Deepfake Detection with Data Privacy Protection

Mingkan Wu
Dalian University of Technology
Dalian, China
390645506@mail.dlut.edu.cn

Fei Wang
Dalian University of Technology
Dalian, China
1727656144@mail.dlut.edu.cn

Xiaohan Wu
Dalian University of Technology
Dalian, China
ssxh@mail.dlut.edu.cn

Fei Yu
Dalian University of Technology
Dalian, China
yf1999@mail.dlut.edu.cn

Bo Wang*
Dalian University of Technology
Dalian, China
bowang@dlut.edu.cn

Zengren Song
National Computer Network Emergency
Response Technical Team/
Coordination Center of China
Beijing, China
songzr0518@163.com

2022 IEEE 24th International Workshop on Multimedia Signal Processing (MMSP) | 978-1-6654-7189-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/MMSP55362.2022.9949458

Abstract—As image and video forgery can be easily used for malicious purposes, the detection of such forgeries has social and technical significance. In this work, we are particularly interested in the detection of Deepfake. For privacy and sensitive data preserving reasons, we engage a flank attack using Federated Learning, a distributed framework-based model which keeps data locally during training while uploading model parameters for aggregate instead. We propose a shallow network for tampering face detection. Also, we made some progress in promoting cross-dataset detection performance which is crucial in Deepfake detection. Our experiments show a well-balanced trade-off result between detection performance and privacy preservation.

Index Terms—Deepfake Detection, Federated Learning

I. Introduction

Deepfake, including DeepFake [1], Face2Face [2], FaceSwap [3], NeuralTexture [4], is a class of techniques which aims to manipulate the facial area on image or video by editing, replacing or moving.

Currently, many people use Deepfake for illegal purposes. Thus, the detection of Deepfake forgery becomes an intriguing topic.

Deepfake detection based on supervised learning, the number, diversity and reliability of data used for training are key to ensure the performance of the trained model. Previous analysis always assumes that the data for training are easy to obtain and sufficient by default. But the changeable methods of Deepfake generation can easily void the detection because different generation methods can produce hardly similar features. Besides, in reality, as the sensitive and private data, the collection of face images is difficult and under strict supervision of law.

In our work, beyond existing works in this field, we also put privacy-preserving into our consideration. To better explain our motivation, let's consider the following scenario. Everyday, numerous individuals (users, clients) produce request to verify the genuineness of locally saved image/video content. A conventional (and intuitive) method might be, clients with requests upload their requests and sample (data) to a server to perform training and classification, and the server feed back to each client with the corresponding result. However, such method could potentially leak sensitive information and data, which causes concerns on the data privacy and security. For solving such issue, we propose an approach from the Federated Learning [11] and the local parameter masking mechanisms. Roughly speaking, each client performs training locally (with local data) and submits updated parameters (i.e., local parameters in our following discussions) to the server, the server aggregates local parameters and updates the global parameters, then feeds the updated global parameters to each client for the next round of training. More details of our model will be given later.

This paper is organized as follows. In Section II, we review the development of face forgery detection and the basic framework of federated learning. In Section III, we depict our implementation in detail. In Section IV, we introduce experimental setup, evaluation, and comparison with other approaches. We conclude and state future works in Section V.

II. Related Work

1) Face Forgery Detection: At the early stage of forgery detection research, as an intuitive approach, image processing methods (e.g., copy-move, splicing, deleting, etc.) are widely applied. Such detection methods mainly focus on the extraction of inherent features from images, and train machine learning classifiers to detect manipulations. For example, people propose features like Speeded-Up Robust Features(SURF) [12], Photo Response Non-Uniformity(PRNU) [13], and use machine learning classifiers to obtain predictors (classifiers).

However, those methods could possibly destroy the intrinsic properties of the image, and easily fail in detecting

*: corresponding author

forged content generated by deep learning (i.e. Gan, VAE etc.) models trained with large-scale dataset. Accordingly, the forgery detection develops following two deep-network-based approaches.

One approach tries to find inconsistent and discontinuous artifacts. [5], [14] For example, Li et al. [5] find that when the forged images are generated, due to the shortage of training data in the closed eye state, the generated image will also be short in the closed eye state. Consequently, the artifact of non-blinking (for long time) can be used as a feature for detection. In [6], Yang et al. propose that after 2D marking the face landmarks, the direction of the center region and the edge region of the face in the real image is consistent after 3D transformation, but the direction of the forged image deviates greatly and the artifacts caused by the discontinuous head posture are detectable.

