Optical Flow Based DeepFake Detection

Bhargav Naraparedy, Ching-Yen Shih, Yuchi Wang, Bo-Hong Cho, Nicholas Wendt
{narbhar, cyshih, yuchiw, bhcho, nwendt}@umich.edu

April 30, 2020

Abstract

The challenge of DeepFake detection is one of the widely researched topics in recent times given the implications of DeepFakes in the age of the internet. DeepFakes are a class of videos that are generated by altering an original video by swapping faces of subjects with others of similar likeness, thus creating an illusion of the latter's authenticity. We explore the possibility of creating a DeepFake detection method by employing optical flow as its primary tool. We explore and compare DeepFake detection efficacy to traditional dense optical flow technique known as Farneback optical flow as well as some of the recent deep learning-based optical flow estimators such as FlowNet, PWC-Net, and SPyNet. We also compare the classification performance of pairwise optical flow-based networks to temporal networks employing RNN, LSTM, and Attention networks.

1 Introduction

The very popular term "DeepFake" is referred to a deep learning-based technique able to create fake images/videos by swapping the face of a person in an image or video by the face of another person. [30] At its worst, it poses a nightmare for security and privacy since the data needed to create DeepFakes are a few well-lit sample photos to create a convincing fake video. This is especially harmful to people with high public exposure since enforcing copyright laws to images is very tedious and hence data acquisition can't be stopped if a bad actor chooses to go after it. Fake political speeches designed to rile political bases, pornographic impersonations of celebrities are a few examples in this regard.

Traditional image manipulation techniques required sophisticated editing software and lots of trivial details need extensive domain knowledge to produce a convincing output. These days however due to advent of deep networks such as Generative Adversarial Networks(GANs) with Convolutional Neural Networks(CNNs) can produce very realistic fake videos within hours of training them.[28]

2 Related Work

2.1 DeepFake Detection

Due to the ramifications of DeepFakes on privacy and security discussed in the previous section, it is therefore vital to develop detection mechanisms to DeepFakes. The most popular database for this face swap problem is Face2Face[29], FaceForensics[22] and FaceForensics++[23]. Face2Face approach transfers the expression of source video while maintaining the identity of the target person. FaceForensics and FaceForensics++ focus on facial expression only and are generated by computer graphics and some learning approach. FaceForensics++ also proposed a rendering approach to learn a neural texture of the target person. As for the face manipulation detection approach, Afchar et al.[1] proposed two networks, both of these networks focus on the mesoscopic properties. In [9], the author used the CNNs to extract frame-level features and these features then feed into RNNs for fake video detection training. For the RNN system, they used the Long Short Term Memory(LSTM) for sequence processing. Yuezun Li and Siwei Lyu[14] showed an approach to solve the issue that the DeepFake algorithm can only generate images of limited resolution. The authors also used four kinds of CNN models: VGG16[26], ResNet50, ResNet101, and ResNet152[10] for training.

2.2 Optical Flow

Optical flow is a vector field to extract motion among two sequential frames and can be used to assist some face detection challenges such as activity recognition and face manipulation. In [25], the authors proposed a two-stream ConvNet to do action recognition in videos. This ConvNet is processed with an RGB video frame and dense optical flow in two separate branches. The result showed that training with the optical flow can have better performance than training on raw stacked frames [13]. Amerini et al. [2] proposed a sequence-based approach that uses the optical flow extracted by the PWC-Net to investigate possible dissimilarity in the inner-frame of a video.

2.2.1 Gunnar-Farneback Optical Flow

The Gunnar-Farneback algorithm[7] is developed to produce dense optical flow. The first step is to approximate each neighborhood of both frames by quadratic polynomials. Afterward, considering these quadratic polynomials, a new signal is constructed by a global displacement. Finally, this global displacement is calculated by equating the coefficients in the quadratic polynomials' yields. This method is widely used in optical flow estimation since it is less computational heavy than the ground truth computational method.

