IsSwap: Deep Fake Detection

Aakriti Aggarwal, Siddhant Wadhwa, Pallav Gupta, Nishit Anand, Rashmi Kushwah

Department of Computer Science & Engineering and Information Technology
Jaypee Institute of Information Technology, Noida-201304 Uttar Pradesh, India
Email: *aakritiaggarwal2k, siddhantwadhwa2010, pallav508, nishitanand01, rashmi.31130@gmail.com*

***Abstract*—In this era of technology, the intake of information through digital media has grown exponentially and it has provided people with personal motives to spread falsified information among the masses to create biased opinions and a sense of unrest. The falsified information is provided to the peoples especially during elections to create political unrest among the masses or simply to spread a rumor. Since most of the information is consumed by people is in the form of videos, it has become a great target spot for people with malicious intent. The explosive growth in deep fake video and its undetected use is a major threat to democracy, justice, and public trust. Due to this, there is an increased demand for fake video analysis, detection and intervention. The paper is aimed at overcoming the given challenge by providing a fast and reliable method to determine the authenticity of a given video.**

***Keywords*—Deep Learning, GAN, Neural Network, Deepfake, Video Manipulation.**

I. Introduction

In the current world scenario the technology has taken over every aspect of human life and has become an essential part of our daily lives. We use technology to complete routine tasks and this has actually made life simpler in more ways than we can imagine. The sheer amount of information we consume from digital media is enormous. As our intake of information through digital media has increased, new problems have arisen. Over time with technological advancements people have found ways to use technology to falsify information and spread it to achieve personal vendetta. One of such practices is called Deep-Fake [1] [4]. Deep-Fake is a video of a person in which their face or body has been digitally altered so that they appear to be someone else, typically used maliciously or to spread false information [2]. Deep Fakes are created by combining and superimposing existing images or videos using a deep learning technique known as Generative Adversarial Networks (GANs) [9] [12]. Figure 1 shows the example of deep fake detection.

Users upload more than 500 hours of fresh video content per minute which roughly translate to 7.2 lakhs of new content uploaded every day and this is just on one platform, namely YouTube [3]. Now the question arises, how do we know that the same technology we love and trust is not being used against us. The explosive growth in deep fake video and its undetected use is a major threat to democracy, justice, and public trust [5] [6]. Due to this, there is an increased demand for fake video analysis, detection and intervention which is the main focus of this paper along with spreading awareness about these deep-fakes and letting people know what serious implications these videos have in their daily life.

IsSwap attempts to protect people from believing in false information which is being spread through these deep-fakes. It helps to identify such videos using different tools and technology. IsSwap is a web application which uses deep learning techniques to achieve the goal of fake video detection so that it becomes possible to determine the authenticity of a given video.

Fig. 1. Examples of Deep Fake

II. BACKGROUND STUDY

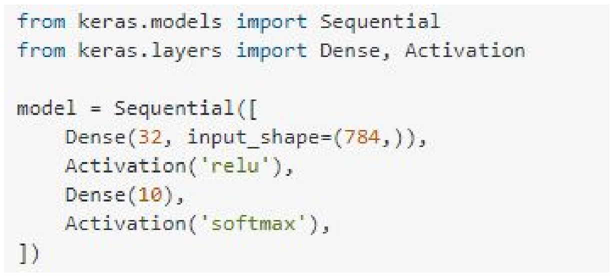
In the proposed work of Deep Fake Detection, following libraries are required:

- Numpy: Array-processing package that provides a high performance multidimensional array object, and tools for working with the arrays.
- Pandas: Fast, powerful, flexible and easy to use open source data analysis and manipulation tool.
- Matplotlib: Plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI, toolkits like tkinter,wxPython, Qt, or GTK+ [7].
- Seaborn: Provides a high level interface to visualize data.
- Keras: There are two ways to build Keras models: sequential and functional [8]. The sequential API allows to create models layer-by-layer for most problems. However, it has the limitation that it does not allow to create models that share layers or have multiple inputs or outputs.

  A Sequential model [10] is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor. A Sequential model is not appropriate when:

  - The model has multiple inputs or multiple outputs.
  - There is a need for layer sharing.
  - When there is a requirement of non-linear topology (e.g. a residual connection, a multi-branch model)

  From the definition of Keras documentation, the sequential model is a linear stack of layers. We can create a sequential model by passing a list of layer instances to the constructor as shown in Figure 2.



```
from keras.models import Sequential
from keras.layers import Dense, Activation

model = Sequential([
    Dense(32, input_shape=(784,)),
    Activation('relu'),
    Dense(10),
    Activation('softmax'),
])
```

Fig. 2. Sequential model

- Patch: A patch is a 2D artist with a face color and an edge color.
- Pytorch: It is a python library based on torch library. it is mainly used in computer vision applications and natural language processing.

