May 14, 2020

DeepFake Detection with Voronoi Regions

Irmak Türköz

A Computational Geometry Project, Bilkent University

Outline

I. Introduction
   A. Problem Definition
   B. Related Work
II. Model
   A. Dataset
   B. Preprocessing
   C. Voronoi Features
   D. Dataset Construction
   E. Detection Model
III. Results and Evaluation
IV. Conclusion

I. Introduction

- What is a DeepFake?

Figure 1: Synthesizing Obama: Learning Lip Sync From Audio[3].

---

[3] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman, “Synthesizing obama: Learning lip sync from audio,” ACM Trans. Graph., vol. 36, July 2017

DeepFakes are created with taking input a video of a specific individual (**'target'**) and outputting another video with the target's faces replaced with those of another individual (**'source'**).

Figure 2: Video: Synthesizing Obama: Learning Lip Sync[3].

[3] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman, "Synthesizing obama: Learning lip sync from audio," ACM Trans. Graph., vol. 36, July 2017

Figure 3: CELEB - DF Dataset DeepFake

[5] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, “Celeb-df: A new dataset for deepfake forensics,” 09 2019.

- What is Voronoi Features?

In mathematical terms, we are given a finite set of points in the plane and, for each point, the corresponding **Voronoi cell** consists of all the **locations** closer to it **than** to any of the other points [1].

Voronoi diagrams used in:

- Biological structures
- Aviation, (nearest airport in case of diversions).
- Mining,
- Epidemiology [1].

---

[1] P. Lynch, "How Voronoi diagrams help us understand our world", The Irish Times, 2020. [Online]. Available: https://www.irishtimes.com/news/science/how-voronoi-diagrams-help-us-understand-our-world-1.2947681. [Accessed: 13- May- 2020].

I.A Problem Definition

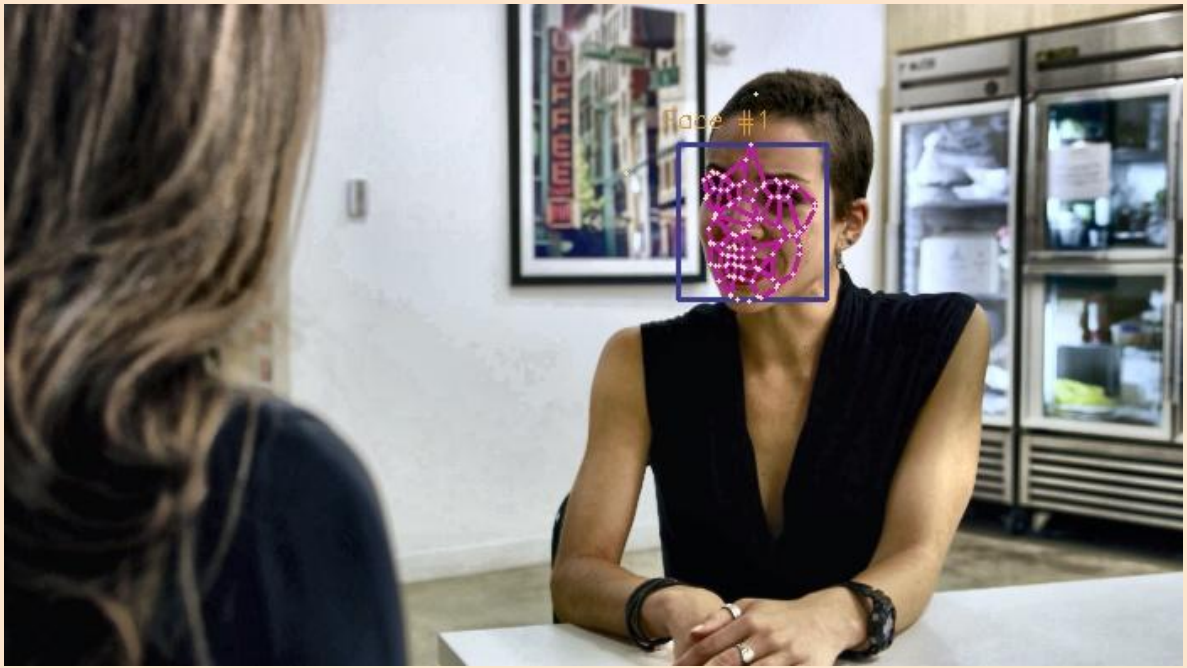


Figure 4: Voronoi Region detection done by current model

Voronoi regions are already used in:

Face recognition, face segmentation, 3D face reconstruction from 2D images

How to detect DeepFakes with Voronoi regions?

