Improving the Efficiency and Robustness of Deepfakes Detection through Precise Geometric Features

Zekun Sun[1] Yujie Han[1] Zeyu Hua[1] ✉Na Ruan[1] Weijia Jia[1,2,3]

[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China

[2] Institute of Artificial Intelligence and Future Networks, Beijing Normal University (BNU Zhuhai), Guangdong, China

[3] Key Lab of AI and Multi-Modal Data Processing, BNU-HKBU United International College, Guangdong, China

{szk037, flora_hua, jiawj}@sjtu.edu.cn borloch@outlook.com naruan@cs.sjtu.edu.cn

Abstract

Deepfakes is a branch of malicious techniques that transplant a target face to the original one in videos, resulting in serious problems such as infringement of copyright, confusion of information, or even public panic. Previous efforts for Deepfakes videos detection mainly focused on appearance features, which have a risk of being bypassed by sophisticated manipulation, also resulting high model complexity and sensitiveness to noise. Besides, how to mine the temporal features of manipulated videos and exploit them is still an open question. We propose an efficient and robust framework named LRNet for detecting Deepfakes videos through temporal modeling on precise geometric features. A novel calibration module is devised to enhance the precision of geometric features, making it more discriminative, and a two-stream Recurrent Neural Network (RNN) is constructed for sufficient exploitation of temporal features. Compared to previous methods, our proposed method is lighter-weighted and easier to train. Moreover, our method has shown robustness in detecting highly compressed or noise corrupted videos. Our model achieved 0.999 AUC on FaceForensics++ dataset. Meanwhile, it has a graceful decline in performance (-0.042 AUC) when faced with highly compressed videos.[1]

1. Introduction

Due to the recent improvement of autoencoders and Generative Adversarial Networks (GAN) [9], synthetic videos are becoming unprecedentedly vivid and difficult to distinguish by either humans or machines. Deepfakes are the most flagrant models among those, which can change a person's identity in the video. Since face videos contain sensitive personal information, abuse of these techniques will grow into a menace. The advent of the forged speech of Barack Obama [24] and manipulated pornographic videos of famous actresses [27] aroused great concern on the Internet. Besides celebrities, ordinary people can also fall victim to Deepfakes on account of the abundant amount of video clips on social platforms and freely fetchable implementations of Deepfakes. Therefore, how to detect Deepfakes videos becomes a matter of urgency.

Deepfakes detecting methods so far can be roughly classified into two types. The first type mainly focuses on defects in one single frame [22, 15, 21, 18, 25, 23, 16]. And the second type takes temporal features into account [17, 2, 26]. Some of the methods mentioned above mainly focus on non-essential defects of Deepfakes techniques (such as abnormal eye blinking or different colors of irises), which in return stimulated the improvement of Deepfakes video synthesis.

In the context of an arms race between Deepfakes generation and detection techniques, there are several challenges need to be encountered. Firstly, advanced manipulation approaches urge detectors to uncover the intrinsic characteristics of Deepfakes videos, which cannot be easily disguised. Secondly, detectors should be more robust, enabling them to perform well on real-world detection missions. For instance, a lot of models [5, 1, 16] witnessed severe performance drop on compressed videos, which reduces their effectiveness in application. Thirdly, the model simplicity should be taken into account. Current detection methods heavily rely on powerful Deep Convolutional Neural Networks (DCNNs) or data augmentation skills, which demands unendurable training costs. Also they are unfriendly for reproduction.

We make a key observation that although manipulated face videos show high fidelity in a single frame, they still reveal some subtle but unnatural expressions or facial organs' movements. This is an inherent defect of Deepfakes techniques because forge videos are generated frame-by-frame, and no strong restriction is imposed on both individual be-

[1] Github: https://github.com/frederickszk/LRNet

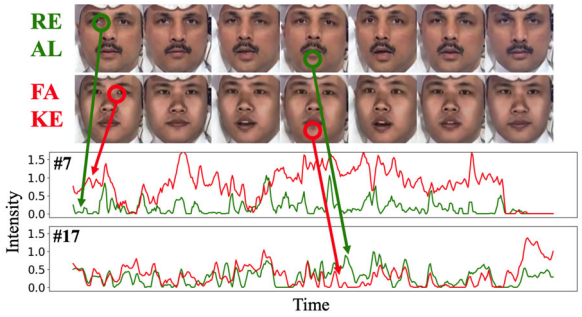

Figure 1. Action units (AU) intensity analysis for pristine and Deepfakes video sequence. AU indicate movements of individual facial muscles that make up the facial expression. We select the two most intense action units: #7 (lid tightener) and #17 (chin raiser). As we can see, although the fake sequence are too realistic to distinguish from appearance, we can still tell their differences on some subtle expressions, even though the faces in this two videos are performing the totally same action.

havior patterns and time continuity (as illustrated in Fig. 1). To better capture these "temporal artifacts", also taking model robustness and simplicity into account, we opt for *geometric features*, *e.g.*, the shape and the position of facial organs. They can be more explicit for modeling facial dynamic behavior. Facial landmarks are a set of points outlining the contours of iconic facial parts, which are sufficient for describing geometric information and suitable for our framework.

Previous works have shown the potential of geometric features (especially the facial landmarks) in exposing synthetic face images or videos [33, 34, 2]. However, they utilized hand-crafted or complex correlation-based features for classification, which are not optimal for capturing all of distinguishable dynamic properties. In contrast, we design a two-stream Recurrent Neural Network to extract deep temporal features on landmarks sequence. Moreover, none of them have considered the influence brought by imprecise facial landmarks, which could be harmful to obtain more meaningful features. We devise a novel landmark calibration module to enhance the descriminative abilities of geometric features by reducing jittering, which enables us to combine the geometric and deep temporal features reliably and construct our detection framework dubbed Landmark Recurrent Network (LRNet).

