Training Strategies and Data Augmentations in CNN-based DeepFake Video Detection

Luca Bondi, Edoardo Daniele Cannas, Paolo Bestagini, Stefano Tubaro
Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, Italy

***Abstract*—The fast and continuous growth in number and quality of deepfake videos calls for the development of reliable detection systems capable of automatically warning users on social media and on the Internet about the potential untruthfulness of such contents. While algorithms, software, and smartphone apps are getting better every day in generating manipulated videos and swapping faces, the accuracy of automated systems for face forgery detection in videos is still quite limited and generally biased toward the dataset used to design and train a specific detection system. In this paper we analyze how different training strategies and data augmentation techniques affect CNN-based deepfake detectors when training and testing on the same dataset or across different datasets.**

I. Introduction

As the number of techniques and algorithms to generate deepfake videos and swap faces grows rapidly, the effort of the forensic community is steering even more towards the development of reliable, robust, and automated deepfake detection methods. Techniques and pipelines for facial manipulation [1] and facial expression transfer between videos [2], [3] are rapidly improving [4], while the availability of source code (Deepfake [5], FaceSwap [6]) and even smartphone apps (Impressions [7], Doublicat [8]) makes face swapping available to a wider audience with either legitimate or harmful intents. Tampered video detection is not a novel task to the forensics community [9]–[11]. Codec history [12], [13], copy-move detection [14], [15], frame duplication or deletion [16], [17] are just a few examples of the many contributions in the last decades. The main drawback of the earlier systems developed by the community is that the exploited traces are inherently subtle and vanish with compression or multiple editing operations [10]. The first generation of deepfake detection methods exploited several semantic traces, including eye blinking [18], face warping [19], head poses [20] or lighting inconsistencies [21]. Due to the improvement of new and more accurate generation techniques, methods based on semantic artifacts began to fail, leading to the proposal of data-driven solutions capable of providing localization information through multi-task learning [22], attention mechanisms [23], and ensembles of CNN [24].

As detecting manipulated faces in videos becomes more important [25], [26], many deepfake detection systems proposed in the literature and in challenges are based on data-driven approaches, often backed by one or more CNNs trained on a specific dataset. However, the black-box model of data-driven CNN-based methods is notoriously prone to a drawback: over-fitting. Oftentimes, a bare train/validation/test split done within a single dataset collected with a uniform methodology and by a single team proves insufficient in avoiding over-fitting on that very same dataset conditions and scenarios. A recent example is shown in [27], where the winning model of the Facebook/Kaggle DeepFake Detection Challenge [28] scored an Average Precision of 82.56% on the public dataset used for the temporary leader board of the challenge, and then dropped to 65.18% on the sequestered dataset used for the final evaluation. Moreover, it is known that data dependency creates the risk of developing solutions unable to generalize over unseen methods or contexts.

While most detectors prove to be very effective on a test subset coming from the same data distribution they are trained on, what are the detection performance in a cross-dataset scenario? What happens when a CNN trained for deepfake detection on a dataset A is tested on dataset B, C, and D? As it is difficult to gain direct insights about what happens inside a CNN black-box model, in this paper we offer a set of preliminary analysis on cross-dataset performance of CNN-based deepfake detection approaches. Rather than focusing on developing a new technique optimized for a specific dataset, we train one of the most popular architectures used by competitors in the DeepFake Detection Challenge [28] and we evaluate how different training approaches [24] and data augmentation techniques [29] affect the intra-dataset and cross-dataset detection performances. We base our experiments on publicly accessible datasets, i.e., FaceForensics++ [30], the DeepFake Detection Challenge Dataset [28], and CelebDF(v2) [31]. We focus on faces extracted from deepfake videos rather than just deepfake images, as video compression is usually stronger than image compression. We also perform some analysis taking into account a limited availability of training data. Far from being an exhaustive evaluation or overview of all the available techniques and datasets, we wish to share with the readers some insights to consider when developing a new deepfake detection system.

