Exploring Temporal Coherence for More General Video Face Forgery Detection

Yinglin Zheng\textsuperscript{1}, Jianmin Bao\textsuperscript{2}, Dong Chen\textsuperscript{2}, Ming Zeng\textsuperscript{1,*}, Fang Wen\textsuperscript{2}

\textsuperscript{1} School of Informatics, Xiamen University
\textsuperscript{2} Microsoft Research Asia

\{zhengyinglin@stu., zengming@\}xmu.edu.cn, \{jianbao, doch, fangwen\}@microsoft.com

Abstract

Although current face manipulation techniques achieve impressive performance regarding quality and controllability, they are struggling to generate temporal coherent face videos. In this work, we explore to take full advantage of the temporal coherence for video face forgery detection. To achieve this, we propose a novel end-to-end framework, which consists of two major stages. The first stage is a fully temporal convolution network (FTCN). The key insight of FTCN is to reduce the spatial convolution kernel size to 1, while maintaining the temporal convolution kernel size unchanged. We surprisingly find this special design can benefit the model for extracting the temporal features as well as improve the generalization capability. The second stage is a Temporal Transformer network, which aims to explore the long-term temporal coherence. The proposed framework is general and flexible, which can be directly trained from scratch without any pre-training models or external datasets. Extensive experiments show that our framework outperforms existing methods and remains effective when applied to detect new sorts of face forgery videos.

1. Introduction

With the development of deep generative models, especially Generative Adversarial Networks (GANs) [29, 40, 7, 38, 31], Current face manipulation techniques [31, 47, 48, 49, 28, 53, 52] are capable of manipulating the attributes or even the identity of face images. These forged images are even difficult to distinguish by humans, and thus may be abused for spreading political propaganda, damaging our trust in online media. Therefore, detecting face forgery is of paramount importance.

Most previous methods [60, 61, 43, 45] are trained for known face manipulation techniques. But they experience a dramatic drop in performance when the manipulation methods are unseen. Some recent works [34, 57, 18, 59, 21, 42, 36, 11] have noticed this problem and attempted to boost the generalization. However, these methods are vulnerable to common perturbations such as image or video compression, noise, and so on. They still show limited generalization capability. A particularly effective work is Face X-ray [32], which proposes to detect blending artifacts instead of generative artifacts. However, the blending artifacts are typically low-level information that is susceptible to post-processing operations. More recent work LipForensics [22] proposes to detect unnatural mouth motion using spatio-temporal neural networks. But they only pay attention to the mouth which may ignore the artifacts in the other region of the face.

In this paper, we propose to leverage temporal coherence for more general face forgery detection. We observe that most face video forgeries are generated in a frame-by-frame manner. Since each altered face is generated independently, it inevitably leads to obvious flickering and discontinuity of
the face area (see Figure 1). So we can leverage the temporal incoherence for more general and robust video face forgery detection. Previous works try to leverage spatio-temporal convolution network [21] or the recurrent neural network [5, 36] to learn temporal incoherence. However, we find that they all failed to learn the general temporal incoherence.

After careful investigation, we find that forged face videos mainly contain two types of artifacts, one is spatially related (e.g., blending boundary, checkboard, blur artifacts), the other is the temporal incoherence. Normally, the spatially related artifacts are more significant than the temporal incoherence. Without any specific design, current video face forgery detection methods [21, 5, 36] may rely more on spatial-related artifacts instead of the temporal incoherence for classification.

To encourage the spatio-temporal convolution network to learn the temporal incoherence, we redesign the convolution operator and propose a fully temporal convolution network (FTCN). The key idea is to restrict the network’s capability for handling spatial-related artifacts. So we set the kernel size of all spatial (height and width) dimensions to 1 and keep the original kernel size of the time dimension in the 3D convolution operator. Due to the extremely low field for spatial dimension, the network learns to classify by temporal-related artifacts and hardly applies the spatial artifacts for detection. Also, we notice that even if the convolution operator is only time-dependent, its capability is sufficient to distinguish between real or fake.

