Image frogery detection

Abstract

With the development of image manipulation software,image tampering is easy.Various solutions including computer vision and deep learning approaches have been proposed to solve this issue.We have applied deep learning technique using CNN to separate the original image from the tempered one.We have measured the performance for different hyperparameter and data augmentation techniques on classification.Huge data set is required for getting higher accuracy.Our experiment s shows that the signifance of data sets for the performance.

1 Introduction

Nowadays, we have very huge amount of digital images are generated using smartphones and tablets.With the fast development of digital image processing technology and the popularity of digital camera, editing or tampering a digital image becomes much easier even for an inexperienced forger with the aid of some user friendly photoediting softwares.These digital images can be manipulated such a way that it is impossible to humans to detect using softwares.These manipulations are widely used in social medias like twitter and facebook. e.g., Adobe Photoshop. In the past few decades, doctored photographs are appearing with a growing frequency and sophistication and various digital forgery tools seemingly emerge in an endless stream.

This images can easily manipulated using wide variety of software's available in the market.These digital images can be manipulated such a way that it is impossible to humans to detect using softwares.These manipulations are widely used in social medias like twitter and facebook .Manipulations like skin tone change and contrast adjustment will not be harmful but there are others that could create serious business or political issues.

Some of the most common manipulations are

• Removal: an image region is removed and the removed part is then in-painted

• Splicing: a region from an authentic image is copied into a different image

• Copy-move: a specific region from the image is copy pasted within the same image

among these, splicing and copy-move are the most common ones that manipulate the images in a way to be hardly perceived by human perceptual system. Therefore, effective detection of these two kinds of forgeries is of great importance for digital image forensics.

More recently, the deep learning based approach has also found applications in passive image forensics. In [, a CNN model was trained for median filtering detection. However, Ying et al. [15] showed that the conventional deep learning framework may not be directly applied to image tampering detection, this because, with elaborated designed tools, the forgery images tend to closely resemble the authentic ones not only visually but also statistically. Therefore, they adopted the

wavelet features of images as input of their deep autoencoder. Motivated by the similar observation, Bayar et al. proposed a new convolutional layer in their CNN model to learn prediction error filters with constraint to discover the traces left by image manipulations.

2 Proposed Method

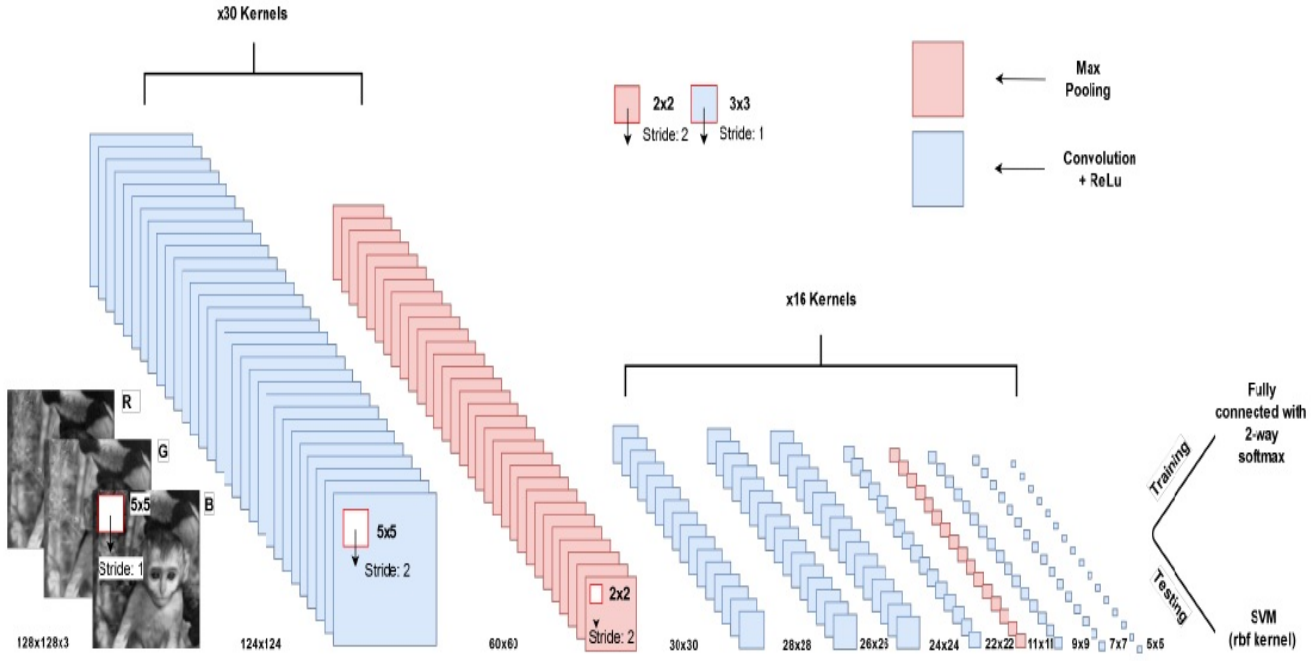


Figure 1: The architecture of 10 layer CNN

One of the main tasks is to create a pipeline that is able to classify images to tampered and authentic images. This image forgery detection approach can automatically learn feature representations based on deep learning framework.
The main step include:

2.1 Network Architecture

We first train a supervised CNN to learn the hierarchical features of tampering operations (splicing and copy-move) with labeled patches (pxp) from the training images. The first convolutional layer of the CNN serves as the pre-processing module to efficiently suppress the effect of image contents. 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 Fn(X) the feature map in the convolution layer n, with the kernel and bias defined by Wn and Bn respectively, the convolutional layer can be computed using the following formula The convolutional layer can be computed using the following formula
$F^n(X) = \text{pooling}(f^n(F^{n-1}X)W^n + B^n))$

2.2 Network weight initialization

Apart from the second convolution, every other convolutional layer in our network is initialized using Xavier initialization.The main concept 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.second convolutional layer are initialized using thirty SRM high-pass filters. The SRM filters used are eight first, four second and eight third order filters.

