Deepfake Detection using Supervised Temporal Feature Extraction model and LSTM

*Chunghwan Lee *Jaihoon Kim **Kijung Yoon
Department of Electronic Engineering, Hanyang University
jungwhan612@gmail.com, jh27kim@gmail.com, kjyoon@hanyang.ac.kr

*These authors contributed equally to this work.
**Kijung Yoon is the corresponding author of the paper.

지도 학습한 시계열적 특징 추출 모델과 LSTM을 활용한 딥페이크 판별 방법

*이정환 *김재훈 **윤기중
한양대학교 융합전자공학부
*위 두 저자는 동일하게 논문에 참여함.
**윤기중 교수는 본 논문의 교신저자임.

SUMMARY

As deep learning technologies becoming developed, realistic fake videos synthesized by deep learning models called “Deepfake” videos became even more difficult to distinguish from original videos. As fake news or Deepfake blackmailing are causing confusion and serious problems, this paper suggests a novel model detecting Deepfake videos. We chose Residual Convolutional Neural Network (Resnet50) as an extraction model and Long Short-Term Memory (LSTM) which is a form of Recurrent Neural Network (RNN) as a classification model. We adopted cosine similarity with hinge loss to train our extraction model in embedding the features of Deepfake and original video. The result in this paper demonstrates that temporal features in the videos are essential for detecting Deepfake videos.

1. Introduction

Deepfake is a technology that can manipulate videos seamlessly. The term "Deepfake" has multiple definitions, however, in this paper we define Deepfake as any videos containing swapped faces created by deep neural networks. This is contrasted with so-called "cheapfakes" - if a fake video was produced by machine learning, it is Deepfake, whereas if it was created with widely-available software with no learning component, it is considered cheapfake [1]. Producing Deepfake videos is not quite difficult because there are lots of available software freely accessible on GitHub, e.g., FakeApp [2], faceswap-GAN [3], faceswap [4], and DeepFaceLab [5].

These Deepfake technologies have the ability to distort reality with fake news and fake pornographies. The development of machine learning is escalating Deepfake technologies to be more sophisticated, and well crafted Deepfake videos might be weaponized by fabricating a specific individual and activities that did not occur in real life.

With development of this potentially harmful Deepfake technique, Deepfake detection methods also evolved. In this paper, we propose a Deepfake detection technique that figures out whether the video is real or fake by analyzing temporal features in the video.

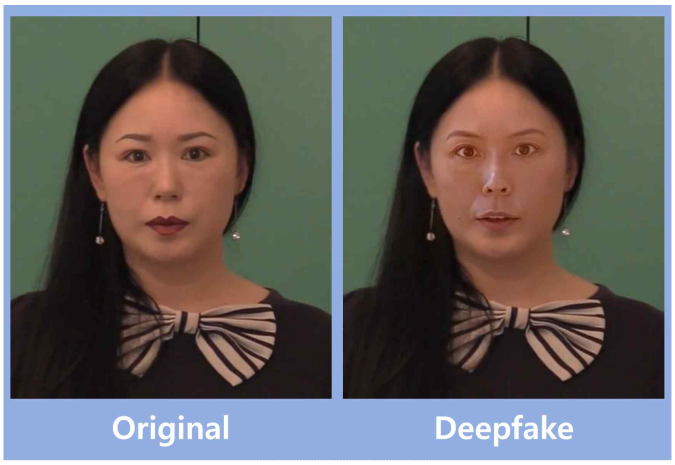

Figure 1. An example of Deepfake face swapping technology

Our work is different from the previous works in the following three ways. First, we divided our model into feature embedding part and detection part. Features from each frame were extracted by our base model CNN [6] and Resnet50 [7],

