Tamper Detection for Images using Deep Learning Methods

Metarya Ruparel
msr9732@nyu.edu

Safwan Mahmood
sm9453@nyu.edu

Abstract

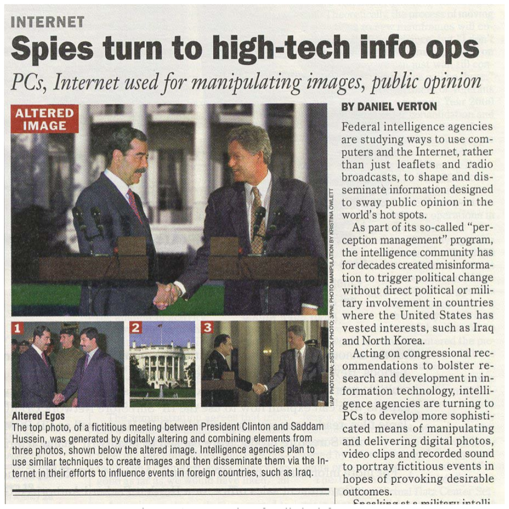
Digital images are easy to manipulate and edit due to availability of powerful image processing and editing software. Nowadays, it is possible to add or remove important features from an image without leaving any obvious traces of tampering. As digital cameras and video cameras replace their analog counterparts, 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 main topic of this project. Throughout the years, various computer vision and deep learning approaches have been proposed to solve the issue. In particular, a few of the CNN architectures suggested, manage to predict images with an accuracy of more than 96%. That said, the images used in those studies are easily recognized by humans. This raises a crucial question: how would CNNs perform on more challenging samples? In this study, we develop a CNN network inspired by a previous study and answer this question by analysing various approaches to measure performance on CASIA2 and MICC datasets. We also measure the effect of a data augmentation technique and different hyperparameters on classification performance. Our experiments show that dataset difficulty can significantly influence the performance obtained.

Keywords—Image Tampering, Machine Learning, CNN, Deep Learning, Neural Networks.

1. Introduction

The availability of powerful digital image processing programs, such as PhotoShop, makes it relatively easy to create digital forgeries from one or multiple images. An example of a digital forgery is shown in Figure 1. As the newspaper cutout shows, three different photographs were used in creating the composite image: Image of the White House, Bill Clinton, and Saddam Hussein.

Manipulated images are presented in such a manner that it is almost impossible to visually discriminate forged data from the original data. It has a noteworthy role in uploading and downloading images to those social media like SNS


ALTERED IMAGE
1
2
3

INTERNET

Spies turn to high-tech info ops

PCs, Internet used for manipulating images, public opinion

1/AP PHOTO/INA; 2/STOCK PHOTO; 3/PNI; PHOTO MANIPULATION BY KRISTINA OWLETT

Altered Egos
The top photo, of a fictitious meeting between President Clinton and Saddam Hussein, was generated by digitally altering and combining elements from three photos, shown below the altered image. Intelligence agencies plan to use similar techniques to create images and then disseminate them via the Internet in their efforts to influence events in foreign countries, such as Iraq.

BY DANIEL VERTON

Federal intelligence agencies are studying ways to use computers and the Internet, rather than just leaflets and radio broadcasts, to shape and disseminate information designed to sway public opinion in the world's hot spots.

As part of its so-called "perception management" program, the intelligence community has for decades created misinformation to trigger political change without direct political or military involvement in countries where the United States has vested interests, such as Iraq and North Korea.

Acting on congressional recommendations to bolster research and development in information technology, intelligence agencies are turning to PCs to develop more sophisticated means of manipulating and delivering digital photos, video clips and recorded sound to portray fictitious events in hopes of provoking desirable outcomes.

Speaking at a military intelli

Figure 1. Example of a digital forgery.

