Image Forgery Detection

Anmol Jain (MT19005) and Bhakti Batra (MT19115)

I. Problem statement and motivation

This project attempt to solve the problem of forgery detection, and tries to differentiate forged image with the normal image.The methodology is to extract the features of images using deep learning models. Further, classifying the images to be forged or authentic with the help of extracted features.

II. Literature review

The problem of image forgery detection has become an urge in the era of boosting technology. This has become a major concern in the age of growing cyber crimes. Forged signatures and inappropriate images can create nuance around. Hence, it is to utmost importance to differentiate the fake images and stop them from spreading.

There are few different types of manipulations listed below[1]:

- Copy-move: a specific region from the image is copy pasted within the same image.
- Splicing: a region from an authentic image is copied into a different image.
- Removal: an image region is removed and the removed part is then in-painted.

Following is an example of tampered image (copy-move) and authentic image extracted from our dataset:

(a) Authentic (b) Tampered

Fig. 1: Image Forgery Problem

Lately, many approaches have been developed for the feature engineering. The traditional approaches use computer vision libraries for feature extraction step. Recent advances in the field of machine learning have lead to the development of models like CNN for feature engineering and feature extraction. Thereby, improving the accuracy of models. In particular, [2][3] use CNNs to classify images as tampered or authentic and achieve an accuracy of more than 98% for CASIA-1 data set as the tampering is easy to recognise. In this project, we have trained neural network for feature engineering on CASIA-2 data set and testing is done on some part of CASIA -2 dataset as well as on some unseen images picked from another source.

III. Dataset details

The CASIA v2.0 dataset contains 12,622 images. Feature engineering is being on 3000 tampered images picked randomly from CASIA-2 dataset and 4000 authentic images. Out of the 4000 original images, 1500 are the ones which are present in the tampered set. Since, tampering has been done on a single aunthentic image in multiple ways. In this project, we have focused on copy-move forgery. The testing has been done on the 1200 tampered images and 1500 authentic images. We have also used the ground truth of the images which contained the mask of tampered region. The images are of different dimensions in the given dataset.

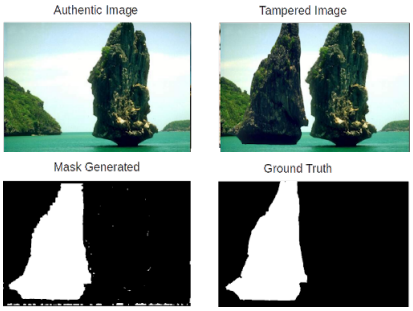


Fig. 2: Manually extracted mask and ground truth mask

Further, the testing has been done on 200 unseen forged images not belonging to CASIA -2 data set and the accuracy has been reported.

IV. Proposed architecture

We have developed a classification pipeline inspired by the work of [2]. While their study achieves high accuracy, the CASIA datasets used to test their network on are manipulated in a way that is easy to recognize by humans.

We have divided the solution into two phases that is Feature extraction and classification. For feature extraction we have implemented the CNN architecture proposed in the paper [2]. This architecture achieves higher accuracy for easy to detect tampered images. We have introduced the novelty in the architecture manipulating some weights in the layers of architecture and hence tried to achieve decent accuracy on the more challenging dataset as well as on unseen images.

A. *CNN architecture*

The CNN has been trained on the patches extracted from:

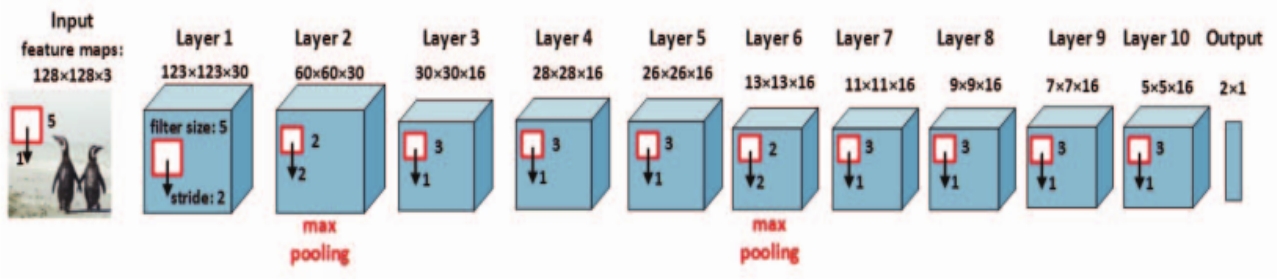


Fig. 3: CNN Architecture

The architecture is composed of 7 convolutional layers, 2 maxpool layers, one fully connected layer and a softmax layer. The softmax layer is used only during the training phase as to create features based on the tampered and authentic images. During the feature extraction phase, the output of fully connected layer was obtained. Xavier initialisation is used for weight initialisation in each of the layer.The kernel size of first 2 layers is 5*5 with filters used as 3 and 30 respectively. A maxpool layer with stride = 2 is embedded after that. Next, there are 7 layers with 16 filters and kernel = 3. Therefore, as the image moves into deep layer, it gets shrinked following : [(n+2p-f)/s]+1 where n= output from previous layer, p = padding, f = filter size in the respective layer and s= stride. Finally, a fully connected layer was added of size 5X5X16 resulting into a single 400 feature vector. In the training phase, softmax layer is used as tampered images were given a label = 0 and authentic images were labelled as 1.

