BOĞAZİÇİ UNIVERSITY

Cmpe 492

DeepFake Detection

WEEK 5

İnci Meliha BAYTAŞ

Baran Deniz KORKMAZ

Doğukan KALKAN

FALL 2020

1 Weekly Summary

1.1 Work Done

1.1.1 Model Extensions: LSTM

This week we have embedded an LSTM model into the end of pretrained CNN model. The CNN model is used to extract meaningful features from each video frame which will then be fed into the LSTM model. Below, you can find the figure illustrating the current version of model:

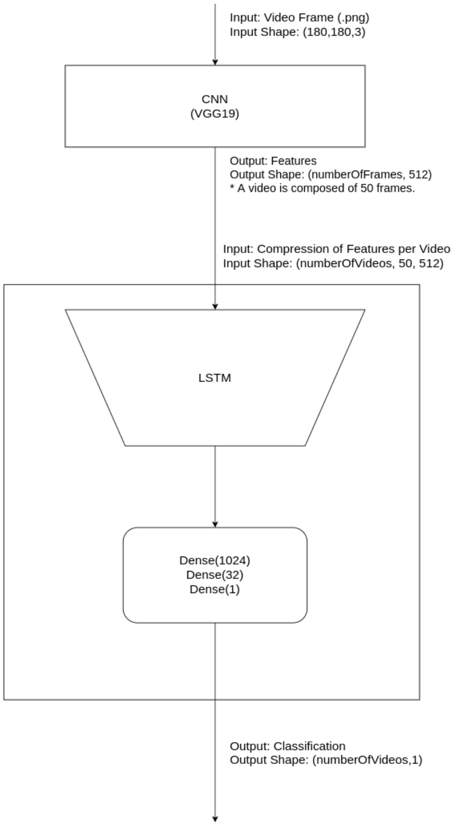
Input: Video Frame (.png)
Input Shape: (180,180,3)
CNN
(VGG19)
Output: Features
Output Shape: (numberOfFrames, 512)
* A video is composed of 50 frames.
Input: Compression of Features per Video
Input Shape: (numberOfVideos, 50, 512)
LSTM
Dense(1024)
Dense(32)
Dense(1)
Output: Classification
Output Shape: (numberOfVideos,1)

Figure 1: Current Version: CNN + LSTM

Please note that the features extracted by CNN are not directly fed into the LSTM. The consecutive 50 frames are extracted from the same manipulated or original video sequence. Therefore they are transformed into a single compound which will be given into the LSTM. Hence, the input fed into LSTM is of shape **(numberOfVideos, 50, 512)**. After the output of LSTM has been obtained, the features extracted per video are used to classify the given video by dense layers.

1.1.2 Constructing Input Pipeline

As mentioned in the previous reports, we have access to FaceForensics++ dataset which contains both original videos and manipulated videos created by using different techniques, namely Deepfakes, Face2Face, FaceShifter, FaceSwap, NeuralTextures and DeepFakeDetection. Below we can see a diagram of our method of constructing input pipeline and also the detailed explanation of the diagram.

