown by several previous works that utilization of an off-the-shelf pretrained CNN network can achieve state-off-the-art performance on major deepfake detection datasets even in the absence of temporal analysis or magnification. However, direct coupling of such approaches with motion magnification would prove insufficient as existing motion enhancement frameworks require specialized and controlled conditions to function effectively. Thus, rather than aiming to exceed the accuracy of overspecialized deepfake detection networks, we aim to show significant and consistent performance improvement with a learnable, dynamic, and spatially aware magnification-inspired manipulator unit.

Our baseline is composed of the CNN-LSTM network. As illustrated in Figure 2, the frames are first fed through the backbone CNN model to obtain the latent representations of the input frames. If plugged into the network, consecutive frames are then fed to the manipulator unit for further feature enhancement. Finally, the features are rectified and fed through two stacks of LSTM networks before the final prediction. When the manipulator unit is plugged out, features are directly fed to the LSTM network. To test if the manipulator unit positively contributes to classification performance, we perform individual experiments both with and without the manipulator unit.

Unlike the original approach on learnable-magnification[13], from which we draw inspiration, our network does not separate the texture and shape representation of the image. This is done to preserve both shape and surface artifacts relevant to the prediction outcome. We argue that the exclusion of texture information prohibits the network from learning texture deformations in the output video, thus inhibiting the learning outcome. Such an approach is also in agreement with our initial argument that the application of motion magnification frameworks as a preprocessing step is suboptimal for the task of deepfake detection. It should be mentioned that, as illustrated by Figure 2, embedding of the manipulator layer in the network yields $K-1$ frames for the LSTM layer since the F_1, the first frame, has no reference frame of its own. Therefore, the number of frames used in the final prediction is always one less for the network with the manipulator unit.

3.3. Feature manipulation unit

In our work, we use the formulation of motion magnification inspired by Oh et al. [13] and Wu et al. [12]. Provided a spatiotemporal interpretation of a moving frame $I(\mathbf{x},t) = f(\mathbf{x}+\delta(\mathbf{x},t))$, where motion image is defined by $I(\mathbf{x},t)$ and the field of motion, $\delta(\mathbf{x},t)$, is expressed as both the function of time and position. Motion magnification then can be expressed by Equation 2 as selective amplification of motion field between consecutive frames.

$$\tilde{I}(\mathbf{x},t) = f(\mathbf{x} + (1+\alpha)\mathcal{J}(\delta(\mathbf{x},t)) \tag{2}$$

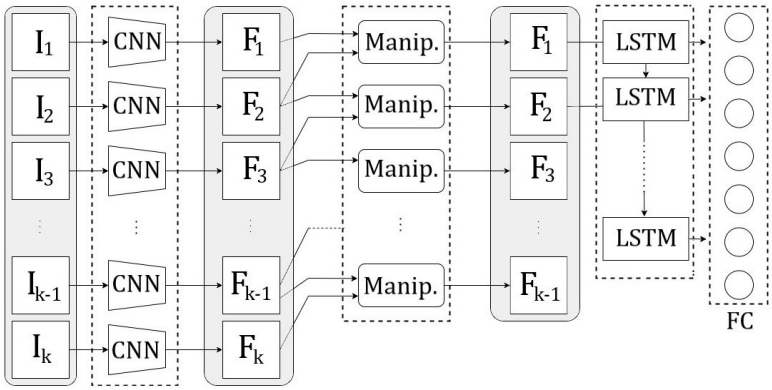

Figure 2. Architecture of baseline model with motion manipulator unit inserted in between. The figure illustrates how frames are fed to the model in a time-distributed manner, with each arrow representing path of the corresponding frame in the processing pipleine. In other words, consecutive frames are processed by the same feature extractor, and the results are then accumulated in the fully connected layer. Here I_k denotes the k^{th} frame of the video, while F_k denotes the feature representation for that corresponding frame. The model consists of CNN spatial feature encoder, manipulator unit, LSTM, and fully connected layer for the final prediction, as detailed in Section 3.2.