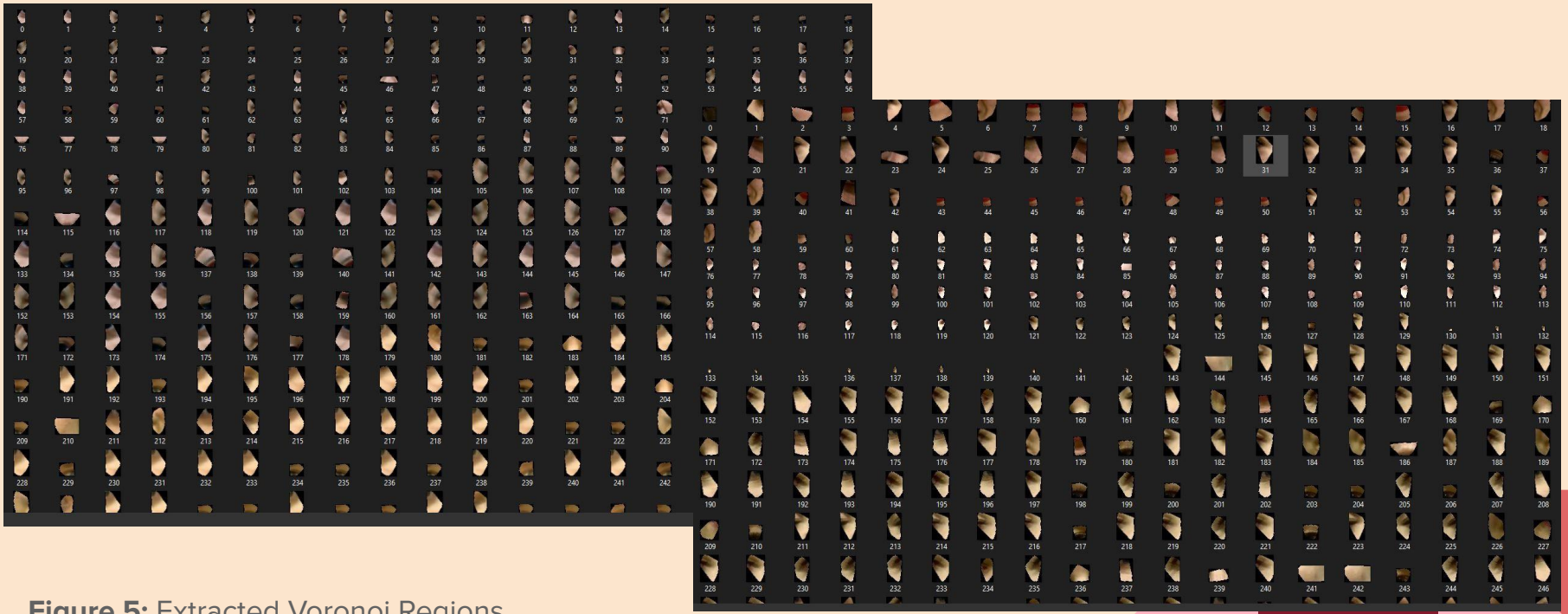


Figure 5: Extracted Voronoi Regions

I.B Related Work

- Face Action Units [8]
- Eye blinking, [9]
- Biological signals [10],
- facial features optical distortions [11][12];
- there are also neural network methods which proposes CNN based models [13],
  - CNN and RNN based models [14],
  - two-stream neural networks [15],
  - autoencoders [16],
  - Long-Short Memory model [17].