2.3 Neural Network Approximations

Since the direct computation of optical flow in the Farneback method is very time-consuming, there exist deep learning methods to approximate. Therefore, it had been a popular topic to find a desirable network structure to best construct the optical flow from the image. The first paper that can compete with the Farneback method in an artificial dataset is FlowNet[6]. FlowNet is mostly composed of convolution layers. The novel architecture that makes FlowNet surpass its previous works is the combination of ”extracting part”, ”expanding part”, and ”refining part”. The ”extracting part” serves as a feature extractor as most of the Convolutional Neural Network (CNN) structure does. It is designed to detect the patterns between original images and extract features from each frame on its feature space. It first produces meaningful representations of the two images separately and then combines the two images and then combine them on a higher level. To aid the matching process of the two images, it introduces a ”correlation layer” that performs multiplicative patch comparisons between two feature maps. For the ”expanding part”, it consists of ”up-convolutional” layers and ”up-pooling” layers. This part of the network is designed to upscale and refine the correlation features to original resolutions. The operations between layers are to apply the "up-convolution" to feature maps, to concatenate it with corresponding feature maps from the "extracting part" of the network and an up-sampled coarser flow prediction. For the "refining part", it uses the variational approach without the matching term.

While FlowNet provides a basic scheme for how a frame of optical flow can be generated smoothly with a deep neural network, its performance on a real-world task such as Sintel[4] is still relatively poor. In this case, FlowNet2.0[12] refines the FlowNet structure and gets competitive results against the variational approach. The network uses the methodology of iterative refinement and stacks original multiple FlowNet models. It found the best way to stack the models is to keep the first network fixed and only train on the second network after warping operations. However, stacking models to form a meta-model raises the computational time in both training and inference. This repeats the problem we mentioned in the traditional optical flow method.

2.3.1 PWC-Net

PWC-Net[27] uses a coarse-to-fine approach scheme and combines it with the previous FlowNet network structure. It has four proposed tricks to enhance its performance in contrast to FlowNet. First, to obtain the shadows and lighting changes from raw images, it replaces the fixed images pyramid with learnable feature pyramids. Second, it succeeds in the traditional method of warping operation as a layer in its network to observe large movements. Third, it proposes a novel cost volume layer to discriminate optical flows. This layer and the warping layer have fixed parameters hence significantly reduced the model size. Last, it post-process the optical flow by using contextual information, such as median-filtering and bi-literal filtering. This method is often used in traditional coarse-to-fine methods. As a whole, PWC-Net possesses the performance competitive against FlowNet2.0 on real-world datasets such as KITTI and Sintel. Nevertheless, it is much lighter on trainable parameters.

2.3.2 SPYNet

SpyNet[20] is also an light-weighted version extended from the original FlowNet. It also uses the coarse-to-fine structure in building the network. The network structure is designed to learn residual flow to each pyramid level. The novel proposal in SpyNet is spatial sampling, which defines a warping operator $w(I, V)$ with bi-linear interpolation where I and V respectively stands for images and optical flows. In overall, SpyNet has almost the same parameter amount as FlowNet does. It has better performance than FlowNet on real-world tasks. When it comes to comparison with other heavy state-of-the-art models such as FlowNet2.0, it has worse performance. Hence, we can conclude that it serves as a trade-off model between computational complexity and performance.

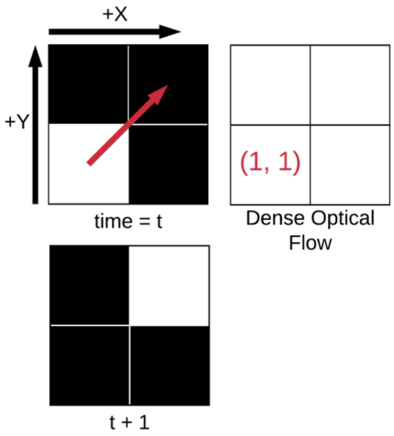

Figure 1: Optical Flow illustration

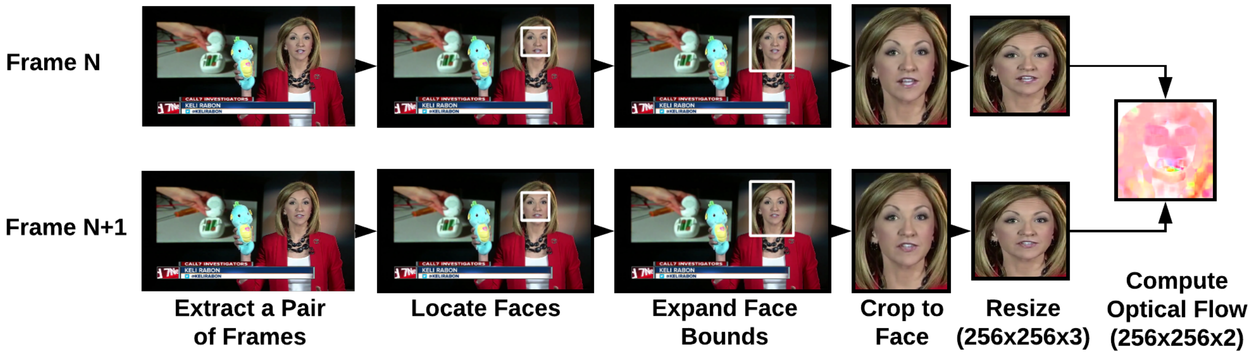

Figure 2: Preprocessing steps used to generate optical flow face images from video. Video frames extracted from [23].