WIFS'2020, December, 6-11, 2020, New York, USA. 978-1-7281-9930-6/20/$31.00 *This work was supported by the PREMIER project, funded by the Italian Ministry of Education, University, and Research within the PRIN 2017 program. Hardware support was generously provided by the NVIDIA Corporation.*

II. Methodology

In order to effectively compare the intra-dataset and cross-dataset detection performances, we first need to define a homogeneous training and testing methodology. The process of determining whether a face in a video is manipulated starts with a face detection and extraction phase. We rely on BlazeFace [32], a fast and GPU-enabled face detector, and we extract the face with the highest confidence from 32 frames for each video, uniformly sampled over time. This choice follows from [24], thus taking into account that time and computational power may be a limited resource. As the extracted faces have different scales and aspect ratios, we crop the faces with a fixed aspect ratio of 1:1 before resizing to a fixed size of 256×256 pixels. Once faces are extracted and uniform in size, we train an EfficientNetB4 [33] architecture as reference CNN, due to its popularity in the DeepFake Detection Challenge. The trained model is used to predict the likelihood of each face being fake. Results are reported at frame level as the Area Under Curve (AUC) of a Receiver-Operating-Characteristic (ROC) curve.

Among the several available datasets, we select the following four, due to their availability and ease of access and download:

- *DF*: FaceForensics [30], in its original version with 1000 real videos and 4000 fake videos generated with four different methods.
- *DFD*: Actors-based videos added to FaceForensics [34], with 363 real and 3068 fake videos.
- *DFDC*: The DeepFake Detection Challenge [28], with 19154 real and 100000 fake videos.
- *CelebDF*: The Celeb-DF(v2) dataset [31], with 890 real and 5639 fake videos.

The four dataset are divided into disjoint train, validation, and test sets at video level. In particular, for *DF* and *DFD* we follow the $720/140/140$ split proportion as suggested in [30]. For *DFDC* we use the folders from 40 to 49 as test set and the folders from 35 to 39 as validation set. The remaining 40 folders are the training set. For *CelebDF* we use the test set provided by the dataset itself, and we randomly select 15% of the videos as validation set, with the remaining 85% for training. For both *DF* and *DFD* we consider only the videos compressed with H.264 at CRF 23.

We run all our experiments with the PyTorch [35] framework on a workstation equipped with two Intel Xeon E5-2687W-v4 and several NVIDIA Titan V.

III. Baseline

As a baseline for the upcoming experiments, we first need to evaluate the deepfakes detection performance of EfficientNetB4 trained as a classifier using the Binary Cross Entropy (BCE) loss. The network is initialized with a model pre-trained on ImageNet, batch of 32 faces, Adam optimizer, initial learning rate of 10^{-4} multiplied by a factor 0.1 after 2000 batch iterations with no reduction in validation loss. The training ends when the learning rate falls below 10^{-8}. The final model is the one at the iteration that minimizes the validation loss. Training and validation batches are always balanced, with randomly selected equal amounts of real and fake faces. No data augmentation is performed at this stage. We train four CNN models on the training sets of the four datasets, then test each model against the test set of each dataset.

TABLE I
ROC AUC FOR BASELINE INTRA AND CROSS-DATASET DETECTION PERFORMANCE.

Train\Test	CelebDF	DF	DFD	DFDC
CelebDF	0.998	0.615	0.708	0.665
DF	0.734	0.960	0.844	0.695
DFD	0.754	0.636	0.987	0.669
DFDC	0.755	0.722	0.891	0.922

TABLE II
ROC AUC FOR TRIPLET TRAINING INTRA AND CROSS-DATASET DETECTION PERFORMANCE.