Here in Equation 2, α is the factor of magnification and $\mathcal{J}(\cdot)$ is the feature selector that isolates the motions of interest relevant to the task at hand. Early approaches rely on hand-crafted decomposition filters to select motions of interest. However, in this research, we use a manipulator block inspired by the architecture proposed by Oh et al. [13]. We further extend this architecture to transform the magnification factor α from a scalar predefined value to a learnable mask. That is, provided a feature vector $\mathbf{F} \in \mathbb{R}^{H\times W\times C}$ with H, W, and C representing the height, width, and number of channels respectively, our multiplication vector can be represented by learned vector $\mathcal{A}(\cdot) \in \mathbb{R}^{H\times W}$. The resulting magnification equation then takes the following form:

$$\tilde{F}(\mathbf{x},t) = f(\mathbf{x} + \mathcal{A}(\cdot) \odot \mathcal{J}(\delta(\mathbf{x},t))) \tag{3}$$

where, similar to Equation 2, $\delta(\mathbf{x},t)$ represents the motion vector, and $\mathcal{J}(\cdot)$ represents the feature selector. In practice, the estimation of motion vector, however, is computationally expensive and also prevents the model from training end-to-end. To reduce the overhead cost, the authors of [13] take a difference-of-two-frames approach defined by Equation 4, where D_{i+1} is a feature vector selected by $\mathcal{L}(\cdot)$ from the difference of two consecutive frame representations, and F_i and F_{i+1} are feature representations from consecutive frames.

$$D_{i+1} = \mathcal{L}(F_{i+1} - F_i) \tag{4}$$

We extend this layout by incorporating learnable magnification damping unit $\mathcal{A}(\cdot)$ as demonstrated by Equation 5 below.

$$F'_{i+1} = F_i + \mathcal{H}(\mathcal{A}(D_{i+1}) \odot D_{i+1}) \tag{5}$$

In Equations 5 and 4, $\mathcal{L}(\cdot)$ represents the first convolution layer, and $\mathcal{H}(\cdot)$ represents the second convolutional layer followed by residual layer illustrated in Figure 3. Here $\mathcal{L}(\cdot)$ network is used to extract the features of interest from the raw difference vector $F_{i+1} - F_i$ while $\mathcal{H}(\cdot)$, as per original research [13], improves results by adding additional nonlinearity to the mask multiplied feature. Mask estimation itself can be formulated as a composition of feature selection, value normalization, and Gaussian smoothing units.

$$\mathcal{A}(D_{i+1}) = \mathcal{G}(\Phi(T(D_{i+1})) \tag{6}$$

$$\Phi(\mathbf{X}) = \frac{exp(\mathbf{X})H_X W_X}{\sum_{i=1}^{H_X}\sum_{j=1}^{W_X} e^{a_{i,j}}} \tag{7}$$

where $T(\cdot)$ serves as mask-specific feature selector, $G(\cdot)$ represents spatial Gaussian smoothing, and $\Phi(\cdot)$ serves as a normalization function that enforces the mean of multiplication array values to 1, as demonstrated by Equation 7. In Equation 7, X represents the input to the normalization equation, $a_{i,j}$ represents individual elements of the input matrix X. H_X and W_X represent the height and with of the input matrix respectively, and $exp(X)$ represents the element-wise application of exponential function to the input matrix. This enforces the enhancement of certain feature regions while dampening others, further ensuring that only features relevant to prediction are included in amplified feature manipulation. The damping feature further creates spatial feature selectivity and improves model performance. Lastly, resulting feature vector is spatially smoothed using Gaussian smoothing $\mathcal{G}(\cdot)$.