Moreover, we find some discontinuity may happen in frames that are not in the neighborhood, for example, the wrinkles or moles of a face may gradually appear or disappear. To handle this issue, we propose to leverage Transformer [51] for capturing long-range dependencies along the time dimension. We add a lightweight Temporal Transformer after the proposed FTCN. The FTCN and the Temporal Transformer are trained end-to-end as the whole framework for general video face forgery detection.

Our approach is general and flexible. It can achieve impressive results without any pre-training knowledge or hand-crafted datasets. In contrast, previous work LipForensics [22] relies heavily on pre-training and Face X-ray relies on hand-crafted dataset. More importantly, without any manual annotations, our method can locate and visualize the temporal incoherence in the face forgery videos.

We conduct extensive experiments to compare its performance with the state-of-the-art in various challenging scenarios. We find that our method significantly outperforms previous methods in terms of generalization capability to unseen forgeries, and robust to various perturbations on videos. Furthermore, we perform ablation studies to validate the design choices of our framework.

Our contributions are summarized as follows:

- We explore to take full advantage of temporal coherence for face forgery detection and propose a framework that combined fully temporal convolution network (FTCN) and Temporal Transformer to explicitly detect temporal incoherence.
- Equipped with our detector, we can locate and visualize the temporal incoherence part of the face forgeries.
- Extensive experiments on various datasets demonstrate the superiority of our proposed methods with respect to generalization capability to unseen forgeries.

2. Related Work

Due to the emergence of high-fidelity face manipulation techniques, the detection of face forgery becomes an increasingly important research area. We will briefly introduce previous work on face forgery detection in this section.

Image Face Forgery Detection. Early studies put more emphasis on the spatial artifacts on the generated images, so they leverage the capability of convolution neural networks to apply deep CNN models [3, 9, 25, 43] to train a binary classifier to distinguish real or fake. Meanwhile, a significant amount of works explores low-level image statistics (e.g., frequency, color) [42, 30, 19, 58, 20] or high-level semantics (e.g., identities) [41] of face images for forgery detection. Some recent studies [8, 4, 14, 27, 56] aim to locate the visual artifacts in the forged images and make predictions based on the location results.

Video Face Forgery Detection. More recently, a great number of works start to take the temporal dimension into consideration and conducting face forgery detection at the video level. Li et al. [33] introduce leveraging eye blinking for detecting generated fake face videos. Amerini et al. [6] suggests using the optical flow between video frames. Mittal et al. [37] use the effective cues between audio and visual appearance for detecting fake videos. In contrast to these methods, our methods put more emphasis on general temporal incoherence. Unlike previous works, they either directly apply 3D convolution network [15, 21] on video or detect a specific kind of incoherence, such as irregular eye blinking [33], lip motions [22] or emotion [37]. In this work, we seek to detect general temporal incoherence, which can be any inconsistent region along time dimension.

Generalization to Unseen Manipulations. With the evolution of novel face manipulation techniques, many current detectors [43, 3] can experience a significant performance drop. Many works have noted this problem and presented some solutions for improving the generalization capability of detectors. FWA [34] explores to detect the resolution differences between altered face and background for general deepfake detection. LAE [18] and Multi-task [39] propose to learn a segmentation mask for the manipulated area in order to get a general detector. PatchForensics [11] suggests that the patch-based classifier can improve the per-
3. Methods

3.1. Motivation

Currently, most face manipulation methods [31, 47, 48, 49, 53, 52] are specifically crafted at the image level. To generate a fake video, current techniques need to apply their methods for each frame independently. However, subtle changes in the appearance (e.g., noise, lighting, motion) often result in temporal incoherent results (e.g., flickering and discontinuous results as shown in Figure 1). Some prior face manipulation techniques [28, 35] have noticed this problem and apply post-processing tools to tackle this issue, but the generated videos still suffer from the problem of temporal incoherence to some extent. Therefore, how to detect temporal incoherence is worthy of more careful study.