(Social Network Service), Facebook, WhatsApp, and Instagram. Due to this, it is very challenging to differentiate between the actual image and the tampered image which is created by using available tools. The fields like forensics, industrial photography, e-commerce, and medical imaging, and substantiating the uniqueness of images is a major challenge. Detecting traces of manipulation of the image is an exciting task and relatively difficult to declare images are trustworthy. Hence, the determination in enhanced image manipulation detection cannot be ignored. Traditional approaches for image manipulation detection typically uses handcrafted structures.

The major problem with these methods is the procedures can categorize a particular type of manipulation by recognizing a definite feature in that image. Image manipulation can be found in several fields like Scientists and researchers manipulate images for their work to get published; medical images are tampered to misrepresent the patients diagnostics, photo and yellow journalists use the trick for creating and giving dramatic effect to their stories, politicians, lawyers, forensic investigators use tampered images to direct the opinion of people, court or law to their favour and so on. Retouching, splicing, copy-pasting, cropping, cloning

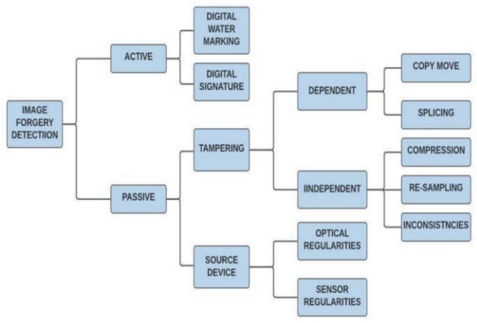


Figure 2. Classification of Image forgery detection approaches [9]

etc are some of the popular techniques used for image manipulations. In additions to these techniques there also exists a wide range of Steganographic methods those use this popular digital media for secret data transmission. The most common categories of non-natural distortions of digital images are splicing [3], copy-move [1,2], and resampling [4]. All of them are used to conceal the geometric transformation existing in the digital image. The copy-move type of distortion means photocopying a section of a digital image, presenting any distortion into this fragment, and implanting the modified fragment into another area of the same image. The most common type of imitation alterations is the resampling-geometric conversion of image chunks and implanting those into other images. The other usually used type of distortions is splicing and this will use fragments of diverse digital images to create a detailed distortion of an existing one or a new image. And finally, an alternative way that attackers use is JPEG compression [5]. In this, after implanting any info in the JPEG file and recompression, there are native transformations in the belongings of JPEG compression. The pronounced procedures of implanting alterations are the most common today, as shown by the enormous amount of publications aimed at evolving solutions to detect such attacks. To find an answer to these issues, the researchers have suggested some methodologies that can be categorized into Active and Passive technologies [6-8] as shown in Figure.2.

In this project, we aim at detecting these forgeries. Firstly, we come up with an approach inspired by [16] involving fine tuning pre-trained models like VGG16,VGG19 and DenseNet on MICC-F220 and MICC-F2000 datasets. To extend the approach, we use feature extraction on pre-trained models mentioned by tapping into the fully-connected layer. We then classify the feature representation of the images using a SVM, achieving around 70-75% validation accuracies. We infer that this is due to restricted amount of datapoints in the dataset and the difficulty to recognize the tampering.

Secondly, we develop a CNN network architecture which is inspired by [10] that achieves an impressive accuracy of nearly 97% on the CASIA v2.0 dataset. Finally, we analyze the effect of hyper-parameter tuning and data augmentation on the network performance.

2. Literature Survey

Recent image tampering work shows using deep learning techniques such as CNN aid in improving tampering detection accuracies. However, existing tampering detection methodologies predominantly focused on identifying a particular type of manipulations such as splicing, re-sampling, copy-move, etc. As a result, some method works well for detecting one kind of attack; however, fails to detect another kind of hybrid attack such as introducing re-sampling attack of copy-move tampered segment. Along with that, it is practically a difficult task to know the tampering type in advance. Then, segmenting only the tampering region is very difficult; especially when there exist multiple forgeries of similar patterns within an image. CNN in object segmentation have attained the very good result; CNN extracts hierarchical feature from the different level to segment meaningful shape of respective objects.

