International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 2 Volume No.7 March - April – 2023
DOI:10.46647/ijetms.2023.v07i02.043 ISSN: 2581-4621

# Image Forgery Detection Using Neural Network In Python

**K. Dhanunjayudu[1], Sabiha Dudekula[2], Sammatha Gowd Pallavi[3], Gandhudi Neethu[4], Vutakonda Bijje Ramakrishna[5], Jayavaram Naga Jyothi[6]**
[1]*Assistant Professor, Dept. of computer science engineering, Gates Institute of Technology, Gooty, A.P*
[2]*UG-computer science engineering, Gates Institute of Technology Gooty, A.P*
[3]*UG-computer science engineering, Gates Institute of Technology Gooty, A.P*
[4]*UG-computer science engineering, Gates Institute of Technology Gooty, A.P*
[5]*UG-computer science engineering, Gates Institute of Technology Gooty, A.P*
[6]*UG-computer science engineering, Gates Institute of Technology Gooty, A.P*

## Abstract

The advancements of technology in every aspect of the current age are leading to the misuse of data. Researchers, therefore, face the challenging task of identifying these manipulated forms of data and distinguishing the real data from the manipulated. Splicing is one of the most common techniques used for digital image tampering; a selected area copied from the same or another image is pasted in an image. Image forgery detection is considered a reliable way to verify the authenticity of digital images. In this study, we proposed an approach based on the state-of-the-art deep learning architecture of ResNet50v2. The proposed model takes image batches as input and utilizes the weights of a YOLO convolutional neural network (CNN) by using the architecture of ResNet50v2. In this study, we used the CASIA_v1 and CASIA_v2 benchmark datasets, which contain two distinct categories, original and forgery, to detect image splicing. We used 80% of the data for the training and the remaining 20% for testing purposes. We also performed a comparative analysis between existing approaches and our proposed system. We evaluated the performance of our technique with the CASIA_v1 and CASIA_v2 datasets. Since the CASIA_v2 dataset is more comprehensive compared to the CASIA_v1 dataset, we obtained 99.3% accuracy for the fine-tuned model using transfer learning and 81% accuracy without transfer learning with the CASIA_v2 dataset. The results show the superiority of the proposed system

**Keywords:** Image, Neural Network, CNN, Forgery

## 1.Introduction

Digital images have an important role in many fields such as in newspapers, digital forensics, scientific research, medicine, and so forth. Nowadays, the usage and sharing of digital images on social media platforms is also widespread. Digital images are considered one of the main sources of information. Considering the excessive use of image sharing through various social media platforms such as WhatsApp, Instagram, Telegram, and Reddit, differentiating between real and forged images is a challenging task. The availability of many image editing software applications is making it more difficult to detect the authenticity of an image day by day. There are generally two approaches that image manipulation can be categorized into, as follows: 1. Active approach; 2. Passive approach. With the active approach, a watermark or digital signature is embedded when the image is created. While using these embeddings, whether the image has been tampered with or not is analyzed at later stages. In the passive approach, any pre-embedded information, such as a watermark em- bedded for the detection of image forgery, cannot be relied upon. This approach is also known as the blind approach because there is no additional information for image forgery detection. This approach is based on features that are extracted directly from the images. Furthermore, the passive approach can be categorized into two types—independent and dependent. The independent approach detects resampling and compression forgeries

## 2.Literature Survey

### Detection of Copy-Move Forgery in Digital Images

**AUTHOR: Fridrich, J.; Soukal, D.; Lukás, J**

**ABSTRACT:**

Digital images are easy to manipulate and edit due to availability of powerful image processing and editing software. Nowadays, it is possible to add or remove important features from an image without leaving any obvious traces of tampering. As digital cameras and video cameras replace their analog counterparts, the need for authenticating digital images, validating their content, and detecting forgeries will only increase. Detection of malicious manipulation with digital images (digital forgeries) is the topic of this paper. In particular, we focus on detection of a special type of digital forgery – the copy-move attack in which a part of the image is copied and pasted somewhere else in the image with the intent to cover an important image feature. In this paper, we investigate the problem of detecting the copy-move forgery and describe an efficient and reliable detection method. The method may successfully detect the forged part even when the copied area is enhanced/retouched to merge it with the background and when the forged image is saved in a lossy format, such as JPEG. The performance of the proposed method is demonstrated on several forged images

### Digital Image Forgery Detection Based on Lens and Sensor Aberration

**AUTHOR: Yerushalmy,**

**ABSTRACT:**

A new approach to detecting forgery in digital photographs is suggested. The method does not necessitate adding data to the image (such as a Digital Watermark) nor require other images for comparison or training. The fundamental assumption in the presented approach is the notion that image features arising from the image acquisition process itself or due to the physical structure and characteristics of digital cameras, are inherent proof of authenticity and they are sensitive to image manipulation as well as being difficult to forge synthetically. Typically, such features do not affect image content nor quality and are often invisible to the inexperienced eye. The approach presented in this work is based on the effects introduced in the acquired image by the optical and sensing systems of the camera. Specifically, it exploits image artifacts that are due to chromatic aberrations as indicators for evaluating image authenticity

## 3.Existing System

The development of deep learning has led to improving methodologies where state-of-the-art methods, such as CNN, Mobile Net, and ResNet50v2, automatically extract the potential features, having been trained on large datasets. Some of the examples of CNN-based feature extractions are deep features utilized for image quality assessment [6], skin lesion classification [7 ], or person re-identification [8 ]. These extracted features are adapted into the inherent structural patterns of the data. This is the main reason behind their non-discriminative and robust architecture compared to the hand-engineered features. In this paper, motivated by the deep learning technique, we propose a transfer learning-based approach. It is an effective architecture with which we incorporated the weights of a model previously trained on a large database, and hence, it benefitted from using the meaningful weights without having to train the model from scratch. We present an architecture based on the ResNet50v2 architecture that employs the use of transfer learning for the detection of tampered images, specifically, spliced images. We used the pre-trained weights of a YOLO CNN model to detect images that were specifically tampered with using the image splicing technique. Furthermore, this study makes the following contributions to this field of research:

**DISADVANTAGES OF EXISTING SYSTEM :**

1) Less accuracy
2)low Efficiency

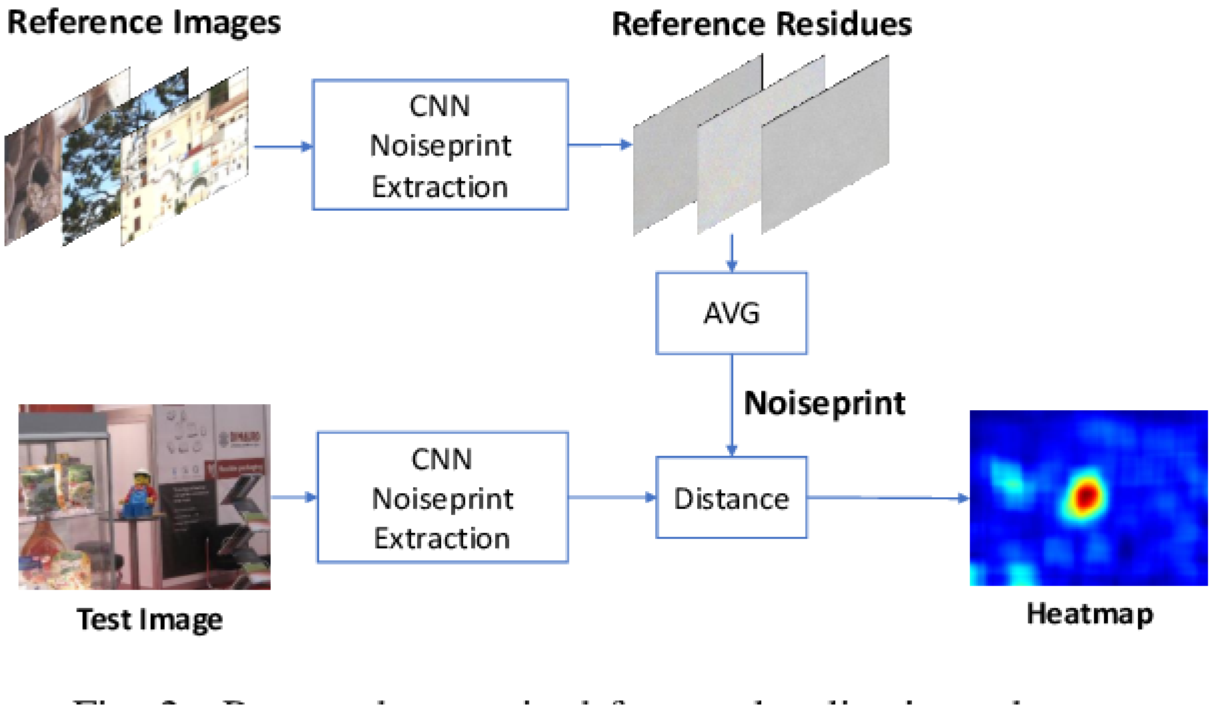
## 4.Proposed System

In this study, we proposed a deep learning-based approach for the identification of forged images. We proposed an architecture using ResNet50v2 as our base model, and we used the YOLO CNN weights for transfer learning. This approach enabled us to train the model with meaningful weights. We used pre-trained weights of the YOLO CNN object detection model to initialize our ResNet50v2-based proposed architecture, which saved a considerable amount of training costs, as we initialized our model with meaningful pre-trained weights. Figure 3 presents the basic architecture of ResNet50v2, in which initially batch normal- ization is performed, followed by an activation function and the weights being updated. Then we performed the batch normalization, ReLU activation function. After the acti- vation function, the weights were optimized. The basic difference from the ResNet50v2 architecture is that we used pre-activation of the weight layers instead of post-activation. ResNet50v2 was developed in such a way that it removes the nonlinearity, hence clearing a path from the input to the output as a means of an identity connection. Version 2 of the ResNet module also applies the batch normalization and the activation function before the weights are multiplied. The overall proposed system

**ADVANTAGES OF PROPOSED SYSTEM :**

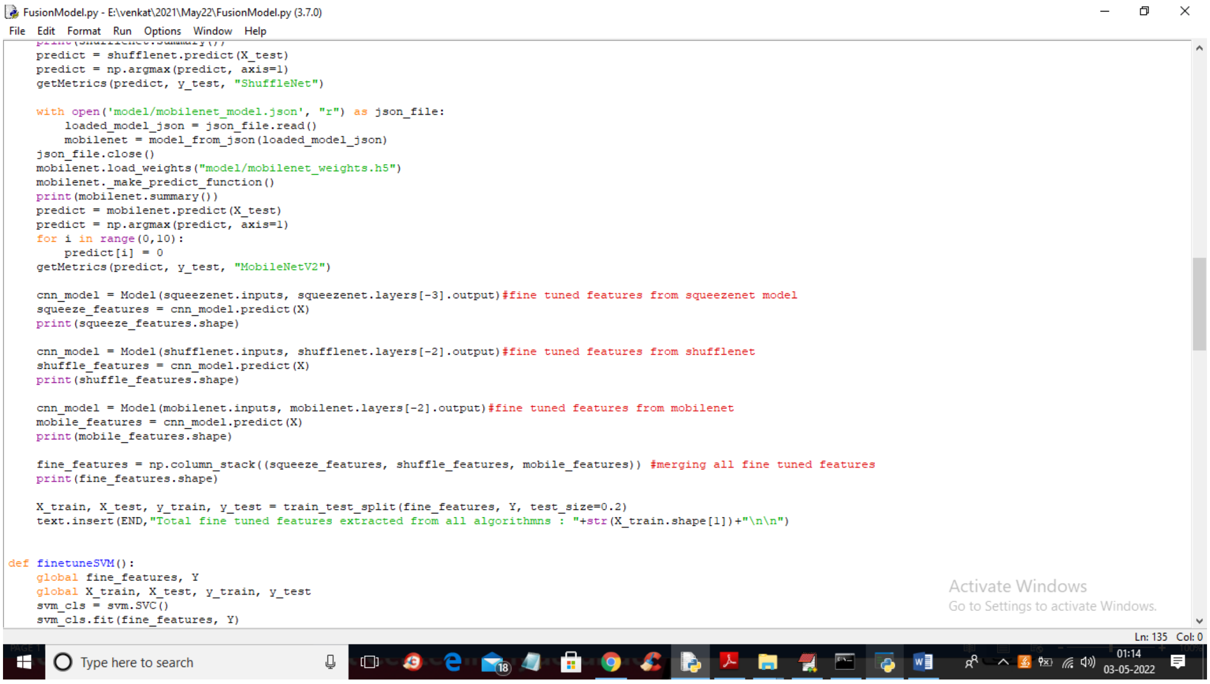
1) High accuracy
2)High efficiency

## 5.System Architecture



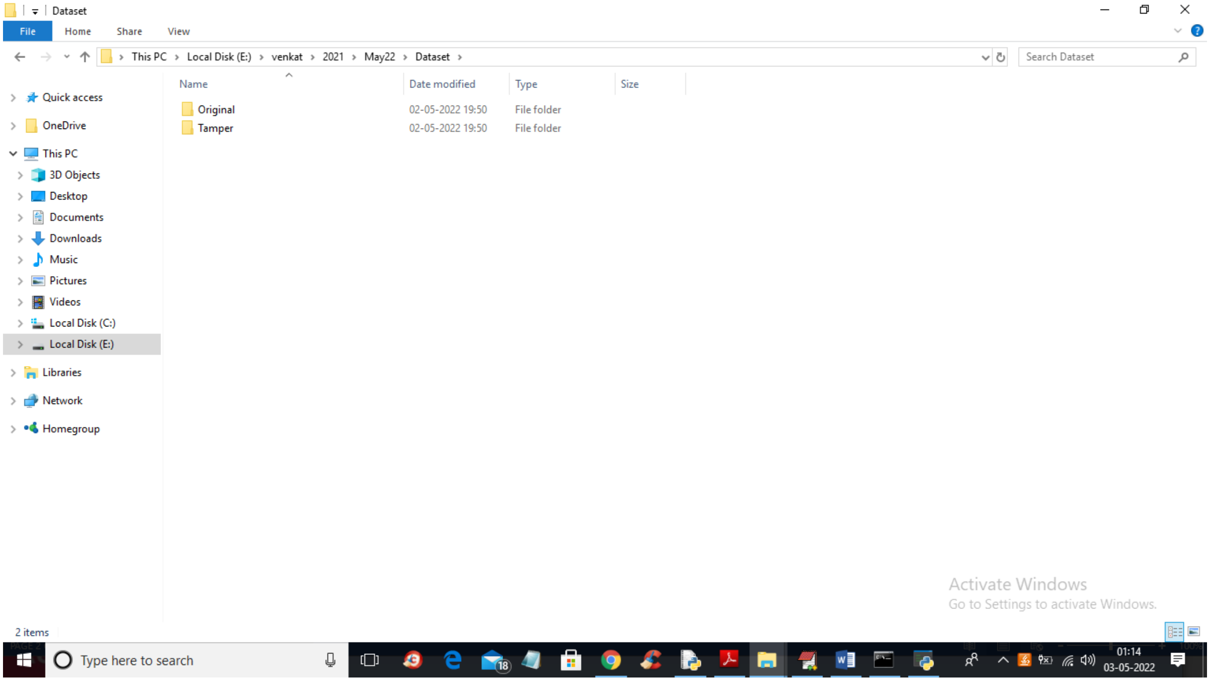
## 6. Results

In below screen code you can see how we are extracting features from all 3 algorithms and then building fusion model