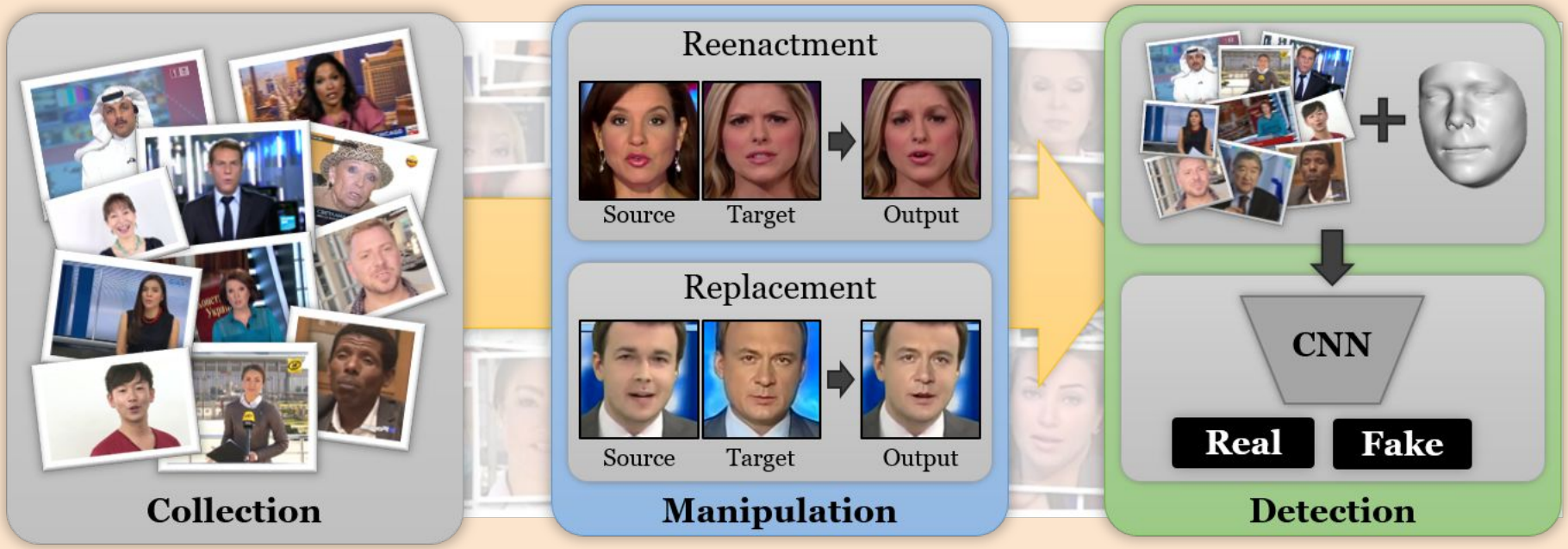


Figure 6: Summary of related work [2]

[2] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "Faceforensics++: Learning to detect manipulated facial images," 01 2019

II. Model

II.A Dataset

Face Forensics++ is used.[2] Including four methods,

- Deepfakes,
- Face2Face,
- FaceSwap, and
- NeuralTextures

Face Forensic (first version) 977 images from YouTube (without consent). New version (Face Forensics++) : Google also contributed to the dataset. 28 paid actors in 16 different scenes and over 3000 manipulated videos

[2] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, “Faceforensics++: Learning to detect manipulated facial images,” 01 2019

-Only a *small* subset of the *dataset* is used.

-Main concern for using Voronoi Regions → train the system with **less data**, gain **robustness** with looking to the Voronoi Regions in different frames.

-Videos are **long**, 100~250 **Voronoi Regions** Images can be extracted from only 1 video. (Voronoi regions with small area is filtered as they do not give much clue)

-Currently using:

- Original Sequences: 30 Actor and 30 Youtube Original videos
- Manipulated Sequences: 60 DeepFakeDetection videos

