Multi-Layer Fusion Neural Network for Deepfake Detection

Zheng Zhao, Xidian University, Xi’an, China
Penghui Wang, National Laboratory of Radar Signal Processing, Xidian University, Xi’an, China
Wei Lu, Sun Yat-Sen University, Guangzhou, China

ABSTRACT

Recently, the spread of videos forged by deepfake tools has been widely concerning, and effective ways for detecting them are urgently needed. It is known that such artificial intelligence-aided forgery makes at least three levels of artifacts, which can be named as microcosmic or statistical features, mesoscopic features, and macroscopic or semantic features. However, existing detection methods have not been designed to exploit them all. This work proposes a new approach to more effective detection of deepfake videos. A multi-layer fusion neural network (MFNN) has been designed to capture the artifacts in different levels. Features maps output from specially designed shallow, middle, and deep layers, which are used as statistical, mesoscopic, and semantic features, respectively, are fused together before classification. FaceForensic++ dataset was used to train and test the method. The experimental results show that MFNN outperforms other relevant methods. Particularly, it demonstrates more advantage in detecting low-quality deepfake videos.

KEYWORDS

Deepfake, Multimedia Security, Neural Network, Video Forensics, Video Forgery

INTRODUCTION

Human face is the most significant identity of human beings. Nowadays, digital videos with human faces are widely used in many serious occasions such as court evidence and news report. Apparently, the validity of them depends on the fact that it is infeasible to forge the faces.

For a long period of time, forging human faces in video has been considered as a time consuming and expensive task. However, the situation has been changed recently. With the development of neural network-based methods like deep learning, more and more new techniques which can support facial tampering and face swap begin to emerge. Based on convolutional autoencoders (Bengio, Lamblin, Popovici, & Larochelle, 2007) and generative adversarial network (GAN) (Goodfellow, Pouget-Abadie, Mirza, Xu, Warde-Farley, Ozair, Courville, & Bengio, 2014), the most famous face manipulation tools under the name of Deepfake (Korshunova, Shi, Dambre, & Theis, 2017; Faceswap Project, 2018; Faceswap-GAN Project, 2018) can replace a human being face in video with a face belongs to anybody else in an easy but effective way. Although the face swapping in video has also been implemented with the methods based on computer graphics, such as Face2Face (Thies, Zollhofer, Stamminger, Theobalt, & Nießner, 2016) and FaceSwap (Kowalski, 2016), Deepfake tools are widely considered as more promising. Moreover, Deepfake technologies have been used by the face forgery software tools designed for common people, such as FaceApp and Deepfakesapp. These tools,
running on either personal computers or on smart phones, have friendly interfaces to guide the people without professional training to forge video faces with a convincing effect. As a result, more and more Deepfake videos have emerged on social networks, and the side effect of them has made such technology a worldwide concern. Apparently, effective ways of detecting them are urgently needed.

To detect Deepfake videos, some methods have been proposed by recognizing the forgery features. Under the traditional framework of pattern recognition, the semantic features such as inconsistent head poses (Yang, Li, & Lyu, 2019), color anomalies (Li, Li, Tan, & Huang, 2019; McCloskey & Albright, 2018), color difference between left and right eyes, shading artifacts, and reflection detail missing in eyes (Matern, Riess, & Stamminger, 2019), have been extracted and classified. In recent two years, deep learning-based methods have been more used. Li and Lyu (2018) proposed a deep learning network to detect the artifacts resulting from face warping transform. Li, Chang, and Lyu (2018) adopted convolutional neural network (CNN) and long short-term memory (LSTM) (Hochreiter, & Schmidhuber, 1997) to detect the anomalies of eye blinking. In fact, LSTM is a typical recurrent neural networks (RNN) used for learning the feature sequence extracted by CNN over each frame. Similarly, Güera and Delp (2018) utilized the CNN named Inception v3 (Szegedy, Vanhoucke, Ioffe, Shlens, & Wojna, 2016) and LSTM (Hochreiter, & Schmidhuber, 1997) to detect the anomalies within and between frames respectively. And Afchar, Nozick, Yamagishi and Echizen (2018) designed the network named MesoNet to detect the so-called mesoscopic features which are considered as the middle-level features between the semantic and statistical ones. More famous neural networks which were previously used for image classification and image forgery detection tasks, such as Xception (Chollet, 2017) and MISLnet (Bayar & Stamm, 2018), have also been applied to detection of Deepfake videos (Rössler, Cozzolino, Verdoliva, Riess, Thies, & Nießner, 2019).