Our framework LRNet achieves complementary strengths. On one hand, replacing face images with landmarks can be seen as an effective data dimensionality reduction. Compare to other deep-learning based model, it not only reduces model redundancy but also is more invariant to corruption in videos. On the other hand, the deep RNNs help to enlarge the feature space and promote the expressiveness competence of facial landmarks. It strikes a better balance between cost and performance.

Our contributions can be concluded in three aspects:

- We propose an efficient and robust framework to classify Deepfakes videos where we model temporal characteristics on precise geometric features.
- We introduce a novel plug-and-use landmark calibration module to promote the precision of geometric features and the effectiveness of temporal modeling while enabling our framework to be more flexible and reproducible.
- We carry out extensive experiments to verify the efficiency and robustness of our method and also explore the influencing factors.

2. Related Work

2.1. Deepfakes Detection

In this part, we introduce the current progress in the field of Deepfakes detection.

Frame-level detection. Up to now, most of the efforts of the deepfakes detection are paid onto the single-frame based approach. Some of these techniques base on simple features selected manually. For instance, Matern et al. [21] focused on simple visual artifacts such as colors of irises, wired shadows on the face and missing details of eyes and teeth. The others turn to the deep features extract by DCNNs. Afchar et al. [1] proposed MesoNet based on mesoscopic properties of images. Rössler et al. [25] successfully transferred Xception [5] into deepfake detection task. Li et al. [16] utilized an advanced architecture named HRNet [28] to detect the blending boundary of Deepfakes manipulated images. These methods trade high costs for good performance thanks to powerful CNNs. Nonetheless, they are short of robustness or difficult to reproduce.

Video-level detection. Recently, the intuition that videos contain more information than images greatly inspires the Deepfakes detection based on temporal features. Some geometric features-based schemes have made valuable attempts. Li et al. [17] captured the abnormality of eye blinking frequency in fake videos. Yang et al. [33] used outer landmarks and central landmarks to compose head direction and face direction respectively and detected the inconsistency between them. The manually selected features are less discriminative which limits their performance, while we turn to exploit expressive deep temporal features. Appearance-based solutions are relatively more prevalent. Güera et al. [10] proposed a framework utilizing a CNN to extract features from frames and a LSTM to process the features sequence. Sabir et al. [26] adopted a similar architecture but replaced the LSTM with bidirectional Gated Recurrent Unit (GRU). These methods rely greatly on CNN as well, thus they suffer from similar problems as those frame-level detectors.

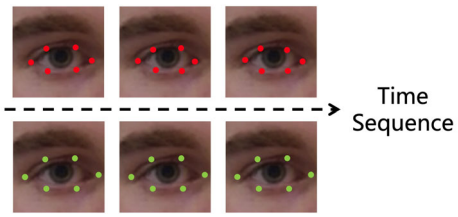

Figure 2. Comparison between accuracy and precision. Red landmarks (upper) are accurate but not precise. They jitter greatly even though they all attach to the contour. Green points (lower) are less accurate but precise, which describe dynamic properties better.

2.2. Landmarks Detection

Facial landmarks are representative and vital geometric features. It's detection methods have been widely studied for years. At the beginning, researchers proposed Active Appearance Model (AAM) [6] and Constrained Local Models (CLM) [7]. Afterwards, the detection methods based on Cascaded Shape Regression (CSR) [32, 13] achieved prominence. These methods make an initial estimate of the landmark locations then refine them iteratively through the ensemble of regression models (e.g., regression trees). They are adopted by widely-used open-source image processing repositories like `Dlib` [14], which are easy to use and fast in detection speed. Recently, abundant deep learning based models are devised such as Cascade CNN [36], Convolutional Pose Machine (CPM) [31], Convolutional Experts CLM [35], etc. Some are also implemented by open-source toolkits like `Openface` [4]. They have better performance while slower in speed. Furthermore, sophisticated architectures are introduced to resolve the problems of face occlusion, extreme head pose, and so on.

It is essential to ensure the *accuracy* and *precision* of detected landmarks because they are the decisive features in geometric-based Deepfakes detection. Specifically, the term "accurate" means the detection results have a low bias while "precise" refers to a low variance (as illustrated in Fig. 2). The precision is relatively more important because the jittering noise would severely disturb the temporal modeling. However, current landmarks detector mostly operate in frame-level and cannot achieve high precision. For this reason, we designed a calibration module to enhance the precision of landmarks detection results.

3. Methodology

Our proposed Deepfakes detection framework LRNet consists of four components (as shown in Fig. 3): face preprocessing module, calibration module, feature embedding procedure and RNN classifying procedure. It exposes manipulated faces by detecting abnormal facial movement patterns and time discontinuities. Unlike most of the proposed methods that need to be trained end-to-end, our framework

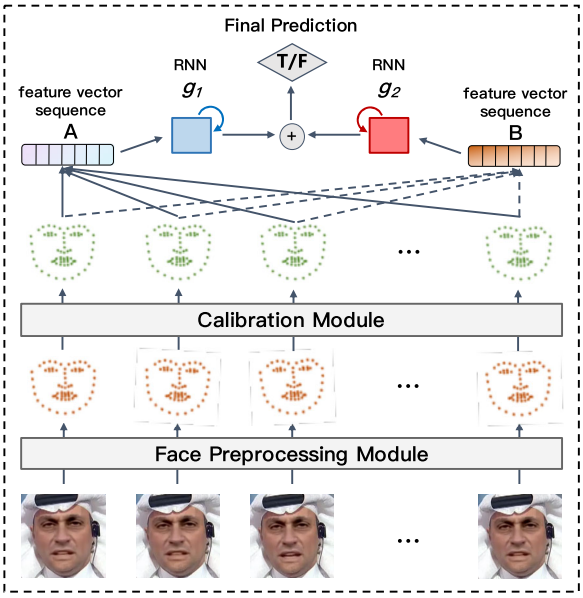

Figure 3. Overview of LRNet detection framework. the video to be detected is split into frames, and passed through the preprocessing procedure along with a carefully designed calibration module to obtain a sequence of more precise facial landmarks. The subsequent embedding procedure embed landmarks into two types of feature vectors, and a two-stream RNN is used to mine the temporal information and judge its authenticity.

only requires the training of the RNNs part. Details of our framework will be demonstrated below.