FusionModel.py - E:\venkat\2021\May22\FusionModel.py (3.7.0)
File Edit Format Run Options Window Help
Ln: 135 Col: 0
Activate Windows
Go to Settings to activate Windows.
Type here to search
01:14
03-05-2022

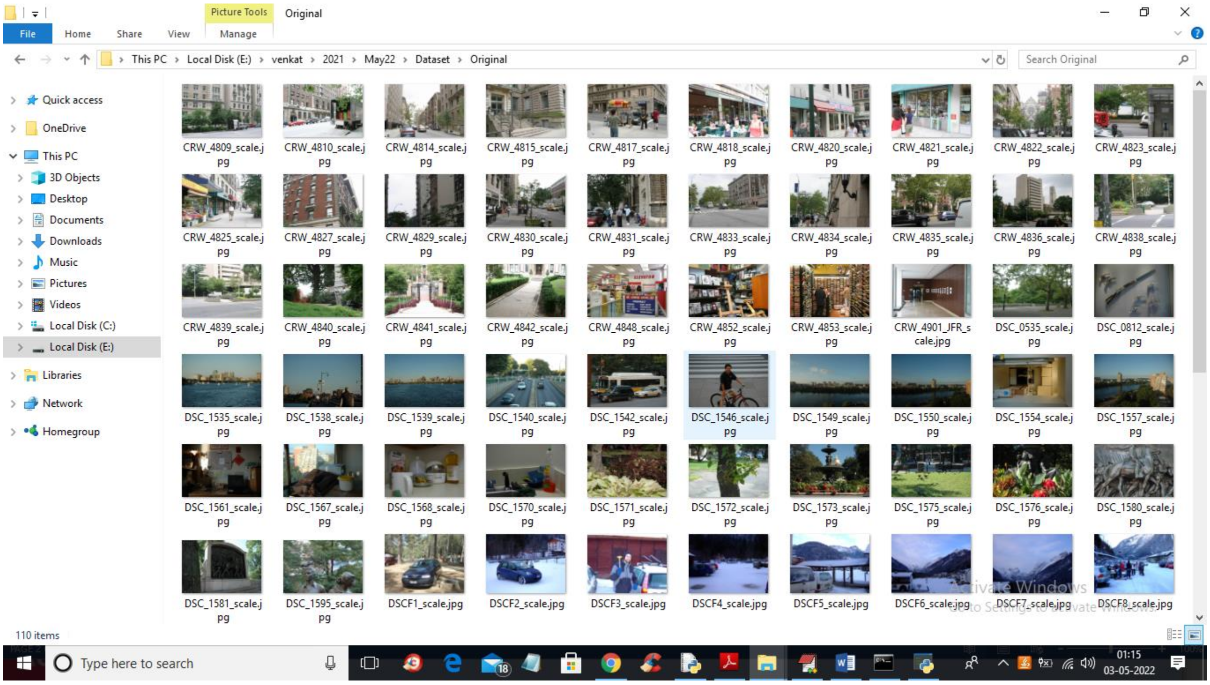
```
    print(shufflenet.summary())
    predict = shufflenet.predict(X_test)
    predict = np.argmax(predict, axis=1)
    getMetrics(predict, y_test, "ShuffleNet")

    with open('model/mobilenet_model.json', "r") as json_file:
        loaded_model_json = json_file.read()
        mobilenet = model_from_json(loaded_model_json)
    json_file.close()
    mobilenet.load_weights("model/mobilenet_weights.h5")
    mobilenet._make_predict_function()
    print(mobilenet.summary())
    predict = mobilenet.predict(X_test)
    predict = np.argmax(predict, axis=1)
    for i in range(0,10):
        predict[i] = 0
    getMetrics(predict, y_test, "MobileNetV2")

    cnn_model = Model(squeezenet.inputs, squeezenet.layers[-3].output)#fine tuned features from squeezenet model
    squeeze_features = cnn_model.predict(X)
    print(squeeze_features.shape)

    cnn_model = Model(shufflenet.inputs, shufflenet.layers[-2].output)#fine tuned features from shufflenet
    shuffle_features = cnn_model.predict(X)
    print(shuffle_features.shape)

    cnn_model = Model(mobilenet.inputs, mobilenet.layers[-2].output)#fine tuned features from mobilenet
    mobile_features = cnn_model.predict(X)
    print(mobile_features.shape)

    fine_features = np.column_stack((squeeze_features, shuffle_features, mobile_features)) #merging all fine tuned features
    print(fine_features.shape)

    X_train, X_test, y_train, y_test = train_test_split(fine_features, Y, test_size=0.2)
    text.insert(END,"Total fine tuned features extracted from all algorithmns : "+str(X_train.shape[1])+"\n\n")


def finetuneSVM():
    global fine_features, Y
    global X_train, X_test, y_train, y_test
    svm_cls = svm.SVC()
    svm_cls.fit(fine_features, Y)
```

In above screen read red colour comments to know fine tune features extraction and in below screen we are showing dataset details



| Name | Date modified | Type | Size |
|---|---|---|---|
| Original | 02-05-2022 19:50 | File folder | |
| Tamper | 02-05-2022 19:50 | File folder | |

In above screen in 'Dataset' folder we have 3 folders where one contains original images and other folder contains TAMPER or FORGE images and just go inside any folder to view its images



So by using above images we will train all algorithms and calculate their performances

