Deepfake Detection with Deep Learning

Marten Thompson
Department of Statistics, University of Minnesota

Introduction

Falsified videos that convincingly replace the original subject with another have emerged as a new form of online misinformation. When they are the product of deep learning algorithms they are known as deepfakes. Significant investments by private companies, academic researchers, and individual hobbyists have produced sophisticated, easily-accessed machine learning tools for producing these manipulated videos.

While traditional photo-editing software typically requires considerable effort on the part of the creator, these tools automatically transfer the facial features of one person onto another. For example, the open source project DeepFaceLab aims to be an "easy-to-use pipeline for people to use with no comprehensive understanding of deep learning frameworks" to create deepfakes on their personal computers [7]. Face-swapping apps like ZAO[11] and FaceApp[5] have also introduced this capability to a mobile environment.

The proliferation of deepfake software has generated a growing interest in discerning whether or not a given image or video has been altered. Both academics and the wider computer science community are developing automatic processes for their detection. Recently, Amazon Web Services, Facebook, Microsoft, and others organized The Deepfake Detection Challenge (DFDC) dataset[4][6], currently the largest publicly-available dataset of face-swapped videos. The DFDC was released with an accompanying Kaggle competition which closed in March, 2020.

With this data we explored the ability for deep convolutional networks and residual networks to correctly classify videos as unchanged or altered by deepfake models (real/fake hereafter). There is abundant literature on image classification (for a recent survey, see Xin et al.[10]), and we applied similar models to each frame of our video data. We refer to these methods as 2D to describe their flat input. For an $H \times W$ pixel image with RGB color channels, these models accept batches of $H \times W \times 3$ tensor input. In order to leverage all dimensions of the data, we also considered 3D models that intook videos comprised of F many frames i.e. $F \times H \times W \times 3$ tensors. Throughout we chose $F = 16$ as de Lima et al. did in similar work on another deepfake dataset [2].

DFDC Dataset

While machine learning methods are instrumental to successful deepfake creation, they may ultimately be credited with their detection. AWS, Facebook, Microsoft, and The Partnership on AI's Media Integrity Steering Committee created the DFDC dataset and public competition in order to motivate research into deepfake detection models. The DFDC dataset contains over 128,000 videos of 3,426 consenting subjects and provides the largest corpus to date for model training and evaluation [6]. Each video is stored as (possibly multiple) 300 frame sequences. We use a subset of the 25+ terabytes of videos, approximately 4Gb.

Of the 128,000 videos, approximately 85% contain faked content. The DFDC creators chose five popular deepfaking methods (including DeepFaceLab) to represent deepfakes of varying quality. Figure 1 shows two typical examples of more believable face swappings. Facial features are clean in both head poses. Figure 2, however, depicts two common telltale signs of forgery: inconsistencies around eyes and reduction in image quality.

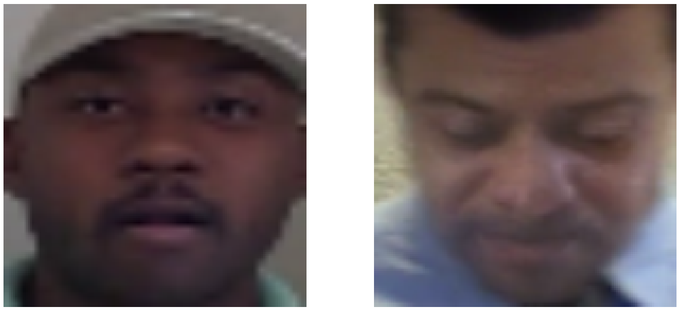


Figure 1: Convincing deepfake frames

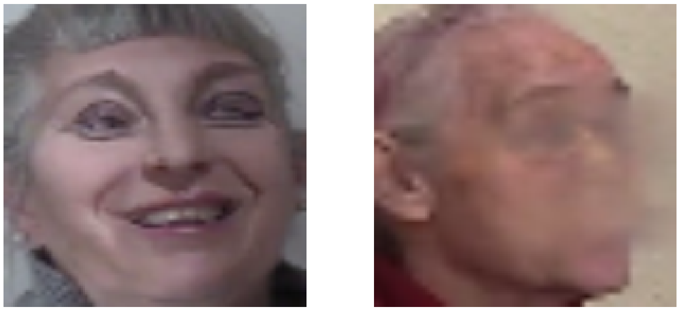


Figure 2: Less-convincing deepfakes

An important aspect of our work was to consider the relationship between sequential frames. We believed the temporal nature of video frames contained information germane to detecting fake content. For example, Figure 3 shows two consecutive frames in a faked video. Considering them individually would discard the uneven treatment of the source video by the generating deepfake algorithm in which the younger man's face was not properly superimposed on its target in the left hand frame.