In [11], offered a CNN-based method to detect interfering were the hints left by various camera models, which abstracts the features regarding the camera model from digital image reinforcements. The Clustering system is employed to study the mined structures and this will classify the digital image as either artificial or not. In [14], the author suggested using noise residual structures for image manipulation detection with localization. CNN is used meant for mining the noise remaining centered landscapes of the digital image and the SVM is used for classifications.

In paper [10], suggested CNN for digital image forgery detection in copy-move and another one image splicing. The primary convolution layer on CNN is involved in preprocessing operations to search for concerns formed by altering processes. Here, the CNN was trained with characterized path illustrations from training images. After this pre-trained CNN was applied on trial images and for the detection of tampering SVM classifier is used.

Bi et al. [13], are recommended a CNN-based method called RRU-Net (Ringed Residual U-Net,) where it is an end-to-end image segmentation network, for digital image splicing detection. The RRU-Net goals to develop the learning approach of CNN over recollection and association with the human brain mechanism. The outstanding propagation is engaged to remember the input feature info to unravel the ruin issue within a deeper network. Finally, the remaining response merges the response feature info to discriminate against the original and fake regions. This RRU-Net tested on two very popular datasets i.e., CASIA and COLUMBIA. Wang et al. [15], A novel model is employed to detect and also to locate the image manipulations. This novel method

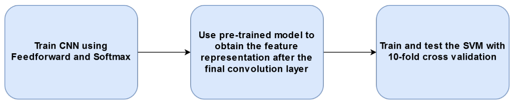

Train CNN using Feedforward and Softmax
Use pre-trained model to obtain the feature representation after the final convolution layer
Train and test the SVM with 10-fold cross validation

Figure 3. System Pipeline

was tested on two datasets i.e., Columbia and Cover. This method was skilled to find equally a copy-move and also splicing falsifications.

Amit Doegar, et al, [16], is proposed to utilize the CNN based pre-trained AlexNet model's deep structures without devoting much time to training. The suggested approach also] exploits the SVM as a classifier. The performance of these deep features mined from that proposed model is satisfactory, even in the occurrence of geometrical and rotational transformation.

3. Technical Approach

One of the main tasks of our study is to create a pipeline [Figure 3][1] that is able to recognize tampered from authentic images. For that reason, we took inspiration from the architecture proposed by Y. Rao et al [10]. In particular, they propose a CNN that is used as a feature extractor that takes an image patch

$$X \in R^{p\times p} \tag{1}$$

as input and outputs a feature representation

$$Y = f(X) \in R^{K} \tag{2}$$

, where K is the number of dimensions. This feature representation is then fed into an SVM classifier that predicts whether the features correspond to an original or tampered image. The network architecture and the procedure via which the CNN and the SVM are trained are detailed in the following sections.

3.1. Network architecture

A Convolution Neural Network (CNN) is a category of deep neural networks mainly used for image analysis. The basic structure of a CNN contains several convolutional layers followed by a fully connected layer(s) and a softmax classifier. Each convolutional layer is composed by a convolution, a non-linear activation, and a pooling. The input and the output of convolution layers are arrays which are called feature maps. If we denote $F^n(X)$ the feature map in the convolution layer n, with the kernel and bias defined by W^n and B^n respectively, the convolutional layer can be computed using the following formula [10]:

$$F^n(X) = \text{pooling}(f^n(F^{\text{n-1}}(X) * W^n + B^n)) \tag{3}$$

[1]GitHub Link: Deep-Learning-Based-Tamper-Detection-on-Images

where $F^0(X) = X$ the input data, $f^n(\cdot)$ a non-linear activation function applied to each element of its input and $pooling(\cdot)$ the pooling operation which reduces the dimensions of the data via a max or mean operation.