To run project double click on ‘run.bat’ file to get below output

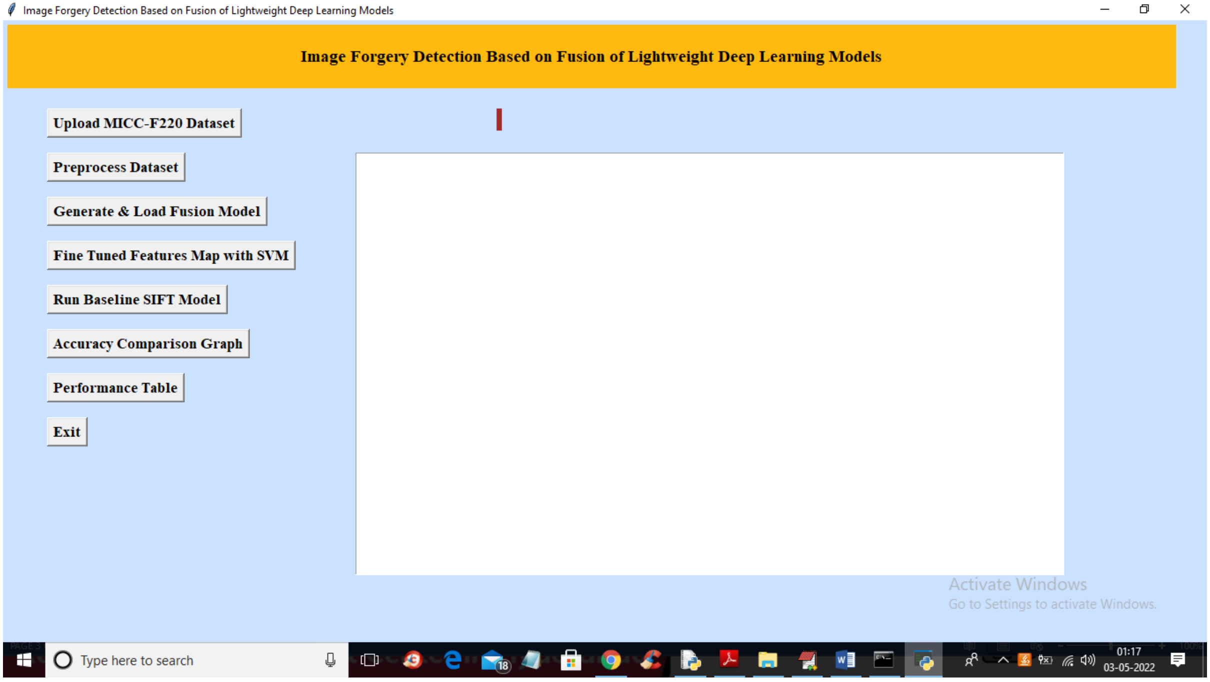
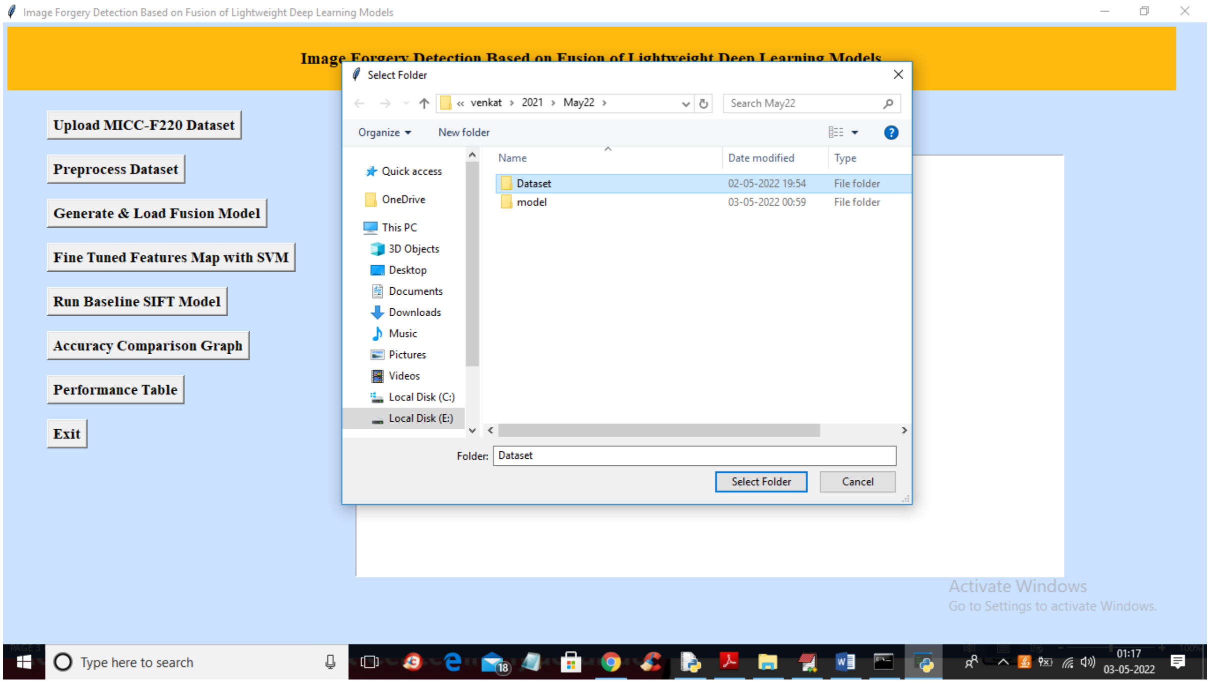


Image Forgery Detection Based on Fusion of Lightweight Deep Learning Models
Image Forgery Detection Based on Fusion of Lightweight Deep Learning Models
Upload MICC-F220 Dataset
Preprocess Dataset
Generate & Load Fusion Model
Fine Tuned Features Map with SVM
Run Baseline SIFT Model
Accuracy Comparison Graph
Performance Table
Exit
Activate Windows
Go to Settings to activate Windows.
Type here to search
01:17
03-05-2022

In above screen click on ‘Upload MICC-F220 Dataset’ button to upload dataset and get below output



| Name | Date modified | Type |
|---|---|---|
| Dataset | 02-05-2022 19:54 | File folder |
| model | 03-05-2022 00:59 | File folder |

Folder: Dataset

In above screen selecting and uploading 'Dataset' folder and then click on 'Select Folder' button to load dataset and get below output

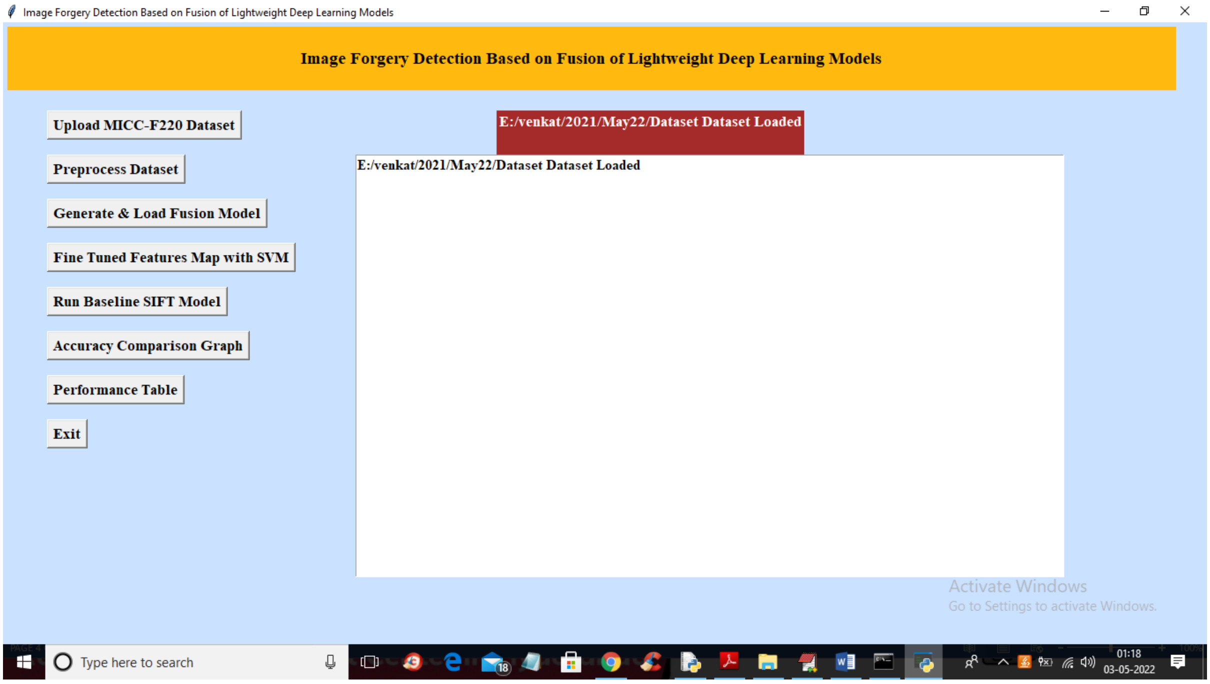


Image Forgery Detection Based on Fusion of Lightweight Deep Learning Models
Image Forgery Detection Based on Fusion of Lightweight Deep Learning Models
Upload MICC-F220 Dataset
Preprocess Dataset
Generate & Load Fusion Model
Fine Tuned Features Map with SVM
Run Baseline SIFT Model
Accuracy Comparison Graph
Performance Table
Exit
Activate Windows
Go to Settings to activate Windows.
Type here to search
01:18
03-05-2022

E:/venkat/2021/May22/Dataset Dataset Loaded

E:/venkat/2021/May22/Dataset Dataset Loaded

In above screen dataset loaded and now click on 'Preprocess Dataset' button to read all images and normalize them and get below output