3.1. Face Preprocessing

The face preprocessing module extracts geometric information from face images. It consists of face detection, facial landmarks detection and landmarks alignment.

In the beginning, face detection is performed on each frame of the video, and we retain the Region Of Interest (ROI) of the face. After cropping out the face images, we detect 68 facial landmarks on them, which outline the iconic profile on the face. Finally, we align landmark points to a preset position implemented by affine transformation.

Noted that our framework is flexible enough to decouple the preprocessing module (more specifically, the landmark detector). Firstly, the landmark detector can be implemented by pre-trained models without any additional training. Secondly, the performance of our whole framework does not heavily rely on the landmark detector. This property is guaranteed by our proposed calibration module which will be discussed below. And we demonstrate it in details by experiments in section 4.3.1.

3.2. Landmark Calibration

The landmarks extract through the preprocessing module can basically meet the demand for accuracy. However, they

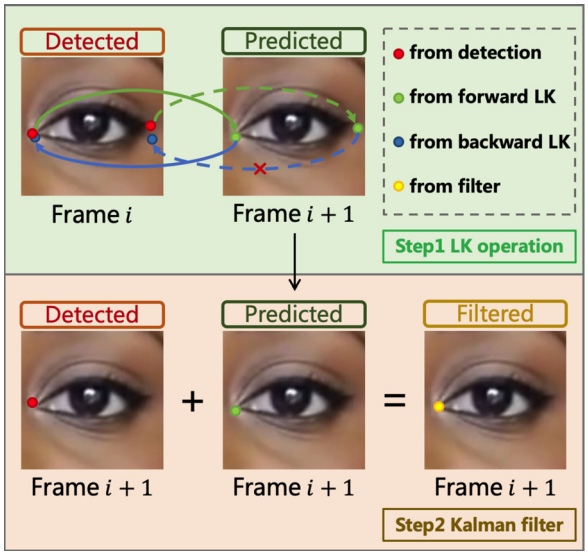

Figure 4. Detailed steps of the calibration module. The first step uses LK operation to track the landmarks point. Also a forward-backward check is performed to eliminate imprecise predictions. The second step uses Kalman filter to merge the detection and prediction results.

are far from precise for they are detected frame-by-frame. From our observation, the detected landmarks will have obvious jittering even if the face is almost not moving. Therefore, we proposed a novel calibration module to tackle this problem (as illustrated in Fig. 4). We use the successive frames to predict the latter positions of landmarks based on Lucas-Kanade optical flow calculation algorithm [20, 3]. And valid predictions are merged with its corresponding detection results by a customized Kalman filter [12] to denoise and obtain the calibrated landmarks with higher precision.

3.2.1 Tracking

For calibrating the landmarks, our intuition lies in that we can adjust their positions by matching small image patches around them. For this reason, Lucas-Kanade algorithm is suitable because it calculates the optical flow, the movement of several feature points between frames, essentially in the same way to this intuition. Motivated by works of Baker et al. [3] and Dong et al. [8], we proposed a pyramidal Lucas-Kanade operation (dubbed LK operation below) to predict the landmark positions, in other words, track the landmarks.

Given a small image patch $\mathbf{P}_i$ from frame$_i$ centered at $\mathbf{x}_i = [x, y]^{\mathrm{T}}$, where another same-size patch $\mathbf{P}_{i+1}$ from frame$_{i+1}$, we try to find a displacement vector $\mathbf{d} = [d_x, d_y]^{\mathrm{T}}$ to minimize the difference between $\mathbf{P}_i$ and $\mathbf{P}_{i+1}$, then we can obtain the tracking prediction $\mathbf{x}_{i+1} = \mathbf{x}_i + \mathbf{d}$. Therefore, we can calculate displacement vector $\mathbf{d}$ by minimizing

$$\sum_{\mathbf{x}\in\Omega} \alpha_{\mathbf{x}} \left\| \mathbf{P}_i(\mathbf{x} + \Delta\mathbf{d}) - \mathbf{P}_{i+1}(\mathbf{x} + \mathbf{d}) \right\|^2, \quad (1)$$

where $\mathbf{d}$ is firstly initialized to be $[0, 0]^{\mathrm{T}}$. From eq. (1) we can solve the $\Delta\mathbf{d}$ and iteratively update $\mathbf{d}$ by

$$\mathbf{d} \leftarrow \mathbf{d} + \Delta\mathbf{d} \quad (2)$$

until convergence. In Eq. (1), Ω refers to the set of all the positions in patch centered at $\mathbf{x}_i$, and $\alpha_{\mathbf{x}} = \exp(-\frac{\|\mathbf{x}-\mathbf{x}_i\|_2^2}{2\sigma^2})$, which is used to reduce the weights of locations far from the center and make a soft prediction.