Detecting the temporal incoherence is challenging since we do not have the location annotation of the incoherence in the video. A naive idea is to adopt the spatio-temporal convolution networks [21, 15] and expect the model to learn to distinguish real or fake by temporal incoherence. However, we find that forged face videos mainly contain two types of artifacts, one is the spatially related (e.g., blending boundary, checkerboard, blur artifacts), the other is the temporal incoherence. Normally, the spatially related artifacts are more significant than the temporal incoherence. Without any specific design, the spatio-temporal convolution network distinguishes real or fake using spatial artifacts instead of temporal incoherence.

So the problem becomes how to encourage the spatio-temporal convolution network to learn the temporal incoherence. We take a fundamentally different approach, we propose a fully temporal convolution network. Concretely, we keep all the temporal-related convolution kernel size as the original but set all the spatial-related convolution kernel size to 1. We find this restriction can encourage the network to learn the temporal incoherence. To prove that, we take ResNet-50(R50) [24] as backbone and compare three types of classifiers:

2. A 2D R50 network structure, which uses 2D convolutions.
3. The proposed 3D R50-FTCN. We use 3D R50 as the backbone and set all the spatial-related convolution kernel size to 1 and keep the temporal-related kernel size.

To ensure fair comparison, all classifiers use the same training set FF++ [43] and the same training and inference settings. We show the t-SNE visualization of features extracted from different classifiers on FF++ test set in Figure 2. We have the following observations: Although all classifiers can distinguish between real and fake data, the distribution of fake data is completely different. Both the 3D R50 and 2D R50 will separate fake data generated by different face manipulation methods, even if we treat all fake data as one class in the training stage. It clearly shows that the features they extract contain the unique artifacts of each face manipulation algorithm. This would affect their generalization ability. On the contrary, the fake data of the 3D R50-FTCN classifier are more mixed together. It proves that the temporal network learns to classify by more general temporal incoherence.

On the other hand, some temporal incoherence exists in the long-range of video frames. However, prior studies [55, 26] indicate that temporal convolution struggles in dealing with the long-range dependencies. To tackle this issue, we add a Temporal Transformer(TT) [17] after the FTCN to detect the long-range temporal incoherence. The Temporal Transformer takes sequences of temporal features extracted by the FTCN as input and apply the class token to make the predictions. We also show the feature distri-
3.2.1 Fully Temporal Convolution Network

3D CNNs [10, 50] are widely used on video-related tasks. Traditional 3D CNN models compute both spatial-temporal correlations via convolution with $K_t \times K_h \times K_w$ kernel, where $K_t \times K_h \times K_w$ are designed to be larger than 1 for most layers. These convolution layers are applied repeatedly, propagating signals progressively through the spatial and temporal dimensions. However, we find such spatial-temporal coupled kernels weaken the model’s ability to capture purely temporal information. To encourage the spatio-temporal convolution network to learn the temporal incoherence, we redesign the convolution operator and propose a fully temporal convolution network (FTCN). The key idea is to restrict the capability of the network for handling spatial-related artifacts. So we set all the spatial sizes of the convolution kernel to 1. As we notice that some convolution layers may involve strides more than 1. In this situation, many locations may be ignored in the input features, thus we also design a rule to handle this. Suppose a 3D convolution is represented as 3DConv($K_t, K_h, K_w, S_t, S_h, S_w$), where $K_t, K_h, K_w$ are the kernel size for time, height, width dimension, $S_t, S_h, S_w$ are the stride for time, height, width dimension. We replace it with 3DConv($K_t, 1, 1, 1, 1, 1$) and add a max-pooling after the convolution operator if $S_t$ or $S_w > 1$.

Take the popular 3D R50 structure [10] as an example, the resulting 3D R50-FTCN model is shown in Table 1. The input video clip has 32 frames each with $224 \times 224$ pixels. Also, we change the last global average pooling to spatial-related average pooling to keep the feature along time dimension unchanged. All the 3D convolutions in Table 1 share a similar kernel shape with like $K_t \times 1 \times 1$, which can be regarded as 1D temporal convolutional filters for temporal dimension. Such networks can be regarded as fully temporal convolutional networks (FTCN), which mainly learn the discriminative features along the temporal dimension.