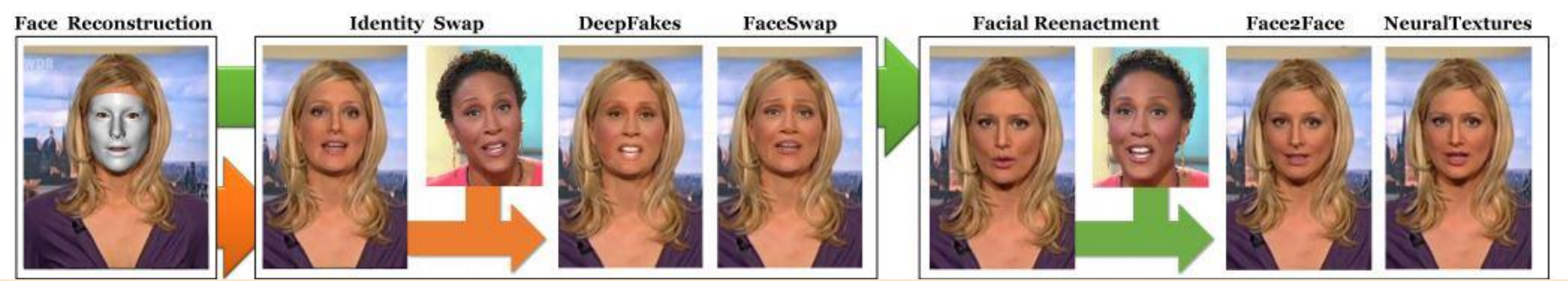


Figure 7: FaceForensics++ example of DeepFake Methods [2]

II.B Preprocessing

Two main preprocessing steps applied:

- To detect face regions : Illumination Normalization.
- To extract voronoi regions from face regions: Gamma Normalization.

Figure 8: Lightning ill affected image (Left) vs preprocessed image (Right)

- To extract voronoi regions from face regions: Gamma Normalization.

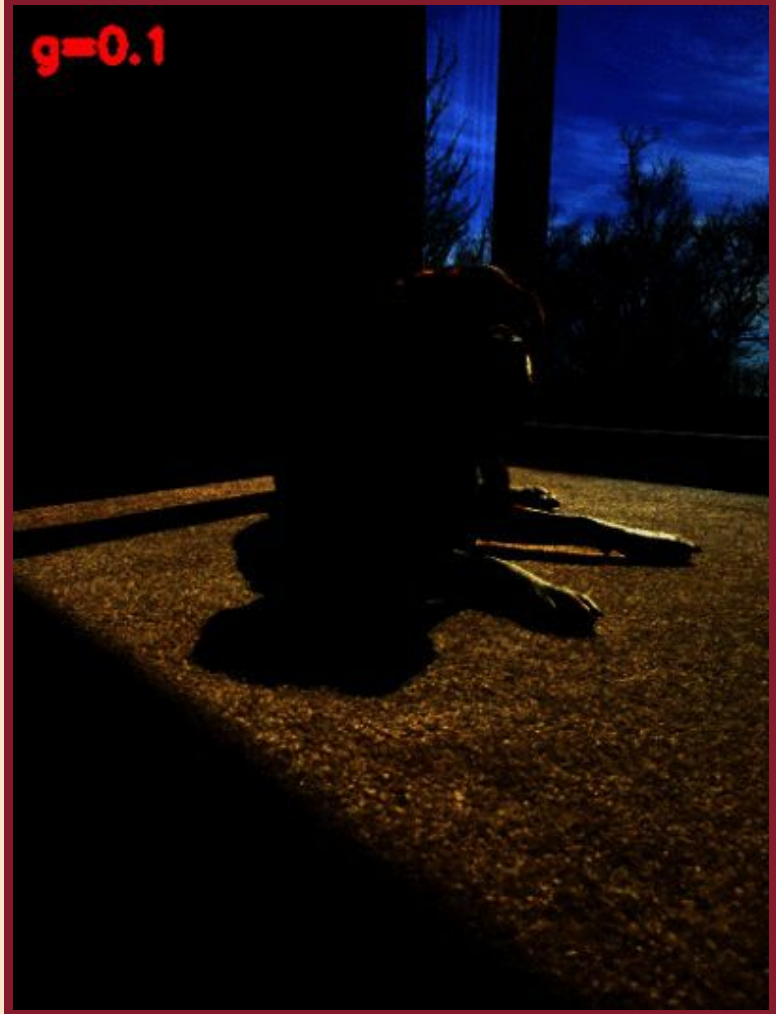


Figure 9: OpenCV Gamma Correction [4]

[4] A. Rosebrock, "OpenCV Gamma Correction - PyImageSearch", PyImageSearch, 2020. [Online]. Available: https://www.pyimagesearch.com/2015/10/05/opencv-gamma-correction/. [Accessed: 13- May- 2020].