3 Preliminaries

Optical flow[11] can be defined as the apparent motion of brightness patterns between images. Optical flow is used to understand the perception of motion between images.

There are two key assumptions that are made to optical flow calculation:

- Brightness constancy - The intensity of a point in an image at time 't' is same at time 't+1' It can be represented by the following equation:

$$I(x,y,t) = I(x+u,y+v,t+1) \tag{1}$$

  where u, v represent the displacement of the point in x, y directions respectively.

- Points of image don't move very far in this timeframe

Differential methods are the most widely used techniques for optical flow computation in image sequences. One of classic technique among them is the Lucas-Kanade Optical Flow[15] proposed by B.D.Lucas and T.Kanade. It assumes that the flow is nearly constant among the neighbor pixels and solve the optical flow equations by the least-squares criterion. Since the computation optical flow is quite time-consuming, some deep learning-based networks which aim to extract the optical flow exist such as FlowNet, PWC-Net, and SPYNet. These three networks are introduced in the Related Work 2.3 section.

4 Method

4.1 Input Preprocessing

The proposed DeepFake detection methods operate on optical flow images. These images are generated by computing the optical flow between pairs of cropped face images that have been extracted from two sequential frames of video. Figure 2 is a diagram of this process.

First, face detection is performed to get the rectangular bounds of a face present in both frames. This work used the open-source *face_recognition* library [8] to locate faces. The resulting bounds are then expanded to ensure that the edges of the face are contained within the bounds. If a bound extends past the edge of the frame after expansion, that bound is clipped to place it on the edge of

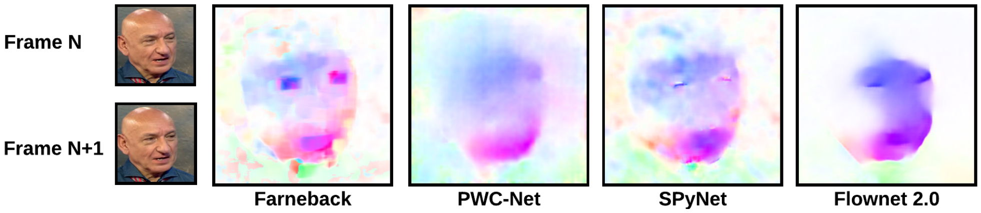

Figure 3: Comparison of optical flows from Farneback [7, 3], PWC-Net [27, 17], SPyNet [20, 18], and Flownet 2.0 [12, 21]. Face images extracted from [23].

the frame. The exact amount of expansion is a tunable parameter and depends on how tightly the face detection method bounds faces. For this work's experiments, the face bounds were enlarged by 50% horizontally and by 80% vertically. After computing the face bounds, each frame is cropped to keep just the located face. The two resulting images are then resized with interpolation to meet the input dimensions of the classifying model. This work's experiments used images of size 256x256 as model inputs.

Next, the two resized face images are used to produce a single optical flow image. The resulting optical flow image contains a 2D vector for each pixel in the first face image. This vector represents the estimated motion for that pixel between the capture times of the first and second images. In this work, each motion vector was then converted to polar form to produce the final input image. Figure 3 demonstrates the optical flow estimation methods used in this work.

4.2 Classification Models

After obtaining optical flows, we have classifiers in discriminating between real and fake corresponding to original videos. We feed single optical flow into convolutional neural network (CNN) to classify if the video is real. Besides, since we have continuous optical flows, we will have a sequence of optical flows as we get a video as input. Consequently, we delve into the usage of temporal models. We introduce the family of recurrent neural network (RNN) models.

The state-of-the-art for RNN models are Long Short-term Memory (LSTM) network, bi-directional LSTM[24], attention[16], and self-attention[31] models. For the attention models, it is motivated by how we pay attention to different regions of an image. Take image caption as an example, when we generate the word 'dog', the model should pay more attention on the 'dog' in the image, and pay less attention on other region. In our task, our input is the optical flow sequence. Therefore, for the self-attention models, it will give an attention to a specific part of optical flow sequence rather than the whole sequence.