3.2.2 Temporal Transformer

The Temporal Transformer aims to learn the long-range discrepancies along the time dimension. With the FTCN, we obtain the temporal features $F \in R^C \times N \times H \times W$ ($C=2048$, $N=16$, $H=1$, $W=1$). The temporal feature $F$ can be represented as a sequence of features $F_t \in R^C$, $t \in \{1, 2, ..., N\}$, which is a 1D sequence of token embeddings in standard Transformer [51]. The $N$ is input sequence length, $C$ is the feature dimension of sequence. The overview of our Temporal Transformer is depicted in Fig-

<table>
<thead>
<tr>
<th>layer</th>
<th>output size</th>
</tr>
</thead>
<tbody>
<tr>
<td>conv1</td>
<td>$5 \times 1 \times 1$, 64, stride 1, 1, 1</td>
</tr>
<tr>
<td>pool1</td>
<td>$1 \times 5 \times 1$, max, stride 1, 1, 4</td>
</tr>
<tr>
<td>res2</td>
<td>$1 \times 1 \times 1$, 64, $3 \times 1 \times 1$, 64</td>
</tr>
<tr>
<td>pool2</td>
<td>$2 \times 1 \times 1$, max, stride 2, 1, 1</td>
</tr>
<tr>
<td>res3</td>
<td>$1 \times 1 \times 1$, 128, $3 \times 1 \times 1$, 128</td>
</tr>
<tr>
<td>res4</td>
<td>$1 \times 1 \times 1$, 512, $3 \times 1 \times 1$, 512</td>
</tr>
<tr>
<td>res5</td>
<td>$1 \times 1 \times 1$, 1, 1, 1, 1, 1, 1, 1</td>
</tr>
<tr>
<td>spatial-related pool</td>
<td>$2048 \times 16 \times 1 \times 1$</td>
</tr>
</tbody>
</table>

Table 1. Our 3D R50-FTCN model for video face forgery detection. Compared with the original 3D ResNet-50 model [10], we follow the rules described in Section 3.2.1 to obtain this structure. The dimension of 3D filter kernels are in $K_t \times K_h \times K_w$, The dimension of output maps are in $C \times N \times H \times W$. The input is $32 \times 224 \times 224$. Residual blocks are shown in brackets.

Figure 3. The Temporal Transformer for our video face forgery detection framework. We split the feature extracted from FTCN along the time dimension, and feed the resulting sequences of features to a standard Transformer encoder. We apply an extra learnable “classification token” to the sequence to learn the final discriminative feature and add an MLP head to distinguish real or fake.
Similar to the settings in ViT [17], we apply a trainable
liner projection \( W \) to map the feature dimension from \( C \) to
\( D \). To enable the classification in Temporal Transformer,
we add a learnable embedding to the sequence of embed-
ded features \( z_0 = F_{\text{class}} \), which serves as the representa-
tive features learned on the input sequences. Following the
settings in ViT[17], we also include learnable 1D position
embeddings to retain positional information. Suppose the
position embedding is \( E_{\text{pos}} \). So the input sequence \( z_0 \) for
the Temporal Transformer can be defined as:

\[
z_0 = [F_{\text{class}}, W_1, W_2, \ldots, W_N]^T + E_{\text{pos}},
\]

where \( F_{\ell} \) is the \( \ell \)-th time slice in feature \( F \), \( W \in \mathbb{R}^{D \times C} \), \( E_{\text{pos}} \in \mathbb{R}^{(N+1) \times D} \).

The Temporal Transformer mainly consists of \( L \) standard
Transformer Encoder blocks [51], each standard Transformer encoder block consists of a multi-head self-
attention(MSA) [51] block and an MLP block. We also
apply the commonly-used LayerNorm(LN) before each
block. Another important structure is the residual connec-
tions [24], which is applied to each block. The activation
function we used for the Temporal Transformer is GELU.
So the features get for the \( \ell \)-th layer can be defined as:

\[
z_\ell = \text{MSA}(\text{LN}(z_{\ell-1})) + z_{\ell-1}, \quad \ell = 1 \ldots L
\]

\[
z_\ell = \text{MLP}(\text{LN}(z_\ell)) + z_\ell, \quad \ell = 1 \ldots L
\]

So based on the class-token output \( \text{LN}(z_0) \) of the last encoder. We can apply an MLP head for the final fake proba-
bility:

\[
y = \text{MLP}(\text{LN}(z_0^T)).
\]

In our experiments, the binary cross-entropy loss is applied
on the final prediction \( y \).