Train\Test	CelebDF	DF	DFD	DFDC
CelebDF	0.995	0.557	0.554	0.619
DF	0.717	0.960	0.829	0.684
DFD	0.759	0.709	0.882	0.666
DFDC	0.773	0.714	0.886	0.907

Results are reported in Table I, where the header column denotes the training dataset, while the header row reports the test dataset. Reading the table by rows, we observe how on *CelebDF* and *DFD* the intra-dataset detection is very accurate, with an AUC above 0.98. This, however, is not reflected on cross-dataset performance, as the model trained on *CelebDF* and tested on *DFD* presents an AUC of just 0.708 (29% gap compared to intra-dataset AUC), while the model trained on *DFD* and tested on *CelebDF* reaches an AUC of 0.754 (23% gap). The model trained on *DF* has a slightly lower AUC when tested on the same dataset (0.960) with a 12% gap when tested on *DFD*. *DFDC* is the dataset presenting the lowest intra-dataset AUC (0.922) being at the same time the one that generalizes better, with 3%, 17%, and 20% gap to *DFD*, *CelebDF*, and *DF*, respectively. The baseline results are in line with what expected from data-driven methods: the largest dataset (i.e., *DFDC*) seems to provide more variety during the training phase, thus better generalization on unseen data.

IV. Training strategy

The first analysis we perform is related to the training strategy adopted for the CNN. Instead of relying on Binary Cross Entropy (BCE) loss, we train the CNN with a triplet loss [36], by running the CNN up to the last-minus-one layer (features layer). Considering triplets as (anchor sample, positive sample, negative sample), we generate the training triplets as (fake face, fake face, real face) and (real face, real face, fake face) in an equal number for each batch, so to balance the batch itself. The training proceeds in a two step fashion.