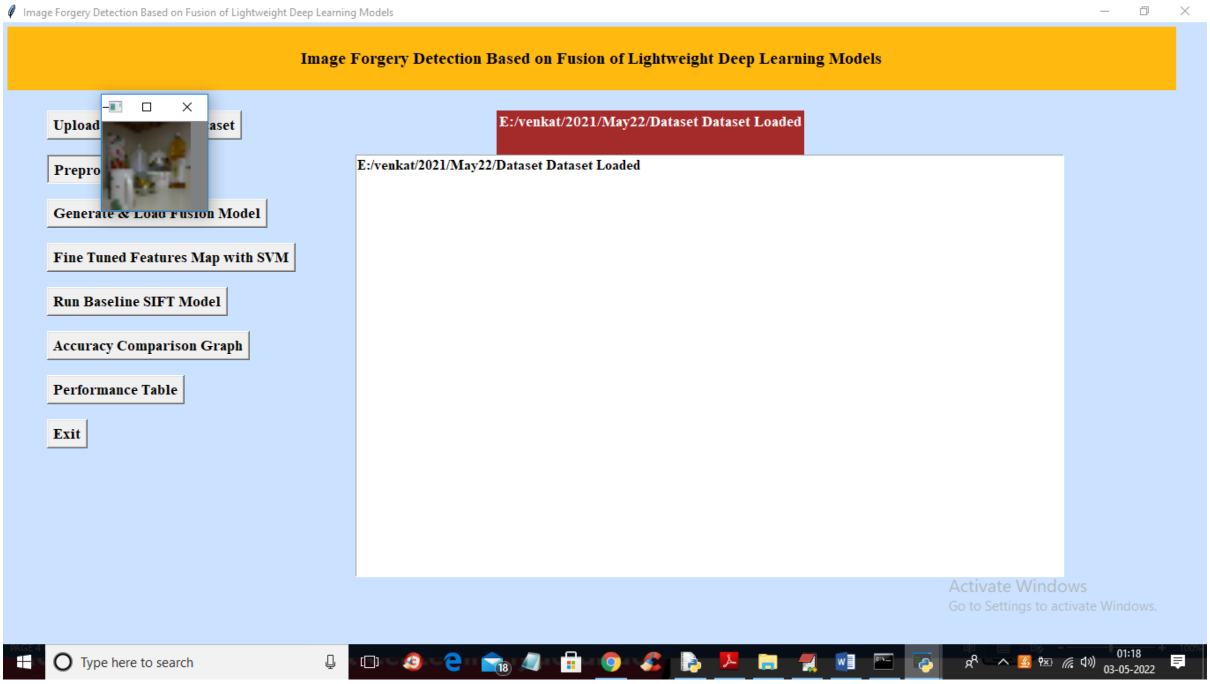


Image Forgery Detection Based on Fusion of Lightweight Deep Learning Models
Image Forgery Detection Based on Fusion of Lightweight Deep Learning Models
Upload
aset
Prepro
Generate & Load Fusion Model
Fine Tuned Features Map with SVM
Run Baseline SIFT Model
Accuracy Comparison Graph
Performance Table
Exit
E:/venkat/2021/May22/Dataset Dataset Loaded
Activate Windows
Go to Settings to activate Windows.
Type here to search
01:18
03-05-2022

E:/venkat/2021/May22/Dataset Dataset Loaded

In above screen all images are processed and to check images loaded properly I am displaying one sample image and now close above image to get below output

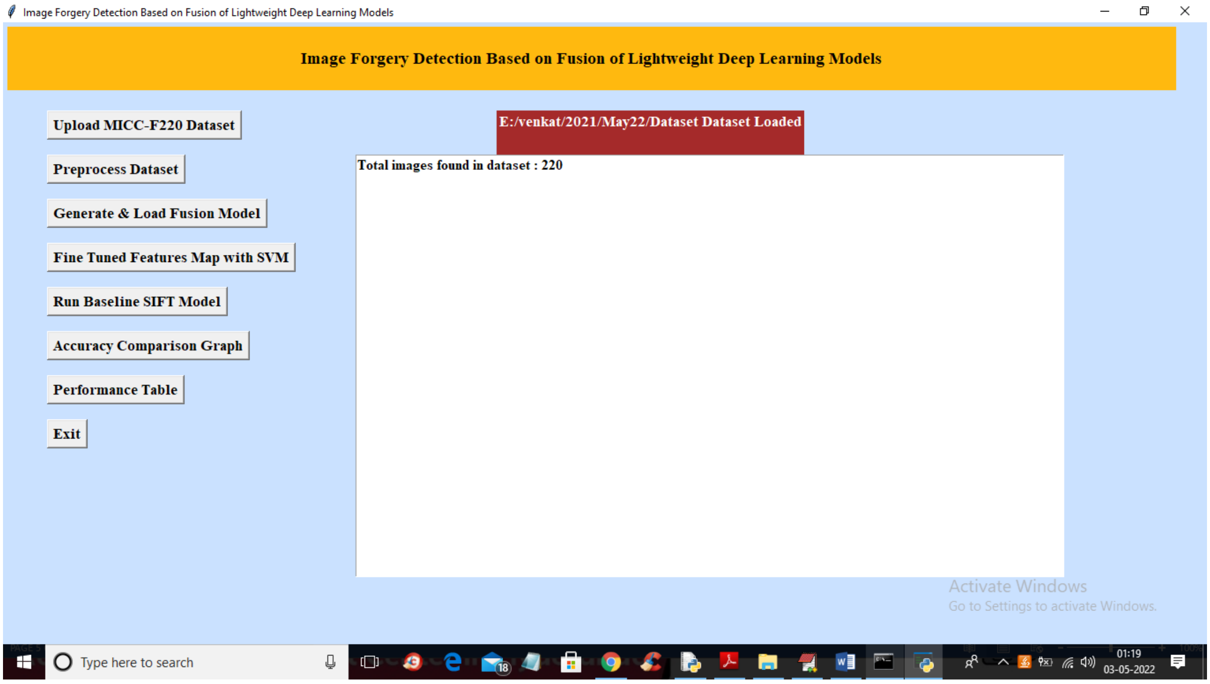


Image Forgery Detection Based on Fusion of Lightweight Deep Learning Models
Image Forgery Detection Based on Fusion of Lightweight Deep Learning Models
Upload MICC-F220 Dataset
Preprocess Dataset
Generate & Load Fusion Model
Fine Tuned Features Map with SVM
Run Baseline SIFT Model
Accuracy Comparison Graph
Performance Table
Exit
E:/venkat/2021/May22/Dataset Dataset Loaded
Activate Windows
Go to Settings to activate Windows.
Type here to search
01:19
03-05-2022

Total images found in dataset : 220

In above screen we can see dataset contains 220 images and all images are processed and now click on ‘Generate & Load Fusion Model’ button to train all algorithms and then extract features from them and then calculate their accuracy