As reviewed above, up to now it has been recognized that Deepfake tools can make at least 3 levels of forgery features, namely microcosmic, mesoscopic, and macroscopic features respectively (Afchar, Nozick, Yamagishi, & Echizen, 2018). Among them, microcosmic features correspond to the unseen artifacts such as the anomalies over small areas. Macroscopic features, also called semantic features, indicate visual defects. And mesoscopic features are in the level between them. Nevertheless, existing ways of detecting Deepfake videos, as reviewed above, have not sufficiently exploited all the 3 levels of features so that their performances have room of improvement. Intuitively, an ensemble classifier (Kodovsky, Fridrich, & Holub, 2012), which combines the decisions of some existing detection methods, can be designed to make use of more features. However, it is often hard to acquire a proper principle for decision fusion which ensures the substantial improvement of accuracy.

Based on the above observation, this paper is to propose a Deepfake detection method by fusing features maps output from different layers in the detection network. We have further observed that Deepfake features, on one hand appear as local anomalies which tend to be captured by shallow layers, and on the other hand appear as semantic anomalies which can be more easily captured by deep layers. And middle layers are more suitable for extracting mesoscopic features. Therefore, we designed a multilayer fusion network capable of fusing features from different layers to capture forgery features of different levels. In constructing the neural network, we created shortcut connections for feature maps from several different layers, downgrade the size of these feature maps, and send them directly to the last layer for concatenation of the feature vectors to be classified. We trained and tested our model on the FaceForensics++ (Rössler, Cozzolino, Verdoliva, Riess, Thies, & Nießner, 2019) dataset. The experimental results show that compared with the existing state-of-the-art methods, our network achieves better accuracy, especially over the Deepfake videos coded in low quality.

The rest of the paper is organized as follows. In next section, some related work will be introduced. In the followed section, the design of our network will be given and explained. Then we will describe the experiments and give their results. Finally, we will draw the conclusion.
RELATED WORK AND OUR THINKING

In this section, we will introduce the Deepfake technologies and give our observation on why it can make artifacts in different levels. Then, we will introduce the hierarchical property of CNNs and explain why it can help us to extract different-level forgery features by using one network.

Deepfake Approach and Its Artifacts in Different Levels

In forging video faces, a face to be manipulated and output is called target, and a face used to manipulate a target is called source. In fact, video face forgery is not something new. People have realized it before the emerge of Deepfake technologies. However, traditional forgery of video faces relies on expensive and professional computer-aided graphics systems. The underlying principle is to perform 3-dimensional modeling on the faces, track the motion and perform animation processing so that a source face can control the facial expression of a target face.

Figure 1. Steps of Deep-Faceswap Forgery Using Autoencoders

Nowadays, Deepfake represents a kind of face swapping technique that replaces one video face with another one by using deep learning technologies. In many occasions, a source face bears resemblance to a target so that the former can control, i.e., enact, the facial expression of the latter. The development of neural network technologies has brought substantial change to multimedia generation and processing. Typically, convolutional autoencoders (Bengio, Lamblin, Popovici, & Larochelle, 2007) are widely used to code and generate facial posture, expression, skin color, scene lighting and other characteristics. Deep-Faceswap (Faceswap Project, 2018), one of the most famous and typical Deepfake tools, is implemented with autoencoder techniques. The steps of its forgery are briefly introduced as follows (Figure 1).

1. **Training.** Train the encoder and decoder for faces of character A and B respectively.
2. **Weight Sharing.** Sharing the weights between the encoders of character A and B, where the encoders are both neural networks. As a result, the encoders are considered the same for both of character A and B.
3. **Face Generation.** To replace a face of character A, i.e. target, with that of character B, i.e. source, just encode the former face but decode it with the decoder of character B.
4. **Face Swapping.** Locate the target face area and cover it with the generated one.
More recently, GAN (Goodfellow, Pouget-Abadie, Mirza, Xu, Warde-Farley, Ozair, Courville, & Bengio, 2014) has been applied to Deepfake technologies. In the most powerful Deepfake tool named Faceswap-GAN (Faceswap-GAN Project, 2018), GAN is used for enhance the forgery effect by adversarial learning so that the faked faces can be more realistic.

