IIVP Course Project

BTech, IT Department, IIIT Allahabad

IIT2018156(Kartik Nema), IIT2018163(Bhupendra), IIT2018172(Prakhar Srivastava),

IIT2018175(Ashish Patel), IIT2018177(Shubham Soni)

Detection of Image Splicing using Image Forgery Techniques

Abstract **-** In this paper we discuss an approach to detect splicing forgery in an image. For this purpose we consider all windows of 64x64 in an image and try to classify those windows as fake or authentic. For the purpose of classification we use the CNN (convolutional neural networks).

Keywords **-** Image forgery, splicing, edgemasks, Convolutional Neural Networks, pickling

I. Introduction

Image forgery refers to the manipulation of the digital image so as to hide or conceal some information in the image. Two common methods are used for image forgery, they are copy-move and splicing.

(i) Copy-move :- A part of picture is picture is copied and pasted over
region of the same image in order to hide details of that region.

(i) Splicing :- It involves composition of 2 or more images as one.

In splicing some part of an image is copied and pasted into some region of another image, it results in loss of image information in the second image (over which the copied section is pasted). It is different from copy-move forgery because the latter involves only one image.

II. Literature Review

Research papers considered

Forgery Detection of Spliced Images Using Machine Learning Classifiers and color Illumination [1]:- In this method the illuminant color inconsistency and machine learning classifiers like Support Vector Machines and Least square Support Vector Machine and Perceptrons are used for forgery detection.

A Review on Splicing Image Forgery Detection Techniques [2]:- Different methods of forgery detection which are

based on statistics, physics or or only algorithmic methods.

Image Forgery Identification using Convolution Neural Network [3]:- For image forgery detection, we project a technique that merges ZM-polar and block discrete cosine transform as motivated by the Works. Over copy-move forgery and splicing.

Image Splicing Detection via Camera Response Function Analysis [4] (base paper) :- Here classifiers like CNN which are used for image classification and recognition and SVM classifiers are used to detect image forgery.

Investigation of image forgery based on multiscale retinex under illumination variations [5]:- The method utilizes statistical region merging (SRM) segmentation method to segment an image. Then, a multiscale retinex (MSR) with color distribution is employed to improve the color of regions

Digital image forgery detection using a deep learning approach [6]:- Patches of images are given as input to classify them as real or fake. The developed approach involves the analysis of the image in a sliding window and the classification of each image fragment corresponding to the position of the window.

Image Forgery Detection Based on statistical features of block DCT coefficients [7]:- Here the image is converted from RGB to YCbCr color space and a feature matrix is formed by creating standard deviation and ones for each component and combined.This is passed to the classifiers to identify the image.

III. Methodology

3.1 *Feature Extraction*:

The CUISDE Dataset used here is not fit for training a model. We are given that for every fake image there is a mask present.This mask is used to sample the fake image along the boundary of the spliced image to ensure that there is a contribution of around 25% from both the non-spliced and the spliced region.To make the coundaris more distinct the image is converted to grayscale and then it is further converted to binary image using Otsu's thresholding implemented in OpenCV module.Further sampling is done by creating a window of 64X64 and moving it through the fake image and counting the number of zero valued pixels(dark pixels).These boundaries are now to be learned by the classifier. Since the data was too large therefore different batches are created .Each batch consists of the data frame for 10 images. So, 16 batches are created from the CUISDE Dataset

3.2 *Classification*:

The data collected in the form of batches from the Feature Extraction stage is now needed to be classified using a CNN classifier. First the data is splitted in Testing and Training data in the ratio of 1:4 respectively.This data is trained on the training dataset. Then to predict the labels in the test dataset it is passed through predict function.

IV. Implementation

4.1 Code Description

4.1.1 Preprocessing.py:

Step 1: The paths for the authentic and forged images are created.

Step 2: All the fake images and their masks are passed to the extractWindowFake function which makes a window of 64X64 and counts the number of dark valued pixels in the mask image.If it is greater than a certain value then it is inserted into the 'data' section.