According to the work of Baker et al. [3], we attain the final solution of Eq. (1) that:

$$\Delta\mathbf{d} = \mathbf{H}^{-1} \sum_{\mathbf{x}\in\Omega} J(\mathbf{x}) \alpha_{\mathbf{x}} (\mathbf{P}_{i+1}(\mathbf{x} + \mathbf{d}) - \mathbf{P}_i(\mathbf{x} + 0)). \quad (3)$$

In Eq. (3), $\mathbf{H} = \mathbf{J}^{\mathrm{T}}\mathbf{A}\mathbf{J} \in \mathbb{R}^{2\times 2}$ is the Hessian matrix. $\mathbf{J} \in \mathbb{R}^{C|\Omega|\times 2}$ is generated by vertically concatenating $J(\mathbf{x}) \in \mathbb{R}^{C\times 2} (\mathbf{x} \in \Omega)$, which is the Jacobian matrix of $\mathbf{P}_i(\mathbf{x} + 0)$. C is the number of channels of $\mathbf{P}_i$ (3 for RGB images). $\mathbf{A}$ is a diagonal matrix whose elements are composed of $\alpha_{\mathbf{x}}$ to weight the corresponding Jacobian of $\mathbf{x}$ in $\mathbf{J}$. The advantage of this solution lies in that $\mathbf{P}_i(\mathbf{x})$ is constant during the iteration, thus complicated $\mathbf{J}$ and $\mathbf{H}^{-1}$ are only computed for once.

Considering that LK operation is sensitive to the patch's size, We introduce pyramidal LK operation (detailedly described in Algorithm 1) that firstly downsamples the image several times (usually halve its size) to build pyramid representation of it, and perform simple LK operation on images of different size with the same patch size.

Noticed that LK operation is not always successful, thus a forward-backward check is introduced as shown in Fig. 4. We perform a forward LK operation (green arrows and points) on the former frame, and a backward LK operation (blue arrows and points) on predicted points from the latter frame back to the former frame. The predicted point with a large difference between its original point and backward LK point will be discarded (dotted arrows).

3.2.2 Denoising

We discover from experimental results that LK operation can also bring in noise, which disturbs the stability of landmarks. Consequently, we devise a customized Kalman filter to integrate the information from both detections and predictions instead of only the LK operation results.

Kalman filter estimates optimal landmark point $\mathbf{x}_{i+1}^{opt}$ in frame$_{i+1}$ through a weighted average of corresponding

Algorithm 1: Pyramidal LK operation

Input: former frame $\mathbf{F}_i$, latter frame $\mathbf{F}_{i+1}$, point in former frame to be tracked $\mathbf{x}_i$
Output: predicted point in letter frame $\mathbf{x}_{i+1}$

Build pyramid representation of F_i and F_{i+1}: $\{F_i^L\}_{L=0,...,L_m}$, $\{F_{i+1}^L\}_{L=0,...,L_m}$;
$\mathbf{g}^{L_m} \leftarrow [0,0]^{\mathrm{T}}$;
for $L = L_m; L \geq 0; L--$ **do**
  $\mathbf{x}_i^L \leftarrow \mathbf{x}_i / 2^L$;
  Extract patch P_i^L from F_i^L centered at $\mathbf{x}_i^L$;
  Compute the Jacobian matrix, $\mathbf{J}$ and $\mathbf{H}$ for P_i^L;
  $\mathbf{d}^L \leftarrow [0,0]^{\mathrm{T}}$;
  for $i = 1 : max$ **do**
    Extract patch P_{i+1}^L from F_{i+1}^L centered at $\mathbf{x}_i^L + \mathbf{d}^L$;
    Compute $\Delta\mathbf{d}^L$ by Eq. (3);
    Update $\mathbf{d}^L$ by Eq. (2);
  end
  $\mathbf{g}^{L-1} \leftarrow 2(\mathbf{g}^L + \mathbf{d}^L)$;
end
Obtain final prediction: $\mathbf{d} \leftarrow \mathbf{g}^0 + \mathbf{d}^0$;
Return: $\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \mathbf{d}$;

Algorithm 2: Landmarks calibration

Input: $\mathbf{L}_i$, $\mathbf{L}_{i+1}$, $\mathbf{F}_i$, $\mathbf{F}_{i+1}$
Output: Calibrated landmarks $\hat{\mathbf{L}_{i+1}}$

for $\mathbf{x}_i \in \mathbf{L}_i$ **do**
  $\mathbf{x}_{i+1} \leftarrow$ Algorithm 1 ($\mathbf{F}_i$, $\mathbf{F}_{i+1}$, $\mathbf{x}_i$);
  $\tilde{\mathbf{x}}_i \leftarrow$ Algorithm 1 ($\mathbf{F}_{i+1}$, $\mathbf{F}_i$, $\mathbf{x}_{i+1}$);
  Perform forward-backward check with $\mathbf{x}_i$ and $\tilde{\mathbf{x}}_i$;
  if $\mathbf{x}_i$ is valid **then** `/* Kalman filter */`
    Calculate D_{i+1}^r by Eq. (7);
    Calculate K_{i+1} by Eq. (5);
    Estimate $\mathbf{x}_{i+1}^{opt}$ by Eq. (4);
    Update P_{i+2} by Eq. (6);
    $\hat{\mathbf{x}_{i+1}} \leftarrow \mathbf{x}_{i+1}^{opt}$;
  else
    $\hat{\mathbf{x}_{i+1}} \leftarrow$ corresponding $\mathbf{x}_{i+1} \in \mathbf{L}_{i+1}$;
  end
end
Return: $\hat{\mathbf{L}_{i+1}} = [\hat{\mathbf{x}_{i+1}^1}, ..., \hat{\mathbf{x}_{i+1}^{68}}]^{\mathrm{T}}$;

landmark detection result $\mathbf{x}_{i+1}^{det}$ and LK operation tracking prediction $\mathbf{x}_{i+1}^{pred}$. This procedure can be represented by:

$$\mathbf{x}_{i+1}^{opt} = \mathbf{x}_{i+1}^{pred} + K_{i+1} \cdot (\mathbf{x}_{i+1}^{det} - \mathbf{x}_{i+1}^{pred}), \tag{4}$$

where K_{i+1} is the Kalman gain when estimating $\mathbf{x}_{i+1}^{opt}$. It can be computed by:

$$K_{i+1} = \frac{P_{i+1}}{P_{i+1} + D_{i+1}}, \tag{5}$$

where P_{i+1} is the variance (indicating the instability) of LK operation when predicting $\mathbf{x}_{i+1}^{pred}$, and D_{i+1} is similarly the variance of landmark detection when detecting $\mathbf{x}_{i+1}^{det}$. Afterwards, we update next LK operation's variance P_{i+2} by:

$$P_{i+2} = (1 - K_{i+1}) \cdot P_{i+1} + Q, \tag{6}$$

where Q is the inherent variance of LK operation.

However, it's difficult to calculate P and D because neither LK operation nor landmark detector is a simply-representable mathematical model. We thus propose the conception of **approximate relative variance** D^r that:

$$D_{i+1}^r = \frac{\mathbf{x}_{i+1}^{det} - \mathbf{x}_i}{\mathbf{x}_{i+1}^{pred} - \mathbf{x}_i}. \tag{7}$$

Due to the fact that each ground-truth landmark point in successive frame won't shift greatly, if the detection result have a apparent vibration, D^r will bigger than 1. We then empirically set $Q = 0.3$ according to visual effect in experiments, and replace D with D^r when computing Eq. (5).

Our calibration module depends on the landmarks of frame$_1$ to calibrate the landmarks of frame$_2$. Then these optimized landmarks of frame$_2$ will be used to calibrate the frame$_3$ and so on. Given the lanmarks $\mathbf{L}_i = [\mathbf{x}_i^1, ..., \mathbf{x}_i^{68}]^{\mathrm{T}}$ and $\mathbf{L}_{i+1}$ extracted from frame $\mathbf{F}_i$ and $\mathbf{F}_{i+1}$ the overall procedure of landmark calibration is detailedly describe in Algorithm 2. For simplicity we only express a single calibration step in it.

3.3. Fake video classification

The landmarks sequence extracted and calibrated in above steps are embedded into two types of feature vectors sequence, and then input to a two-stream RNN for fake video classification.

Each landmark point $\mathbf{x}^a$ can be represented by $\mathbf{x}^a = [x^a, y^a]^{\mathrm{T}}$, thus the first type of feature vector $\boldsymbol{\alpha}_i$ embedded from landmarks $\mathbf{L}_i = [\mathbf{x}_i^1, ..., \mathbf{x}_i^{68}]^{\mathrm{T}}$ can be generated by:

$$\boldsymbol{\alpha}_i = \left[x_i^1, \quad y_i^1, \quad x_i^2, \quad y_i^2, \quad ..., \quad x_i^{68}, \quad y_i^{68}\right],$$

which can be seen as directly flatten from $\mathbf{L}_i$. Then the second type feature vector $\boldsymbol{\beta}_i$ can be computed by:

$$\begin{aligned}\boldsymbol{\beta}_i &= \boldsymbol{\alpha}_{i+1} - \boldsymbol{\alpha}_i \\ &= \left[x_{i+1}^1 - x_i^1, \quad ..., \quad y_{i+1}^{68} - y_i^{68}\right],\end{aligned}$$

representing the difference of landmarks' positions between successive frames.

Through embedding we obtain two feature vectors sequences $\mathbf{A} = [\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_n]^{\mathrm{T}}$ and $\mathbf{B} = [\boldsymbol{\beta}_1, ..., \boldsymbol{\beta}_{n-1}]^{\mathrm{T}}$. Afterwards, one RNN g_1 models facial shape movement pattern on $\mathbf{A}$, and the other RNN g_2 model landmarks difference pattern (or can be seen as the speed pattern, which is used to capture time discontinuity) on $\mathbf{B}$. Fully-connected layers are attached to each RNN's output to make its own prediction and the two streams are averaged as the final prediction. We conclude these prediction-making operation in one function $f(\cdot, \cdot)$. Therefore the final prediction, i.e., the real or fake possibility of a video clip, can be noted as:

$$f(g_1(\mathbf{A}), g_2(\mathbf{B})). \tag{8}$$

To perform the video-level detection, each video sample is segmented into clips with a fixed length. The predicted labels for clips are aggregated for the prediction of video.

4. Experiments

In this section, we firstly declare the experiment settings. Then we evaluate the efficiency of our proposed LRNet framework on several benchmarks. Furthermore, we analyze the influencing factors of LRNet.

4.1. Experiment Setting

4.1.1 Datasets

Several challenging datasets have been proposed throughout the research progress of Deepfakes. To make the evaluation representative and comprehensive, we selected 4 typical datasets.

UADFV [17] contains 49 pristine videos and 49 manipulated videos. It represents the early dataset and adopted by a lot of classical works.

FaceForensic++ (FF++) [25] contains 1000 videos as well as their manipulated version. Each video has a original version (raw), slightly-compressed version (c23) and heavily-compressed version (c40). It's the most typical recent dataset and has been widely adopted.

Celeb-DF [19] and DeeperForensics-1.0 (DF1.0)[11] are two newly proposed datasets with high visual quality. Celeb-DF contains 5639 fake videos and 540 real videos, and DF1.0 contains 1000 real and corresponding fake videos similar to FF++. Each work also provide a benchmark which facilitates our evaluation.