4.3 Single Classification

The first part of this investigation focused on detecting DeepFaked faces by observing single optical flow images. For this task, CNN models are trained to classify individual optical flow face examples as “real” or “fake”. To evaluate the effectiveness of this detection method, the same models were also trained to classify RGB face images, and concatenated RGB and optical flow pairs. Figure 4 contains diagrams of these experiments.

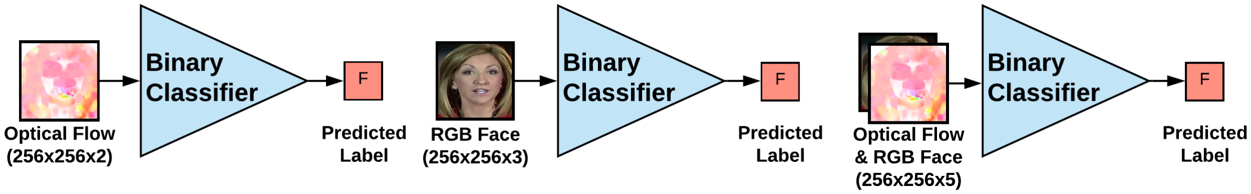

Figure 4: Single-input DeepFake detection was evaluated for optical flow inputs, RGB inputs, and concatenated flow and RGB inputs.

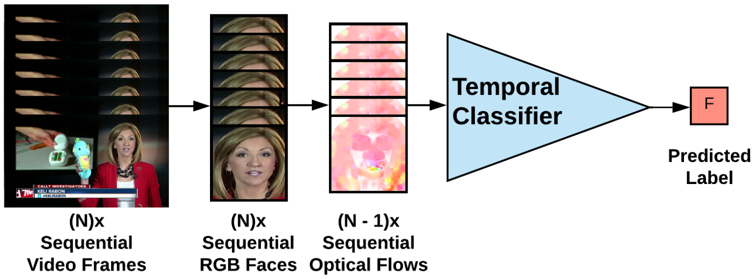

Figure 5: The temporal models classify a sequence of optical flow face images that are generated from a sequence of video frames.

4.4 Temporal Classification

For the second portion of this work, models were trained to classify runs of sequential optical flow images. These sequential runs of flow images are constructed by preprocessing back-to-back pairs of frames from a single video. The models used for this task are given entire sequences of flow images and they are trained to classify the entire sequence as “real” or “fake”. Figure 5 outlines this classification process.

5 Experiments

5.1 Dataset

We evaluate our methods on FaceForensics++ dataset [23]. In this dataset, there are 1000 real videos collected from Youtube. The fake videos are generated by four different algorithms on each real video, namely, Face2Face, DeepFakes, FaceSwap, and NeuralTextures. Also, this dataset provides videos with different quality (raw, light compression, high compression). In consideration of training time, we used real videos from Youtube and fake videos generated by Face2Face with light compression only. The training, validation, and test set in the dataset consist of 720, 140, and 140 videos respectively. Therefore, we train our model on the training set, and use the validation set to select the model. Finally, we evaluated the tuned models on the test set.

5.2 Single Classification

The single classification approach to optical flow based DeepFake detection was evaluated using PyTorch’s [19] implementations of ResNet18 [10], VGG11 [26], and VGG11 with batch normalization (VGG11bn). For each CNN, experiments were run for optical flow face images generated by

OpenCV's Farneback implementation [3, 7], and open-source PyTorch implementations of FlowNet 2.0 [21, 12], SPyNet [18, 20], and PWC-Net [17, 27].

As a baseline for comparison, classifiers were also directly trained on RGB images extracted from the video dataset. A hybrid approach was also experimented with by having ResNet18 and VGG11bn train on stacked RGB and optical flow images.

The classifiers were trained on a subset of the training split of the dataset. The subset was generated by selecting a random sequence of 20 frames from each video in the training split. Table 1 summarizes the results of the single classification experiments by listing the final test accuracies for each classifier/input combination. Note that the RGB+Farneback result for VGG11 without batch normalization is omitted because that model was found to perform poorly when an RGB image was part of the input.

For classifier, VGG performs better than ResNet18 a lot whatever the input is. In addition, batch normalization can improve accuracy for Farneback and SPyNet flow a lot. For flow-generating methods, Flow generated by SPyNet outperforms all the other deep learning based algorithm, and Farneback perform the best.