The architecture implemented in this study is a CNN comprising of nine convolutional and two max-pooling layers as shown in Figure 4. The input size of the network is a 128x128x3 patch, where 3 represents the RGB color channels. The first two convolutions have a 5x5 kernel size and output 3 and 30 kernels respectively. These layers are followed by a pooling operation with a 2x2 filter. The next eight layers have 16 kernels, with a 3x3 kernel size for the convolutions and a 2x2 filter for the max pooling. These seven convolutions have stride one whereas the pooling operation has stride two. Additionally, every convolutional layer uses the ReLU activation function. It has to be mentioned that local response normalization is applied to every feature map before the pooling operation to improve generalization [10]. In particular, the central value in each neighborhood is normalized by the values of its three neighboring channels.

3.2. Network weight initialization

All convolutional layers, except second layer, in our model are initialised using Xavier initialization [12]. The reason behind such an initialisation is that it avoids high values or values that vanish to zero. This is achieved by keeping the variance the same with each passing layer. The kernels of the second convolutional layer are initialized using thirty SRM high-pass filters proposed in [17]. We have eight first, four second and eight third order SRM filters. We observed that the tampered images could have abnormal edges which do not blend in with their surroundings. As a result, the change between one-pixel-region to the other would be drastic. Thus to enable our model to better detect forged images, apart from the aforementioned, two square high pass filters of size 3x3 and 5x5 are used which can detect pixels that have different values from their neighbors. Finally, we use eight edge detection filters 3x3 and 5x5 that are the best in finding edges. We also shuffle the filters on all channels leading every RGB channel to have a different filter in every dimension. This has been empirically proven to increase performance due to its regularization effect.

3.3. CNN Training

We wanted to focus on the local regions of the artifacts and learn to recognize them, hence we used image patches extracted from the dataset used. The size of the extracted patches is 128x128x3, meaning that there is a 128x128 patch for every color channel. The extraction was performed by applying a patched-size sliding window with stride equal to eight for the whole image. Following that, the tampered patches are discriminated from the non- tam-

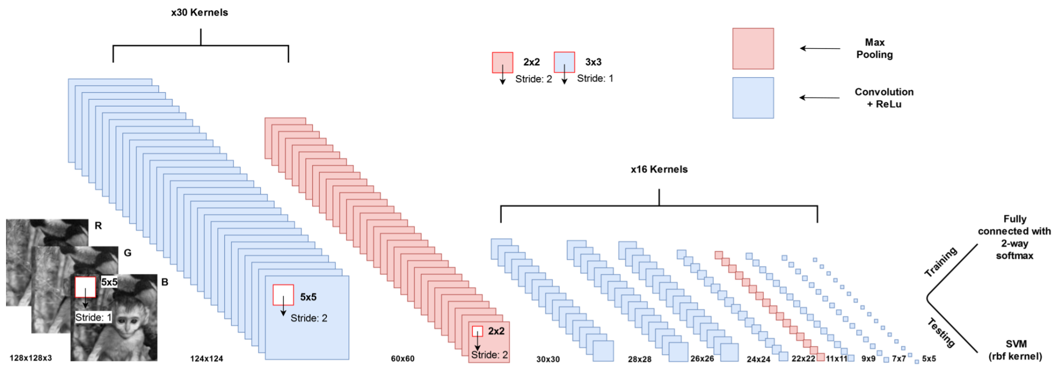


Figure 4. Network Architecture

pered ones. As far as the tampered patches are concerned, we compare each patch with the equivalent patch (from the same region of the image) of the mask of this image and keep the ones that contain part of the tampered region. Training the CNN with a huge amount of extracted patches would be computationally expensive, hence we randomly choose a couple of patches and use them to train the CNN. When it comes to the non tampered patches, we apply the same technique but now on the equivalent authentic image and randomly select two of these patches. Finally, in order generalization better and avoid overfitting, we augment the patches extracted by rotating them four times by a step of 90 degrees.

Following the aforementioned procedure, the patches are fed into the CNN which extracts a 5x5x16 (400-D) feature representation of the patches. These features are then passed to a fully-connected layer with a 2-way softmax classifier that uses dropout.