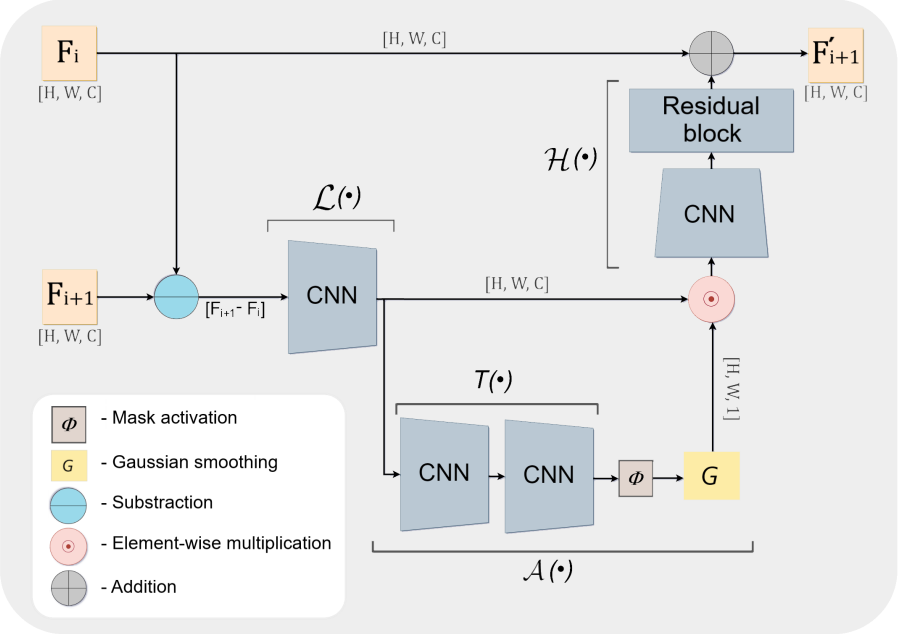

Figure 3. Architecture of the manipulator unit with a learnable dynamic multiplication mask. The symbols and characters used here are consistent with Section 3.3, where a comprehensive explanation of the functions $\mathcal{L}(\cdot)$, $\mathcal{A}(\cdot)$, and $\mathcal{H}(\cdot)$ is provided. Here H, W, C, and 1 in square brackets indicate the dimensions of the input. First, the difference of consecutive feature vectors is obtained (as shown by the operation $F_{i+1} - F_i$) and then passes through the convolutional feature selector $\mathcal{L}(\cdot)$. $\mathcal{A}(\cdot)$ is used to obtain the dynamic multiplication mask, while $\mathcal{H}(\cdot)$ is used to add nonlinearity to the final output.

4. Experiments

In this section, we define our experimental setup and discuss the obtained results. We start by describing the dataset in 4.1. In Subsection 4.2, we provide quantitative network parameters and framework details, followed by an overview of the learning schedule with details of hardware and software platform. Finally, in Subsections 4.3 through 4.6, we present and discuss our experimental results. In our experiments, we test and validate several variations of the manipulator unit to arrive at the optimally performing learning framework. We evaluate the effect of static, dynamic, and smoothed dynamic, manipulator units on the performance of our model, and compare the results with the baseline architecture. To verify the consistency of the results, for each configuration, we perform seven identical experiments and report average accuracy, area under the ROC curve (AUC), and F-1 results. In addition, we visualize dynamic mask values from the manipulator unit and comment on their relation to the input frame. We test our results on the publicly available Celeb-DF [39] dataset that is composed of more than 6000 total video samples. In our experiments, we aim to demonstrate that the manipulator unit can be flexibly inserted in and out of the standard baseline network and have a significant contribution to the output performance. Therefore, rather than focusing on maximizing performance across several benchmark datasets, we conduct a controlled set of experiments that demonstrate the potential applicability of magnification-inspired manipulator unit in manipulation detection tasks.

4.1. Dataset

To test the performance of the proposed framework, we are using a publicly available Celeb-DF dataset [39]. The dataset consists of 590 real and 5639 fake videos with a standard frame rate of 30 and an average length of 13 s, as well as 300 additional YouTube videos. It can be pointed out that the dataset at hand is class-imbalanced. This is due the fact that the same original video clip is often processed by several deepfake generation methods resulting in overrepresentation of corresponding class. However, we refrained from using data augmentation or dataset enrichment for two reasons. First, we cautioned that addition of more 'real' samples without 'fake' counterparts could result in video memorization, and secondly, we refrained from data augmentation to prevent learning of other forms of data alteration. Original real videos of the dataset are collected from a diverse set of publicly available YouTube recordings of individuals of different ages, genders, and ethnicities. To obtain deepfakes from original videos, authors use a modified and improved version of the DeepFakes algorithm. First, they achieve higher image resolution by expanding the encoder-decoder to have additional layers and larger output dimensions. Then, the authors manage to reduce the number of color deformations in resulting images by adding color perturbations to the training frames, in addition to using color transfer between output and input images. Furthermore, to reduce boundary distortions and inconsistencies between original and synthesized faces, the authors devise a more accurate face region estimation along with boundary blurring. Finally, authors reduce temporal flickering between consecutive frames by using Kalman filtering along the temporal dimension of generated face landmarks.