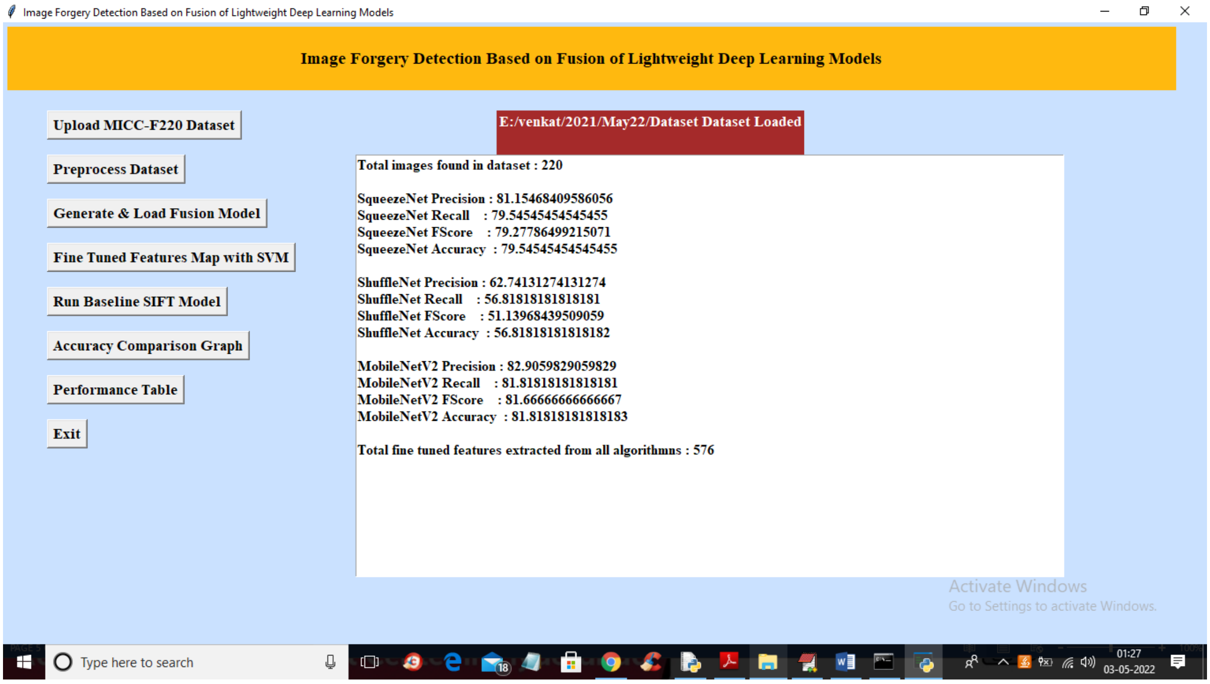


Image Forgery Detection Based on Fusion of Lightweight Deep Learning Models
Image Forgery Detection Based on Fusion of Lightweight Deep Learning Models
Upload MICC-F220 Dataset
Preprocess Dataset
Generate & Load Fusion Model
Fine Tuned Features Map with SVM
Run Baseline SIFT Model
Accuracy Comparison Graph
Performance Table
Exit
E:/venkat/2021/May22/Dataset Dataset Loaded
Activate Windows
Go to Settings to activate Windows.
Type here to search
01:27
03-05-2022

```
Total images found in dataset : 220

SqueezeNet Precision : 81.15468409586056
SqueezeNet Recall    : 79.54545454545455
SqueezeNet FScore    : 79.27786499215071
SqueezeNet Accuracy  : 79.54545454545455

ShuffleNet Precision : 62.74131274131274
ShuffleNet Recall    : 56.81818181818181
ShuffleNet FScore    : 51.13968439509059
ShuffleNet Accuracy  : 56.81818181818182

MobileNetV2 Precision : 82.9059829059829
MobileNetV2 Recall    : 81.81818181818181
MobileNetV2 FScore    : 81.66666666666667
MobileNetV2 Accuracy  : 81.81818181818183

Total fine tuned features extracted from all algorithmns : 576
```

In above screen we can see accuracy of all 3 algorithms and then in last line we can see from all 3 algorithms application extracted 576 features and now click on 'Fine Tuned Features Map with SVM' to train SVM with extracted features and get its accuracy as fusion model

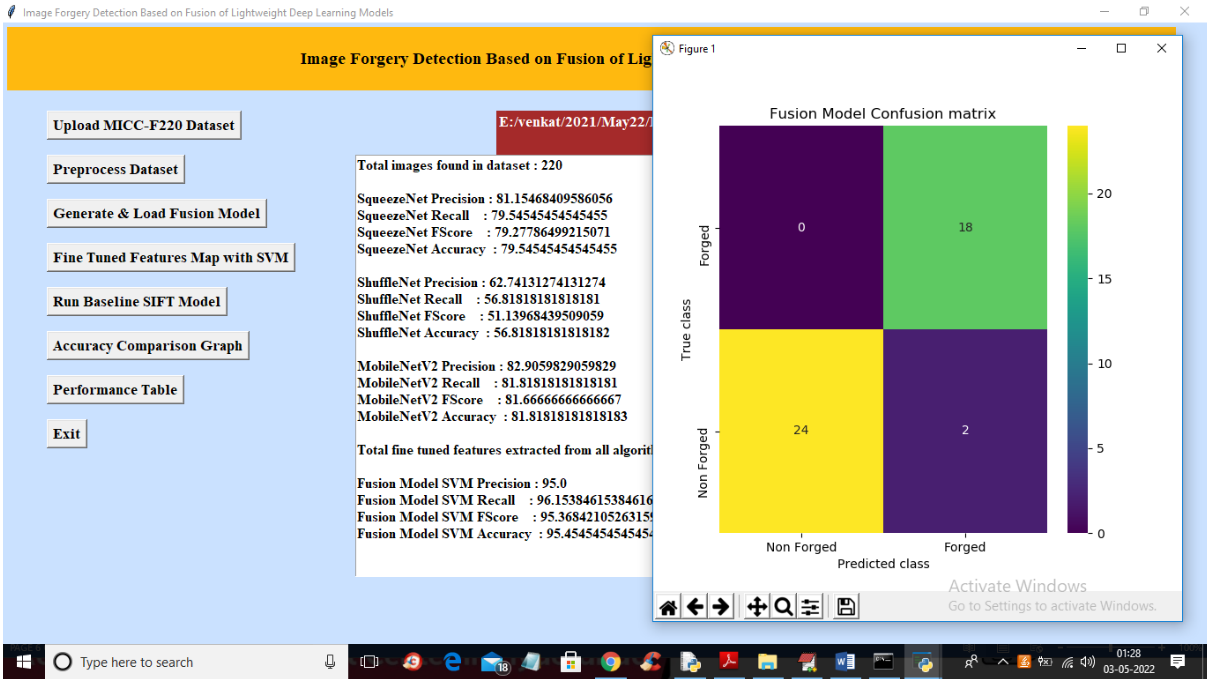


Image Forgery Detection Based on Fusion of Lightweight Deep Learning Models
Image Forgery Detection Based on Fusion of Lig
Upload MICC-F220 Dataset
Preprocess Dataset
Generate & Load Fusion Model
Fine Tuned Features Map with SVM
Run Baseline SIFT Model
Accuracy Comparison Graph
Performance Table
Exit
E:/venkat/2021/May22/
Figure 1
Fusion Model Confusion matrix
True class
Forged
Non Forged
Predicted class
Non Forged
Forged
0
18
24
2
20
15
10
5
0
Activate Windows
Go to Settings to activate Windows.
Type here to search
01:28
03-05-2022

```
Total images found in dataset : 220

SqueezeNet Precision : 81.15468409586056
SqueezeNet Recall    : 79.54545454545455
SqueezeNet FScore    : 79.27786499215071
SqueezeNet Accuracy  : 79.54545454545455

ShuffleNet Precision : 62.74131274131274
ShuffleNet Recall    : 56.81818181818181
ShuffleNet FScore    : 51.13968439509059
ShuffleNet Accuracy  : 56.81818181818182

MobileNetV2 Precision : 82.9059829059829
MobileNetV2 Recall    : 81.81818181818181
MobileNetV2 FScore    : 81.66666666666667
MobileNetV2 Accuracy  : 81.81818181818183

Total fine tuned features extracted from all algorit

Fusion Model SVM Precision : 95.0
Fusion Model SVM Recall    : 96.15384615384616
Fusion Model SVM FScore    : 95.36842105263159
Fusion Model SVM Accuracy  : 95.4545454545454
```

In above screen with Fine tune SVM fusion model we got 95% accuracy and in confusion matrix graph x-axis represents PREDICTED LABELS and y-axis represent TRUE labels and we can see both X and Y boxes contains more number of correctly prediction classes. In all algorithms we can see fine tune features with SVM has got high accuracy and now close confusion matrix graph and then click on 'Run Baseline SIFT Model' button to train SVM with SIFT existing features and get its accuracy