Following are the technologies which are required in the proposd work:

- Neural Networks (NN): Neurons in the NN are inspired from biological neurons. The NN would be able to do various tasks like classifying images and predictions. A Perceptron [11] is a single-layer neural network used for supervised learning of binary classifiers. Binary classifiers decide whether an input, usually represented by a series of vectors, belongs to a specific class.
- Flask (Flux Advanced Security Kernel): It is a web framework which is suitable for the development of web applications. We have chosen flask over Django because it is lighter and much more explicit.
- Tensorflow: Tensorflow [13] is a python library which is used for training and inference of deep neural networks.
- Haar Cascade Classifier: Haar cascade [14] is a program which is useful for identification of various objects or faces or hand gestures in an image or video. The algorithm can be explained in four stages:
  - Calculating Haar features
  - Creating integral images
  - Using Adaboost
  - Implementing Cascading Classifiers
- Long Short Term Memory [15] [16]: usually called LSTMs are a special kind of recurrent neural networks, capable of learning long-term dependencies. LSTMs are designed to avoid the long term dependency problem.
- ResNeXt [17]: It is a simple, highly modularized network architecture for image classification. It is constructed by repeating a building block that collects a set of transformations with the same topology. It is a simple design which results in a homogeneous, multi-branch architecture that has only a few hyper-parameters to set.

III. PROPOSED WORK

A. REQUIREMENT ANALYSIS

There are two phases of the proposed work. One from the user side i.e. user will upload a video to our model and secondly from the developer side i.e. developer will train the model based on the datasets. Here we are using a dataset which consists of video and the corresponding labels in a .csv file. This dataset is fed into the model_preprocess notebook. This notebook preprocess the data using a haar cascade frontal face classifier. The preprocessed data will split into training and testing data for measuring the performance in 80:20 ratio. From here we will train the model for 80% of the training data. We label the train videos with the corresponding label and then feed the data into the model.

For Deepfake detection, we are using Deep Learning, NNs and Convolutional neural networks. These require a good amount of computational power and GPU power to do the required ML tasks quickly and efficiently. Figure 3 shows the system description of the proposed work.
We need training and testing data for training the model and validation as follows:

- For training the model training data is needed. For this we need labelled data that tells which data is real and which is fake, so that the model can learn from it and then be able to identify deep fake videos on its own. The data we are using [18] is composed of mp4 files, split into compressed sets of $\sim 10GB$ apiece. A metadata json accompanies each set of mp4 files, and contains filename, label (REAL/FAKE), original and split columns.
- For validation purposes, we also need testing data, so that we can check the performance of our model and see how well it can identify and differentiate deep fake videos from real ones. test_videos.zip is a zip file containing a small set of videos to be used as a public validation set.

Following tools and technologies are used for the proposed work:

- Programming Language: Python, JavaScript, HTML/CSS
- Programming Framework: PyTorch, Flask
- Neural Networks: Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Long Short Term Memory(LSTM)
- Libraries: Numpy, Pandas, Matplotlib, Seaborn, Keras, Tensorflow, Scikit Learn, OpenCV
- IDE: Google Colab , Jupyter Notebook
- Version Control : Git

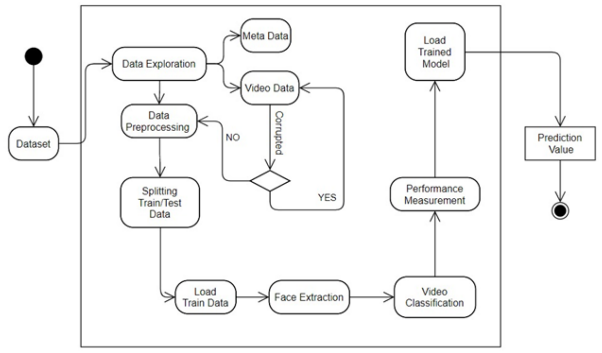


Fig. 3. System Description

B. *Model Architecture*

Following are the layers available in proposed model for training a deep neural network.

- ResNeXt-50, 32*4 dimension pre trained model for feature extraction. It consists of 50 layers with 32 nodes in each layer which is capable of learning a large number of parameters.
- The output of ResNeXt is a pooling layer which gives us a feature vector which is then fed into a sequential layer.
- Sequential layer fed the input into the LSTM layer. We have used one LSTM layer with 2048 hidden dimensions and with the chance of Dropout of 0.4.
- The output of LSTM is further processed by linear and adaptive layers.
- Finally the softmax layer is implemented which gives the output whether the video is real or fake.
- Train_epoch trains the model based on the given number of epochs. Epochs is a hyperparameter that defines the number of times that the learning algorithm will work through the entire training dataset. Each epoch consists of a number of batches. In our function we are defining a range of parameters for each epoch. It will compute the best possible epoch value and batch size.