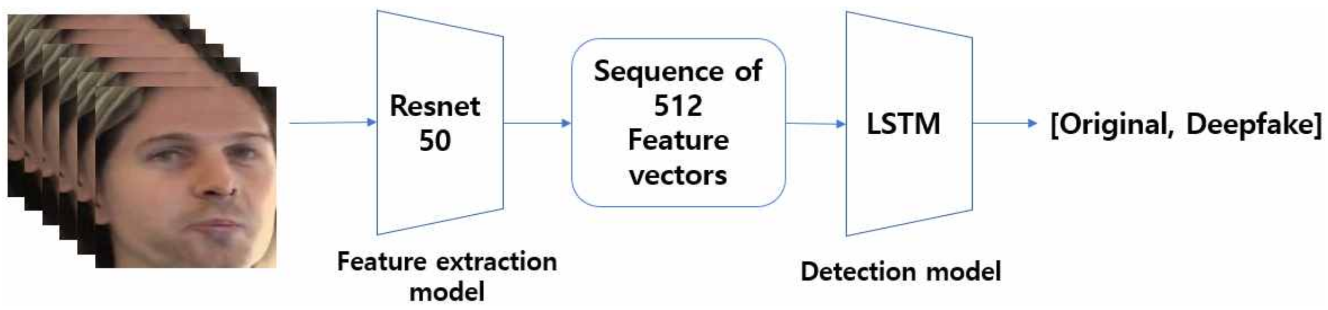

Figure 2. The overview of our model. The feature extraction model is trained to map features of the input sequences. Detection model examines the sets of the input features and classifies if the video is original or Deepfake.

then, we classified the original videos from fake videos by examining the continuity of the frames using Long Short-Term memory(LSTM) [8] which is a form of recurrent neural network(RNN)[9]. Separating these two models enables extraction model to preserve distinct features from Deepfake and real videos. Second, loss function for our extraction model set the features of fake video and pristine video apart. Third, we clearly proved that time-domain features from Deepfake videos helped to discriminate fake by comparing LSTM and MLP using as our detection model.

2. Method

Our model is divided into two parts, the former one is feature embedding part and the latter one is detecting part (Figure 2).

Model architecture

A. Feature extraction model

Considering the large size of videos, an effective encoding model extracting features of the videos to minimal information was crucial. Our first trial for feature extraction model was simply designed 4-layer CNN. Next, we used ResNet50 as our extraction model to extract more sophisticated features. Inputs are sequences of frames from videos, and outputs are 512 dimensional vectors. We used hinge loss to enhance the performance of our model. Hinge loss will be discussed in greater detail below.

B. Detection Model

We used 512 dimensional feature vectors from extraction model for classification. We needed to select a model that can learn temporal relations between the frames in a meaningful manner. Our model for detection is a 4-layer LSTM with 512 hidden units and 0.3 chance of dropout. Input for our model is 512 dimensional embedded feature and it will output a prediction value activated by sigmoid function.

Loss

We adopted Hinge Loss with cosine similarity as loss function for the extraction model. Cosine similarity returns 1 for two vectors pointing at the same direction, and 0 for two vectors that are orthogonal. We intended adjacent frames of real images to have cosine similarity as close as possible to 1 and 0 for the Deepfake images. Therefore, identity matrix was an ideal option as our target matrix. In case of the Deepfake images, the model will try to minimize $\mathcal{L}1$ (1) between the cosine similarity matrix of the images with the identity matrix, and it will maximize for real images which are not synthesized.

$$\mathcal{L}_1 = \frac{1}{N}\sum_{i=1}^{N}\left|y_i - \hat{y_i}\right| \tag{1}$$

Comparing the similarity among features from continuous frames with cosine similarity is beneficial for two major reasons. First, it will extract common face features that are shared among the frames of real videos. Second, it will emphasize abrupt changes between adjacent frames of Deepfake videos, making features of Deepfake images even more distinct.

In feature extraction model, hinge loss (2) applies different losses to Deepfake videos and real videos. For Deepfake videos, we used cosine similarity matrix of the feature vectors to target identity matrix in $\mathcal{L}1$ norm. For real videos, we picked the max values between 0 and 1 minus $\mathcal{L}1$ Loss. In detection model, we used Binary Cross Entropy(BCE) Loss (3) to detect whether the sequence of the videos are fake or real.

$$\mathcal{L}_{HINGE} = \begin{cases} \mathcal{L}_1, & \text{if} y_n = 1, \\ \max\{0, \Delta - \mathcal{L}_1\}, & \text{if} y_n = -1 \ (\Delta = 1) \end{cases} \tag{2}$$

$$\mathcal{L}_{BCE} = \frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y_i}) + (1-y_i)\log(1-\hat{y_i})] \tag{3}$$