4.1.2 Parameters and implementation details

In preprocessing step, we adopt `Dlib` [14] to carry out face and landmark detection (another detector `Openface`[4] is adopt in the ablation study). In classification procedure, Each RNN in our two-stream network is bidirectional and consists of GRU (Gated Recurrent Unit) whose number of output units is set to be $k = 64$. And two fully-connected layers with the number of units to be 64 and 2 are connected to RNN layer's output. A dropout layer with drop rate $dr_1 = 0.25$ is inserted between input and RNN, and another 3 dropout layers with $dr_2 = 0.5$ are used to separate the rest of the layers. These settings are partially based on existing reserach results [26]. In addition, we adopt an 8:2 dataset split, i.e., 80% for training and 20% for testing. Each video is segmented into clips with a fixed length of 60, which sum up to 2 seconds when the fps is 30. We adopt Adam optimizer with $lr = 0.001$, and batch size is set to be 1024. This classification model will be trained up to 500 epochs.

Methods	Configurations			Testing Datasets		
	Size	Aug.	Training	UADFV	FF++	Celeb-DF
Meso4 [1]	0.03 M	×	Unpub.	84.3	84.7	54.8
FWA [18]	26 M	√	Unpub.	97.4	80.1	56.9
DSP-FWA [18]	28 M	√	Unpub.	97.7	93.0	**64.6**
Xception [25]	20.8 M	×	FF++	80.4	99.7	48.2
Capsule [23]	15 M	×	FF++	61.3	96.6	57.5
CNN+RNN [26]	24.3 M	×	FF++	70.9	98.3	61.5
LRNet (ours)	0.18 M	×	FF++	**98.5**	**99.9**	56.9

Table 1. General performance evaluation by AUC socres (%) on different testing datasets. "Aug." refers to if the method adopts data augmentation. Our proposed LRNet is relatively lightweight in the model's size and not in need of data augmentation, while performs the best on FF++.

4.2. Performance Evaluation

4.2.1 General evaluation

We firstly make a general evaluation of LRNet based on Celeb-DF benchmark [19]. Because a big part of currently proposed detection methods didn't open-source, making themselves difficult to be reproduced, we follow the evaluation setting of Celeb-DF benchmark that train our model on one dataset (mostly FF++) and test on different datasets. The evaluation metric is AUC scores (Area Under ROC Curve) and the results are shown in Table 1. We only show the results of part of the best-performance methods. Our method achieves an almost full AUC score on its training dataset FF++ (99.9), showing that it can effectively capture the abnormal movements and discontinuities. Besides, it can also generalize to other datasets (unseen samples).

4.2.2 Robustness to video compression

We further test our methods robustness to video compression, which is overlooked by the majority of current works. On FF++, we compare the best detector on its benchmark, Xception, as well as newly-proposed and advanced X-Ray [16]. Each detector is trained on original video (raw) and tested on three version of videos with different compression rates. On Celeb-DF, we used its benchmark settings

that Xcpetion trained with FF++(c23) and FWA(DSP-FWA) trained with data augmentation. While our method directly train on FF++(raw). The results are shown in Table 2. We can draw from the results that the performance our methods is relatively more invariant to video compression.

Methods	FF++			Decline
	raw	c23	c40	
Xception [25]	99.7	93.3	86.5	6.4/13.2
X-Ray [16]	99.1	87.3	61.6	11.8/37.5
LRNet (ours)	**99.9**	**97.3**	**95.7**	**2.6/4.2**
Methods	Celeb-DF			Decline
	raw	c23	c40	
Xception-c23 [25]	**65.3**	**65.5**	52.5	**-0.2**/12.8
FWA [18]	56.9	54.6	52.2	2.3/4.7
DSP-FWA [18]	64.6	57.7	47.2	6.9/17.4
LRNet (ours)	57.4	56.3	**55.4**	1.1/**2.0**

Table 2. AUC scores (%) of different methods when encountering video compression.

4.2.3 Robustness to video noise

We in addition challenge our proposed LRNet on videos corrupted by different noise. We select DF1.0 benchmark, which is suitable for this evaluation. Because it provides this setting and tests on various advanced video-level detection methods that are ignored by other benchmarks. The results are shown in Table 3. We can see that all of the methods perform well on the same training and testing dataset (include our LRNet who achieves 97.74% acc. and 99.2% AUC). While our methods suffer from the least performance decline when faced with noise.

Methods	Train/Test		Decline
	std/std	std/noise	
C3D [29]	98.50	87.63	10.87
TSN [30]	99.25	91.50	7.75
I3D [29]	**100.00**	90.75	9.25
CNN+RNN [26]	100.00	90.63	9.37
Xception [25]	100.00	88.38	11.62
LRNet (Ours)	97.74	**96.83**	**0.91**

Table 3. Comparison of different method's robustness to video noise, which can are evaluated by binary classification accuracy (%). "std" refers to clean samples and "noise" refers to samples with several types of strong noise as described in benchmark [11].

4.2.4 Efficiency in training

To better demonstrate our framework's efficiency, we carry out evaluations over several representative baseline external methods and show the results in Table 4. All models require similar pre-processing operations and our proposed LRNet consumes acceptable time, i.e., only 2h more than the basic requirement (6h). While our model is significantly faster and less memory intensive in training.