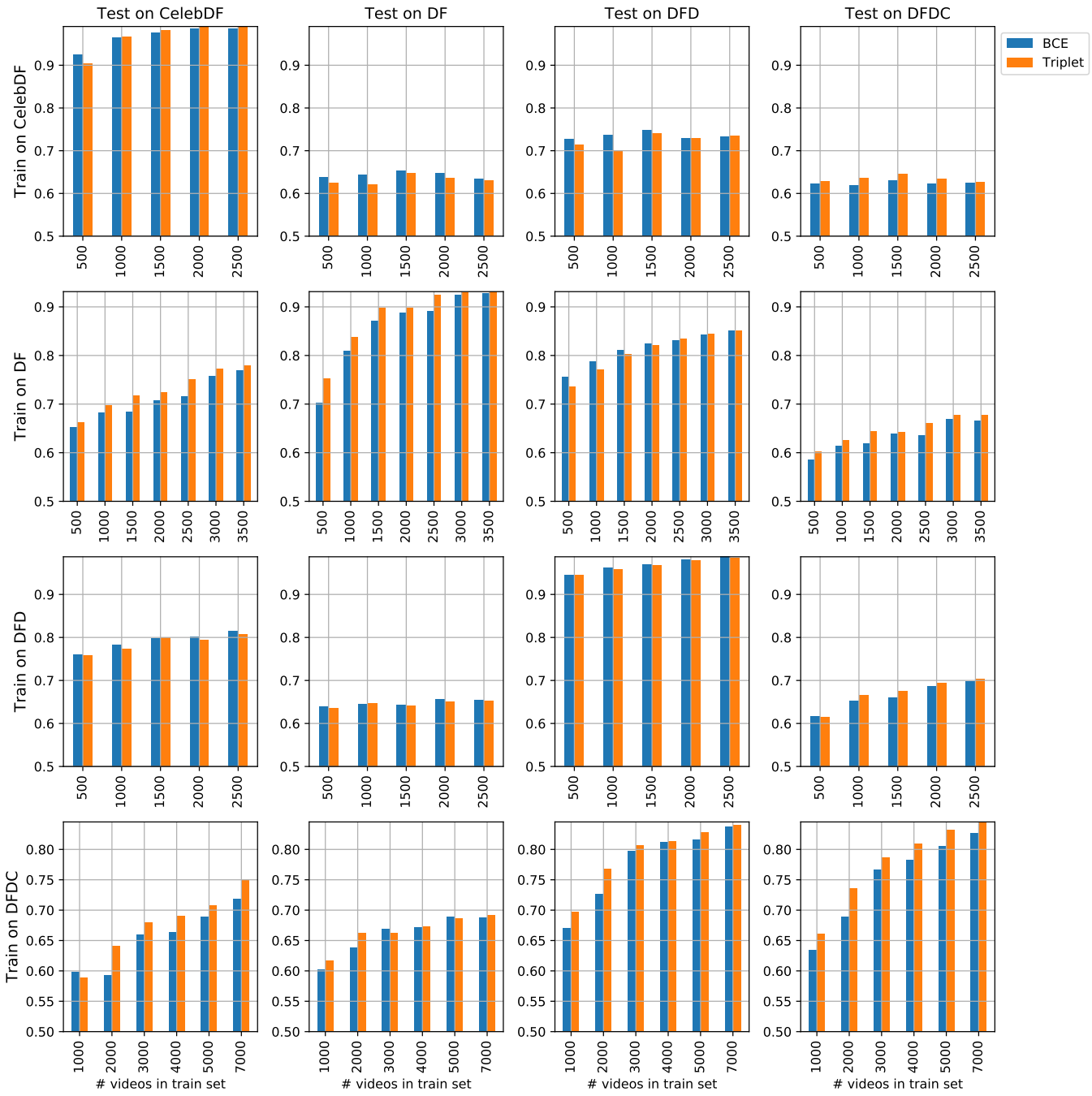
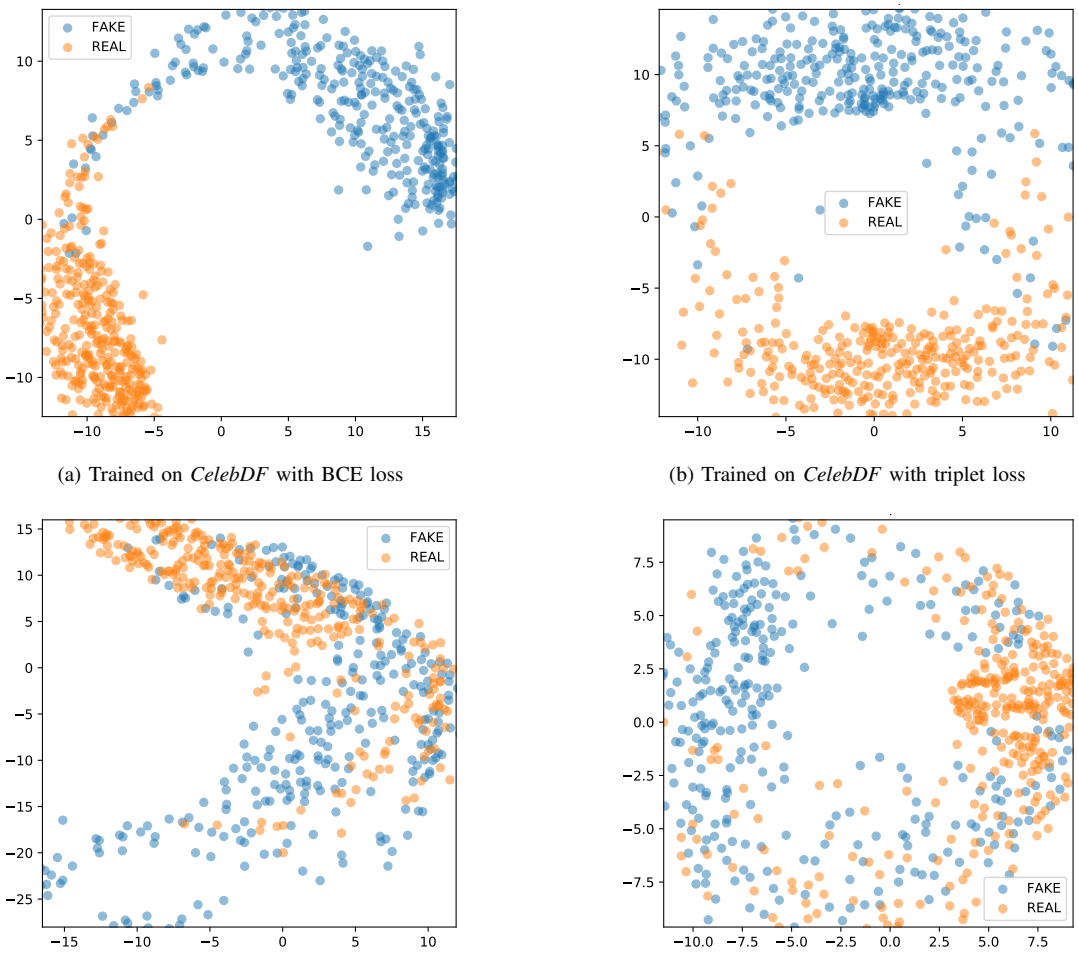