4. Experiments

4.1. Experimental Settings

Training Datasets. We adopt the most commonly used
benchmark dataset FaceForensics++(FF++)[44] for training.
It contains 1000 original videos and 4000 fake videos.
The fake videos are manipulated by four meth-
ods: Face2Face(F2F) [49], FaceSwap(FS) [2], NeuralTex-
tureNT) [48], and Deepfake(DF) [1]. We trained on the
high-quality (HQ) subset of the FF++, which is the light
compression version.

Testing Datasets. To evaluate the generalization capability
of our framework, we test our model on the follow-
ing datasets: 1) FF++ that contains four types of manip-
ulation as mentioned above; 2) FaceShifter(FFS) and
3) DeeperForensics28(DFo) employ the real videos from
FF++ for high-fidelity face swapping; 4) DeepFake De-
tection Challenge Preview dataset[16](DFDC), where each
original video is filmed in the challenging environment; and
5) Celeb-DF-v2[35](CDF) is a new DeepFake dataset in-
cluding 518 videos from different sources.

Evaluation Metrics. Following the evaluation metrics in
previous works [32, 22], we report the area under the re-
ciever operating characteristic curve (AUC). Since most
previous works are image-based, following the setting in
LipForensics [22], we report video-level AUC for a fair
comparison. For the image-based method, we average
the model predictions for each frame across the entire video.
Therefore, all models utilize the same number of frames for
classification.

Implementation. We take the 3D R50 as the basic structure
of our proposed FTCN. The Temporal Transformer we use
is one layer of standard Transformer Encoder [51], whose
self-attention heads, hidden size, and MLP size are set to 12,
1024, 2048, respectively. For the training setting, we use
a batch size of 32 and SGD optimizer with momentum, and
the weight decay is set as 1e-4. We apply a warm-up strat-
egy for the training of our methods. Concretely, the learning
rate first increases from 0.01 to 0.1 in the first 10 epochs and
then cosinelty decayed to 0 for the last 90 epochs. For more
details, please refer to the supplementary material.

Baselines. We mainly compare our methods with various
state-of-the-art methods. These methods are mainly about
boosting the generalization capability as well as some pop-
ular baselines. 1) Xception [43] explores the performance
of face manipulation detection by the popular Xception [13]
model. 2) CNN-aug [54] find that current CNN-generated
images can be easily detected by a CNN model. 3) Patch-
Forensics [11] suggests that the patch-based classifier can
achieve impressive results for face forgery detection. 4) Face X-ray [32] aims to improve the generalization capa-
bility by detecting the blending boundary artifacts instead
of the artifacts from generative models. 5) CNN-GRU [45]
introduces GRU [12] into CNN model to model tempo-
ral coherence. 6) Multi-task [39] applies an autoencoder-
like architecture for deepfake detection. 7) FW A [34]
explorres to detect the resolution differences between altered
face and background for improving generalization capabil-
ity. 8) Two-branch [36] presents multi-task learning on
FaceForensics++ dataset. 9) LipForensics [22] is a recent
work that studies the irregular mouth motions for general
and robust face forgery detection.

4.2. Generalization to Unseen Manipulations

The differences between face forgery datasets mainly
lie in the variations of source videos and face manipula-
tion methods. To evaluate the cross-manipulations gener-
alization capability of different face forgery detectors and
prevent the possible bias introduced by different source
videos, we conduct experiments on FF++, as it provides
fake videos created by multiple face forgery methods for
the same source videos. Following the setting in [22], we
evaluate face forgery detectors with the leave-one-out strat-
Table 2. Generalization to unseen manipulations. We report the video-level AUC(%) on the FF++ dataset, which consists of four manipulation methods (DF, FS, F2F, NT). We train on three methods and test on the other one method. The results of other methods are from [22].

Table 3. Comparison with state-of-the-art methods. We report the number of parameters, pre-training, extra data usage, and the average video level AUC(%) on four unseen manipulation methods (DF, FS, F2F, NT), all the models are trained on the remaining three methods in FF++.