Although GAN has made a great improvement on face forgery effect, the whole face swapping function is implemented by the abnormal coding model illustrated in Figure 1. That is why current Deepfake tools can inevitably make forgery artifacts. In fact, in encoding the face of character A but decoding the output with the decoder of character B, the final output face under the name of character A will has the characteristics of both character A and B. It can make some visual semantic and local anomalies such as we have showed in Figure 2 and reviewed in the section of introduction. Of course, there must be some features between the semantic level and local level, which we have called them mesoscopic features in the middle level.

Figure 2. Example of Deepfake Artifacts: (a) Faked face; (b) Original face; (c) Abnormal edges of cheek contours; (d) Shading detail Missing; (e) Purple halo around lips; (f) Distorted face.
Hierarchical Property of CNN and Our Thinking on Exploiting It

Based on the observation that Deepfake tools make 3-level artifacts, we think the work for effective detection of Deepfake videos is to give a good way of extracting and classifying them. And we have found that such a task is related to exploiting the different feature maps in different layers of CNN. Zeiler and Fergus (2014) used deconvolutional network to visualize CNN, and discovered many interesting properties. One of them is the hierarchical nature of CNN which is illustrated in Figure 3.

In Figure 3, we show that the convolutional kernel in shallow layers of CNN learns the local pattern, and as the network go deeper, the convolution kernel can capture more semantic and integrated pattern. So, we can think that the local representation of edges and texture can be learned in shallow layer, and the representation of more global semantic features is to be learned in the deeper layers. Between them, the patterns over middle-sized location such as eyes and lips can be learned by middle layers.

The hierarchical property of CNNs has made us consider combining feature maps from different layers for more effective detection of Deepfake videos. As we have mentioned in the introduction section, it is widely recognized that current Deepfake forgery features can be categorized into macroscopic, mesoscopic, and microcosmic ones. Largely, they correspond to the above 3 levels of forgery features respectively. In fact, an example of the 3-level features has been shown in Figure 2. As we see in it, Deepfake forgery traces can appear as the abnormal local edges, abnormal local shading, abnormal lip halo, inconsistent noise model within middle-sized locations, and distortion over the larger face area. They tend to be captured by different-depth convolutional layers. Consequently, to better make use of all the 3-level features, our idea is to output feature maps from some different-depth layers respectively and fuse them together before classification. In the sequel, our task is to design a proper framework and a detailed network for doing it.
Based on the above observation, we have designed a so-called multi-layer fusion neural network (MFNN) to better capture different levels of Deepfake forgery features for classification. In this section, we will first give the framework of MFNN, and then give the explanation on the key part of it.

**Framework of MFNN**

The framework of our MFNN supports different convolutional layers to output feature maps respectively. Not only are the deep-layer feature maps used for final classification, but also the shallow-layer and middle-layer ones. The specially designed shortcut connections start from different layers are responsible for outputting the features from different layers respectively. And these different-layer features are finally concatenated and accordingly prepared for classification.

In order to take full advantage of different levels of features, we need to specially design the network structure for each layer and shortcut connection so that these features can have positive contribution to the final decision-making. Next, we will introduce the most important details about the design of network structure and parameter setting.
Structure of Multi-layer Feature Combination

In this subsection, we will give and explain the vital part of our network structure, that is, the multi-layer shortcut connection structure. Our idea is to combine multi-layer feature maps to make decision. There are two different ways to do it. One is to use different classifiers to classify separately over different-layer features and then make all decisions fused into one decision. The other is to combine all different-layer features first and then classify them once. Since the method of separated classification and joint decision-making cannot train the network end-to-end, it is difficult to make full use of the features extracted at different layers and exploit the internal connection between them. In many occasions, the separated decisions can be counteractive with each other so that it is more difficult to improve the final accuracy in this way. Therefore, we choose the way of combining all features before classification. Figure 4 shows a sketch of our solution for multi-layer feature combination. Shortcut connections for outputting feature maps from several layers are created and then the feature maps are concatenated for classification. Feature map transforms are designed to further process the feature maps and adjust their dimensionality. Each feature map transform is to be trained in deep learning, and the design details of it will be given in the sequel.