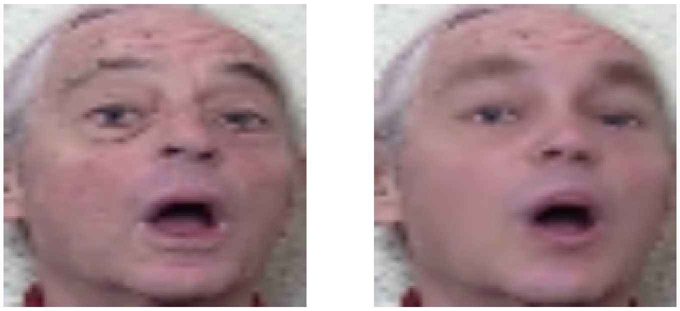
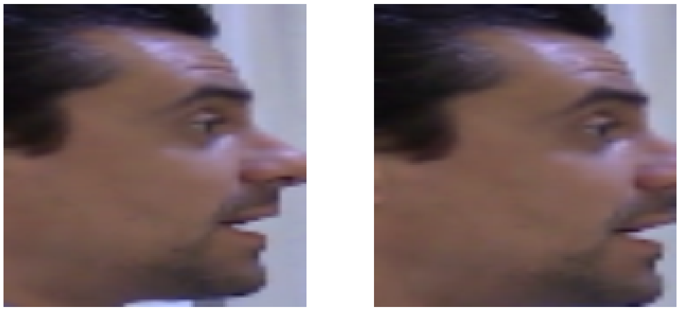


Figure 3: Consecutive frames exhibiting temporal flickering

This inconsistent generation of deepfakes for DFDC also presented a challenge. While we did not perform an exhaustive search across all available data, we found several examples of seemingly identical videos labeled both real and fake. Figure 4 shows two frames from two different 300 frame clips with differing labels. It appears as if a video can appear in the dataset twice if one copy was passed through a deepfake model. However, in this case the man's face was not replaced, possibly due to its turned angle. Since this was the case for an entire 300 frame sequence, it resulted in essentially identical input being labeled both real and fake.



(a) Unaltered video frame (b) Purported deepfake

Figure 4

While this certainly presented a difficulty, it was not without its value. The classifiers the DFDC motivates intend to be applicable to real-world deepfakes. Since such videos are themselves imperfect at times, we did not consider it inappropriate to reflect this in training data.

Results

Our original assumption was that basic image classification frameworks like convolutional neural networks and residual networks would be able to correctly label individual frames like Figure 2. It was also possible that they would detect characteristics of deepfake images too subtle to see in Figure 1. These 2D methods treated each frame of a video as a separate image for training. Since the basic components of these networks generalize to 3D input, we applied similar models to video input. Finally, we augmented pre-existing models with custom classification layers to suit our classification problem. All results can be found in Figure 8 and Table 1. Due to the imbalance of real and fake data (85:15), models needed to exceed 0.85 testing accuracy to make meaningful contributions to the classification problem.

Convolutional Neural Networks

We created several convolutional neural networks that stacked variations of the basic block depicted in Figure 5. We considered blocks with one, two, and three convolutions before pooling and convolutions with 8, 16, and 32 filters. Network depths ranged from one to four blocks. All of these networks ended in one, two, or three fully connected layers before the final output node. These fully connected layers were trained with dropout with probabilities ranging from zero to 0.5.

Surprisingly, one of simplest models was among the best. Its performance is detailed in Figure 8 and Table 1; it performed only $k = 1$ convolution per block, had three such blocks all with 32 filters, one fully connected layer with 64 nodes, and used a dropout probability of 0.5. Since deeper models did not perform better, we took this as evidence that we had reached the ceiling that combinations of these basic network elements could achieve.