Table 4. Generalization to unseen datasets. We report the video-level AUC(%) on four unseen datasets: Celeb-DF-v2 (CDF), Deepfake Detection Challenge Preview (DFDC), FaceShifter (FSh), and DeeperForensics (DFo). We train on FF++ and test on these unseen datasets. The results of other methods are from [22].

4.3. Generalization to Unseen Datasets

In a real-world scenario, suspicious videos are likely to be created by unseen methods from unseen source videos, thus the cross-dataset generalization would be crucial. To evaluate the cross-dataset generalization capability, we trained the face forgery detectors on all the four types of fake data in FF++, and perform the evaluation on four unseen datasets, including Celeb-DF-v2 (CDF) [35], DFDC [16], FaceShifter [31] and DeeperForensics [28]. As shown in Table 4, our model achieves the best performance on every dataset, with especially strong results on FaceShifter and DeeperForensics. On CDF and DFDC dataset, all the methods obtain relatively low scores, one possible explanation is the scenario gaps between different datasets.

4.4. Robustness to Unseen Perturbations

For real-world scenarios, it is of great importance for face forgery detectors to be robust to unseen perturbations. We conduct experiments to verify the robustness of our methods. Following [28], we consider four popular perturbations: 1) Block-wise distortion; 2) Change of color saturation; 3) Gaussian Blur; 4) Resize: downsample the image by a factor then upsample it to the original resolution. Each perturbation is divided into five intensity levels as [28]. The results are reported in Figure 4. On average, our methods achieve better robustness to unseen perturbations. It is worth mentioning that our method does not apply any pre-training knowledge during training, which is obviously helpful for robustness.

4.5. Ablation study

We perform comprehensive studies on the FF++ [43] dataset to validate our design of the overall framework. We start by verifying the design of 3D R50-FTCN. Why remove the spatial convolution? To validate why we remove all the spatial convolution in the proposed 3D R50-FTCN, we construct multiple variants of 3D ResNet-50 (3D R50), including the following models:

1. 3D R50: The original model of 3D R50, with spatio-temporal convolutions.
2. 3D R50-Spatial: Based on 3D R50, replace all the 3DConv($K_t$, $K_h$, $K_w$, 1, 1, 1) with 3DConv(1, $K_h$, $K_w$, 1, 1, 1).
3. 3D R50-FTCN-FK3: We replace the first 3D convolution layer of 3D R50-FTCN with 3DConv(5, 3, 3, 1, 1, 1), which involves spatial-related convolution.
4. 3D R50-FTCN-FK5: We replace the first convolution of 3D R50-FTCN with 3DConv(5, 5, 5, 1, 1, 1),
which involves more spatial-related convolution than 3D R50-FTCN-FK3.

5. 3D R50-FTCN-Shuffle: We use the same network structure as 3D R50-FTCN, but take a “spatial-shuffled” clip as input. The spatial-shuffle operation shuffles the pixel order in the clips along the spatial dimensions while keeping the pixel-wise temporal continuity (all images in the clip share the same shuffle pattern), we show the shuffled clips in the Supplementary Material.

6. 3D R50-FTCN: The proposed FTCN, its structure is described in Section 3.2.1, and shown in Table 1.

We apply these models to train a binary classifier and evaluate the performance under the setting of training only on F2F, and testing on all four methods of FF++ (F2F, FS, DF, NT). All the variants are trained with exactly the same setting except the model architecture as described above. For each model, we report the performance of the best setting among four methods on the validation set.

The results are shown in Table 5. We can identify some important conclusions from the results: 1) Compare the results of 3D R50 and 3D R50-Spatial, involving temporal information into face forgery detection can benefit the performance of generalization. 2) For 3D R50, 3D R50-FTCN-FK5, 3D R50-FTCN-FK3, 3D R50-FTCN, the spatial-related convolution involves less and less, but the generalization capability gets better and better, so less spatial-related convolution leads to better results. 3) Even if we damage the spatial information by pixel shuffle, the 3D R50-FTCN can still achieve a reasonable result, this suggests that the 3D R50-FTCN mainly learns to distinguish by temporal-related information.