CNN	**Farneback**	**FlowNet2.0**	**SPyNet**	**PWC-Net**	**RGB**	**RGB+Farne.**
ResNet18	68.90	62.26	72.79	61.21	90.94	92.15
VGG11	72.83	63.30	70.83	**62.31**	49.95	
VGG11bn	**78.87**	**63.80**	**77.05**	62.01	**95.23**	**94.60**

Table 1: Test accuracy for CNN-based models trained to classify single units of input as "real" or "fake".

5.3 Temporal Classification

The temporal classification approach was evaluated using RNN models with pre-trained CNNs as feature extractors. These feature extractors are derived from the CNNs trained on Farneback flow images in the single classification experiments. ResNet18 was modified to be a feature extractor by removing its final, fully-connected layer. VGG11bn was adapted to be a feature extractor by removing its final two layers. The feature extractors' parameters were frozen during RNN training for all but one experiment (indicated in Table 2 as w/ Fine-tuning).

The temporal models were trained on Farneback-generated face flow images for the first 50 frames of each video in the training split.

The experiment result shows features extracted by VGG are better than features extracted by ResNet18. In addition, BiLSTM didn't improve the experiment result but increase the number of parameters. Furthermore, self-attention model perform the best, it is because that self-attention model can know where should pay more attention on, and give more weight on the important part of the sequence.

Model	Feature Extractor	Train Acc.	Valid. Acc.	Test Acc.
LSTM	ResNet18 w/o Fine-tuning	84.65	88.57	87.50
BiLSTM	ResNet18 w/o Fine-tuning	81.94	88.21	87.50
LSTM	VGG11bn w/o Fine-tuning	96.81	97.50	95.36
LSTM	VGG11bn w/ Fine-tuning	93.75	95.00	96.07
Self-Attention	VGG11bn w/o Fine-tuning	96.53	96.79	**96.43**

Table 2: Accuracies for models trained to classify sequences of 50 Farneback flow images as "real" or "fake".

6 Conclusion

Farneback flow produced best classification score under VGG11 with batchnorm for single classification models. SPyNet performed the best among the deep learning-based flow methods, followed by FlowNet2.0 and PWC-Net.

Since Farneback flow performed particularly well on the single classification task, it was employed as the flow input to the temporal model where the best results were achieved with the Self-Attention model. This marks a significant improvement over the single classification models and improves even upon [2] where even our single classification models showed better results when compared.

With these results at hand, we're able to conclude that while optical flow on itself might not be producing patterns that meet the best classification criteria, combining them in conjunction with RGB images produced the best results in DeepFake detection. Flow only input however provided the best results when fed to a temporal network leading us to conclude that hypothesis of optical flow serving a good DeepFake detection tool.

For future work, encoding optical flow vectors in cartesian and tuning temporal model input sequence range may be explored. Apart from the MPI Sintel dataset current deep learning based flow generation models don't have a good facial dataset to train on and hence perform inferior compared to Farneback. They may be revisited when better facial datasets are available.

Author Contributions

- Bhargav: Worked on 3DCNN and temporal networks to train on FaceForensics dataset
- Ching-Yen: Worked on VGG11 and ResNet based single classification models.
- Yuchi: Worked on LSTM based temporal networks for FaceForensics dataset
- Bo-Hong: Worked on RNN based networks for DFDC and Faceforensics dataset before and after modifications and training them.
- Nicholas: Preprocessed DFDC dataset and handled data aquistion for the group and training for models on DFDC dataset.

All co-authors were involved in writing this report, and all co-authors have equally contributed to this project.

References

[1] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen. Mesonet: a compact facial video forgery detection network, 09 2018.

[2] I. Amerini, L. Galteri, R. Caldelli, and A. Del Bimbo. Deepfake video detection through optical flow based cnn. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.

[3] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[4] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, Oct. 2012.

[5] Deepfake detection challenge. `https://www.kaggle.com/c/deepfake-detection-challenge`, December 2019.

[6] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. v. d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2758–2766, 2015.

[7] G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In J. Bigun and T. Gustavsson, editors, *Image Analysis*, pages 363–370, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[8] A. Geitgey. face_recognition. `https://github.com/ageitgey/face_recognition`, 2017.

[9] D. Güera and E. J. Delp. Deepfake video detection using recurrent neural networks. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, 2018.

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[11] B. K. Horn and B. G. Schunck. Determining optical flow. In *Techniques and Applications of Image Understanding*, volume 281, pages 319–331. International Society for Optics and Photonics, 1981.