Fig. 1. ROC AUC for BCE and triplet training in data-limited conditions. For each dataset two CNNs are trained selecting an increasing number of videos with BCE and triplet loss. Interestingly, we can see that the cross-dataset performances are generally higher on *DFD*. This might be related to the overall quality of the dataset: while *DFD* consists generally of high resolution videos, the other ones are more various and present also low quality samples. Training for detection in such difficult settings therefore might be helpful in generalizing on different, yet of higher quality, datasets.



(a) Trained on *CelebDF* with BCE loss

(b) Trained on *CelebDF* with triplet loss

(c) Trained on *DFDC* with BCE loss

(d) Trained on *DFDC* with triplet loss

Fig. 2. MDS projection of 10 pairs of REAL/FAKE videos from the *CelebDF* test dataset. Each point represent a frame in a video, 32 frames are extracted from each video. Projections are produced starting from the features extracted by an EfficientNetB4 architecture trained on different datasets with binary cross entropy (BCE) or triplet loss.

In the first step the CNN is trained with triplet loss up to the features layer. In the second step, only the last layer (classifier) of the CNN is trained (fine tuned) with binary cross entropy. In the context of the EfficientNetB4 architecture, feature vectors are 1792 elements while the classifier has 1793 weights (1792 multipliers and one bias coefficient). This means the classification layer accounts for less than 0.01% of the net coefficients. Triplet training is initialized with the model trained through BCE from the baseline, as this provides a faster convergence and prevents the model from failing into a trivial solution (all-zeros feature vector). The batch size is 10 triplets to fit into 12GB of GPU memory, the initial learning rate is set to 10^{-5} and it is dropped by a factor 10 after 500 batch iterations with no improvements on the validation loss.

The fine tuning of the classifier is initialized with the triplet-trained model, with an initial learning rate of 10^{-6} dropped by a factor 10 after 100 iterations with no validation loss improvements. Both the triplet training and the fine-tuning process are stopped when the learning rate falls below 10^{-8}. For both steps, the model at the iteration with the smallest validation loss is selected as the final one.

As for the baseline, we are interested in understanding both the intra and cross-dataset detection performance, as reported in Table II. The results for *DF*, *DFD*, and *CelebDF* show almost the same intra-detection AUC as with BCE training, with a loss in generalization capability more marked for the *CelebDF* dataset. For the model trained on *DFDC*, the intra-detection AUC is similar to the BCE training, with slightly better cross-dataset AUC (with a modest 2% increase in AUC with respect to the same combination in BCE training) only when testing on *CelebDF*.

A different perspective on the differences between BCE and triplet losses is offered in Figure 1, where EfficientNetB4 is trained in data-limited conditions by sub-sampling the training dataset. In this context, triplet loss proves beneficial in intra-dataset detection (*DFDC*, *CelebDF*, and *DF*) as well as in cross-dataset detection and outperforms BCE.