The other approach focuses on the autonomous detection/classification of hidden features. Gradually optimize the network structure to get a more optimal performance. [8], [9], [16]

In [8], Afchar et al. propose to do meso-scopic(between micro-scopic and macro-scopic) analysis with small number of layers network and achieve the best accuracy as far as they know. In [19], Dong et al. uses the method of face recognition in which each forged image and its corresponding real image (reference image) with erased inner face is put in during training, and make the network find the fuzzy difference between real and forged outer face.

III. Our Method

A. Data Preprocessing

Data pre-processing (e.g., normalization) is essential for training a neural network. In the previous work of federated learning, training data is usually normalized by (local) statistics (e.g., mean and variance) independently at each client. However, when data is heterogeneous, such as hue, brightness or various forgery method et.al, unusual normalized data deviation may occur, and it may cause unexpected biased to the global gradient by local stochastic gradients [20], [21]. Thus, we propose a global data pre-processing method under the condition of federated learning privacy protocol, and we name it as federated Z-score normalization (FZSN).

To be more explicit, assuming we have a server and m clients in our federated setting and $n = \sum_{i=1}^{m} n_i$ data in total, the number of data held by each client are $\{n_1, ..., n_m\}$. First, every client uses their local data to calculate their local statistics$\{mean, variance\}$, $\{\{\overline{X}_1, S_1^2\}, ..., \{\overline{X}_m, S_m^2\}\}$. Then each client uploads its own statistics, which is hard to infer from, to the central server for aggregating. The statistics (without any data) which represent the whole data of our federated learning framework is calculating by the following formulas:

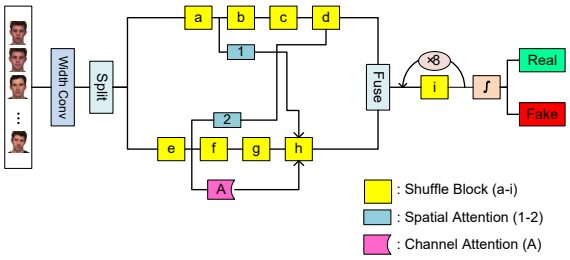

Fig. 1. Pipeline of our Deepfake Detection model

$$\text{mean} = \frac{\sum_k n_k \overline{X}_k}{\sum_k n_k}, \tag{1}$$

$$S^2 = \frac{\sum_k n_k S_k^2 + \sum_k n_k d_k}{\sum_k n_k}, \tag{2}$$

where $d_i = (mean - \overline{X}_i)^2$.

B. Network Structure

The pipeline of our network is depicted in Figure 1. We refer to the idea of channel shuffle and group convolution from ShuffleNet [22], and take it as the basic repeatable cell in our network. Our model is built with shallow layers for reasons as follows. For the feature perspective, the artifacts express saliently in mesoscopic [8] while the high semantic feature always fails in cross-dataset detection. Regarding the framework perspective, clients are always assumed to have low computing capacity and such that shallower model is more suitable.

Regarding the proposed network, in the input phase, we implement the large-width convolution to make all features fully expressed and to reduce the information loss in pooling. Then the input is mapped into a latent space Π. In the space Π, we split the channel into two branch F_a and F_b and note that each branch consists of some repeatable shuffle-blocks. The shuffle-block is a residual structure, and composed of single kernel group convolution, channel shuffle and DWConv [23].

We use an attention mechanism to supervise the two branches and perform intersection technique in branch crossing. At the beginning of the second shuffle-block, we introduce the channel attention and connect it with the last block. For our width network, we only need limited extra computation resource to calculate the importance of different channels. Another is cross-branch spacial attention, since the information of the two branches is incomplete, the spatial information of each other is used as supervision. After applying fusion, we can get the most representative spatial flow and channel flow information. We conclude the block as:

$$[f_{ca}(X_{branchA}) + f_{sa}(X_{branchB})] \cdot g(h(x_{branchA})) \tag{3}$$

$$[f_{ca}(X_{branchB}) + f_{sa}(X_{branchA})] \cdot g(h(x_{branchB})) \tag{4}$$

The idea of attention mechanism is derived from [24]. For the channel attention, we compress the input in space dimension and consider both average pooling and max pooling. For the spatial attention, we also compress the channel to one based on max pooling and average pooling and merge the two attention map. The tail of our model is followed by eight shuffle-blocks the same as described above, at last it is a full connection layer to get the alternative result real or fake.