Feature Map Transforms on Shortcut Connection

In this subsection we will give the detail of features processing on the shortcut connections. Intuitively, the learning capability of neural network cannot be fully exploited if we directly combine and send these different-layer feature maps with different characteristics and dimensionalities to the final decision-making part, that is, the fully connected layer. These feature maps need to be further processed at different stages before they arrive at the decision-making part. Therefore, we further process the output feature maps at 2 stages, and the processing has been called feature map transform in Figure 4. In the first stage, the feature maps output by a convolutional layer are directly filtered with their
dimensionality controlled. In the second stage, the combined feature maps are further filtered. It is worth mentioning that all the filter kernels participate in the deep learning.

An example of the processing is shown in Figure 5. The input 16×16×256 feature map is divided into 8×8×256 regions, and each region contains 2×2×256 samples. Therefore, we set a 2×2×256 convolution with 2×2 strides to conduct the transform to acquire an 8×8 feature maps. Compared with using pooling operation, using convolution to reducing the feature dimensionality must learn a larger number of parameters, but it can retain more characteristics of the input features. So, we use the convolution to conduct the feature map transform. In order to control the dimensionality, we divide the input feature map into regions with particularly sized and numbered channels, and set a convolution which has the according filter size, channel numbers and strides as illustrated in Figure 5.

**Overall Structure and Convolutional Layers**

We design the MFNN under the framework of the above shortcut connection and feature maps combination. The overall network structure is shown in detail in Figure 6. The network has 14 convolution layers for feature extraction and can be divided into 6 modules, including input module (1), feature extraction module (2-5) and decision module (6). We set batch normalization (Loffe & Szegedy, 2015) after each convolutional layer, which is not shown in the figure due to the space limitation. The design rule of this network is like ResNet (He, Zhang, Ren, & Sun, 2016) and VGGnet.
(Simonyan & Zisserman, 2014). Some design is applied to overcome the degradation of deep neural network. In Figure 6, the second module has been designed to have linear residual connection. This linear residual connection create shortcut which bring features from shallower layer to deeper layer to avoid network degradation. As for other feature extraction modules, the shortcut connections can serve the similar purpose as linear residual connections since these shortcut connections also create shortcut for parameters. Inside each module, every convolution layer has the same output feature map size and the same number of filters. Between adjacent modules, the feature map size is halved and the number of filters in the next module is doubled so that the time complexity of each convolutional layer is close. We use maxpooling layer to conduct downsampling because it is better at preserving texture information, which often contains the local anomalies of Deepfake forgery.

To more precisely locate and detect the forgery, our method, as many others for detecting Deepfake videos, extracts one frame from every 20 frames and let MFNN make one decision on each selected frame. Our test focused on the face and its surrounding areas, and face action is complex, fast and irregular. Therefore, it is not feasible to find the key frames of video as training data. We choose to extract frames at equal intervals. The video clip in our dataset is 25 frames per second, given that training requires a large amount of data, but too close a gap can lead to data sameness, we extracting one frame from every 20 frames, and eventually get a series of facial image data with different expressions. Before inputting a selected frame to MFNN, it must be preprocessed as shown in Figure 7. Because the faked region can only be the face area, our method selects and cuts the face area in each selected frame by the CNN face detector in Dlib library with pre-trained weights. In order to maintain some background information and forging information around the face edge, our method expands the selecting box by 1.4 times. Finally, it cuts the boxed area and resizes it to 256×256 as the final image to be input to MFNN. Because the preprocessing may have side impact on the input frames, in the input module three convolution layers are used to enhance the input signals with different coding qualities.

In designing MFNN, we consider the feature maps on each shortcut connection and those on the main path as equal importance. Because the dimensionality of the feature maps on the main path is 8×8×256, feature map transforms are applied to the 3 shortcut connections respectively to make the dimensionality on each connection be also 8×8×256. As a result, the final dimensionality of the concatenated feature maps is 8×8×1024, which is the input of module 6. As shown in Figure 6, the final dimensionality of feature maps input to the fully connected layer is 1024.

EXPERIMENTS

In this section, we will first give the details on our experimental database and testing settings, and then we will analyze and compare the results.