Even though the triplet training procedure is not revolu-

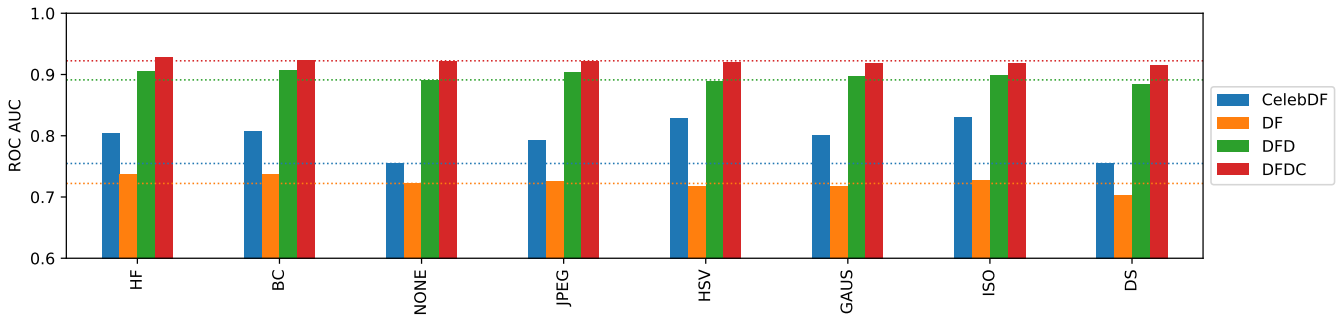


Fig. 3. ROC AUC of EfficientNetB4 trained with BCE on *DFDC* with different augmentation techniques. HF: Horizontal Flip. BC: Brightness and Contrast change. HSV: Hue, Saturation and Value changes. ISO: Addition of ISO noise. GAUS: Addition of Gaussian noise. DS: Down-scaling. JPEG: JPEG compression. MIX: baseline mix of all the other single augmentations. NONE: no augmentations. The horizontal lines are the AUC values when no augmentations are used.

TABLE III
ROC AUC FOR EFFICIENTNETB4 TRAINED WITH BCE AND SELECTED AUGMENTATIONS.

Train\Test	CelebDF	DF	DFD	DFDC
CelebDF	0.998	0.616	0.795	0.673
DF	0.764	0.966	0.847	0.691
DFD	0.842	0.650	0.990	0.690
DFDC	0.826	0.733	0.923	0.919

TABLE IV
ROC AUC FOR FOR EFFICIENTNETB4 TRAINED WITH TRIPLET LOSS AND SELECTED AUGMENTATIONS.

Train\Test	CelebDF	DF	DFD	DFDC
CelebDF	0.995	0.570	0.604	0.595
DF	0.779	0.963	0.858	0.682
DFD	0.809	0.658	0.982	0.694
DFDC	0.777	0.725	0.905	0.889

tionary in terms of AUC, we are interested in analyzing the differences in representations learned at feature level with BCE and triplet loss on the same dataset and across different datasets. To this end, Figure 2 shows the Multidimensional Scaling (MDS) projection on two components of the features extracted with four differently trained EfficientNetB4 models. All four subplots project faces from the very same 10 pairs of real/fake videos randomly extracted from the *CelebDF* test set. Figure 2a uses features extracted with the CNN trained on the *CelebDF* dataset with BCE, while Figure 2b uses features extracted with the CNN trained on the same dataset with triplet loss instead. While in both cases the separation between real and fake frames is quite evident, in the triplet case the overlapping frames are less in number. This improvement could prove useful when aggregating the predictions from several frames at video level. Figure 2c and 2d are generated with features extracted by CNNs trained on *DFDC* with BCE and triplet loss respectively. While certainly the overlap between real and fake frames is more evident than in Figures 2a and 2b, the triplet loss seems to offer a bit more separation between the two classes, despite the feature extractor being trained on a different dataset.

V. DATA AUGMENTATION

The second batch of experiments is devoted to understanding the effect of different data augmentation techniques. It is known that for deepfake images [29] some type of data augmentation techniques prove beneficial in terms of robustness and cross-dataset generalization. Among the many possible data augmentations techniques, we focus on the subset that could represent the transformations a face undergoes in the wild. The following augmentations are considered:

- HF: Horizontal Flip
- BC: Brightness and Contrast changes
- HSV: Hue, Saturation and Value changes
- ISO: Addition of ISO noise
- GAUS: Addition of gaussian noise
- DS: Downscaling with a factor between 0.7 and 0.9
- JPEG: JPEG compression with a random quality factor between 50 and 99

We test the aforementioned augmentations independently, training with BCE on the *DFDC* dataset. All the proposed experiments are performed with the Albumentations [37] framework. Results are reported in Figure 3, ordered left to right in decreasing order of AUC on the *DFDC* test set. Two interesting considerations can be drawn in light of these results.