4.2. Framework parameters

In preprocessing the dataset for our network, we use nonoverlapping segments of fixed length. The number of frames (K) per segment is set to 32, and the shape of input frames is set to 256×256, this decision is made in light of the compromise between input resolution and hardware limitations, and initial experimental findings. We conduct all of our experiments on `GeForce GTX 1080 Ti` and `GeForce RTX 2080 Ti` graphic processing units. Average inference time for 32-frame video samples, measured in a batch of 1000 samples is around 520 ms with

feature manipulation unit attached. However, in our experiments, we noticed that processing speed fluctuates significantly based on the load of the system and temperature of the server room. In addition, measured time does not include preprocessing described in Section 3.1, which could create additional computational overhead.

The architecture of the baseline network is outlined in Figure 2. The backbone of the CNN network is made of three convolutional layers followed by max pooling operation. The leaky rectified linear unit (LeakyReLU) function is used as activation for the convolutional layers and the leaking factor is at 0.4. In addition, each max pooling layer is followed by a batch normalization and dropout layer that has a rate of 0.2. The sequence of convolutional and pooling layers is followed by a manipulator layer. All convolutional, pooling, and manipulator layers are time-distributed, meaning each frame of the sequence passes through them sequentially. All convolutional layers of the manipulator unit have the number of filters equal to the number of filters in the last convolutional layer of the main network, which in our case, is equal to 16. We use linear and ReLU activation functions for the convolutional layers of the manipulator network. In addition, all the convolutional layers of the manipulator unit have a kernel size of 3x3 and padding to ensure that the feature dimensions are not altered by the manipulator unit. In the manipulator unit, we employ spatial Gaussian smoothing with a filter size of 3 and a standard deviation of 1.

During experiments, we train and test the network, first without and then with the magnification layer, while the remaining parameters of the network remain intact. We do not alter the training parameters between experiments and conduct a total of seven experiments for each configuration. At the start of each epoch, we randomly shuffle training samples. We use the Stochastic Gradient Descent optimization function at a learning rate of 0.0001 with no decay. During each epoch, we conduct validation accuracy checks and stop training if the validation accuracy does not increase for 5 consecutive checks. The interval for validation accuracy is set at 2000 samples where each video segment is counted as a single sample. Validation checks for early stopping are performed only after the first epoch of the network. Network weights with the highest validation accuracy were recorded and used to obtain test results. To obtain single video predictions from individual segment probabilities, we use two probability accumulation techniques. The first method determines the class of prediction by majority voting where the sample is categorized as being fake if the majority of segments are predicted to be fake. This relation is expressed by Equation 8, where P_v corresponds to the probability of the entire video, S_v is the subset of all segments for a video v, and $P_{v,i}$ is the probability for an individual segment.

$$P_v = \frac{1}{|S_v|} \sum_{i \in S_v} d_i, \text{ where}$$
$$d_i = \begin{cases} 1, & \text{if } P_{v,i} \geq 0.5 \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

$$P_v = \frac{1}{|S_v|} \sum_{i \in S_v} P_{v,i} \tag{9}$$

The second method obtains the mean of all segment probabilities for a single video, and treats the obtained mean as the output probability for that video. Relation for the second aggregation method is expressed in Equation 9, where the variable denomination is identical to Equation 8.

4.3. Effect of manipulator unit

In the first experiment, we compare the baseline network with a network that has a manipulator unit. We first train the network from scratch without the manipulator unit and report the obtained results. We then repeat the same set of experiments for the network with the manipulator unit. The precise configuration of the

manipulator unit includes dynamic multiplication mask and Gaussian smoothing as illustrated in Figure 3. For each configuration, we conduct seven individual runs to ensure the highest possible consistency in the results. The corresponding results are presented in Table 2.

From the comparison in Table 2, we can observe consistent improvement over the baseline architecture in both aggregation methods and also in predictions for individual segments. From the obtained average results, we can observe that two aggregation methods yield very similar probability outcomes and outperform each other on different subsets.

Table 2. Effect of the manipulator unit on performance.