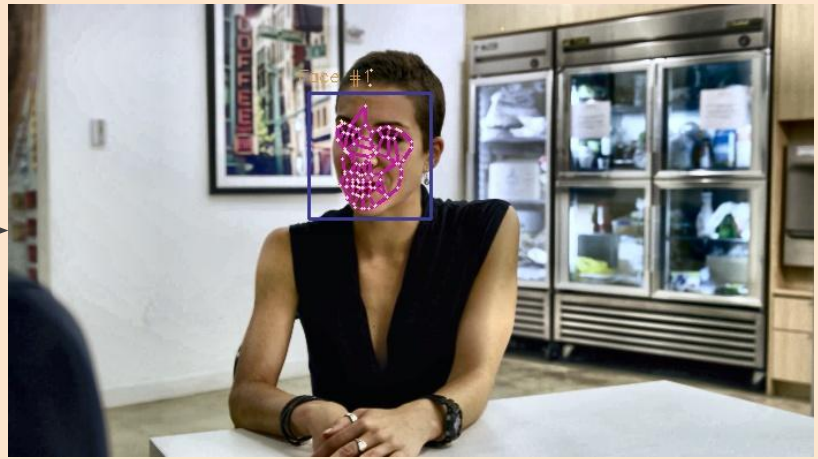
II.C Voronoi Features

1. Apply Illumination Normalization
2. Use a pre-trained model to detect the face **region**. (dlib [6])
3. Apply Gamma Normalization
4. Use a pre-trained model to detect the facial **landmarks**. (dlib [6])
5. Generate Voronoi Mapping with facial landmarks.

[6]D. King, "Classes — dlib documentation", *Dlib.net*, 2013. [Online]. Available: http://dlib.net/python/index.html. [Accessed: 30- Mar- 2020]

Figure 9: Voronoi regions without and with post processing are illustrated.

- scipy.spacial's Voronoi to generate Voronoi [7]
- Voronoi regions might have points outside the bounding box.
- Post Processing required.



[7]"scipy.spatial.Voronoi — SciPy v0.18.1 Reference Guide", *Docs.scipy.org*, 2016. [Online]. Available: https://docs.scipy.org/doc/scipy-0.18.1/reference/generated/scipy.spatial.Voronoi.html. [Accessed: 30- Mar- 2020].

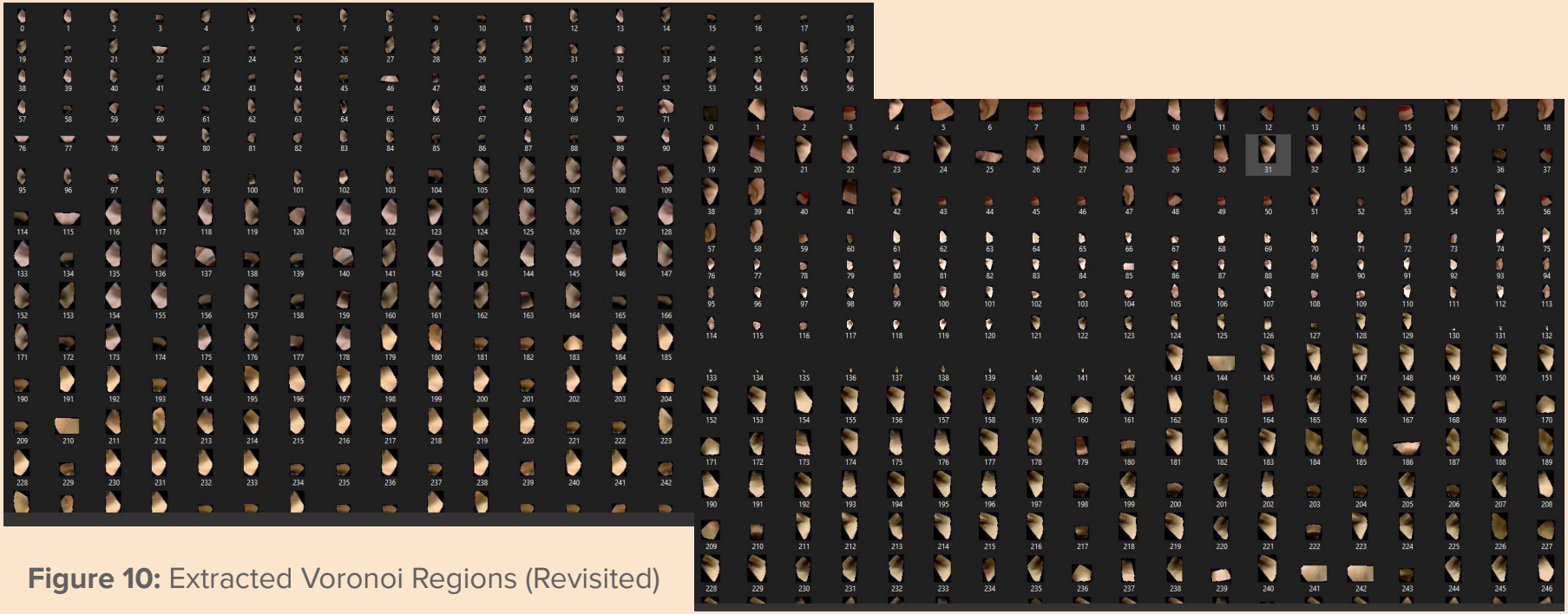
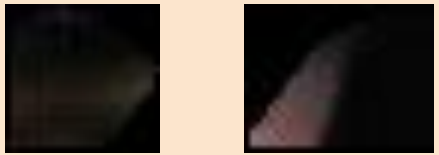