C. *DETAILED INFORMATION OF DATASET*

The data is composed of .mp4 files, split into compressed sets of $\sim 10GB$ apiece. A metadata .json accompanies each set of .mp4 files and contains filename, label (REAL/FAKE), original and split columns, listed below under columns.

1) Files
   a) train_sample_videos.zip - a ZIP file containing a sample set of training videos and a metadata .json with labels.
   b) sample_submission.csv - a sample submission file in the correct format.
   c) test _videos.zip - a zip file containing a small set of videos to be used as a public validation set.
2) Columns
   a) filename - the filename of the video.
   b) label - whether the video is REAL or FAKE.
   c) original - in the case that a train set video is FAKE, the original video is listed here.
   d) split - this is always equal to "train".

D. *Data Preprocessing*

1) Split the video into the frame.
2) Validate the video to check if the video is corrupted or not. If it is then delete the video.
3) Haar Cascade frontal face classifier is used to detect the faces in the frame.
4) Remove the face from the frame. (cropping)
5) Resize the images so as to have fixed pixel size for better output.

E. *PROPOSED ALGORITHM*

Following are the steps of proposed algorithm:

1) Collect the dataset consisting of real and fake videos.
2) Collect the mp4 file and target the face.
3) Preprocess the input file.
   a) Extract faces histogram of ordered gradients.
   b) Reduce every image to grayscale.
4) Detect features.
   a) Resolution differences.
   b) Angle of face
   c) Facial feature scale with the rest of the head, ears, jawline and hairline.
   d) Inconsistent skin tone.
5) Train the deep fake detection model.
6) Evaluate the model
7) Load preprocessed input to the trained model.
8) Confusion metric is used for model evaluation to get the accuracy of the model.

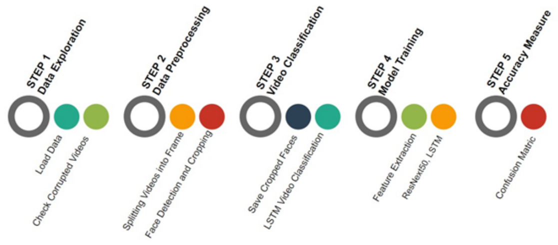


Fig. 4. Algorithm Designed

Figure 4 shows the various phases of proposed algorithm. The frontend part consists of the UI design and other various web pages which will provide a nice experience to the user. Standard codes are written using HTML, CSS and JavaScript frameworks. Bootstrap, a sleek and intuitive CSS framework that boasts of multiple CSS, JS and other front files that kick start the web development task. There are different components in Bootstrap that are used in creation of dropdown boxes, alert boxes, and such other features.

IV. Implementation Results

Following system specifications are required for the proposed work:

- A laptop or computer running Windows 10 or mac OS Sierra or above or Linux, or Ubuntu 16.04 LTS
- A dual core processor 2 GHz or more
- 8GB of RAM or more
- NVidia GPU 1000 series or above
- Jupyter Notebook
- Google Colab

A. Data Preprocessing Result:

- Frame Extraction from video

Videos are a collection of images. Here we need to extract frames from the input video. We have used the OpenCV library to obtain image frames from the input video. We take input as a video file and the output is the set of images which composed up the video given as input. Figure 5 shows the frame extraction from video.

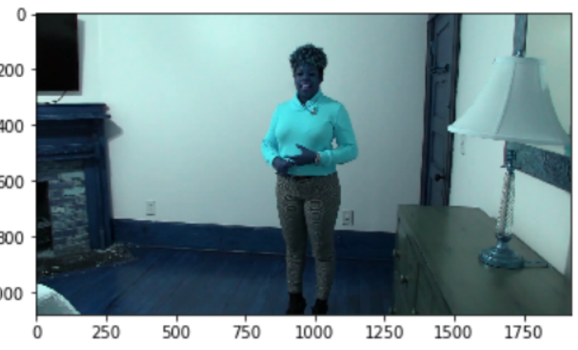


Fig. 5. Frame Extraction from video

- ***Face Extraction from frame***

Now to extract faces from frames HaarCascade Classifier is used. It recognizes the facial features and draws a bounding box around any face in an image. This tells us the location of a face in an image. It also tells us the coordinates of the bounding box drawn around the face. Figure 6 shows the face extraction from frame.