Is limited model capability that benefits generalization capability? For 3D R50, 3D R50-FTCN-FK5, 3D R50-FTCN-FK3, 3D R50-FTCN, the spatial-related convolution involves less and less, and the model capability becomes weaker and weaker. Thus it is natural to ask whether the generalization capability benefits from the limited model capability. We conduct experiments to verify this. We design several variants of our proposed 3D R50-FTCN and trained on the F2F in FF++:

<table>
<thead>
<tr>
<th>Model</th>
<th>Train on F2F</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D R50</td>
<td>80.0 89.5</td>
<td>90.3</td>
</tr>
<tr>
<td>3D R50-Spatial</td>
<td>77.9 53.6</td>
<td>77.7</td>
</tr>
<tr>
<td>3D R50-FTCN-FK5</td>
<td>97.4 94.1</td>
<td>96.8</td>
</tr>
<tr>
<td>3D R50-FTCN-FK3</td>
<td>94.2 93.0</td>
<td>95.1</td>
</tr>
<tr>
<td>3D R50-FTCN-Shuffle</td>
<td>97.3 92.5</td>
<td>95.8</td>
</tr>
<tr>
<td>3D R50-FTCN</td>
<td>98.0 95.9</td>
<td>97.5</td>
</tr>
</tbody>
</table>

Table 5. Ablation study of variants design of FTCN Video-level AUC(%) is reported on FF++ datasets.

<table>
<thead>
<tr>
<th>Model</th>
<th>Train on F2F</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D R50</td>
<td>80.0 89.5</td>
<td>90.3</td>
</tr>
<tr>
<td>3D R50-SP</td>
<td>86.2 85.3</td>
<td>89.6</td>
</tr>
<tr>
<td>3D R50-FHCN</td>
<td>76.1 49.5</td>
<td>77.1</td>
</tr>
<tr>
<td>3D R50-FWCN</td>
<td>84.8 73.2</td>
<td>83.4</td>
</tr>
<tr>
<td>3D R50-FTCN</td>
<td>98.0 95.9</td>
<td>97.5</td>
</tr>
</tbody>
</table>

Table 6. Ablation study of 3D R50 variants with different model capability. Video-level AUC(%) is reported.

1. 3D R50-SP: 3D R50 with the same amount of parameters as 3D R50-FTCN, this model is created by reducing the number of channels in 3D R50.

2. 3D R50-FHCN: Replace 3DConv(K\_h, K\_w, S\_h, S\_w) with 3DConv(1, K\_h, 1, 1, 1) and add a MaxPool(1, S\_h, S\_w) if S\_h > 1 or S\_w > 1.

3. 3D R50-FWCN: Replace 3DConv(K\_t, K\_h, K\_w, S\_h, S\_w) with 3DConv(1, 1, K\_w, 1, 1) and add a MaxPool(1, S\_h, S\_w) if S\_h > 1 or S\_w > 1.

The results are reported in Table 6. 3D R50-SP and 3D R50-FTCN share a similar number of parameters, but 3D R50-FTCN presents better results. This validates that the performance gain is mainly from the fully temporal design. Moreover, 3D R50-FHCN and 3D R50-FWCN share exactly the same amount of parameters and computation cost with 3D R50-FTCN but present lower performance. This further indicates that temporal artifacts are more general as well as the effectiveness of our proposed 3D R50-FTCN.

Influence of video clip size. To find the optimal clip size, we trained 3D R50-FTCN with clip sizes of 8, 16, 32, 64. All the models are trained on F2F and test on all four methods in FF++. We change only the clip size and keep the
Table 7. Ablation study of using different clip sizes for the training of FTCN. Video-level AUC(%) is reported.