	Train			Validation			Test		
	ACC	AUC	F-1	ACC	AUC	F-1	ACC	AUC	F-1
Baseline	0.84	0.94	0.87	0.80	0.90	0.84	0.79	0.86	0.73
Manipulator	0.88	0.95	0.91	0.85	0.90	0.88	0.79	0.88	0.76
Baseline voting	0.85	0.94	0.89	0.82	0.92	0.86	0.81	0.86	0.77
Manipulator with voting	0.91	0.95	0.94	0.89	0.92	0.92	0.82	0.89	0.81
Baseline prob. mean	0.86	0.96	0.89	0.84	0.93	0.87	0.81	0.89	0.77
Manipulator with prob. mean	0.92	0.96	0.94	0.89	0.93	0.92	0.82	0.91	0.82

4.4. Effect of scalar multiplication

In the second set of experiments, we compare the results of the dynamic, Gaussian smoothed manipulator unit to that of the scalar motion manipulator unit proposed by Oh et al. [13]. In this case, multiplication array $\mathcal{A}(\cdot)$ is replaced with a scalar value. This configuration of the manipulator is identical to the one appearing in the original research. Here we set the multiplication factor to 2, and similar to the previous configuration we conduct a total of seven experiments for each configuration and report results for aggregate and individual segment results.

As can be seen from Table 3, a manipulator unit with a static scalar multiplier adversely affects the results of classification. This observation further confirms the initial claim that motion magnification approaches cannot be applied to classification tasks in isolation and need to be specifically designed for the task at hand. By incorporating dynamic feature learning for the multiplication matrix $\mathcal{A}(\cdot)$, and ensuring both amplification and dampening by normalization function $\Phi(\cdot)$, we enable the network to generalize the multiplication mask to previously unseen samples. In the absence of dynamic and sample-specific mask generation, there's a higher likelihood of overfitting to the intrinsic characteristics of the samples in the training data. As evident from Table 3, results using a static multiplication factor tend to overfit strongly to the training set and fail to generalize to the test subset.

4.5. Effects of Gaussian smoothing

In the third set of experiments, we compare the effects of Gaussian smoothing on the performance of manipulation mask. We compare the model accuracy, AUC and F-1 score with the nonsmoothed dynamic multiplier. During training, we turn off the Gaussian smoothing and train the network from scratch. As usual, we perform seven runs with each configuration and report average results.

In Table 4, we can observe that the performance of the manipulator unit improves as we add Gaussian smoothing to the manipulator. In fact, this improvement manifests itself not just in accuracy but also across AUC and F-1, further proving the robustness of the model. This can be explained by the fact that enforcing

spatial smoothness across frames ensures that the multiplication does not overfit to specific pixel values in the input frame but rather distributes attention over the larger regions of interest. Similar to previous comparisons, we observe that two probability aggregation methods reveal somewhat similar results and do not vary drastically.

Table 3. Performance of the dynamic and static manipulator units.

	Train			Validation			Test		
	ACC	AUC	F-1	ACC	AUC	F-1	ACC	AUC	F-1
Static	0.68	0.56	0.80	0.66	0.58	0.79	0.42	0.57	0.58
Dynamic	0.86	0.93	0.91	0.81	0.88	0.87	0.71	0.85	0.71
Static voting	0.73	0.54	0.84	0.71	0.56	0.83	0.45	0.55	0.62
Dynamic voting	0.89	0.90	0.93	0.84	0.88	0.90	0.74	0.82	0.76
Static prob. mean	0.73	0.60	0.84	0.72	0.63	0.83	0.45	0.62	0.62
Dynamic prob. mean	0.89	0.95	0.93	0.84	0.92	0.90	0.74	0.87	0.76

Table 4. Effect of Gaussian smoothing on performance.

	Train			Validation			Test		
	ACC	AUC	F-1	ACC	AUC	F-1	ACC	AUC	F-1
No Gaussian	0.86	0.93	0.91	0.81	0.88	0.87	0.71	0.85	0.71
Gaussian	0.88	0.95	0.91	0.85	0.90	0.88	0.79	0.88	0.76
No Gaussian voting	0.89	0.90	0.93	0.84	0.88	0.90	0.74	0.82	0.76
Gaussian voting	0.91	0.95	0.94	0.89	0.92	0.92	0.82	0.89	0.81
No Gaussian prob. mean	0.89	0.95	0.93	0.84	0.92	0.90	0.74	0.87	0.76
Gaussian prob. mean	0.92	0.96	0.94	0.89	0.93	0.92	0.82	0.91	0.82

4.6. Visualization of multiplication mask

The utilization of a dynamic multiplier mask enables us to attend differently to each section of the feature vector D_i, while Gaussian smoothing ensures smooth spatial variation. In addition, by generating a frame-specific multiplication mask, we ensure optimal performance across samples. In our experiments we are curious to observe the explainable visualization of the multiplication mask. Therefore, similar to spatial attention frameworks, we visualize the multiplication mask along the corresponding input frames. In Figure 4, from left to right,we present the original frame, multiplication mask, and the mask overlaid with the original frame for the deepfake-synthesized samples in the dataset.