3.4. SVM Training

We extract every possible p × p patch from both the original and the tampered images using a sliding-window with stride s to scan the whole image. This process results in n new patches per image which are passed through the CNN resulting in n feature representations Y_i (400-D). That said, these representations need to be fused into a single $\hat{Y}[k]$ representation for each image before being passed as an input to the SVM. Similarly to [10], max or mean pooling is applied on each dimension of Y_i over all the n patches extracted from each image:

$$\hat{Y}[k] = \text{Mean or Max}\{Y_1[k]...Y_n[k]\} \qquad (4)$$

where k ∈ [1, 400] dimensions. The resulting 400-D feature vector is then used by the SVM to classify images either as original or tampered.

4. Experiments

In order to run an experiments, we first extract the CNN training patches according to Section 3.3 and use them to train the neural network. Then, we extract new patches and the corresponding image features with the use of mean fusion as described in Section 3.4. Having obtained the features, the SVM is trained and tested using stratified 10-fold cross-validation. We resorted to computing our own masks using the tampered and authentic images. As for the hyper-parameters used in our experiments, the patches required to extract the image features are obtained using a stride s of 128 for CASIA v2.0. All the CNNs are trained for 250 epochs using the cross-entropy loss and optimizing the network via Stochastic Gradient Descent (SGD). The SGD implementation uses momentum equal to 0.99, a weight decay of 5×10^{-4} and a decaying learning rate that is reduced by 10% for every ten epochs. the batch size and the initial learning rate were tweaked accordingly for each experiment since they greatly influenced the network's ability to train. The network training was carried out on our local machines, and required a considerable amount of time to be trained. The SVM uses RBF kernel for every run and uses optimized values of C and γ to better regularize the model and avoid over fitting. The optimized values were chosen based on a exhaustive grid search. We used the following values in grid search $C = [0.001, 0.01, 0.1, 1, 10, 100, 1000]$ and $\gamma = [0.001, 0.0001]$.

We will discuss the results of the different models, the affect of hyper-parameters, as well as how the complexity of the dataset affects the accuracy of the model. The hyper-parameters chosen for each different dataset are presented in Table 1.

Dataset	Batch Size	LR
CASIA v2.0	200	0.0005
CASIA Aug	128	0.001
CASIA v2.0	200	0.0001

Table 1. Hyperparameters

CASIA v2.0	Auth. - Pred	Tamp. - Pred
Authentic - Actual	1,308	193
Tampered - Actual	15	1,013

Table 2. Confusion matrix - CASIA v2.0

4.1. Models

We used the network architecture described in Section 3 to train four different networks and compare their classification performance. More specifically, we trained a network with the patches from CASIA v2.0, both with augmented (four rotations) and non-augmented data, using different Learning Rate and Batch Size. The training loss for each of the four aforementioned configurations is depicted in Figure 5a. We see that loss has sharply decreased following the first epoch and already reached its minimum value after roughly 50 epochs. This behavior suggests that we could have applied early stopping. We notice that data augmentation did help the model and learning rates of 0.001 and batch size of 128 worked most effective.

We compare the accuracies in Figure 5b of various tweaks on the model as discussed above. We notice that data augmentation produced the best accuracy.

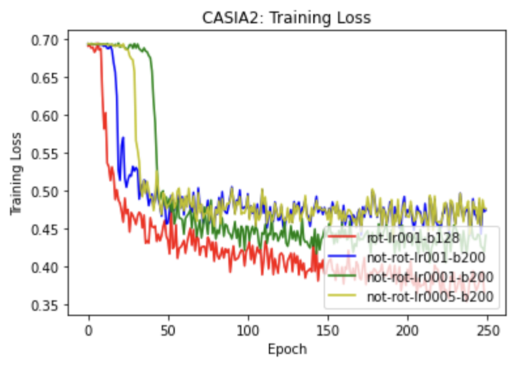
4.2. Dataset Complexity

In this section, we make a note of the classification performance.