B. *Patch Extraction*

The CNN was trained on the patches extracted from the images. Given that our data set has varying sizes of images so extracted 128X128X3 sized patches from each image. From the tampered images, we have extracted only those patches which contains the tampered region. Since our architecture is heavy enough, so have randomly sampled the patches from each image and took only 7 patches per image for training purpose.

Fig. 4: Training Patches

During the phase of tampered patches extraction, masks were required. We tried two approaches, firstly by creating the masks by writing the code and secondly by using the provided ground truth value from here. Using the ground truth masks has achieved higher accuracy because by creating the masks on our own has introduced noise in the masks leading to the extraction of unsuitable patches from tampered images thereby deteriorating the performance.

C. *SRM Filters*

Taking ideas from[4]. we have used the SRM filters for weight initialisation in the second conv 2D layer. SRM filters are used to extract low noise features, extracts the nearby co-occurrence information as the final features. Along with SRM filters, we have also applied the low pass filters and rotated them for better edge extraction. This has lead to the addition of 30 filters each having kernel size =5. For better performance the permutations of these filters is applied.

$$\frac{1}{4}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 2 & -4 & 2 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{1}{12}\begin{bmatrix} -1 & 2 & -2 & 2 & -1 \\ 2 & -6 & 8 & -6 & 2 \\ -2 & 8 & -12 & 8 & -2 \\ 2 & -6 & 8 & -6 & 2 \\ -1 & 2 & -2 & 2 & -1 \end{bmatrix} \quad \frac{1}{2}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Fig. 5

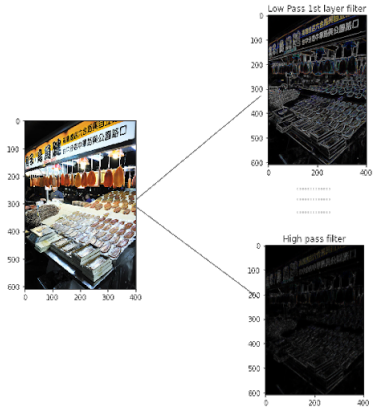


Fig. 6: SRM Filters

D. *Feature Fusion*

After the training phase, the feature extraction of each image was done as follows:

1) Extracting all the patches from the given image by sliding the window of size 128X128X3 and setting the stride to 128.

2) Taking output of the the CNN model from the final fully connected layer, the output is of 400D.
3) Each patch of the image will be having a 400D output, therefore stacking those to form a single feature vector.
4) Feature stacking is done by taking the mean of each value of feature vectors.

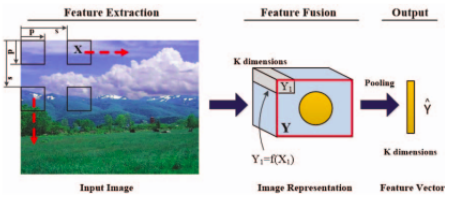


Fig. 7: Feature Fusion

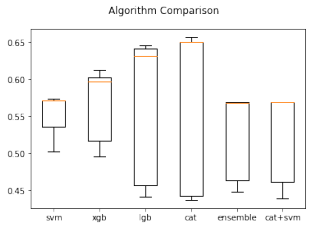
5) Following is the formula used: $\hat{Y}$ [k] = Mean Y1[k]...Yn[k] where Y1[K] is the 400D vector for 1st patch of an image (github).

The resulting 400D feature vector is fed to SVM for training. Similary, for test dataset also the features are extracted and used for predicting the labels.

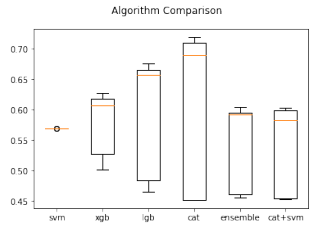
V. Results

We have used pytorch for implementing the architecture of CNN. The performance extracted 400 features was evaluated on different classifiers using 10 fold cross validation and the evaluation metric used is accuracy. The analysis has been done on the following three approaches:

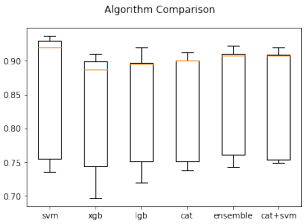
- Tampered regions detected using self extracted masks.
- Tampered regions detected using ground truth masks without data augmentation(4 patches for CNN Training).
- Tampered regions detected using ground truth masks with the data augmentation(8 patches for CNN training)