<table>
<thead>
<tr>
<th>clip size</th>
<th>Train on F2F</th>
<th></th>
<th></th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DF</td>
<td>FS</td>
<td>F2F</td>
<td>NT</td>
</tr>
<tr>
<td>8</td>
<td>79</td>
<td>86.2</td>
<td>99.8</td>
<td>85.6</td>
</tr>
<tr>
<td>16</td>
<td>95.4</td>
<td>95.3</td>
<td>100</td>
<td>94.8</td>
</tr>
<tr>
<td>32</td>
<td>98.0</td>
<td>95.9</td>
<td>100</td>
<td>96.0</td>
</tr>
<tr>
<td>64</td>
<td><strong>98.2</strong></td>
<td><strong>96.6</strong></td>
<td><strong>100</strong></td>
<td><strong>96.7</strong></td>
</tr>
</tbody>
</table>

Table 8. Ablation study of using different layers of Transformer encoders in our framework. Video-level AUC(%) is reported.

<table>
<thead>
<tr>
<th>Model</th>
<th>Train on F2F</th>
<th></th>
<th></th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DF</td>
<td>FS</td>
<td>F2F</td>
<td>NT</td>
</tr>
<tr>
<td>3D R50 FTCN</td>
<td>98.0</td>
<td>95.9</td>
<td>100</td>
<td>96.0</td>
</tr>
<tr>
<td>3D R50 FTCN+TT L×1</td>
<td><strong>98.1</strong></td>
<td><strong>99.6</strong></td>
<td>100</td>
<td><strong>98.0</strong></td>
</tr>
<tr>
<td>3D R50 FTCN+TT L×2</td>
<td>97.8</td>
<td>98.6</td>
<td>100</td>
<td>97.7</td>
</tr>
<tr>
<td>3D R50 FTCN+TT L×3</td>
<td>97.0</td>
<td>98.4</td>
<td>100</td>
<td>97.4</td>
</tr>
</tbody>
</table>

unrelated hyper-parameters unchanged.

Table 7 shows that as clip size increases, the performance boost. There is a tiny performance gain when clip size changes from 32 to 64, the possible reasons could be 1) There are not enough temporal convolution layers to capture such a long clip. 2) Video face alignment suffers from large clip size and large motion, as it would be hard to find a crop region that covers all the faces in the clip. As clip size growth brings performance boost along with more computation cost, a good trade-off would be clip size 32.

**Effectiveness of Transformer.** To validate the effectiveness of our Temporal Transformer, we perform an ablation study on the framework. We train three variants of our framework: 1) we train a model which only has the 3D R50 FTCN; 2) based on our framework, we change the number of encoder layers in Temporal Transformer to 2 (3D R50 FTCN+TT L×2); 3) based on our framework, we change the number of encoder layers in Temporal Transformer to 3 (3D R50 FTCN+TT L×3). The results are presented in Table 8. We can find the following observations: 1) the Temporal Transformer can improve the performance of generalization capability. 2) more layers of Standard encoder for Temporal Transformer can not further improve the performance, which shows that one layer of standard Transformer encoder is sufficient in our framework.

**4.6. Localization of Temporal Incoherence**

Without training with any explicit annotations, our method could be readily extended to localize temporally incoherent regions. In test time, for an input clip, we slide a window across the spatial domain. For regions outside the sliding window, we remove their content by replacing their RGB values with zeros. The modified clip is then fed into our forgery classifier to estimate the fake probability of the window area. Figure 5 shows that our method could robustly distinguish real and fake clips, and accurately localize the regions even with subtle temporal defects. For better visualization of temporal incoherence, please check the video results in the supplementary material.

**5. Conclusion**

This paper investigates the effectiveness of temporal cues for more robust and general video face forgery detection. We propose to first encode short-term flickering with a Fully Temporal Convolution Network, then explore more subtle long-term incoherence with a Temporal Transformer. Extensive experiments evident the significant effects of the temporal information for video face forgery detection, and show the superior capabilities both on robustness and generalization of our proposed solution against previous methods. We hope our study will attract the community’s attention to the temporal incoherence in deepfake detection.

**6. Acknowledgements**

Yinglin Zheng and Ming Zeng were partially supported by NSFC(No.62072382), Fundamental Research Funds for Central Universities, China(No.20720190003).
References


[59] Tianchen Zhao, Xiang Xi, Mingze Xu, Hui Ding, Yuanjun Xiong, and Wei Xia. Learning to recognize patch-