Databases

Our experimental dataset is from FaceForensics++ (Rössler, Cozzolino, Verdoliva, Riess, Thies, & Nießner, 2019), a widely used face-swapping database. Its 1000 source videos, most of which are
news broadcast clips with fix-positioned faces and constant background, come from YouTube. The video file format in this dataset is mp4, with a resolution of 640×480 in all level of compression rate. These video clips are around 15 seconds with 25 frames per second. And its Deepfake videos were made by first forging the 1000 source videos by Deepfake-GAN (Faceswap-GAN Project, 2018) and then coding them in H.264 with 3 qualities denoted C0, C23, and C40 respectively. As a result, there are totally 3000 Deepfake videos, and each 1000 are under one compression quality. C0 denotes non-lossy compression, C23 denote the compression ratio of 23:1, and C40 the ratio of 40:1. The bit rate of video is 15237kbps for c0, 397kbps for c23, 53kbps for c40. Figure 8 shows an example of the original video frame and the corresponding forgery video frame of the video data in FaceForensics++. In the training and testing process, we always took the same number of source and accordingly faked video samples, and used accuracy to evaluate the effectiveness of MFNN. The videos under different compression qualities were trained and tested separately because in practice the quality factor of a video can be recognized before detection. Videos, either source or faked, with the same compression rates were collected and divided into the training set, validation set and testing set according to the respective proportions of 70%, 15%, and 15%. The specific processing process of the image has been introduced. After processing, a total of 44392 image data are obtained for each compression rate.

**Settings for Training and Testing**

We implemented MFNN by using Keras deep learning library within TensorFlow platform. We selected cross entropy as the loss function, and adopted the auto adaptive algorithm Adam (Kingma & Ba, 2017) in the optimization process. The batch size of the input data is 32. The policies of learning rate decay and early termination were both used. The initial learning rate is 1e-4. If the value loss of the verification set does not decrease after every six rounds of iteration, the learning rate is reduced to the original 0.1. If the loss of the verification set does not decrease after 15 consecutive rounds of iteration, the training is finished. The rest of the model settings are default values. We used Intel Xeon Processor E5-2609 v4 for CPU and NVIDIA Quadro P5000 for GPU. It took two and a half hours to train according to the training settings in the paper. For comparison, Inception v3 took two hours and twenty minutes, and Xception took three hours.

The number of positive and negative samples in any part of our dataset is the same. Therefore, accuracy is used to judge the performance of MFNN in the testing. Here, accuracy is the quotient of the number of correctly predicted samples and the total number of samples containing both positive and negative ones, in our test dataset.

**Results and Comparison with Other Methods**

The experimental results of MFNN will be compared with the following existing methods.
Inception v3 (Szegedy, Vanhoucke, Ioffe, Shlens, & Wojna, 2016). The network adopts four filters of different sizes to process input data simultaneously. It takes factorization strategy on convolution kernels to receive more disentangled parameters and therefore can be trained fast.

Xception (Chollet, 2017). The design of the network is inspired by Inception. However, some of the convolution modules had been replaced by depth-wise separable convolutions to improve the detection performance.

Meso-4 (Afchar, Nozick, Yamagishi, & Echizen, 2018). It is a neural network specially designed for detecting face swapping in videos. Less convolutional layers are used to focus on extracting the mesoscopic features of forged images.

MesoInception-4 (Afchar, Nozick, Yamagishi, & Echizen, 2018). It is an alternative network designed for extracting the mesoscopic features of forged images. Most part of the network is the same as Meso-4 but the first two convolutional layers are replaced by a module like that in Inception.

MISLnet (Bayar & Stamm, 2018). The neural network, which adopts learnable high-pass filters, is designed to suppress the interference of the image content and focus on learning manipulation traces.