Figure 10: Extracted Voronoi Regions (Revisited)



➜ **Eliminate these**

II.D Dataset Creation

Extracting features every time → Slow

Saved **voronoi regions** as features to create new dataset.

- Train: ~1600 Manipulated, ~2400 Original
- Validation: ~800 Manipulated, ~900 Original
- Test: ~300 Manipulated, ~400 Original

II.E Detection Model

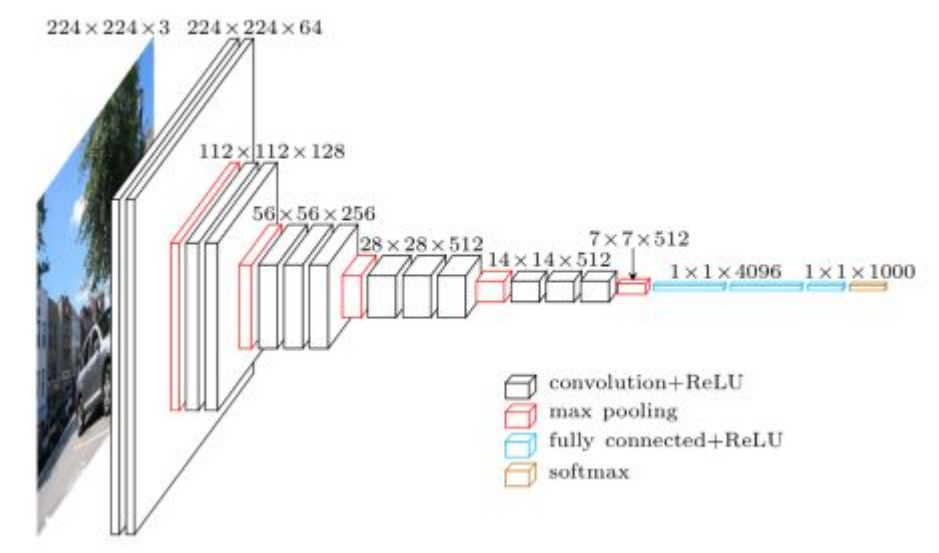


Figure 11: VGG16 Model [19]

Drawbacks:

- **Slow** to train → However, we have features, small images (100 x 100).
- The network architecture **weights** themselves are quite large (concerning disk/bandwidth). Still not a problem, we already have features.

[19]R. Thakur, "Step by step VGG16 implementation in Keras for beginners", towardsdatascience, 2020. [Online]. Available: https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c. [Accessed: 13- May- 2020].