Algorithm 1

Input: model G, initialized model parameters θ_0, learning rate l_r, batch size bs, the number of clients n, global epoch t, local epoch e, global loss L, local batch b_{nm}
Output: model parameters θ^*

while $|L_{ep_t} - L_{ep_{t-1}}| > threshold$ do
  for each client $i \in [1, n]$ do
    central sever send global model parameters θ_i to client
    $\tilde{\theta} = \theta_i$
    for each $j \in [1, e]$ do
      compute $g = \sum_m \nabla L(\tilde{\theta}, b_{nm})$
      if $j == e$ then
        compute DP-SGD $\tilde{g} = g + N(0, \sigma^2)$ [25]
      else
        $\tilde{g} = g$
      end if
      compute $\tilde{\theta} = \tilde{\theta} - l_r * \tilde{g}$
    end for
    add random mask to model parameters $\theta_i = Pertubation(\tilde{\theta})$
    send local parameters, local loss to central server
  end for
  compute $\theta = \frac{1}{n} \sum_{i=1}^{n} \theta_i$
  compute $L = \sum_{i=1}^{n} L_i$
end while

C. Random Mask and Differential Privacy

Addressing our concern on privacy-preserving, we implement the differential privacy and random masking mechanisms locally for each client. The pseudo-code of our method is shown in Algorithm 1.

Differential privacy [26], [27] has been widely accepted as a standard method and applied widely for privacy preserving.

For every global epoch, we apply a differential-privacy-style mechanism to the gradients, which is called DP-SGD [25], in the last local epoch of every client. That is, we add small noise to the updated parameter (gradients) $W^k_{t,iter=i}$ and obtain a randomized version $\hat{W}^k_{t,iter=i}$ for next iteration of training. For example, $\hat{W}^k_{t,iter=i} = \mathcal{M}(W^k_{t,iter=i})$ and if $\mathcal{M}$ is an additive mechanism $\mathcal{M}(W) = W + N$ and N can be taken as Gaussian or Laplace noise. We set the maximum gradient l_2 norm being 1, so the scale of gradient keeps unchanged, and δ is equal to 1e-5 which is close to the reciprocal of the dataset size in magnitude. The noise scale is set as $\sigma = 2$ which is thought to be small noise.

For data safety and communication efficiency, we also apply random masking to the model, in the last epoch of local iteration. Before uploading the model to the central server, we set a part of parameters to be null with a certain probability. When sending it to the server, the model parameters are uploaded in a semi-completion form, which reduces the amount of parameters and increases the difficulty of getting data from the gradient. With differential privacy styled randomness adding mechanism, the connection between model parameters and training data can be blurred such that privacy is preserved. And the evaluation of privacy will be introduced in the subsequent section.

IV. Experiments

A. Experimental Settings

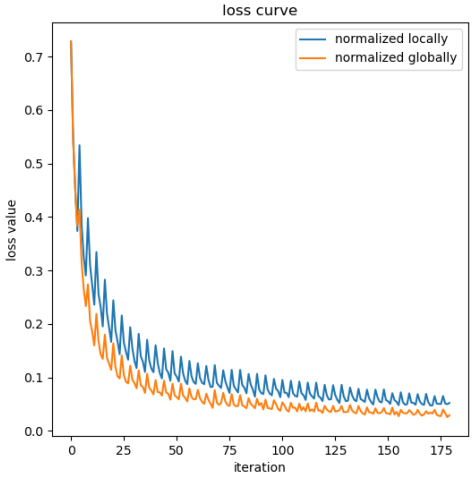

Fig. 2. The loss curve of non-i.i.d data in federated learning settings, drown from the loss value of client-a. Peak occurs after global aggregation

1) Datasets: Deepfake detection can be considered as a classification problem with two categories - real and fake. As different forgery methods may leave different forgery artifacts in the generated image or video contents, there are impacts on the generalization ability or cross-dataset ability of current Deepfake detection mechanisms. If the training set and testing set are generated with the same forgery method, the detection accuracy is high naturally for most state-of-the-arts, but they may probably fail in cross-database detection, which is one of the most intriguing problems of Deepfake detection. Therefore, it is wise to evaluate different methods in term of both intra-dataset and inter-dataset performances.