Model	Pre-processing		Training			
	Operations	Time	#Param	GPU	DISK	Time
Xception[5]	F+L+A	6h	20.8M	12G	64G	21h
X-Ray[16]	F+L+A+D	24h	37.7M	>12G	>180G	>30h
CNN+RNN[26]	F+L+A	6h	24.3M	9G	64G	22.5h
TSN[30]	F+L+A+O	10h	22.5M	>12G	>120G	>30h
LRNet(Ours)	**F+L+A+C**	**8h**	**0.18M**	**3G**	**1.1G**	**0.2h**

Table 4. Quantitative comparisons for training cost (on FF++). The operations include face detection(F), landmark detection(L), alignment(A), data augmentation(D), optical flow(O) and proposed landmark calibration(C). #Param refers to the trainable parameter size of the model. We also evaluate the GPU memory occupation (GPU), and disk memory occupied by training data (DISK).

	train	Dlib	
test	calib	Y	N
OpenFace	Y	98.5	85.5
	N	81.6	90.2

	train	OpenFace	
test	calib	Y	N
Dlib	Y	98.2	90.6
	N	71.6	78.4

Figure 5. The confuse matrices for train on one kind of landmarks and test with the other. We evaluate the accuracy (%) of each setting on FF++(raw). "Y" means that detecting landmarks with our calibration module while "N" means not.

$\times 10^{-2}$	jaw	eyebrow (left)	eyebrow (right)	nose	eye (left)	eye (right)	lips
raw	5.87	3.26	3.45	2.49	1.16	1.15	3.67
raw+calib	5.34	3.04	3.25	2.21	0.89	0.89	3.08

Figure 6. The average distance between the landmarks detected by different detectors (Dlib and OpenFace here). "+calib" means utilize our calibration module. We merge the 68 landmarks into 7 groups by the organs they belong to.

4.3. Framework Analysis

4.3.1 Effect of calibration module

Landmarks calibration is the core component of LRNet. We perform an ablation study on it (as shown in Tabel 5). We can see that the calibration module enhances the overall performance of the detection framework as well as keeping its robustness.

We take a step further in evaluation. Firstly we try to detect landmarks with a better-performing CNN-based detector, `Openface` [4], and retrain the RNNs part of LRNet. The performances are very similar to the results of using

Methods	FF++(raw)		FF++(c40)	
	Acc	AUC	Acc	AUC
LRNet	99.7	99.9	91.2	95.7
w/o Kalman filter	98.5	99.4	87.2	94.3
w/o calibration	92.8	97.4	84.3	92.6

Table 5. Ablation study of calibration module in LRNet by evaluating the binary detection accuracy (%) and AUC scores (%). All the models are only trained on FF++(raw).

Methods	FF++(raw)		FF++(c40)	
	Acc	AUC	Acc	AUC
LRNet ($g_1 + g_2$)	99.7	99.9	91.2	95.7
g_1	83.4	89.3	80.4	86.1
g_2	98.3	99.2	85.2	93.9

Table 6. Ablation study of network architecture by evaluating the binary detection accuracy (%) and AUC scores (%). g_1 refers to the RNN which models abnormal facial movements and g_2 refers to the other RNN which captures time discontinuity.

`Dlib` and we do not list them in detail here. Then we explore the influence of not-retraining the model, i.e., feeding `Openface` landmarks into LRNet trained by `Dlib` landmarks, vice versa. As shown in Fig. 5, only with the help of calibration module can it avoid a performance drop. The reason lies in that model trained by calibrated landmarks can better capture the abnormal facial movements instead of the noise brought by landmark detectors. We further demonstrate this by calculating the differences between landmarks from different detectors as shown in Fig. 6. Calibration module helps shorten the gap of different landmarks detections, bring them closer to the ground-truth positions.

4.3.2 Effect of network architecture

We demonstrate the effectiveness of our two-stream RNN architecture by comparing it with only using its one stream (as shown in Table 6). The results show that this structure has a superior ability. The information from two roads promotes each other, where time discontinuity clues detected by g_2 contribute more to final accuracy and abnormal movements messages recognized by g_1 provide more robustness to the prediction.

4.3.3 Influence of input length

Input length refers to how many successive frames we feed it into the model as a single sample. We evaluate different input lengths with other conditions controlled (shown in Fig. 7). Despite the fact that input length hardly affects the performance when training and testing on the same dataset, a suitable input length (60 adopted by us) can improve both the effectiveness and robustness on different data distribution (e.g., compressed samples).

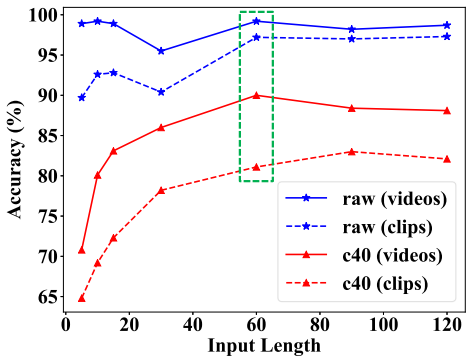

Figure 7. Accuracy (%) of LRNet when the input length varies. All the models are trained on FF++(raw). "(clips)" refers to the sample-wise detection accuracy and "(videos)" is the result of video-level classification.

5. Discussion

In this section, we discuss the our limitations and future works at first. From general evaluation results, there is still room for improvement in the generalizability of our framework. Besides, it's difficult to interpret the temporal features captured by our model and visualize the difference of movement pattern between real and fake faces. In future we will carry out a more in-depth research and analysis of these abnormal dynamic patterns of different face manipulation techniques, and then promote the model's generalizability.

We also expound the effect of appearance and geometric features. From the results of current works, appearance features are high-dimensional, more expressive to generalize, but less robust and high-cost. While geometric features are more robust, low-cost, but harder to generalize. So it's a trade-off of performance (especially generalization ability) and cost. While we make great efforts to promote the efficiency of Deepfakes detection only rely on the geometric features, it's worthy to explore if we can combine the appearance and geometric together at the same time avoiding high cost to improve the efficiency.