3. Experiments

Dataset

We used 4670 videos of Deepfake dataset from Facebook AI

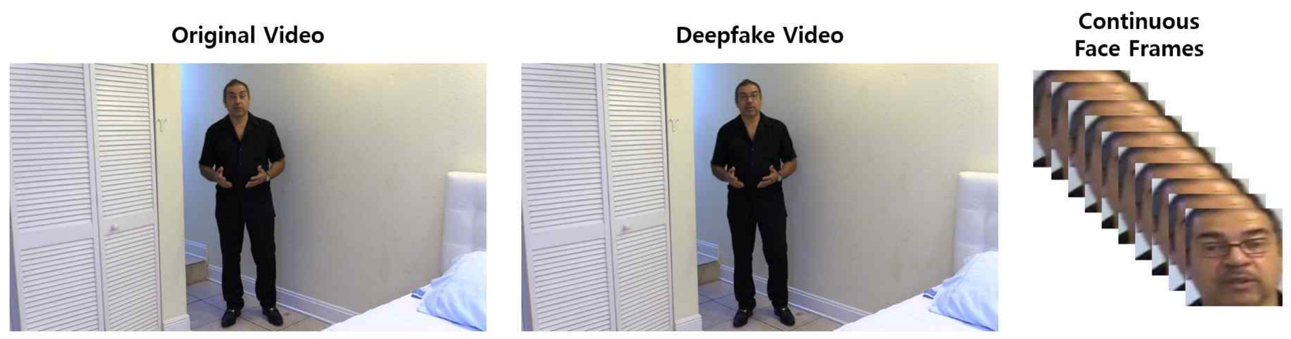

Figure 3. Deepfake Detection Challenge Dataset(DFDC) and cropped the face part from each frame from dataset.

(The Deepfake Detection Challenge Dataset, DFDC)[10] which is roughly 3.6% of the total dataset due to the limitation of computing power. We extracted cropped faces (256 x 256) from the videos using MTCNN [11].

We confronted three major difficulties during the face extraction: miss detection, false detection, multi-face recognition. In case of miss detection, when no face was detected, we just skipped the frame. This resulted in inconsistent time gaps. When there were more than one face detected, we compared Mean Square Error (MSE)(4) between the current faces and previously detected faces to make pairs with the lowest MSE. Moreover, we only used the frames that showed MSE below 0.5 to eliminate possibility of using non facial images.

$$\mathcal{L}_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \qquad (4)$$

Our key intention when designing our system was to map embedding vectors from real images and Deepfake images as discretely as possible. We noticed that adjacent images do not vary abruptly since a time gap between the frames is very small. We made an assumption that sudden changes between adjacent images can only be caused by Deepfake technique. Indeed, this is only true when the time intervals between the frames are constant. In our dataset, the time gap between the frames is not constant because we omitted some misdetected frames during data preprocessing. Therefore, we used two-pointer algorithm to mark the intervals of consecutive frames and selected frame sets that have constant interval into our dataset.

As a result, we constructed dataset with sequences of 100 frames with consistent time gaps for our training. The total number of sequences used was 5908. We trained the model with 5500 sets, 308 sets as a validation set, and 100 sets as a test set.

Training

A. Feature extraction model

We experimented with two different extraction models. One is a simple 4-layer CNN and the other is Resnet50 (Figure 4). The number of epochs we trained is 100. Optimizer is set to Adam [12] with learning rate of 0.001. We used Hinge Loss using $\mathcal{L}1$ as our loss for feature extraction model.

B. Detection model

We tested a performance of our LSTM detection model by changing the training datasets. The number of epoch we trained is 20 to avoid over-fitting as can be seen in (Figure 5).

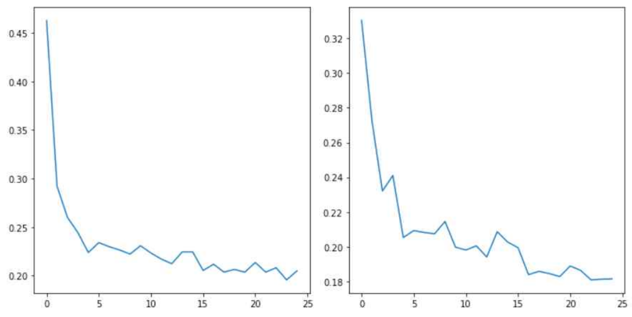

Figure 4. Training loss (Right) and Validation loss (Left) graph for training Feature extraction model (Resnet50)

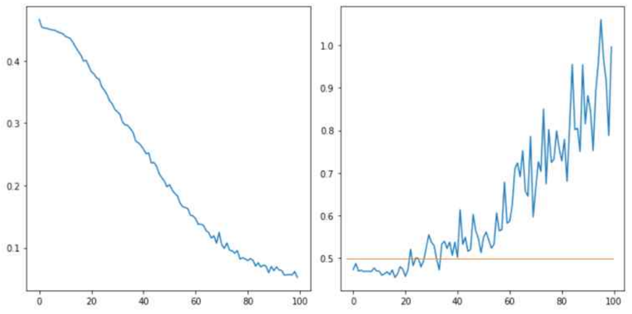

Figure 5. Traing loss graph (Left) and Validation loss graph (Right) for training Detecting model (LSTM)