First, augmentations do not seem to help much increasing intra-dataset detection, maybe due to the cross-contamination between train, validation, and test set in terms of video settings and scenarios. The only exception is the HF augmentation, that provides a boost of just 0.7% in AUC.

Second, some augmentations are beneficial (at times by a large margin) in terms of cross-dataset generalization. In particular, HF, BC, HSV, and JPEG provide for an AUC increase on networks trained on both *CelebDF* and *DFD*.

DF does not seem to benefit much from augmentations, maybe due to the very different scenes depicted in *DFDC* compared to the ones in *DF*. While the former has actors at distance, moving in the scene, often two actors, the latter has almost only a single actor, in the center of the scene, in a TV studio or during an interview with studio-level lights.

In light of the results in terms of single augmentation,

we build a data augmentation pipeline based on HF, BC, HSV, and JPEG, and re-train the CNN with both BCE and triplet loss. Table III reports results for BCE loss. The fusion of augmentations brings important improvements in terms of cross-dataset detection AUC, with up to $+9\%$ when training on *DFD* and testing on *CelebDF*, and when training on *CelebDF* and testing on *DFD*. The intra-dataset detection performances are instead mostly unaffected. With augmentations applied to the CNN trained with triplet loss, Table IV shows how the few beneficial effects of triplet loss when training on full dataset are not visible anymore. In facts, triplet loss with data augmentations provides lower AUC for almost all combinations compared to BCE loss with data augmentation.

VI. Conclusions

Two are the main conclusions we can draw from the experiments presented in this paper. First, a carefully built and tested data-augmentation pipeline can prove useful in increasing the generalization of a CNN model for deepfake video detection across different datasets. Not all augmentations are beneficial though, and checking the usefulness of each type of augmentation could be an important step in the workflow of developing a detection pipeline. Second, triplet loss proves to be helpful in terms of both intra-dataset and cross-dataset detection performances under limited availability of training data. When large datasets are available, data augmentation on a BCE-trained CNN architecture proves to be the winning combination.

References

[1] M. Zollhöfer, J. Thies, P. Garrido, D. Bradley, T. Beeler, P. Pérez, M. Stamminger, M. Nießner, and C. Theobalt, "State of the art on monocular 3D face reconstruction, tracking, and applications," *Computer Graphics Forum*, vol. 37, pp. 523–550, 2018.

[2] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2face: Real-time face capture and reenactment of RGB videos," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] J. Thies, M. Zollhöfer, and M. Nießner, "Deferred neural rendering: Image synthesis using neural textures," *ACM Transactions on Graphics (TOG)*, vol. 38, pp. 1–12, 2019.

[4] L. Li, J. Bao, H. Yang, D. Chen, and F. Wen, "Advancing High Fidelity Identity Swapping for Forgery Detection," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[5] "Deepfakes github," https://github.com/deepfakes/faceswap.

[6] "Faceswap," https://github.com/MarekKowalski/FaceSwap/.

[7] "Impressions," https://impressions.app/.

[8] "Doublicat," https://doublicat.com/.

[9] A. Rocha, W. Scheirer, T. Boult, and S. Goldenstein, "Vision of the unseen: Current trends and challenges in digital image and video forensics," *ACM Computing Surveys*, vol. 43, pp. 1–42, 2011.

[10] S. Milani, M. Fontani, P. Bestagini, M. Barni, A. Piva, M. Tagliasacchi, and S. Tubaro, "An overview on video forensics," *APSIPA Transactions on Signal and Information Processing*, vol. 1, p. e2, 2012.

[11] M. C. Stamm, Min Wu, and K. J. R. Liu, "Information forensics: An overview of the first decade," *IEEE Access*, vol. 1, pp. 167–200, 2013.