(a) Self Extracted Mask



(b) Without Data Augmentation



(c) With Data Augmentation

Fig. 8: BoxPlot Analysis of Classifiers

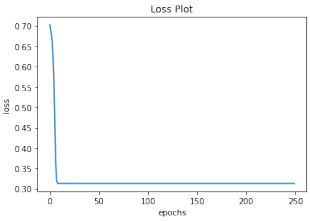


Fig. 9: CNN Training on Augmented Data

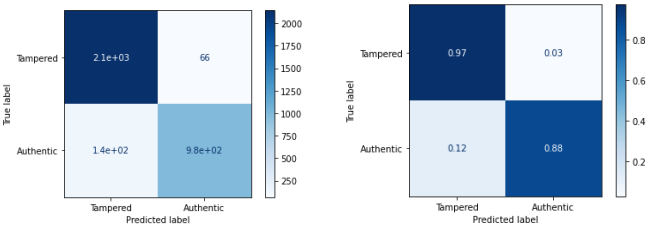
CNN training is done on patches generated by above 3 approaches, on parameters- epochs=250, learning-rate=0.0001, batch-size=200. Min Loss Achieved =0.3 shown in Fig 7. on Data Augmentation approach. After feature extraction, classification is done on 6 classifiers - SVM, CatBOOST, XGBoost, LightGBM, StackingClassifier(layer 1- SVM, CatBoost, XGBoost, LGBM, layer 2- Logistic Regression) , StackingClassifier(layer 1-SVM, CatBoost, layer 2- Logistic Regression). We have also checked accuracy on CASIA1 dataset after training CNN on CASIA2 patches , accuracy reported was 57.8%,57%,62% respectively on the three CNN models. BoxPlot Analysis is shown in Fig.8. Best Validation Accuracy is reported on SVM - 85.7% on tuned Hyperparameter C=1000 , Gamma=0.0001 Fig 10. Final Accuracy achieved 93% on SVM Classifier.

Approach	SVM	XGBoost	LGBM	CatBoost	Stack1	Stack2
Mask	55.63	53.43	52.11	54.7	53.65	54.37
Without Data Augmentation	56.97	55.43	54.56	56.92	54.55	54.67
With Data Augmentation	85.55	82.73	83.55	84.12	83.55	84.7

Fig. 10: Validation Accuracies

Approach	SVM
Mask	60.4
Without Data Augmentation	62.67
With Data Augmentation	93

Fig. 11: Accuracies on SVM classifier

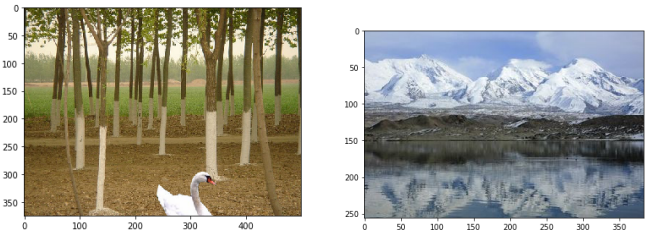


(a) (b)

Fig. 12: Confusion Matrix of Best Approach(3)

VI. Inferences

The confusion matrix is result of the model which has achieved the highest accuracy. It can be seen that most of



(a) (b)

Fig. 13: Misclassified Images

the tampered images are recognised appropriately.

Following are the few misclassified images by the best model obtained. The reason we can understand behind it that the mountains image of mountains has a very little tampering therefore the tampered regions were not properly detected. In the first image of forest, proper smoothening has been done which failed our model to detect the tampered region.

VII. Individual Contribution

After project inception, we read up on research papers and medium articles to obtain information on how we can do CNN based Forgery detection route for the project along with our future plans. The writing of project proposals, presentations and this final report has been a joint effort. We tackled the this problem with two different approaches and each team member took one approach. The model training with ground truth images has been done by Bhakti and the model training along with self extracted masks was done by Anmol. While we coded on separate solutions to the same end result, we always discussed the problems we faced and the approaches we should make in our respective problems. Each team member is aware of the intricacies of their own problem and simultaneously has a good grasp on the whole project as a whole.

References

[1] Peng Zhou, Xintong Han, Vlad I Morariu, and Larry S Davis. Learning rich features for image manipulation detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1053–1061, 2018.

[2] Yuan Rao and Jiangqun Ni. A deep learning approach to detection of splicing and copy-move forgeries in images. In 2016 IEEE International Workshop on In- formation Forensics and Security (WIFS), pages 1–6. IEEE, 2016.

[3] Belhassen Bayar and Matthew C Stamm. A deep learning approach to universal image manipulation de- tection using a new convolutional layer. In Proceed- ings of the 4th ACM Workshop on Information Hiding and Multimedia Security, pages 5–10. ACM, 2016.

[4] P. Zhou, X. Han, V. I. Morariu and L. S. Davis, "Learning Rich Features for Image Manipulation Detection," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, 2018, pp. 1053-1061, doi: 10.1109/CVPR.2018.00116.

[5] https://medium.com/@vvsnikhil/image-forgery-detection-d27d7a3a61d