As demonstrated by Figure 4, a common region where the mask of the manipulator unit attends to are the peripheries of the face and sections surrounding the eyes and the mouth. Such a result is not completely surprising, since the most revealing synthesis artifacts are often found at the boundaries of the swapped face. These regions usually contain color blending and alignment artifacts which are valuable to manipulation detection. In addition, across samples, we observe that the network often focuses on the hair region of the image which represents another section where deepfake generation methods display significant color inconsistencies. As intended by the design of the multiplication vector, certain values, expressed by darker tones in Figure 4 are dampening the corresponding sections of the feature vector D_i, while others are amplified.

Figure 4. Values of the multiplication mask $\mathcal{A}(\cdot)$ represented along with frames that generated them. Original frame is displayed on the left, multiplication mask in the middle, and overlaid picture of the two is on the right.

4.7. Comparison to other methods

In this section, we compare our method with some of the existing works in the literature. It is important to emphasize that the main objective of this research is to investigate custom spatio-temporal feature manipulation units that can yield interpretable improvements in tampered media detection. In addition, we retain a compact architecture so as to fully train our model without employing a pretrained backbone. For a fair comparison, in Table 5, we compare our model with a selection of relatively compact architectures from the recent state-of-the-art literature. As observed, with the help the proposed manipulator unit, our model provides comparable results with the state-of-the-art models while retaining a much simpler architecture and without using a pretrained backbone.

Table 5. Comparison with existing models on CelebDF dataset.

Method	ACC	AUC
Frame inference-based detection [41]	0.90	0.93
Temporal Dropout 3D CNN [42]	0.81	0.89
Two-stream temporal analysis [43]	0.90	-
Self-supervised learning [44]	0.80	-
Multi-attentional network [45]	-	0.67
Proposed: Spatio-temporal feature manipulation	0.86	0.91

5. Conclusion and discussion

In this work, we present a magnification-inspired learning-based attention network for deepfake classification. Our model demonstrates the possibility of utilizing learnable magnification-like feature manipulation for the task of video classification, and unlike off-the-shelf magnification approaches, our approach utilizes learnable magnification-like network architecture for the task of deepfake classification. We believe that our work serves as a valuable first step toward understanding the utilization of magnification for the tasks of classification. In

this paper, we present and test several variations of the network and demonstrate consistent improvements over the baseline architecture. We demonstrate that the proposed manipulator unit can be easily swapped in and out of the entire network architecture and can be trained end to end. In addition, we observe improvements over several metrics such as accuracy, AUC, and F-1, reinforcing our conclusions of statistically significant improvement.

As future work, we aim to test the applicability of the manipulator unit for the broader task of video classification and possibly extend the scope of the project to tasks such as object detection and segmentation. We also aim to improve the performance of the unit on deepfake detection and experiment with multilayer manipulator units.

References

[1] Macon MW, Jensen-Link L, Oliverio J, Clements MA, George EB. A singing voice synthesis system based on sinusoidal modeling. In: 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing; Munich, Germany; 1997. pp. 435-438.

[2] Macon M, Jensen-Link L, George EB, Oliverio J, Clements M. Concatenation-based midi-to-singing voice synthesis. In: Audio Engineering Society Convention 103; 1997.

[3] Shan S, Gao W, Yan J, Zhang H, Chen X. Individual 3d face synthesis based on orthogonal photos and speech-driven facial animation. In: Proceedings 2000 International Conference on Image Processing; Vancouver, BC, Canada; 2000. pp. 238-241.

[4] Boulkenafet Z, Komulainen J, Hadid A. Face spoofing detection using colour texture analysis. IEEE Transactions on Information Forensics and Security 2016; 11 (8): 1818-1830. https://doi.org/10.1109/TIFS.2016.2555286

[5] De Souza GB, da Silva Santos DF, Pires RG, Marana AN, Papa JP. Deep texture features for robust face spoofing detection. IEEE Transactions on Circuits and Systems II: Express Briefs 2017; 64 (12): 1397-1401. https://doi.org/10.1109/TCSII.2017.2764460

[6] Komulainen J, Hadid A, Pietikäinen M. Context based face anti-spoofing. In: 2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS); Arlington, VA, USA; 2021. pp. 1-8.