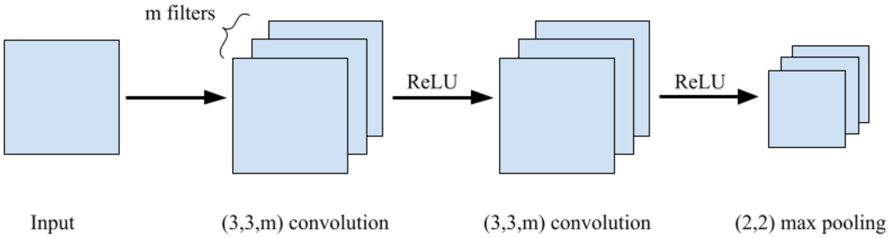


Figure 5: Basic convolution network element

We also created versions of these models that accepted 3D input and performed 3D convolutions and pooling. They were largely inferior to those of their 2D counterparts. A small 3D CNN with a single block of $k = 3$ convolutions all with 8 filters feeding a dense layer of 64 nodes performed best with a test accuracy of 0.853.

With this relative performance in mind, we also created a model that blended 3D and 2D elements. Referred to as MC2 below, it passed input through two 3D convolutions before pooling across the temporal axis and feeding 2D convolutions. We expected this network to extract the important temporal information before leveraging the more successful 2D architectures. Unfortunately, this did not result in an increase in accuracy (0.849 on test data).

Residual Networks

We next considered 2D and 3D residual networks in an effort to promote learning on deeper models. Their basic unit was the combination of convolutions and pass-through shown in Figure 6, of which we explored the following variations: two or three convolutions per block, filters of size 16, 32,. . . , 128, varying the number of filters between blocks, and only applying the ReLU activation after the last convolution. All of these ended in a global average pooling layer before feeding fully connected layers as in our CNNs above. Disappointingly, these did not exceed the performance of either the 2D or 3D CNNs.

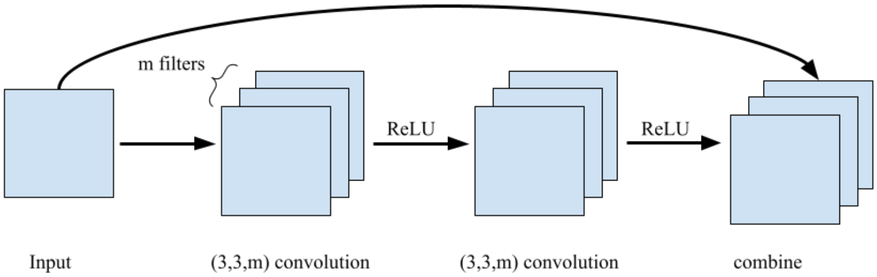


Figure 6: Basic residual network element

Transfer Learning

Finally, we took two well-known classifiers: MobileNetV2 [9] and I3D [1], the former for image classification and the latter for 3D input. MobileNetV2 is a 20 layer CNN that employs bottleneck layers to limit its number of parameters (2,258,000 in total). These convolution layers are similar to those in Figure 5, with the first convolution replaced with a 1×1 kernel. This creates the appropriate number of channels for later convolutions and preserves depth without requiring as many parameters. The I3D model employs a new basic network unit: the inception block (drawn below for 3D input, not depicting m filters). This architecture employs many different sized kernels in a single layer to detect features at different scales without increasing depth.

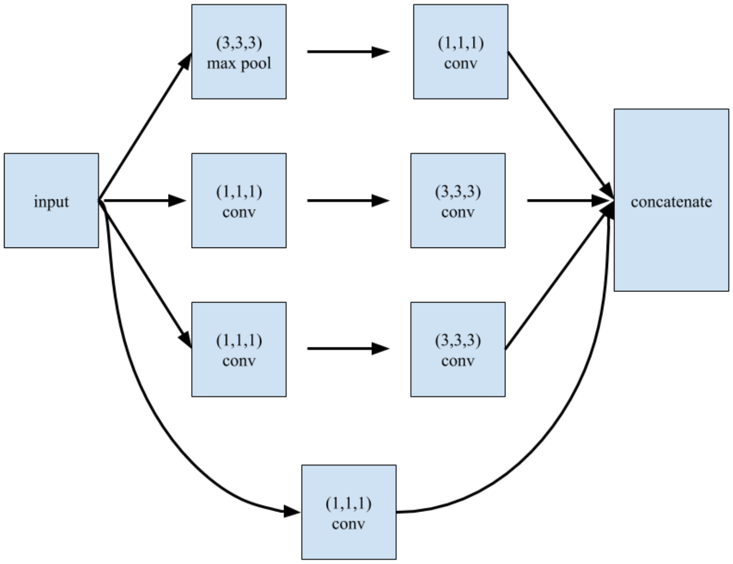


Figure 7: Basic inception block

MobileNetV2 and I3D were used as-is with their convolution parameters frozen. We removed their final fully connected layers and replaced them with our own for training. This leveraged the