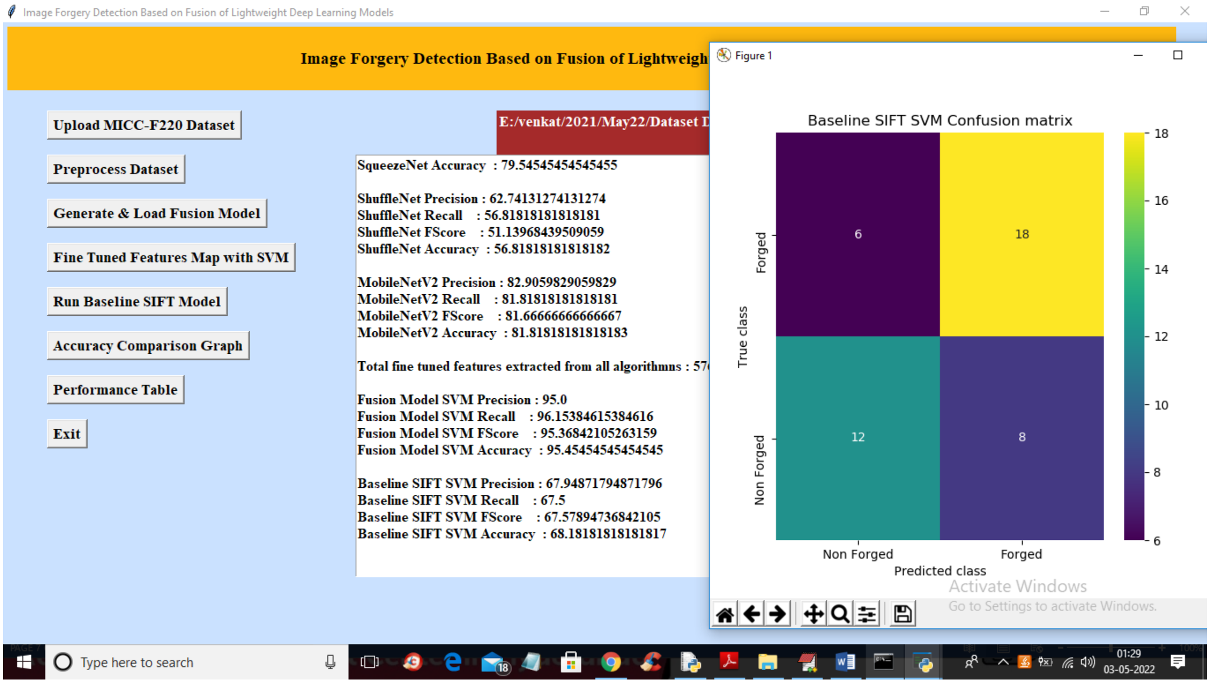
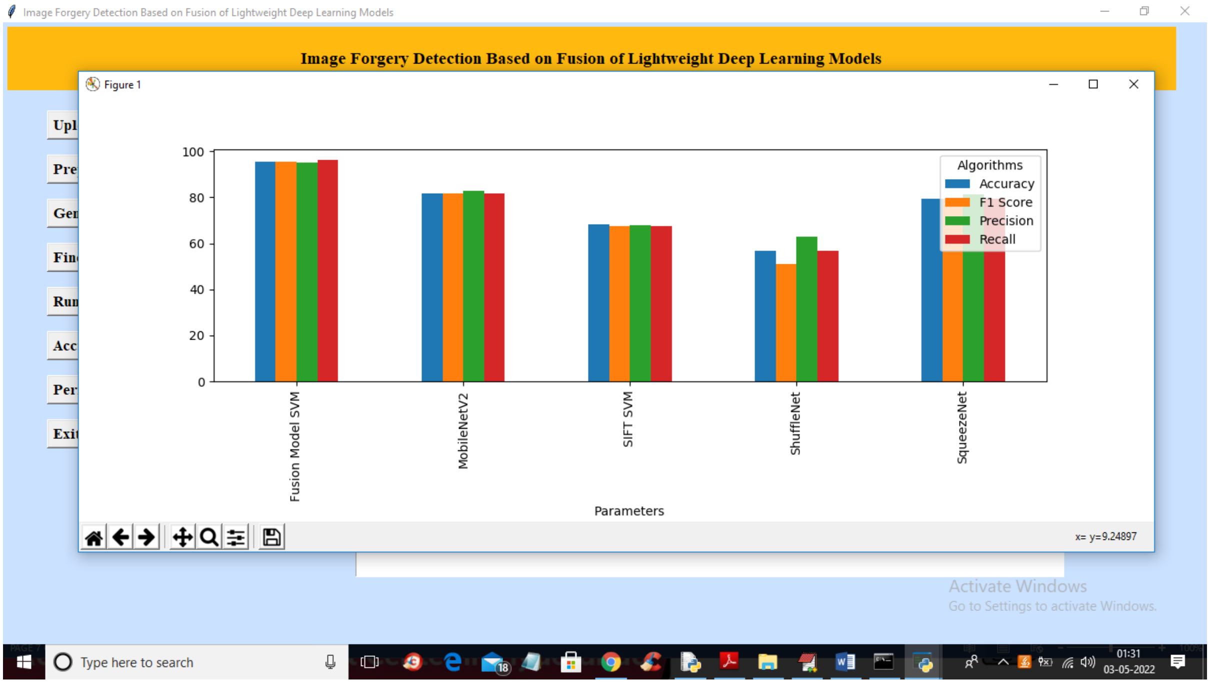


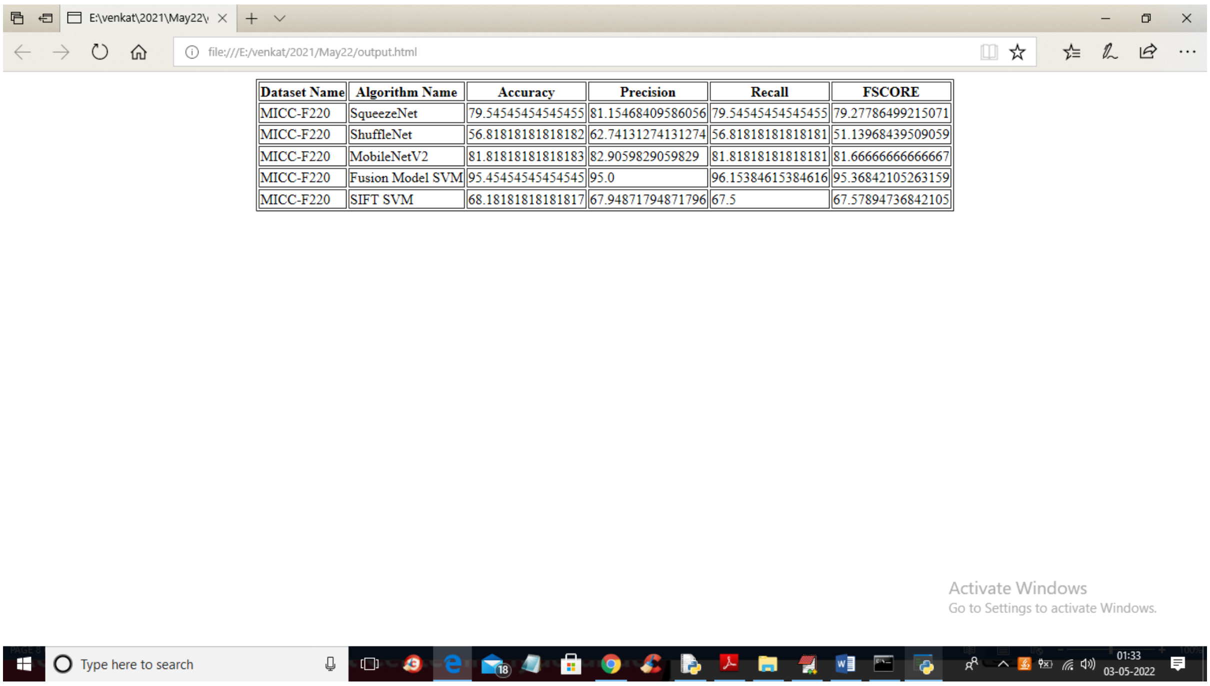
Image Forgery Detection Based on Fusion of Lightweight Deep Learning Models
Image Forgery Detection Based on Fusion of Lightweigh
Upload MICC-F220 Dataset
Preprocess Dataset
Generate & Load Fusion Model
Fine Tuned Features Map with SVM
Run Baseline SIFT Model
Accuracy Comparison Graph
Performance Table
Exit
E:/venkat/2021/May22/Dataset D
Figure 1
Baseline SIFT SVM Confusion matrix
Forged
Non Forged
True class
6
18
12
8
Non Forged
Forged
Predicted class
18
16
14
12
10
8
6
Activate Windows
Go to Settings to activate Windows.
Type here to search
01:29
03-05-2022

```
SqueezeNet Accuracy  : 79.54545454545455

ShuffleNet Precision : 62.74131274131274
ShuffleNet Recall    : 56.81818181818181
ShuffleNet FScore    : 51.13968439509059
ShuffleNet Accuracy  : 56.81818181818182

MobileNetV2 Precision : 82.9059829059829
MobileNetV2 Recall    : 81.81818181818181
MobileNetV2 FScore    : 81.66666666666667
MobileNetV2 Accuracy  : 81.81818181818183

Total fine tuned features extracted from all algorithmns : 57

Fusion Model SVM Precision : 95.0
Fusion Model SVM Recall    : 96.15384615384616
Fusion Model SVM FScore    : 95.36842105263159
Fusion Model SVM Accuracy  : 95.45454545454545

Baseline SIFT SVM Precision : 67.94871794871796
Baseline SIFT SVM Recall    : 67.5
Baseline SIFT SVM FScore    : 67.57894736842105
Baseline SIFT SVM Accuracy  : 68.18181818181817
```