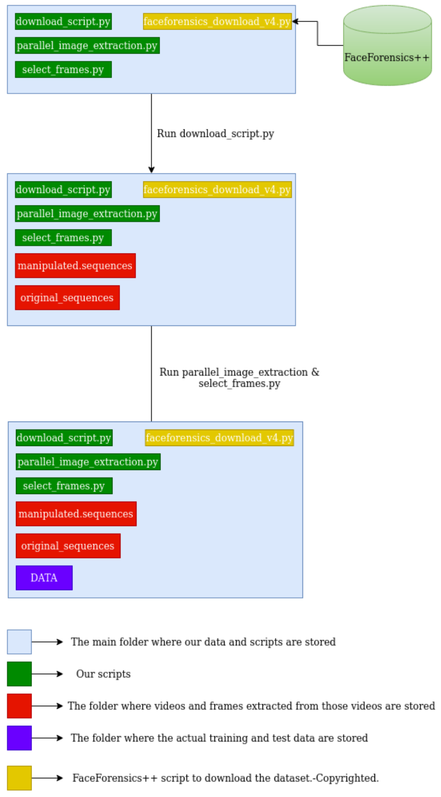
download_script.py
faceforensics_download_v4.py
parallel_image_extraction.py
select_frames.py
FaceForensics++
Run download_script.py
download_script.py
faceforensics_download_v4.py
parallel_image_extraction.py
select_frames.py
manipulated.sequences
original_sequences
Run parallel_image_extraction & select_frames.py
download_script.py
faceforensics_download_v4.py
parallel_image_extraction.py
select_frames.py
manipulated.sequences
original_sequences
DATA
The main folder where our data and scripts are stored
Our scripts
The folder where videos and frames extracted from those videos are stored
The folder where the actual training and test data are stored
FaceForensics++ script to download the dataset.-Copyrighted.

Figure 2: Construction of Input Pipeline

First of all, we must have a folder which contains these four Python scripts: **faceforensics_download_v4.py**, **download_script.py**, **parallel_image_extraction.py** and **select_frames.py**.

The script **faceforensics_download_v4.py** is the script that is given to us by FaceForensics++, so it should not be accessible by others. To run this script, we have another script called **download_script.py** which actually runs the first script with certain configurations. These configurations specify the names of the datasets that we would like to download, such as DeepFakeDetection or Face2Face. The **download_script.py** downloads the videos from different datasets as well as the original videos and puts them in two different folders, namely **manipulated_sequences** and **original_sequences**. Under **original_sequences**, we can find the original videos, and under **manipulated_sequences**, we can find different folders each of which contains the videos from different datasets.

After downloading the dataset, to extract the frames from videos, we need to run the script **parallel_image_extraction.py**. For each video, the script creates a folder with the name of the video under the folder called **images** and puts all the frames in that folder.

At this point, we have all the frames extracted from the videos. Now, we need to create our own dataset from these frames. To do that, we need to run the script **select_frames.py**. The script creates a folder named **DATA** and two folders under this folder, namely **Fake** and **Real**. The creation of these two folders comes in handy, because we use a **import_data** in **data_handler.py** which uses the function **image_dataset_from_directory** in TensorFlow.This function imports our data with labels, for example, giving the label 0 to data from the folder **Fake** and the label 1 to data from the folder **Real**. The script fills the the folder **Fake** with all the frames extracted from fake videos. Likewise, the folder **Real** is filled with the frames extracted from the original videos. We select 50 frames from all the frames extracted from a video. In other words, we have 50 frames from each video in our dataset.

Thus, we have finished the construction of our dataset.

1.2 Learning Outcomes

1.2.1 LSTM

1. We learned the underlying mechanism of LSTM, an artificial recurrent neural network architecture including internal networks called gates.

2. We learned the use of Tensorflow framework for implementing an LSTM model.

1.2.2 Input Pipeline

1. We learned how to download the datasets using the script provided by the FaceForensics++.

2. We learned how to extract frames from the videos and create our own dataset from those frames.

2 Challenges

1. Learning the use of Tensorflow framework for LSTM was challenging as it requires the analysis of plenty of practical applications.

2. Constructing a pipeline that enables us to create our own dataset using different scripts was a challenge by itself.

3 What's Next?: Upcoming Week

1. The proper execution of input pipeline, model training, and prediction of test data will be tested and delivered into the supervisor.

2. The collaborators will study on the concept of **One-Class Classification** and the prospected uses on the deepfake detection.

3. As explained above in **1.1.2**, we create a folder named **DATA** where our own dataset is stored. Currently, under this folder, we have two folders called **Fake** and **Real**. However, we need to have 2 folders, one of them is for training and the other is for testing. So we need to create two folders named **Training** and **Test**. Also, under each of these folders, there should be two folders named **Fake** and **Real**. So we will modify the script **select_frames.py** to work as explained.