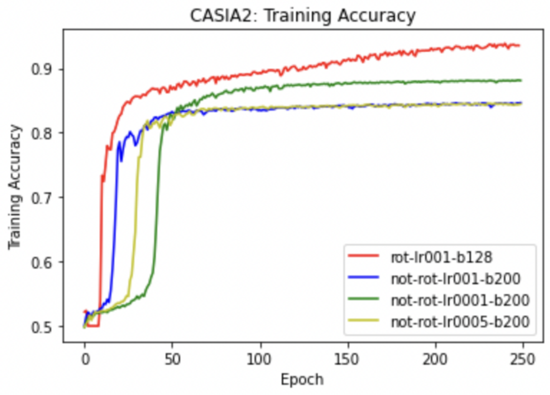
First, we trained the CNN by using the CASIA v2.0 dataset with an initial learning rate of 0.0005 and a batch size of 200 images. The best SVM hyperparameters that we trained on were $C = 1$ and $\gamma = 0.0001$. The previous settings resulted in a classification accuracy of 92.54±3%. Moreover, the corresponding confusion matrix was computed using a random 80-20 split and can be found in Table 2. In particular, the SVM managed to correctly classify 1,013 tampered and 1,308 original images, while it misclassified only 15 tampered images and 193 original images.

Confusion matrix indicates that with this network architecture we obtain more FP than FN. However, given the nature of the image forgery detection problem we cannot conclude which of the two is more important since it depends entirely on the case study.

Following that, we performed the same tests using the augmented dataset, where the images are rotated four times by a step of 90 degrees. The use of the augmentation improved classification accuracy. In particular, for CASIA v2.0, accuracy increased from 92.34±3.5% to 96.74±1.2%.



(a) Training loss on CASIA dataset.



(b) Training Accuracies on CASIA dataset.

Figure 5. Comparing affects of hyperparameters

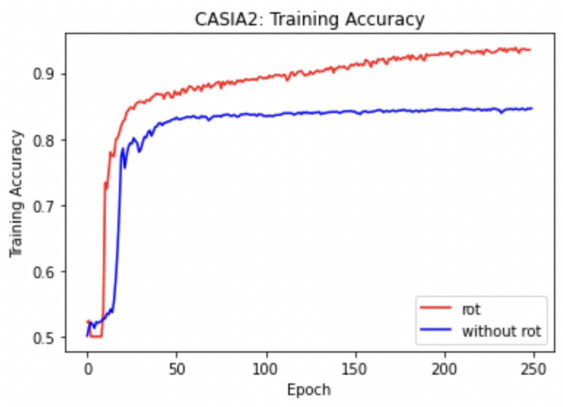
Data Augmentation	CASIA2
With	96.74 ± 1.2%
Without	92.34 ± 3.5%

Table 3. Augmentation effect

4.3. Hyperparameters effect

In this section, we discuss the effect of different hyper parameters on our model. To investigate the effect of the CNN learning rate in the system performance, we train three networks with different learning rates. We used the patches extracted from CASIA2 without data augmentation. The results for learning rate values are demonstrated in Table 4.

The next parameter we experimented with, is the method we use to combine the values of all the patches for each of the 400 features, i.e, mean/max fusion as explained in Section 3.4. We used the augmented version of CASIA2.



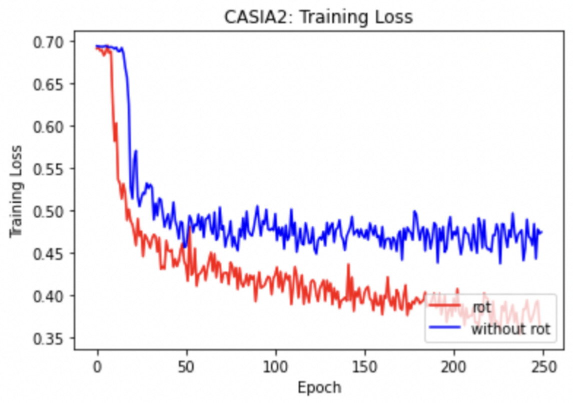


Figure 6. Comparing affects of augmentation

LR	CNN accuracy (%)	Test accuracy (%)
0.0001	88.27	92.42 ± 3.79
0.0005	84.47%	92.54 ± 3.81

Table 4. Effect of Learning rate

Fusion method	Test accuracy (%)
Mean	96.82 ± 1.19
Max	96.90 ± 0.40

Table 5. Effect of Mean/Max Fusion

As we can observe, the difference in terms of accuracy is negligible (0.08%).