4. Results

In Deepfake Detection Challenge Dataset, training and validation sets were released publicly and we used a part of the dataset for training. Since the public and private test sets were not released, we constructed our own test set. We randomly chose and excluded 100 sets of frames 6 times from the 5908 training dataset and used them as our test set. The 6 different training sets and test sets are equally trained and tested in a same environment. We evaluated accuracy and log loss from each test set. Mean and standard deviation calculated from our 6 different test sets can been seen in (Table1).

Table 1. Accuracy and LogLoss of our own 6 test set

	Mean	Standard Deviation
Accuracy	0.81	0.028867513
Logloss	0.51470457	0.056360475

One purpose of our experiment is to figure out whether extraction model trained by our method extracts optimal features in discerning the Face-swapped videos. (Table 2). In the first experiment, we simply verified our hypothesis using the pretrained InceptionResNet which was not trained by our hinge loss. Then, we used 4-layer CNN as feature extraction model. This result demonstrates that using hinge loss improved classification accuracy. At last, we used a more complex model, Resnet50, which scored the lowest LogLoss (5). Log loss estimates performance of models in DFDC. Smaller log loss indicates a more accurate model.

$$\mathcal{L}_{Logloss} = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y_i}) + (1-y_i)\log(1-\hat{y_i})] \quad (5)$$

Table 2. LogLoss comparison in feature extraction model.

	Trained with Hinge Loss	LogLoss
InceptionResnetV2 (Pretrained)[13]	X	0.4479743
4-layer CNN	O	0.4426801
Resnet50	O	0.4410648

The next purpose of our experiment was to figure out how much temporal features influence in discerning Deepfake. We supposed that temporal features would play a key role when dealing with Deepfake classification task. We verified our assumption by comparing the performance of classification model between LSTM and Multi-Layer Perceptron (MLP) [14]. As can be seen in (Table 3), temporal features from extraction model played a key role in discriminating Deepfake.

Table 3. Comparison between LSTM and MLP model

	LogLoss for our Test set
LSTM	0.4410648
4-layer MLP	0.5137186

5. Conclusion

The top performing models from DFDC rated accuracy of 82.56% for public test dataset. Comparing to state-of-the-art models, the best accuracy of our model tested by our own testset is 82.56%. We can conclude that our method for reinforcing temporal feature helped classifying Deepfake videos.

Since we only used 3.6% of the whole DFDC dataset, our model underperformed for unseen data. We expect to overcome this limit if we provide more computing powers to handle the entire dataset.

Reference

[1] Paris, B., & Donovan, J. (2019). Deepfakes and cheap fakes. United States of America: Data & Society.

[2] Guilloux, L. (n.d). *FakeApp*. Malavida. Retrieved August 30, 2021, from https://www.malavida.com/en/soft/fakeapp/

[3] Shaoanlu. (2018, August 27). *faceswap-GAN*. Github. https://github.com/shaoanlu/faceswap-GAN

[4] Torzdf. (2019, Octorber 22). *faceswap*. Github, https://github.com/Deepfakes/faceswap

[5] Iperov (2019, January 24). *DeepFaceLab*. Github," https://github.com/iperov/DeepFaceLab

[6] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems, 25*, 1097-1105.

[7] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

[8] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation, 9*(8), 1735-1780.

[9] Zaremba, W., Sutskever, I., & Vinyals, O. (2014). Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329.*

[10] Dolhansky, B., Bitton, J., Pflaum, B., Lu, J., Howes, R., Wang, M., & Ferrer, C. C. (2020). The deepfake detection challenge (dfdc) dataset. *arXiv preprint arXiv:2006.07397.*

[11] Xiang, J., & Zhu, G. (2017, July). Joint face detection and facial expression recognition with MTCNN. In *2017 4th international conference on information science and control engineering (ICISCE)* (pp. 424-427). IEEE.

[12] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

[13] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2818-2826).

[14] McClelland, J. L., Rumelhart, D. E., & PDP Research Group. (1986). *Parallel distributed processing* (Vol. 2, pp. 20-21). Cambridge, MA: MIT press.