[7] Guarnera , Giudice O, Battiato S. Deepfake detection by analyzing convolutional traces. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops; 2020. pp. 666-667.

[8] Güera D, Delp EJ. Deepfake video detection using recurrent neural networks. In: 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS); Auckland, New Zealand; 2018. pp. 1-6.

[9] Mittal T, Bhattacharya U, Chandra R, Bera A, Manocha D. Emotions don't lie: an audio-visual deepfake detection method using affective cues. In: Proceedings of the 28th ACM International Conference on Multimedia; Seattle, WA, USA; 2020. pp. 2823-2832.

[10] Sabir E, Cheng J, Jaiswal A, AbdAlmageed W, Masi I. Recurrent convolutional strategies for face manipulation detection in videos. Interfaces (GUI) 2019; 3 (1): 80-87.

[11] Liu C, Torralba A, Freeman WT, Durand F, Adelson EH. Motion magnification. In: ACM Trans. Graph; New York, NY, USA; 2005. pp. 519–526.

[12] Wu H-Y, Rubinstein M, Shih E, Guttag J, Durand F et al. Eulerian video magnification for revealing subtle changes in the world. In: ACM Trans. Graph; New York, NY, USA; 2012. pp. 1–8.

[13] Oh T-H, Jaroensri R, Kim C, Elgharib M, Durand F et al. Learning-based video motion magnification. In: Proceedings of the European Conference on Computer Vision (ECCV); 2018. pp. 633-648.

[14] Fei J, Xia Z, Yu P, Xiao F. Exposing AI-generated videos with motion magnification. Multimedia Tools and Applications 2021; 80: 30789-30802. https://doi.org/10.1007/s11042-020-09147-3

[15] Qi H, Guo Q, Juefei-Xu F, Xie X, Ma L et al. DeepRhythm: exposing deepfakes with attentional visual heartbeat rhythms. In: Proceedings of the 28th ACM International Conference on Multimedia (MM '20); New York, NY, USA; 2020. pp. 4318–4327.

[16] Fernandes S, Raj S, Ortiz E, Vintila I, Salter M et al. Predicting heart rate variations of deepfake videos using neural ODE. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops; 2019. pp. 0-0.

[17] Mehra A, Agarwal A, Vatsa M, Singh R. Motion magnified 3-D residual-in-dense network for deepfake detection. IEEE Transactions on Biometrics, Behavior, and Identity Science 2023; 5 (1): 39-52. https://doi.org/10.1109/TBIOM.2022.3201887

[18] Lin Y, Wang S, Lin Q, Tang F. Face swapping under large pose variations: a 3D model based approach. In: 2012 IEEE International Conference on Multimedia and Expo; Melbourne, VIC, Australia; 2012. pp. 333-338.

[19] Dale K, Sunkavalli K, Johnson MK, Vlasic D, Matusik W et al. Video face replacement. In: Proceedings of the 2011 SIGGRAPH Asia Conference (SA '11); New York, NY, USA; 2011. pp. 1–10.

[20] Korshunova I, Shi W, Dambre J, Theis L. Fast face-swap using convolutional neural networks. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV); 2017. pp. 3677-3685.

[21] Bansal A, Ma S, Ramanan D, Sheikh Y. Recycle-GAN: unsupervised video retargeting. In: Proceedings of the European Conference on Computer Vision (ECCV); 2018. pp. 19-135.

[22] Zhu Y, Li Q, Wang J, Xu CZ, Sun Z. One shot face swapping on megapixels. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2021; 4834-4844.

[23] Rossler A, Cozzolino D, Verdoliva L, Riess C, Thies J et al. Faceforensics++: learning to detect manipulated facial images. In: Proceedings of the IEEE/CVF International Conference on Computer Vision; 2019: 1-11.

[24] Chen HS, Rouhsedaghat M, Ghani H, Hu S, You S et al. Defakehop: a light-weight high-performance deepfake detector. In: 2021 IEEE International Conference on Multimedia and Expo (ICME); Shenzhen, China; 2021: 1-6.

[25] Wu X, Xie Z, Gao Y, Xiao Y. Sstnet: detecting manipulated faces through spatial, steganalysis and temporal features. In: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing; Barcelona, Spain; 2020. pp. 2952-2956.