2.3 CNN training

We then extract the features for an image with the pre-trained CNN on the basis of pxp patch by applying a patch-sized sliding-window to scan the whole image. The generated image representation is then condensed by a simple feature fusion technique, i.e. regional pooling, to obtain the final discriminative feature.In order to train the aforementioned CNN architecture in a way that it can focus on the local regions of the artifacts and learn to recognize them, image patches have to be 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 nontampered 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, as demonstrated in Figure 5. Moreover, we only keep two random tampered patches per image, as training the CNN with a huge amount of extracted patches would be computationally expensive. When it comes to the nontampered patches, we apply the same technique but now on the equivalent authentic image and randomly select two of these patches. Finally, in order to improve the generalization ability of CNN and avoid overfitting, we augment the patches extracted by rotating them four times by a step of 90 degrees.

2.4 SVM Training

Finally, a SVM classifier is trained based on the resulting feature representation for binaryAfter the training of the CNN network, the next step is to train the SVM classifier. For that purpose, we extract every possible p X 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 Yi (400-D). That said, these representations need to be fused into a single Y [k] representation for each image before being passed as an input to the SVM. Similarly to [9], max or mean pooling is applied on each dimension of Yi over all the n patches extracted from each image:

max or mean pooling is applied on each dimension of Yi over all the n patches extracted from each image:

$\widehat{Y}$ = Mean or Max {Y1[k] . . . Yn[k]}

3 Results

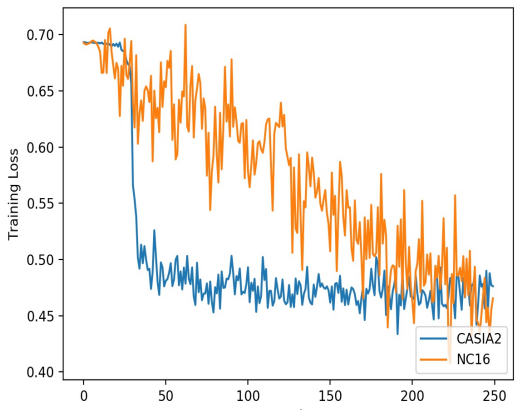
We have trained the CNN by using the CASIA v2.0 dataset and NC16 dataset with an initial learning rate of 0.0005 and a batch size of 200 images. The best SVM hyperparameters that we trained on were C = 1 and = 0:0001. The confusion matrix was computed using a random 80-20 split and can be found in the below Table .

Confusion matrix		
CASIA v2.0	predicted authentic	predicted tempered
Actual authentic	1203	185
Actual tempered	11	998

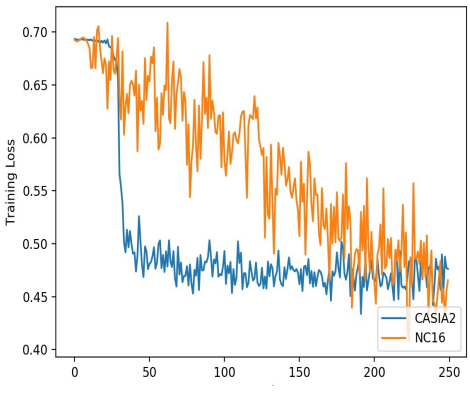
we trained the feature extractor (CNN) with the second dataset(NC16) with a learning rate of 0.001 and a batch size of 32 images. The optimal SVM parameters chosen after the grid search were C = 100 and γ= 0:001. An interesting thing is that the classification accuracy of our system on the more difficult NC16 is 10matrix in a similar fashion to CASIA v2.0, the output of table is obtained as

Confusion matrix		
CASIA v2.0	predicted authentic	predicted tempered
Actual authentic	88	24
Actual tempered	13	100

we trained a network with the patches from each of the two datasets, CASIA v2.0 and NC16, both with augmented (four rotations) and non-augmented data.The training loss for each of the four aforementioned configurations is depicted in the Figure below.



(a) Non-augmented data - CASIA v2.0 vs NC16



(b) Augmented data - CASIA v2.0 vs NC16

Figure 2: Training loss comparison of CASIAv2.0 vs NC16 with non-augmented and augmented data

First off, to investigate the effect of the CNN learning rate in the system performance, we train three networks with different learning rates.The results for learning rate values of 0.0001, 0.0005 and 0.001 are demonstrated in Table

Learning rate comparison - CASIA v2.0(C = 1 and $\gamma = 0.0001$)		
Learning Rate	training accuracy of CNN	test accuracy
0.0001	88.14	92.42
0.0005	84.53	92.54
0.001	84.65	92.35

4 advantages

- The proposed image frogery detection gives better accuracy compared to the traditional computer vision approaches.
- It can be improved further by tuning different hyperparameters.

5 disadvantages

- The accuracy depends on the data sets we have used . some datasets gives better results than the other.
- Huge data sets are required for getting better accuracy
- Time taken for feature extraction using CNN is higher.

6 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 the datasets namely CASIA v2.0 an. The extracted features were then used to train and test an SVM, achieving an accuracy of 96.82 percentage 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 percentage 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.