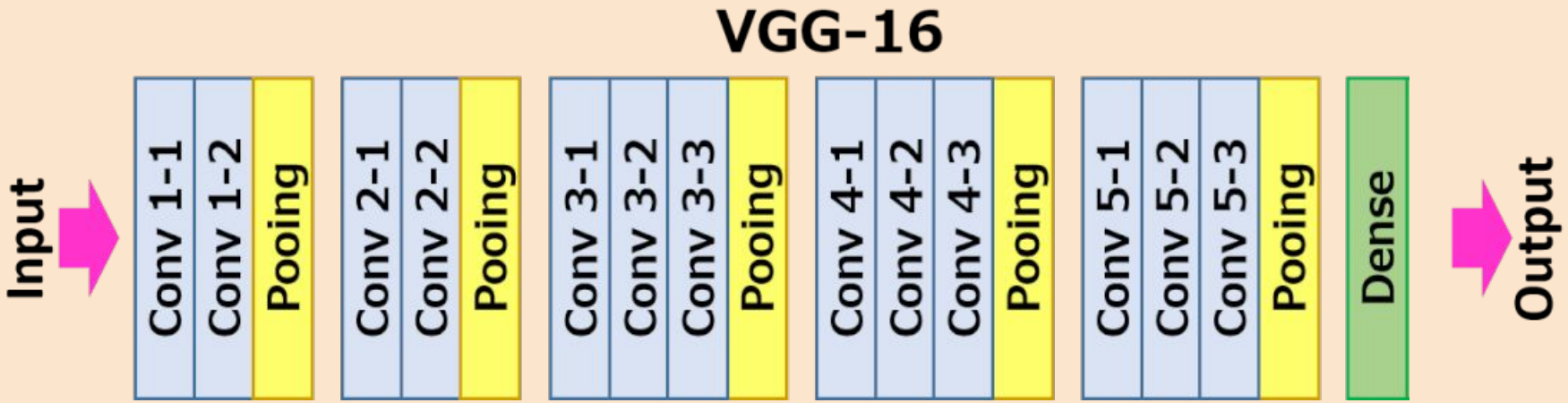


Figure 11: VGG16 Model Architecture

Advantages:

- Working good with images, especially binary classification.
- Convolutions followed by pooling layers help generalize the data

[19]R. Thakur, "Step by step VGG16 implementation in Keras for beginners", towardsdatascience, 2020. [Online]. Available: https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c. [Accessed: 13- May- 2020].