[12] P. Bestagini, S. Milani, M. Tagliasacchi, and S. Tubaro, "Codec and gop identification in double compressed videos," *IEEE Transactions on Image Processing (TIP)*, vol. 25, pp. 2298–2310, 2016.

[13] D. Vázquez-Padín, M. Fontani, D. Shullani, F. Pérez-González, A. Piva, and M. Barni, "Video integrity verification and GOP size estimation via generalized variation of prediction footprint," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 15, pp. 1815–1830, 2020.

[14] P. Bestagini, S. Milani, M. Tagliasacchi, and S. Tubaro, "Local tampering detection in video sequences," in *IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2013.

[15] L. D'Amiano, D. Cozzolino, G. Poggi, and L. Verdoliva, "A patchmatch-based dense-field algorithm for video copy-move detection and localization," *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, vol. 29, pp. 669–682, 2019.

[16] M. C. Stamm, W. S. Lin, and K. J. R. Liu, "Temporal forensics and anti-forensics for motion compensated video," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 7, pp. 1315–1329, 2012.

[17] A. Gironi, M. Fontani, T. Bianchi, A. Piva, and M. Barni, "A video forensic technique for detecting frame deletion and insertion," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014.

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

[19] Y. Li and S. Lyu, "Exposing deepfake videos by detecting face warping artifacts," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

[20] X. Yang, Y. Li, and S. Lyu, "Exposing deep fakes using inconsistent head poses," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

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

[22] H. H. Nguyen, F. Fang, J. Yamagishi, and I. Echizen, "Multi-task learning for detecting and segmenting manipulated facial images and videos," *CoRR*, vol. abs/1906.06876, 2019.

[23] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. Jain, "On the detection of digital face manipulation," *CoRR*, vol. abs/1910.01717, 2019.

[24] N. Bonettini, E. D. Cannas, S. Mandelli, L. Bondi, P. Bestagini, and S. Tubaro, "Video Face Manipulation Detection Through Ensemble of CNNs," in *International Conference on Pattern Recognition (ICPR)*, 2020.

[25] S. Agarwal, H. Farid, Y. Gu, M. He, K. Nagano, and H. Li, "Protecting world leaders against deep fakes," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

[26] L. Verdoliva, "Media forensics and deepfakes: an overview," *CoRR*, vol. abs/2001.06564, 2020.

[27] "DeepFake Detection Challenge Results," https://ai.facebook.com/blog/deepfake-detection-challenge-results-an-open-initiative-to-advance-ai.

[28] B. Dolhansky, J. Bitton, B. Pflaum, R. Lu, Jikuo ans Howes, M. Wang, and C. Canton Ferrer, "The deepfake detection challenge dataset," *CoRR*, vol. abs/2006.07397, 2020.

[29] S.-Y. Wang, O. Wang, R. Zhang, A. Owens, and A. A. Efros, "Cnn-generated images are surprisingly easy to spot... for now," *CoRR*, vol. abs/1912.11035, 2019.

[30] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "FaceForensics++: Learning to detect manipulated facial images," in *International Conference on Computer Vision (ICCV)*, 2019.

[31] Y. Li, P. Sun, H. Qi, and S. Lyu, "Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics," in *IEEE Conference on Computer Vision and Patten Recognition (CVPR)*, 2020.

[32] V. Bazarevsky, Y. Kartynnik, A. Vakunov, K. Raveendran, and M. Grundmann, "Blazeface: Sub-millisecond neural face detection on mobile gpus," *CoRR*, vol. abs/1907.05047, 2019.

[33] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning (ICML)*, 2019.

[34] "FaceForensics++," https://ai.googleblog.com/2019/09/contributing-data-to-deepfake-detection.html.

[35] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems (NIPS)*, 2019.

[36] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[37] A. V. Buslaev, A. Parinov, E. Khvedchenya, V. I. Iglovikov, and A. A. Kalinin, "Albumentations: fast and flexible image augmentations," *CoRR*, vol. abs/1809.06839, 2018.