Step 3: All the authentic images are passed to the extractWindow function again in which a window is created and is traversed through the image. But here the R,G,B components of the image(in a window) are stored and returned.

Step 4: The data generated by these two functions are stored in a pickle file.Since the data was too large therefore the preprocessing stage code was implemented batch wise.

Step 5: After generating the pickle files for all the batches they were stored in Google Drive.

4.1.2 Classify.ipynb:

Step 1: The folder which contains all the pickle files is connected to the runtime of the Colab.

Step 2: The dataset is loaded into the dictionary dic from all the pickle files.The data and tag contains the dic['data'] and dic['tag'] respectively.

Step 3: The dataset is then splitted into training and testing data in the ratio of 4:1.

Step 4: The CNN classifier is further trained on the training data in 50 iterations and 32 batch size.

Step 5: For prediction x_test is passed to the predict function which will return an array.The values are rounded off.If it comes out to be 0 then the image is authentic else forged.

Step 6: To measure the accuracy predicted array is compared to the y_test and counting the number of correctly predicted values.

We observe that the accuracy by this method comes out to be 68.43%

4.1.3 sample_output.py

To enable easy access we created a graphical user interface.

Step 1: All the pickle files which were created are downloaded and stored in the folder.

Step 2: All the data in the pickle files are loaded into the dataset.Then our model is trained using the random forest classifier.

Step 3: The image to be classified is selected in the GUI.

Step 4: Using our model that image is classified as authentic or spliced.

Step 5: The message is displayed on the GUI

4.2 Dataset description

Our group found the Columbia Uncompressed Image Splicing Detection Dataset (CUISDE Dataset) [9] to be more insightful than the others, it contains two folders one having 183 authentic images and another containing 180 spliced images. The images cover both indoor and outdoor settings. They were taken using a variety of cameras like canon, nikon etc.

CUISDE contains 183 authentic and 180 spliced images and the masks.

Following are samples from the CUISDE dataset, the one on left is a spliced image present in the 4cam_splc folder, while the one on right is authentic present in the 4cam_auth folder.

Also each folder contains a sub-folder named edgemasks which stores edgemasks for all the images in the original directory, i.e. the dataset contains edgemasks for all the images be they spliced or authentic.

4.3 Technologies used

Language :- Python (Python 3)

Tools and Libraries :-

For development purposes we used coding environments like Google Colab, PyCharm, Jupyter Notebook.

A number of text editors like sublime text, Atom, Notepad++, VS code etc. were used.

Libraries include openCV, PIL, sklearn, numpy, pandas, matplotlib, pickle, tkinter, keras etc.

4.4 Software & Hardware Requirements

The code was developed (in addition to Google Colab) on local system having the following specifications :-

RAM :- 8 GB

Intel Core i5 processor

1 TB hard disk drive

Operating Systems :- Windows 10 (64 bit), Linux Ubuntu (LTS 18.04)

V. Analysis

5.1 Accuracy

To test our model we divided the dataset into two parts one containing 80% of samples used for training and one having 20% for testing. For this purpose of splitting the dataset into 2 parts we use sklearn's train_test_split. On testing our model on the above mentioned test set (20% of entire dataset), we get an accuracy of 68.43%, i.e. our model was able to correctly sample 644 out of the total 941 samples.

5.2 Performance measures

Time complexity for loading the image data:-

O(file_row_size0) + O(file_row_size2).....O(file_row_size15)

If we generalize the size say n then:-

$O(n*16) \Rightarrow O(n)$.

Since each feature is associated with each node in the hidden layer the number of weights to calculate is => m(number of nodes in hidden layer)*n(number of features).

So if there are m number of nodes in the hidden layer , we need to calculate the bias parameter for each node ,So the total weights to calculate are:-

m*n + m.

So for epochs time complexity will become => epochs(m*n + m).