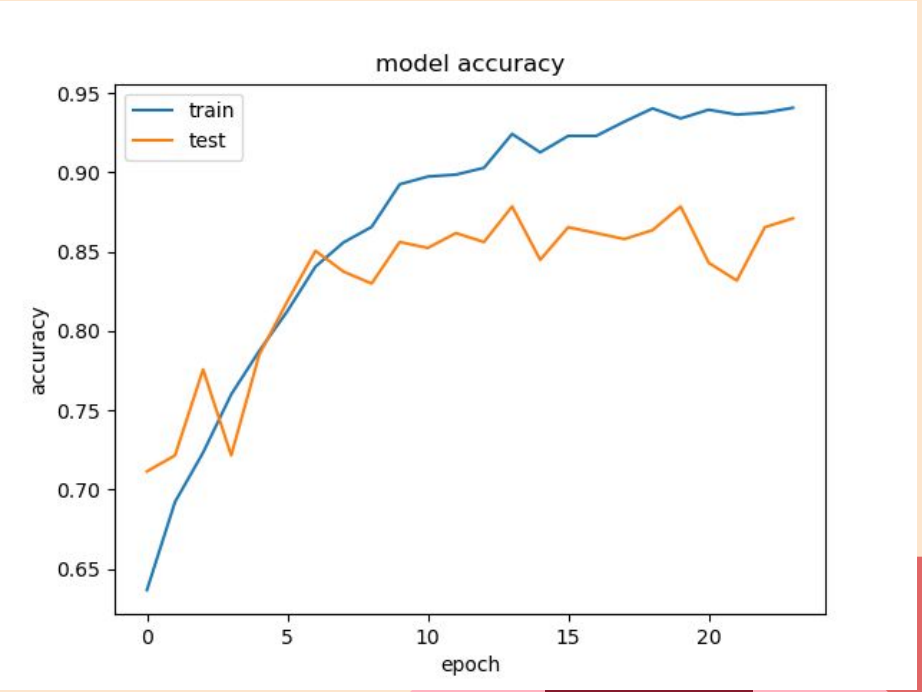
Hyper Parameters

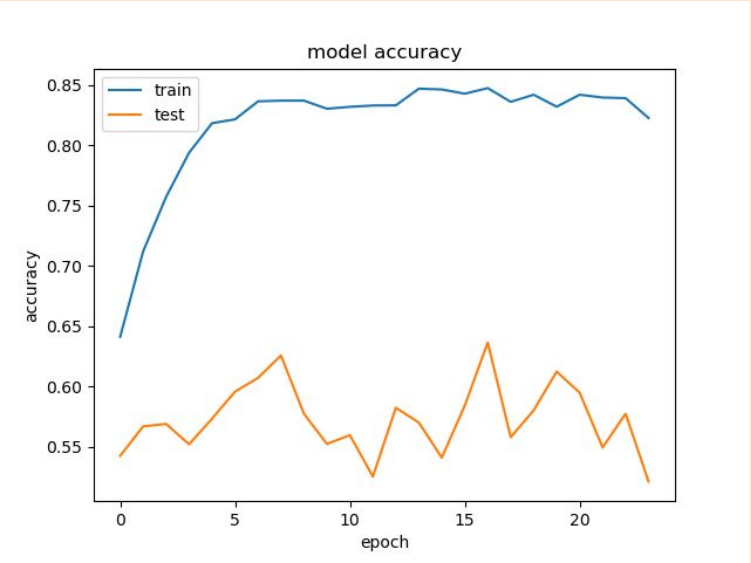
- Using RELU activation in the CONV - 2D layers.
- Sigmoid activation in Dense layers. (Also tested with Softmax)
- RMSProp as optimizer
- Binary Cross Entropy as loss function.
- Input shape = (80 x 100 x 3)
- 24 Epoch
- Batch size is 16

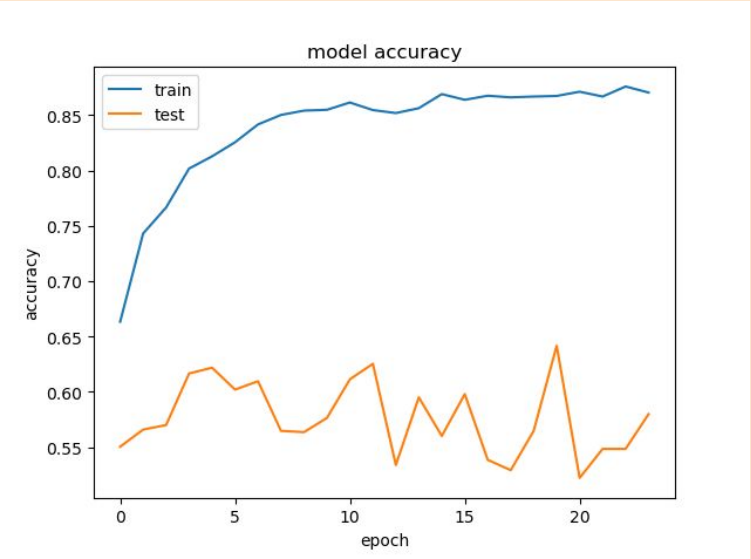
III. Results and Evaluation

First trial:

- Worked with only 6 original images and 6 manipulated images.
  - (Because feature extraction takes time)
- Divided train, val and test **after** feature generation.
- **loss: 0.1780 - accuracy: 0.9377 - val_loss: 0.0980 - val_accuracy: 0.8654**
- Learned too fast.
- Overfit after 8th epoch.







Second Trial:

- Trained with 30 manipulated, 30 original.
- Not working.
- Problem with Model? Problem with Features?
  - To be answered in the final report.

Evaluation

- Looking only a region of Face might not be *enough* to detect DeepFakes.
- Looking every voronoi region on a face to train?
  - Requires more data,
  - More time for feature extraction.
- Looking only the length of the edges of Voronoi?
  - Not meaningful.
- The quality of the video?
  - Quality of the videos used height= 1200,width=1600.
  - Quality of Voronoi Regions height= 80 ,width=100
- Future work:
  - Use only one person’s manipulated and original data.

Conclusion