6. Conclusion

Deepfakes are huge threats to human society and their rapid development is calling for efficient solutions. Our work reveals that integration of facial landmarks and temporal features can be a fast and robust test of Deepfakes. We also explore how to enhance the landmark detection results and make full use of temporal features. We found that the facial geometric information and its dynamic characteristic are essential clues and worth exploring in future work for a more efficient and robust in-wild Deepfakes detection.

References

[1] Darius Afchar, Vincent Nozick, Junichi Yamagishi, and Isao Echizen. Mesonet: a compact facial video forgery detection network. In *IEEE International Workshop on Information Forensics and Security*, pages 1–7, 2018. 1, 2, 6

[2] Shruti Agarwal, Hany Farid, Yuming Gu, Mingming He, Koki Nagano, and Hao Li. Protecting world leaders against deep fakes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 38–45, 2019. 1, 2

[3] Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004. 4

[4] Tadas Baltrusaitis, Amir Zadeh, Yao Chong Lim, and Louis-Philippe Morency. Openface 2.0: Facial behavior analysis toolkit. In *IEEE International Conference on Automatic Face & Gesture Recognition*, pages 59–66, 2018. 3, 6, 7

[5] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1251–1258, 2017. 1, 2, 7

[6] Timothy F. Cootes, Gareth J. Edwards, and Christopher J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001. 3

[7] David Cristinacce and Timothy F Cootes. Feature detection and tracking with constrained local models. In *British Machine Vision Conference*, pages 929–938, 2006. 3

[8] Xuanyi Dong, Shoou-I Yu, Xinshuo Weng, Shih-En Wei, Yi Yang, and Yaser Sheikh. Supervision-by-registration: An unsupervised approach to improve the precision of facial landmark detectors. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 360–368, 2018. 4

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 1

[10] David Güera and Edward J Delp. Deepfake video detection using recurrent neural networks. In *IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 1–6, 2018. 2

[11] Liming Jiang, Ren Li, Wayne Wu, Chen Qian, and Chen Change Loy. Deeperforensics-1.0: A large-scale dataset for real-world face forgery detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2889–2898, 2020. 6, 7

[12] Rudolph Emil Kalman et al. A new approach to linear filtering and prediction problems [j]. *Journal of basic Engineering*, 82(1):35–45, 1960. 4

[13] Vahid Kazemi and Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1867–1874, 2014. 3

[14] Davis E King. Dlib-ml: A machine learning toolkit. *The Journal of Machine Learning Research*, 10:1755–1758, 2009. 3, 6

[15] Haodong Li, Bin Li, Shunquan Tan, and Jiwu Huang. Detection of deep network generated images using disparities in color components. *arXiv preprint arXiv:1808.07276*, 2018. 1

[16] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5001–5010, 2020. 1, 2, 6, 7

[17] Yuezun Li, Ming-Ching Chang, and Siwei Lyu. In ictu oculi: Exposing ai generated fake face videos by detecting eye blinking. *arXiv preprint arXiv:1806.02877*, 2018. 1, 2, 6

[18] Yuezun Li and Siwei Lyu. Exposing deepfake videos by detecting face warping artifacts. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 46–52, 2019. 1, 6, 7

[19] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, and Siwei Lyu. Celeb-df: A large-scale challenging dataset for deepfake forensics. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3207–3216, 2020. 6

[20] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, pages 674–679, 1981. 4

[21] Falko Matern, Christian Riess, and Marc Stamminger. Exploiting visual artifacts to expose deepfakes and face manipulations. In *IEEE Winter Applications of Computer Vision Workshops*, pages 83–92, 2019. 1, 2

[22] Scott McCloskey and Michael Albright. Detecting gan-generated imagery using color cues. *arXiv preprint arXiv:1812.08247*, 2018. 1

[23] Huy H Nguyen, Junichi Yamagishi, and Isao Echizen. Capsule-forensics: Using capsule networks to detect forged images and videos. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2307–2311, 2019. 1, 6

[24] Aja Romano et al. Jordan peele's simulated obama psa is a double-edged warning against fake news. *Australasian Policing*, 10(2):44, 2018. 1

[25] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. *arXiv preprint arXiv:1901.08971*, 2019. 1, 2, 6, 7

[26] Ekraam Sabir, Jiaxin Cheng, Ayush Jaiswal, Wael AbdAlmageed, Iacopo Masi, and Prem Natarajan. Recurrent convolutional strategies for face manipulation detection in videos. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 80–87, 2019. 1, 2, 6, 7

[27] Russell Spivak. "deepfakes": The newest way to commit one of the oldest crimes. *The Georgetown Law Technology Review*, 3(2):339–400, 2019. 1

[28] Ke Sun, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang. High-resolution representations for labeling pixels and regions. *arXiv preprint arXiv:1904.04514*, 2019. 2

[29] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *IEEE International Conference on Computer Vision*, pages 4489–4497, 2015. 7

[30] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision*, pages 20–36, 2016. 7

[31] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016. 3

[32] Xuehan Xiong and Fernando De la Torre. Supervised descent method and its applications to face alignment. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 532–539, 2013. 3

[33] Xin Yang, Yuezun Li, and Siwei Lyu. Exposing deep fakes using inconsistent head poses. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8261–8265, 2019. 2

[34] Xin Yang, Yuezun Li, Honggang Qi, and Siwei Lyu. Exposing gan-synthesized faces using landmark locations. *arXiv preprint arXiv:1904.00167*, 2019. 2

[35] Amir Zadeh, Yao Chong Lim, Tadas Baltrusaitis, and Louis-Philippe Morency. Convolutional experts constrained local model for 3d facial landmark detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 2519–2528, 2017. 3

[36] Erjin Zhou, Haoqiang Fan, Zhimin Cao, Yuning Jiang, and Qi Yin. Extensive facial landmark localization with coarse-to-fine convolutional network cascade. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 386–391, 2013. 3