Now for predicting the output of the testing data we are using for loop for len(test_data) times. If the size of the testing data is n :-

Then its complexity is O(n) .

Actual time taken by code to execute is:-

In sec (for python program) => 79.9553234577179 sec.

VI. Conclusion

In this paper we discussed a solution to the problem of "Detection of image splicing using image forgery techniques", for this purpose we decided to use the CNN classification model on the CUISDE dataset. The method involves considering all 64*64 windows in the image. In case of authentic images all windows are considered while in case of spliced images only the ones with a certain threshold of 1s are considered. Finally we split the data for training and testing purposes and classify it using the CNN algorithm achieving an *accuracy of* 68.43%. We aim to utilize the time before final submission to work on some remaining (some optional) factors like :-

(i) Training and testing model on a larger dataset like the CASIA [8] dataset.

(ii) Improving performance of the model by experimenting with more algorithms and parameters configurement.

(ii) Optionally trying to provide a gui for the project.

VII. References

[1] Tamana Sharma, Er.Mandeep Kaur "Forgery Detection of Spliced Images Using Machine Learning Classifiers and color Illumination", International Journal of Innovative Research in Science, Engineering and Technology http://www.ijirset.com/upload/2016/june/186_54_Forgery.pdf

[2] Chitwan Bhalla Surbhi Gupta, "A Review on Splicing Image Forgery Detection Techniques", International Journal of Computer Science and Information Technology & Security (IJCSITS), ISSN: 2249-9555 Vol.6, No.2, Mar-April 2016 https://pdfs.semanticscholar.org/f6a9/877a29d2b90601d320cabe491c16ddc0270a.pdf

[3] N. Hema Rajini, "Image Forgery Identification using Convolution Neural Network" ,International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-8, Issue-1S4, June 2019 https://www.ijrte.org/wp-content/uploads/papers/v8i1s4/A10550681S419.pdf

[4] Can Chen, Scott McCloskey, Jingyi Yu, "Image Splicing Detection via Camera Response Function Analysis", Computer Vision foundation https://openaccess.thecvf.com/content_cvpr_2017/papers/Chen_Image_Splicing_Detection_CVPR_2017_paper.pdf

[5] Diaa M. Uliyan, Mohammad T.Alshammari, "Investigation of image forgery based on multiscale retinex under illumination variations", Department of Information and Computer Science, College of Computer Science and Engineering, University of Ha'il, Ha'il, 81481, Saudi

Arabia
https://www.researchgate.net/publication/341728677_Investigation_of_image_forgery_based_on_multiscale_retinex_under_illumination_variations

[6] A Kuznetsov, “Digital image forgery detection using deep learning approach”, Journal of Physics: Conference Series https://www.researchgate.net/publication/337550730_Digital_image_forgery_detection_using_deep_learning_approach

[7] Shilpa Duaa, Jyotsna Singha, Harish Parthasarathya, “Image forgery detection based on statistical features of block DCT coefficients”, Third International Conference on Computing and Network Communications (CoCoNet’19) https://www.sciencedirect.com/science/article/pii/S1877050920310048

[8] Jing Dong, Wei Wang and Tieniu Tan, “CASIA IMAGE TAMPERING DETECTION EVALUATION DATABASE”, Institute of Automation, Chinese Academy of Sciences P.O.Box 2728, Beijing, 10190 http://ir.ia.ac.cn/bitstream/173211/12318/1/06625374.pdf

[9] Y.-F. Hsu and S.-F. Chang. Detecting image splicing using geometry invariants and camera characteristics consistency. In 2006 IEEE International Conference on Multimedia and Expo, pages 549–552. IEEE, 2006”. Columbia Uncompressed Image Splicing Detection Dataset is available here.

[10] https://towardsdatascience.com/image-forgery-detection-2ee6f1a65442

[11] https://github.com/vishu160196/image-forgery-detection/blob/master/top_model.ipynb

[12] https://www.geeksforgeeks.org/python-gui-tkinter/