- ***Cropped faces from the frame and removed an extra part of the image***

The bounding box coordinates we got from the last step are used here. Using the coordinates we will crop each image so that it only contains the face in each frame obtained from the video. Thus, we cropped faces from the frame and extracted faces of every frame (one video with at most 20 frames). These images are then resized. Frame size of each cropped image is (100,100,3). If there is no face captured, then we do not run the command count+=1 and we go on to capture the next face and crop it. We will repeat this until the 20th frame for each second of the video. Figure 7 shows the cropped faces from the frame.

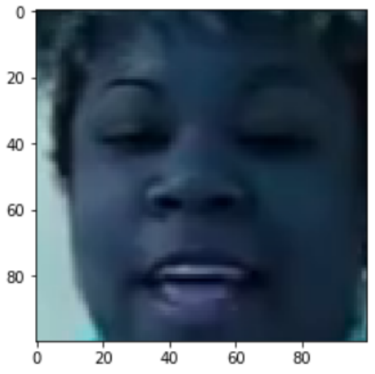


Fig. 6. Face Extraction from frame

Fig. 7. Cropped faces from the frame

B. ***Data Modelling Result:***

We trained the model on the input images and fine tuned different parameters to achieve the highest accuracy on the validation dataset. An important observation after training the final model is that as we are increasing the number of frames, i.e. as the sequence length is increasing, the accuracy also increases. We have checked the sequence length upto 100 frames. The highest accuracy was achieved when 100 frame length is used, and the highest accuracy achieved is 74.2%.

V. Conclusion and Future Work

IsSwap is proposed to detect the fake face and general images generated by the state-of-the-art GANs. The proposed CFFN can be used to learn the middle- and high-level and discriminative fake features by aggregating the cross-layer feature representations. The proposed feature enables the fake feature learning, which allows the trained fake image detector to have the ability to detect the fake image generated by a new GAN, even if it was not included in the training phase. The model developed uses ResNeXt, followed by a sequential layer. This is then passed to an

LSTM layer and softmax layer which predicts whether the video is fake or real. The experimental results demonstrated that the proposed method outperformed other state-of-the-art methods in terms of precision and recall rate and performed well in predicting whether the video is fake or real. As a future work the proposed method can be extended for fake video detection, incorporating the object detection and model accuracy considering several factors i.e. Sequence Length, Epochs and boxes.

REFERENCES

[1] M. Westerlund. The emergence of deepfake technology: A review. Technology Innovation Management Review, vol. 9, no. 11, 2019.

[2] B. Zi, M. Chang, J. Chen, X. Ma, and Y. G. Jiang, ”WildDeepfake: A Challenging Real-World Dataset for Deepfake Detection”, International Conference on Multimedia, pp. 2382-2390, 2020

[3] https://www.tubefilter.com/2019/05/07/number-hours-video-uploaded-to-youtube-per-minute/: :text=The%20platform%27s%20users%20upload%20more,of%20new%20content%20per%20day

[4] N. Ruchansky, S. Seo, and Y. Liu, ”A hybrid deep model for fake news detection”, International Conference on Information and Knowledge Management, pp. 797-806, 2017.

[5] https://www.forbes.com/sites/robtoews/2020/05/25/deepfakes-are-going-to-wreak-havoc-on-society-we-are-not-prepared/?sh=2ab780597494

[6] L.Guarnera, O. Giudice, and Battiato, S. Battiato, ”Deepfake detection by analyzing convolutional traces”, IEEE International Conference on Computer Vision and Pattern Recognition, pp. 666-667, 2020

[7] https://matplotlib.org/stable/index.html

[8] https://stackoverflow.com/questions/57751417/what-is-meant-by-sequential-model-in-keras

[9] B.Dolhansky, R. Howes, B. Pflaum, N. Baram, and C. C. Ferrer, ”The deepfake detection challenge preview dataset”, arXiv preprint arXiv:1910.08854, 2019.

[10] https://keras.io/guides/sequential _model/

[11] https://deepai.org/machine-learning-glossary-and-terms/perceptron: :text=A%20Perceptron%20is%20an%20algorithm, a%20single%2Dlayer%20neural%20network

[12] S. Lyu, ”Deepfake detection: Current challenges and next steps” International Conference on Multimedia and Expo Workshops, pp. 1-6, 2020

[13] https://www.tensorflow.org/tutorials

[14] https://www.cs.cmu.edu/ efros/courses/LBMV07/Papers/viola-cvpr-01.pdf

[15] https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[16] A. Mehra, ”Deepfake detection using capsule networks with long short-term memory networks”, Master’s thesis, University of Twente, 2020.

[17] https://github.com/facebookresearch/ResNeXt: :text=ResNeXt%20is%20a%20simple%2C%20highly,transformations%20with%20the%20same%20topology

[18] https://www.kaggle.com/c/deepfake-detection-challenge/data