In above screen with existing SIFT SVM features we got 68% accuracy and in confusion matrix graph we can see existing SIFT predicted 6 and 8 instances incorrectly. So we can say existing SIFT features are not good in prediction and now close above graph and then click on ‘Accuracy Comparison Graph’ button to get below graph



In above graph x-axis represents algorithm names and y-axis represents accuracy and other metrics where each different colour bar represents different metrics like precision, recall etc. Now close above graph and then click on ‘Performance Table’ button to get result in below tabular format


E:\venkat\2021\May22\
file:///E:/venkat/2021/May22/output.html
Activate Windows
Go to Settings to activate Windows.
Type here to search
01:33
03-05-2022

| Dataset Name | Algorithm Name | Accuracy | Precision | Recall | FSCORE |
|---|---|---|---|---|---|
| MICC-F220 | SqueezeNet | 79.54545454545455 | 81.15468409586056 | 79.54545454545455 | 79.27786499215071 |
| MICC-F220 | ShuffleNet | 56.81818181818182 | 62.74131274131274 | 56.81818181818181 | 51.13968439509059 |
| MICC-F220 | MobileNetV2 | 81.81818181818183 | 82.9059829059829 | 81.81818181818181 | 81.66666666666667 |
| MICC-F220 | Fusion Model SVM | 95.45454545454545 | 95.0 | 96.15384615384616 | 95.36842105263159 |
| MICC-F220 | SIFT SVM | 68.18181818181817 | 67.94871794871796 | 67.5 | 67.57894736842105 |

In above screen we can see propose fusion model SVM with fine tune features has got 95% accuracy which is better than all other algorithms

## 7.Conclusion

Image forgery detection is a very challenging problem. In this era of technological advancement, we need to be able to distinguish between real and tampered images. In this study, we proposed a deep learning-based approach for image forgery detection. The proposed model is based on ResNet50v2 architecture, which uses residual layers; thus, using this architecture increases the detection rate of tampered images. Using this approach also provides the benefit of transfer learning by using the pre-trained weights of the YOLO CNN model. The use of transfer learning enabled us to train our model more efficiently, as we initialized our proposed model by meaningful assigning weights. This reduced the training time and complexity of the model and makes the architecture more efficient. We evaluated our proposed architecture on benchmark datasets, CASIA_v1 and CASIA_v2. We also compared the performance of our system with and without the use of transfer learning. We obtained an accuracy of 99.30% with the CASIA_v2 dataset for the forgery detection problem. The results of the comparison with the existing methods show the superiority of the proposed system. The proposed system will help in the image manipulation detection domain and also paves the way for future research in detecting multiple types of image forgery manipulations.

## References

1. Fridrich, J.; Soukal, D.; Lukás, J. Detection of Copy-Move Forgery in Digital Images. Int. J. Comput. Sci. 2003, 3, 55–61.
2. Yerushalmy, I.; Hel-Or, H. Digital Image Forgery Detection Based on Lens and Sensor Aberration. Int. J. Comput. Vis. 2011, 92, 71–91. [CrossRef]
3. Dirik, A.E.; Memon, N. Image tamper detection based on demosaicing artifacts. In Proceedings of the 2009 16th IEEE International Conference on Image Processing (ICIP), Cairo, Egypt, 7–10 November 2009; pp. 1497–1500. [CrossRef]
4. Christlein, V.; Riess, C.; Jordan, J.; Riess, C.; Angelopoulou, E. An Evaluation of Popular Copy-Move Forgery Detection Approaches. IEEE Trans. Inf. Forensics Secur. 2012, 7, 1841–1854. [CrossRef]
5. Zampoglou, M.; Papadopoulos, S.; Kompatsiaris, Y. Large-scale evaluation of splicing localization algorithms for web images. Multimed. Tools Appl. 2016, 76, 4801–4834. [CrossRef]

6. Varga, D. Multi-Pooled Inception Features for No-Reference Image Quality Assessment. Appl. Sci. 2020, 10, 2186. [CrossRef]
7. Kawahara, J.; Bentaieb, A.; Hamarneh, G. Deep features to classify skin lesions. In Proceedings of the 2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI), Prague, Czech Republic, 13–16 April 2016; pp. 1397–1400.
8. Bai, X.; Yang, M.; Huang, T.; Dou, Z.; Yu, R.; Xu, Y. Deep-Person: Learning discriminative deep features for person Re-Identification. Pattern Recognit. 2020, 98, 107036. [CrossRef]
9. Dong, J.; Wang, W.; Tan, T. CASIA Image Tampering Detection Evaluation Database. In Proceedings of the 2013 IEEE China Summit and International Conference on Signal and Information Processing, Beijing, China, 6–10 July 2013.
10. Mahdian, B.; Saic, S. A bibliography on blind methods for identifying image forgery. Signal Process. Image Commun. 2010, 25, 389–399. [CrossRef]
11. Farid, H. Image forgery detection. IEEE Signal Process. Mag. 2009, 26, 16–25. [CrossRef]
12. Lanh, T.V.; Chong, K.; Emmanuel, S.; Kankanhalli, M.S. A Survey on Digital Camera Image Forensic Methods. In Proceedings of the 2007 IEEE International Conference on Multimedia and Expo, Beijing, China, 2–5 July 2007; pp. 16–19. [CrossRef]
13. Barad, Z.; Goswami, M. Image Forgery Detection using Deep Learning: A Survey. In Proceedings of the 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 6–7 March 2020; pp. 571–576.
14. Wu, Y.; Abd-Almageed, W.; Natarajan, P. BusterNet: Detecting Copy-Move Image Forgery with Source/Target Localization. In Lecture Notes in Computer Science; Springer: Berlin, Germany, 2018; pp. 170–186.
15. Manjunatha, S.; Patil, M.M. Deep learning-based Technique for Image Tamper Detection. In Proceedings of the 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV), Tirunelveli, India