[26] Chintha A, Rao A, Sohrawardi S, Bhatt K, Wright M et al. Leveraging edges and optical flow on faces for deepfake detection. In: 2020 IEEE International Joint Conference on Biometrics (IJCB); Houston, TX, USA; 2020: 1-10.

[27] Nassif AB, Nasir Q, Talib MA, Gouda OM. Improved optical flow estimation method for deepfake videos. Sensors 2022; 22(7): 2500-2500. https://doi.org/10.3390/s22072500

[28] Bharadwaj S, Dhamecha TI, Vatsa M, Singh R. Computationally efficient face spoofing detection with motion magnification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops; 2013. pp. 105-110

[29] Raja KB, Raghavendra R, Busch C. Video presentation attack detection in visible spectrum iris recognition using magnified phase information. IEEE Transactions on Information Forensics and Security 2015; 10 (10): 2048-2056. https://doi.org/10.1109/TIFS.2015.2440188

[30] Ge H, Tu X, Ai W, Luo Y, Ma Z et al. Face anti-spoofing by the enhancement of temporal motion. In: 2020 2nd International Conference on Advances in Computer Technology, Information Science and Communications (CTISC); Suzhou, China; 2020. pp. 106-111.

[31] Das R, Negi G, Smeaton AF. Detecting deepfake videos using euler video magnification. arXiv Preprint arXiv:2101.11563 2021. https://doi.org/10.48550/arXiv.2101.11563

[32] Zhao H, Zhou W, Chen D, Wei T, Zhang W et al. (2021). Multi-attentional deepfake detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2021. pp. 2185-2194.

[33] Liu H, Li X, Zhou W, Chen Y, He Y et al. Spatial-phase shallow learning: rethinking face forgery detection in frequency domain. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2021. pp. 772-781.

[34] Liu X, Yu Y, Li X, Zhao Y. Magnifying multimodal forgery clues for deepfake detection. Signal Processing: Image Communication 2023; 118: 117010. https://doi.org/10.1016/j.image.2023.117010

[35] Baltrusaitis T, Zadeh A, Lim YC, Morency L-P. Openface 2.0: facial behavior analysis toolkit. In: 2018 13th IEEE International Conference on Automatic Face and Gesture Recognition; Xi'an, China; 2018. pp. 59-66.

[36] Zadeh A, Chong Lim Y, Baltrusaitis T, Morency LP. Convolutional experts constrained local model for 3d facial landmark detection. In: Proceedings of the IEEE International Conference on Computer Vision Workshops; 2017. pp. 2519-2528.

[37] Baltrusaitis T, Robinson P, Morency LP. Constrained local neural fields for robust facial landmark detection in the wild. In: Proceedings of the IEEE International Conference on Computer Vision Workshops; 2013. pp. 354-361.

[38] Dang H, Liu F, Stehouwer J, Liu X, Jain AK. On the detection of digital face manipulation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern recognition; 2020. pp. 5781-5790.

[39] Li Y, Yang X, Sun P, Qi H, Lyu S. Celeb-df: a large-scale challenging dataset for deepfake forensics. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2020 pp. 3207-3216.

[40] Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2016. pp. 2818-2826.

[41] Hu J, Liao X, Liang J, Zhou W, Qin Z. Finfer: frame inference-based deepfake detection for high-visual-quality videos. In: Proceedings of the AAAI Conference on Artificial Intelligence; 2022. pp 951-959.

[42] Zhang D, Li C, Lin F, Zeng D, Ge S. Detecting deepfake videos with temporal dropout 3DCNN. In IJCAI 2021: 1288-1294.

[43] Hu J, Liao X, Wang W, Qin Z. Detecting compressed deepfake videos in social networks using frame-temporality two-stream convolutional network. In IEEE Transactions on Circuits and Systems for Video Technology 2022; 32 (3): 1089-1102. https://doi.org/10.110/TCSVT.2021.3074259.

[44] Chen L, Zhang Y, Song Y, Liu L, Wang J. Self-supervised learning of adversarial example: towards good generalizations for deepfake detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition 2022;18710-18719.

[45] Zhao H, Zhou W, Chen D, Wei T, Zhang W et al. 2021. Multi-attentional deepfake detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition 2021;2185-2194.