Finally, we experimented with different stride sizes while extracting the patches in testing phase. We used the augmented patches of CASIA2 and the model hyperparameters that provided the best results so far. We compared 64 and 128 stride values. Based on the results above, we can conclude that these two stride values do not influence the performance of the classifier significantly since the accuracy and standard deviation are almost the same for both cases.

Stride size	Test accuracy (%)
64	96.83 ± 0.56
128	96.90 ± 0.40

Table 6. Effect of Strides

5. Discussion

Given that neural networks are known to require a lot of data to yield a high accuracy, this could be part of the reason for the performance difference noticed with our first implemenation with MICC and second implementation with CASIA2. Moreover, the image size of the two datasets is significantly different.

Another interesting observation in our study is that data augmentation does indeed increase the performance of the network regardless of the dataset used. This finding is in line with past studies which use such techniques to artificially create more training data. However, designers have to also weight the extra time and cost required to train the CNN against the performance increase achieved.

Moving on to the system hyperparameters, we realized first hand their crucial importance on the classification performance. What became apparent during our efforts to train the neural networks is that without selecting the proper hyper-parameters, the CNN either initially converges and then diverges or that it does not converge at all. To elaborate, settings such as batch size, learning rate and the use of dropout created serious issues when trying to train the CNN since small changes in their values resulted in different behaviors. On the contrary, the stride and the fusion method selected have an insignificant effect in the classification accuracy. Moreover, the hyperparameters of the SVM were also shown to significantly influence the classification performance (Refer to tables 4,5,6).

6. Future Work

The CASIA v2.0 CNNs trained underlined the need to implement early stopping during the network training. To be more specific, all of them converged fairly quickly and plateaued at a specific cost value, suggesting that the use of early stopping would save precious training time/cost while achieving the same or even better performance by avoiding overfitting. Over and above, the use of two datasets with varying difficulty but similar sample and image sizes would assist in validating the results presented in this study. Another option would be to combine different datasets with forged and original images to create a larger training dataset for the CNN. As a result, more data with varying underlying distributions would be available, thus improving the generalization ability of the CNN model. That said, the differences in image sizes and image tampering techniques pose a challenge in creating a ”one size fits all” network that can learn the feature representations for all types of sample variations. Last but not least, it would be interesting to look into ways to extend CNNs such that they can detect tampered images irrespective of the manipulation technique applied to them. An example of one such study is the one performed by [18].

7. Conclusion

In this work we experimented with using a CNN in the image forgery detection task. More specifically, we used a CNN network to extract features from two datasets of varying difficulty, namely CASIA v2.0 and MICC using different approaches. Furthermore, the extracted features in the second implemented were then used to train and test an SVM, achieving an accuracy of nearly 97% on CASIA v2.0. These results validate our intuition that the classification performance decreases the more challenging the samples are. What is more, our study has shown that image tampering can be detected with an accuracy of more than 84% even if done by professionals. However, according to our findings the implemented architecture does not easily generalize to datasets with different underlying distributions. To conclude, while there is surely a lot of work still to be done in the image forgery detection domain, we believe that neural networks will be able to detect tampered images regardless of their difficulty in the near future.

8. References

[1] R. Dixit, R. Naskar, and A. Sahoo. Copy-move forgery detection exploiting statistical image features, 2017 International Conference on Wireless Communications, Signal Processing, and Networking (WiSPNET), Chennai, 2017, pp. 2277-2281.