[12] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1647–1655, 2017.

[13] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

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

[15] B. D. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. 1981.

[16] M.-T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation, 2015.

[17] S. Niklaus. A reimplementation of PWC-Net using PyTorch. `https://github.com/sniklaus/pytorch-pwc`, 2018.

[18] S. Niklaus. A reimplementation of SPyNet using PyTorch. `https://github.com/sniklaus/pytorch-spynet`, 2018.

[19] 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 H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[20] A. Ranjan and M. J. Black. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4161–4170, 2017.

[21] F. Reda, R. Pottorff, J. Barker, and B. Catanzaro. flownet2-pytorch: Pytorch implementation of flownet 2.0: Evolution of optical flow estimation with deep networks. `https://github.com/NVIDIA/flownet2-pytorch`, 2017.

[22] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner. Faceforensics: A large-scale video dataset for forgery detection in human faces. *arXiv preprint arXiv:1803.09179*, 2018.

[23] 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 International Conference on Computer Vision*, pages 1–11, 2019.

[24] M. Schuster and K. Paliwal. Bidirectional recurrent neural networks. *Trans. Sig. Proc.*, 45(11):2673–2681, Nov. 1997.

[25] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[26] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[27] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8934–8943, 2018.

[28] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman. Synthesizing obama: Learning lip sync from audio. *ACM Trans. Graph.*, 36(4), July 2017.

[29] 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*, pages 2387–2395, 2016.

[30] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia. Deepfakes and beyond: A survey of face manipulation and fake detection, 2020.

[31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. 2017.

Appendix

6.1 DeepFake Detection Challenge Dataset

We had planned to utilize the large DeepFake Detection Challenge Dataset provided by Kaggle [5] for training, validation, and testing. We did end up preprocessing much of the dataset into a usable, Farneback flow form, but the data ended up being difficult to train with for a number of reasons. Models seemed to either overfit (see figures 6 and 7) or underfit, and due to the size of the dataset we ran out of time before we could debug its preprocessing parameters.

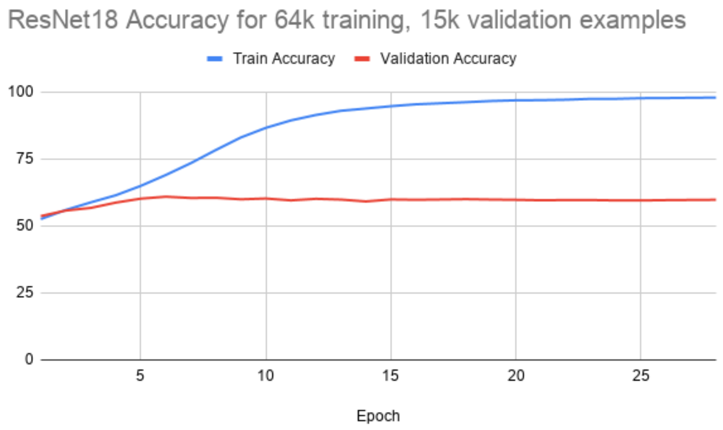

Figure 6: Plot of training and validation accuracies for ResNet18 training on a large subset of the DeepFake Detection Challenge Dataset.

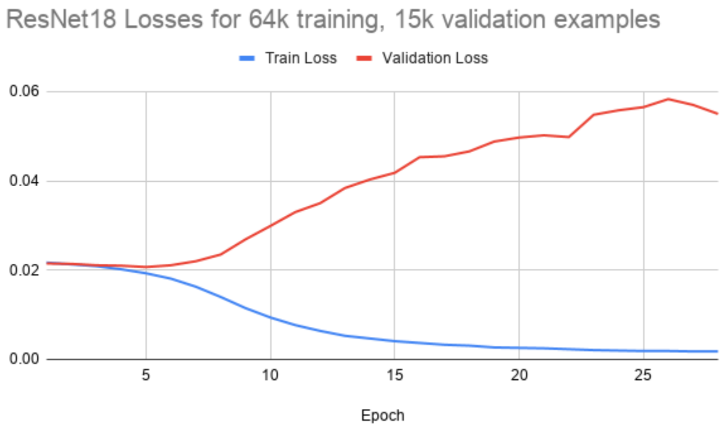

Figure 7: Plot of training and validation losses for ResNet18 training on a large subset of the DeepFake Detection Challenge Dataset.