TABLE I
The performance of our method and other classical Deepfake detection method

method	resolution	DeepFake		Face2Face		FaceSwap		NeuralTexture	
		ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC
Steg.Feature	c23	78.15	-	75.32	-	80.35	-	78.46	
	c40	67.20	-	58.26	-	60.34	-	59.25	-
cozzolino et.al	c23	81.07	-	79.26	-	82.25	-	80.38	-
	c40	70.17	-	60.90	-	62.22	-	63.64	-
MesoNet	c23	93.20	-	78.05	-	87.38	-	71.44	-
	c40	85.90	-	72.33	-	72.73	-	63.85	-
Fed-MesoNet	c23	90.27	-	84.89	-	77.42	-	71.13	-
	c40	83.65	-	71.80	-	71.42	-	60.87	-
our method	c23	93.97	98.17	89.17	94.90	91.46	96.15	83.97	94.35
	c40	92.28	97.74	84.13	93.09	88.12	94.44	79.96	94.15

TABLE II
generalization result based on federated learning under different data distributions in clients

training set \ testing set	DeepFake	Face2Face	FaceSwap	NeuralTexture	RealImage
total mixed	87.44	79.60	79.58	72.38	92.32
one for every client	87.42	77.03	72.16	63.19	89.27
two for every client	79.41	75.64	73.86	66.60	94.75

TABLE III
Acc under different datasets which share the same forgery method, every client holds one dataset

datasets	accuracy
FF++DF	95.22
Celeb-DFv2	93.37
UADFA	97.14
DFD	95.65

In this work, we import five datasets (FaceForensics++ [28], Celeb-DF [29], DFD [30], DFDC [31], UADFA [6]) to evaluate our method (as shown in Table 1). The datasets are generated with two categories of forgery method (face identity replacement, face attribute edit), and four sub-categories (DeepFake , Face2Face, FaceSwap, NeuralTexture).

B. Experimental Result

The evaluation results of our model are given in Table 1. For comparison, we also trained a federated learning based variation of the MesoNet model. Under the same-database setting, the performance of federated learning models is a bit lower than (conventional) centralized learning models. As experimental results show, we can shrink this performance degradation under 3% in the settings when training data and evaluating data has the same distribution. We use FaceForensic++ dataset with the image of $C23$ and $C40$ compression rate to evaluate.

We also evaluate on other four datasets (results are shown in Table 3) with the Fed-MesoNet and our method. Our method obviously outperforms Fed-MesoNet.

Table 2 and Table 3 are the generalization ability experimental results in two different aspects.

Table 2 shows the detection accuracy when each client holds data of different forgery methods in the FF++ dataset. Total mixed means we mixed the data of four forgery methods to get a new dataset; one for every client means every client holds the data of one forgery method; two for every client means every client holds the data of two forgery methods, for example, client-a holds DF and FS, client-b holds DF and NT, and so on. While in inference phase, we evaluate and get the results according to the forgery method.

Figure 2 shows the loss curve of this experimental setting, and the loss curve is plotted according to the loss of each local epoch of client-a. We can learn from the curve that if we normalize the data using our federated Z-score normalization, we can get our method convergence faster and the loss value is smaller. Curve fluctuates less, represents that our method has stronger convergence and more stable performance.

Table 3 shows the result of that: every client holds the data which is generated by the same forgery method - DeepFake, but comes from different datasets. They are coarsely consistent in the forgery method, but differ in the implementation details.

From Table 2 and Table 3, we can learn that different forgery methods bring in more difference in artifacts detected by our method. Therefore, in Table 2, the detection accuracy of all methods has decreased by a margin, while the result in Table 3 has hardly decreased.

V. Conclusion

In this paper, federated learning is introduced into deep forgery detection to ensure data security in deep learning training. Privacy mechanism prevents our data from being accessed by malicious participants. Adopting federated learning makes multi-party joint training possible, which has positive practical significance.

VI. Acknowledgement

This work is supported by the National Natural Science Foundation of China (No. U1936117, No. 62106037, No. 62076052), the Science and Technology Innovation Foundation of Dalian (No. 2021JJ12GX018), the Open Project Program of the National Laboratory of Pattern Recognition (NLPR)(No.202100032), and the Fundamental Research Funds for the Central Universities (DUT21GF303, DUT20TD110, DUT20RC(3)088).

References

[1] (2018) Deepfake. [Online]. Available: https://github.com/deepfakes/faceswap

[2] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2face: Real-time face capture and reenactment of rgb videos," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2387–2395.