[2] A. J. Fridrich, B. D. Soukal, and A. J. Luk s, Detection of copy-move forgery in digital images, in Proceedings of Digital Forensic Research Workshop, Citeseer 2003.

[3] B. Patil, S. Chapaneri, and D. Jayaswal. Improved image splicing forgery localization with first digits and Markov model features, IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS), Srivilliputhur, 2017, pp. 1-5.

[4] A. C. Popescu and H. Farid. Exposing digital forgeries by detecting traces of re-sampling. IEEE Trans. Signal Processing, vol. 53, no. 2, pp. 758–767, 2005.

[5] Bo Liu, Chi-Man Pun, and Xiao-Chen Yuan. Digital Image Forgery Detection Using JPEG Features and Local Noise Discrepancies.

[6] Arun Anoop M, ”Image forgery and its detection: A survey,” 2015 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), Coimbatore, 2015, pp. 1-9.

[7] Nikhil Kumar P. Joglekar1, Dr. P. N. Chatur. A Compressive Survey on Active and Passive Methods for Image Forgery Detection, International Journal Of Engineering And Computer Science ISSN:2319-7242 Volume 4 Issue 1 January 2015, Page No. 10187-10190.

[8] Gajanan K. Birajdar, Vijay H. Mankar. Digital image forgery detection using passive techniques: A survey. Digital Investigation. https://doi.org/10.1016/j.diin.2013.04.007.

[9] Nor Bakiah Abd Warif, Ainuddin Wahid Abdul Wahab, Mohd Yamani Idna Idris, Roziana Ramli, Rosli Salleh, Shahaboddin Shamshirband, Kim-Kwang Raymond Choo, Copy-move forgery detection: Survey, challenges and future directions, Journal of Network and Computer Applications, Volume 75, 2016, Pages 259-278, ISSN 1084-8045, https://doi.org/10.1016/j.jnca.2016.09.008

[10] 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.

[11] L. Bondi, S. Lameri, D. Güera, P. Bestagini, E. J. Delp and S. Tubaro. Tampering Detection and Localization Through Clustering of CameraBased CNN Features, 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, 2017, pp. 1855-1864, DOI: 10.1109/CVPRW.2017. 232.

[12] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the thirteenth international conference on artificial intelligence and statistics, pages 249–256, 2010.

[13] X. Bi, Y. Wei, B. Xiao, and W. Li, ”RRU-Net: The Ringed Residual UNet for Image Splicing Forgery Detection,” 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Long Beach, CA, USA, 2019, pp. 30-39, DOI: 10.1109/CVPRW.2019.00010.

[14] D. Cozzolino and L. Verdoliva. Single-image splicing localization through autoencoder-based anomaly detection, 2016 IEEE International Workshop on Information Forensics and Security (WIFS), Abu Dhabi, 2016, pp. 1-6, DOI: 10.1109/WIFS.2016.7823921.

[15] Xinyi Wang, He Wang, Shaozhang Niu, and Jiwei Zhang. AIMS/MBE. http://www.aimspress.com/journal/MBE 16(5): 4581–4593.

[16] Amit Doegara, Maitreyee Dutta, Gaurav Kumar. CNN based Image Forgery Detection using pre-trained AlexNet Model. Proceedings of International Conference on Computational Intelligence & IoT (ICCIIoT) 2018. https://www.ssrn.com/link/ijcciiot-pip.html.

[17] Jessica Fridrich and Jan Kodovsky. Rich models for steganalysis of digital images. IEEE Transactions on

Information Forensics and Security, 7(3):868–882, 2012.

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

[19] Manjunatha. S and Malini M Patil Deep learning-based Technique for Image Tamper Detection. In Proceedings of the Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV 2021).

[20] MICC Dataset - http://lci.micc.unifi.it/labd/2015/01/copy-move-forgery-detection-and-localization/

[21] CASIA2 Dataset - https://ieeexplore.ieee.org/document/6625374