The testing results of our method and the above comparative methods are shown in Table 1 and Table 2. As shown in Table 1, our method achieved the best accuracy performance under all the three different compression ratios. Over C0 videos, MFNN has the highest accuracy of 100% while

<table>
<thead>
<tr>
<th>Compression Ratio Detection Methods</th>
<th>C0</th>
<th>C23%</th>
<th>C40%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception v3</td>
<td>98.88%</td>
<td>92.72%</td>
<td>81.03%</td>
</tr>
<tr>
<td>Xception</td>
<td>99.93%</td>
<td>98.23%</td>
<td>87.11%</td>
</tr>
<tr>
<td>Meso-4</td>
<td>98.01%</td>
<td>93.50%</td>
<td>83.14%</td>
</tr>
<tr>
<td>MesoInception-4</td>
<td>98.61%</td>
<td>95.36%</td>
<td>84.41%</td>
</tr>
<tr>
<td>MISLnet</td>
<td>99.91%</td>
<td>98.00%</td>
<td>82.05%</td>
</tr>
<tr>
<td>MFNN (Our method)</td>
<td><strong>100%</strong></td>
<td><strong>98.97%</strong></td>
<td><strong>91.82%</strong></td>
</tr>
</tbody>
</table>

Table 2. The AUC (Area under Curve) result of our method and the comparative methods

<table>
<thead>
<tr>
<th>Compression Ratio Detection Methods</th>
<th>C0</th>
<th>C23</th>
<th>C40</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception v3</td>
<td>0.9999</td>
<td>0.9911</td>
<td>0.9215</td>
</tr>
<tr>
<td>Xception</td>
<td>0.9999</td>
<td>0.9963</td>
<td>0.9434</td>
</tr>
<tr>
<td>Meso-4</td>
<td>0.9996</td>
<td>0.9929</td>
<td>0.9252</td>
</tr>
<tr>
<td>MesoInception-4</td>
<td>0.9999</td>
<td>0.9921</td>
<td>0.9277</td>
</tr>
<tr>
<td>MISLnet</td>
<td>0.9992</td>
<td>0.9909</td>
<td>0.8923</td>
</tr>
<tr>
<td>MFNN (Our method)</td>
<td><strong>0.9999</strong></td>
<td><strong>0.9961</strong></td>
<td><strong>0.9703</strong></td>
</tr>
</tbody>
</table>
Xception achieves the second-best performance of 99.93%. Over C23 videos, MFNN also has the highest accuracy of 98.97% while Xception still achieves the second-best performance of 98.23%. The improvement of MFNN is the most substantial over the C40 videos which have the lowest quality. In this case, MFNN has an accuracy of 91.82%, and Xception, still the second best, only has 87.11%. The AUC (Area under Curve) result of our method and the comparative methods, as shown in Table 2, shows the similar advantage of MFNN. Because it is more difficult to detect low-quality Deepfake videos, the above results have apparent significance in detecting the videos in the real world. The ROC curves of the top three methods, MFNN, Xception, and MesoInception-4, acquired on C40 dataset, are shown in Figure 9. On C0 and C23 videos, the three curves almost overlap due to little room of improvement and we do not give them.

**CONCLUSION**

This work proposes a new approach to more accurate detection of face forgery in Deepfake videos. It has given a multi-layer fusion neural network (MFNN) to capture various artifacts in three different levels left by Deepfake forgery. The features maps acquired by shallow, middle, and deep layers are respectively output and fused together before classification. FaceForensics++ dataset was used for training and testing the network. And the experimental results show that the new detection method outperforms the existing methods in detecting Deepfake videos. In particular, the accuracy has been substantially improved by MFNN in detecting low-quality Deepfake videos, which is often more difficult for existing methods.

**ACKNOWLEDGMENT**

This work was supported by the National Natural Science Foundation of China (61701379).
REFERENCES


Zheng Zhao is currently an undergraduate student from Xidian University. He is studying electronic and information engineering in the school of electronic engineering.

Penghui Wang received the B.S. degree in communication engineering from National University of Defense Technology (NUDT), Changsha, China, in 2005 and Ph.D. degree in signal processing from Xidian University, Xi’an, China, in 2012. He is now an associate professor at the National Laboratory of Radar Signal Processing, Xidian University. His research interests include radar signal processing and automatic target recognition.

Wei Lu received the B.S. degree in Automation from Northeast University, China in 2002, the M.S. degree and the Ph.D. degree in Computer Science from Shanghai Jiao Tong University, China in 2005 and 2007 respectively. He was a research assistant at Hong Kong Polytechnic University from 2006 to 2007. He is currently a Professor with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China. His research interests include multimedia forensics and security, data hiding and watermarking, privacy protection. He is an Associate Editor for the Signal Processing and the Journal of Visual Communication and Image Representation.