[3] (2018) Faceswap. [Online]. Available: ttps://github.com/MarekKowals-ki/FaceSwap/

[4] J. Thies, M. Zollhöfer, and M. Nießner, "Deferred neural rendering: Image synthesis using neural textures," ACM Transactions on Graphics (TOG), vol. 38, no. 4, pp. 1–12, 2019.

[5] Y. Li, M. Chang, and S. Lyu, "In ictu oculi: Exposing ai created fake videos by detecting eye blinking," in 2018 IEEE International Workshop on Information Forensics and Security (WIFS), 2018, pp. 1–7.

[6] X. Yang, Y. Li, and S. Lyu, "Exposing deep fakes using inconsistent head poses," ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 8261–8265, 2019.

[7] S. Agarwal, H. Farid, O. Fried, and M. Agrawala, "Detecting deep-fake videos from phoneme-viseme mismatches," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 660–661.

[8] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "Mesonet: a compact facial video forgery detection network," in 2018 IEEE International Workshop on Information Forensics and Security (WIFS). IEEE, 2018, pp. 1–7.

[9] H. H. Nguyen, F. Fang, J. Yamagishi, and I. Echizen, "Multi-task learning for detecting and segmenting manipulated facial images and videos," in 2019 IEEE 10th International Conference on Biometrics Theory, Applications and Systems (BTAS). IEEE, 2019, pp. 1–8.

[10] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. K. Jain, "On the detection of digital face manipulation," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 5781–5790.

[11] J. Konecnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," CoRR, 2016.

[12] Y. Zhang, L. Zheng, and V. L. Thing, "Automated face swapping and its detection," in 2017 IEEE 2nd International Conference on Signal and Image Processing (ICSIP). IEEE, 2017, pp. 15–19.

[13] M. Koopman, A. M. Rodriguez, and Z. Geradts, "Detection of deepfake video manipulation," in The 20th Irish machine vision and image processing conference (IMVIP), 2018, pp. 133–136.

[14] Y. Li and S. Lyu, "Exposing deepfake videos by detecting face warping artifacts," arXiv preprint arXiv:1811.00656, 2018.

[15] F. Matern, C. Riess, and M. Stamminger, "Exploiting visual artifacts to expose deepfakes and face manipulations," in 2019 IEEE Winter Applications of Computer Vision Workshops (WACVW). IEEE, 2019, pp. 83–92.

[16] H. H. Nguyen, J. Yamagishi, and I. Echizen, "Capsule-forensics: Using capsule networks to detect forged images and videos," in ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019, pp. 2307–2311.

[17] H. Khalid and S. S. Woo, "Oc-fakedect: Classifying deepfakes using one-class variational autoencoder," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 656–657.

[18] D. M. Montserrat, H. Hao, S. K. Yarlagadda, S. Baireddy, R. Shao, J. Horváth, E. Bartusiak, J. Yang, D. Guera, F. Zhu et al., "Deepfakes detection with automatic face weighting," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 668–669.

[19] X. Dong, J. Bao, D. Chen, W. Zhang, N. Yu, D. Chen, F. Wen, and B. Guo, "Identity-driven deepfake detection," arXiv preprint arXiv:2012.03930, 2020.

[20] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in in COMPSTAT, 2010.

[21] S. Ghadimi and G. Lan, "Stochastic first-and zeroth-order methods for nonconvex stochastic programming," SIAM Journal on Optimization, vol. 23, no. 4, pp. 2341–2368, 2013.

[22] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018.

[23] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1251–1258.

[24] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in Proceedings of the European conference on computer vision (ECCV), 2018, pp. 3–19.

[25] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, 2016, pp. 308–318.

[26] C. Dwork, "A firm foundation for private data analysis," Communications of the ACM, vol. 54, no. 1, pp. 86–95, 2011.

[27] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," Found. Trends Theor. Comput. Sci., vol. 9, no. 3–4, p. 211–407, Aug. 2014. [Online]. Available: https://doi.org/10.1561/0400000042

[28] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "Faceforensics++: Learning to detect manipulated facial images," in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 1–11.

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

[30] A. G. Nick Dufour, "Contributing data to deepfake detection research," Google AI blog, 2019.

[31] B. P. J. L. R. H. M. W. C. C. F. Brian Dolhansky, Joanna Bitton, "The deepfake detection challenge dataset," 2020.