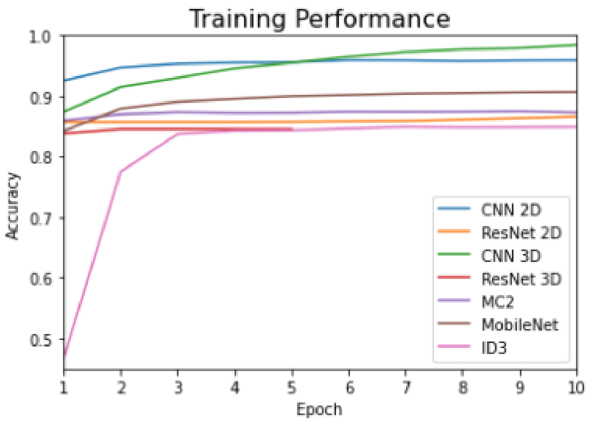


Figure 8: Training was largely complete in the first 3 epochs

Model	Accuracy
MobileNetV2	0.915
CNN 2D	0.884
I3D	0.855
CNN 3D	0.853
ResNet 2D	0.849
MC2	0.849
ResNet 3D	0.843

Table 1: Models must exceed 0.85 accuracy to make meaningful contributions

feature extraction of each existing network and tailored the output to suit our data. Again, the 2D model bested the 3D model, with MobileNetV2 producing the most accurate predictions overall.

Discussion

Overall, our results confirmed the difficulty of the problem posed by the DFDC dataset. While we described some of its inconsistencies and resulting difficulties, we do not consider these a flaw in the challenge. It typifies real-world video data where potential deepfakes may not be perfect.

Leveraging temporal information with 3D models did not appear to improve the performance of any architecture. The CNNs, ResNets, and existing models all found better accuracy when applied to 2D input. Other authors have found more success applying similar methods to alternative deepfake datasets. For example, the I3D model input achieved an accuracy of 0.923 on the Celeb-DF deepfake data [2]. There, authors pre-trained on the Charades data [8], whereas ours was trained on Kinetics [3]. Rather than employing MobileNetV2 and I3D as-is, future work could explore the effect of different pre-training data on final performance.

References

[1] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[2] Oscar de Lima, Sean Franklin, Shreshtha Basu, Blake Karwoski, and Annet George. Deepfake detection using spatiotemporal convolutional networks. *arXiv preprint arXiv:2006.14749*, 2020.

[3] Kinetics — deepmind. `https://deepmind.com/research/open-source/kinetics`.

[4] Brian Dolhansky, Joanna Bitton, Ben Pflaum, Jikuo Lu, Russ Howes, Menglin Wang, and Cristian Canton Ferrer. The deepfake detection challenge dataset. *arXiv preprint arXiv:2006.07397*, 2020.

[5] FaceApp. `https://www.faceapp.com`, (accessed 12/10/2020).

[6] Inc. Facebook. Deepfake detection challenge. `https://www.kaggle.com/c/deepfake-detection-challenge`, (accessed 12/10/2020).

[7] Ivan Petrov, Daiheng Gao, Nikolay Chervoniy, Kunlin Liu, Sugasa Marangonda, Chris Umé, Jian Jiang, Luis RP, Sheng Zhang, Pingyu Wu, et al. Deepfacelab: A simple, flexible and extensible face swapping framework. *arXiv preprint arXiv:2005.05535*, 2020.

[8] Perceptual reasoning and interaction research charades. `https://prior.allenai.org/projects/charades`.

[9] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[10] Mingyuan Xin and Yong Wang. Research on image classification model based on deep convolution neural network. *EURASIP Journal on Image and Video Processing*, 2019(1):40, 2019.

[11] ZAO. `https://zaodownload.com`, (accessed 12/10/2020).