STEGANALYSIS OF IMAGES

Jaidev Chittoria-181IT119
Information Technology
NITK, Surathkal
Karnataka, India
jaidev.chittoria02@gmail.com

Shraddha Gole - 181IT145
Information Technology
NITK, Surathkal
Karnataka, India
shraddhagole10sg@gmail.com

Naman Vijayvargiya-181IT129
Information Technology
NITK, Surathkal
Karnataka, India
namanvj26@gmail.com

Abstract—**Steganography is the art of hiding the fact that communication is taking place, by hiding information in other information. Many different carrier file formats can be used, but digital images are the most popular because of their frequency on the Internet. For hiding secret information in images, there exists a large variety of steganography techniques some are more complex than others and all of them have respective strong and weak points. Different applications have different requirements of the steganography technique used.For example, some applications may require absolute invisibility of the secret information,while others require a larger-secret message to be hidden. This project allows revels of message which is hidden using 3 different algorithms either in LSB or DCT and builds a software to find the chance of a message that is hidden.**

***Keywords*- Steganography, LSB,DCT, Steganalysis**

I. Introduction

Today the growth in Information Technology, especially networks such as mobile communication, Internet and digital multimedia applications has opened new opportunities for steganography and information hiding techniques. Steganography is a method of hiding secret messages into an innocent-looking cover media known as stegogramme such that an unintended observer will not be aware of the existence of the hidden messages. With steganographic techniques, it is possible to hide information within images, audio, video files or text which is perceptually and statistically undetectable.The same technology employed for digital watermarking and for digital markings/copyrights .But it's not limited to it , anyone can use it to hide information, to bypass securities, contact a specific person, send malicious content etc .The software build can be used by companies for this purpose.

II. Literature Survey

[1] provides a comparison of some of the steganalysis methods proposed in the literature, and using these comparison results, a global steganalysis methodology is proposed. The secret message detection capacities of these steganalysis methods are evaluated using stego images generated by typical data hiding algorithms. The evaluation of steganalysis methods is realized in terms of false negative and false positive error rates using 100 images. There is not any steganalysis that can detect presence of secret messages in all types of stego images. Therefore, to realize a reliable analysis about a suspicious image, several steganalysis methods must be efficiently combined. In this paper, some considerations about steganalysis are provided using the results obtained of the comparison of steganalysis methods.

[2] evolves deep learning technique in steganalysis of image.This shows how the development of Deep Learning has made it possible to unify and automate the two traditional stages into an end to end approach with promising results.[2] shows the evolution of steganalysis in recent years using the Deep Learning techniques. The results of these techniques have surpassed those obtained with conventional methods - Rich Models with Ensemble Classifiers - both in the spatial and frequency (JPEG) domains. Since 2014, researchers have used The Convolutional Neural Networks to solve this problem generating diverse architectures and strategies to improve the detection percentages of steganographic images on the last generation algorithms (WOW, S-UNIWARD, HUGO, J-UNIWARD, among others). Deep Learning, being applied to steganalysis, is now in the process of construction and results so far are encouraging for researchers that are interested in the topic.

[3] presents an observational ponder on applying convolutional neural systems (CNNs) to recognizing J-UNIWARD -- one of the foremost secure JPEG steganographic strategies. Tests directing the engineering plan of the CNNs have been conducted on the JPEG compressed BOSSBase containing 10,000 covers of measure 512×512. Comes about have confirmed that both the pooling strategy and the profundity of the CNNs are basic for execution. Comes about have too demonstrated that a 20-layer CNN, in common, outflanks the foremost advanced feature-based strategies, but its advantage continuously reduces on hard-to-detect cases. To appear that the execution generalizes to large-scale databases and to distinctive cover sizes, one exploration has been conducted on the CLS-LOC dataset of ImageNet containing more than one million covers trimmed to bound together measures of 256×256. The proposed 20-layer CNN has cut

the blunder accomplished by CNN as of late proposed for large-scale JPEG steganalysis by 35%.

III. Proposed Methodology

We have used the WaterFall Model as an abstract representation of the process.The software is divided into stages which consists of

A. Exploratory Data Analysis

This module includes the process of finding out whether the data is hidden in least significant bits or discrete Cosine coefficients. If data is hidden in LSB then the image will be prone to noise and hence easily identified so most of the time data is hidden in DCT. Hence software build is such that it identifies that data is hidden in DCT and LSB This step involves visualisation step-

Visualize YCbCr Channels: The Y or Luminous component is the brightness of the color. That means the light intensity of the color. The human eye is more sensitive to this component.Cb and Cr is the blue component and red component related to the chroma component.Filter out the Y, Cb and Cr components for each image separately using the convert function with the path of the image as the parameter.

Visualize DCT coefficients: DCT is a transformation technique for data compression and probably a good domain where the data can be hidden, and it can be seen that in the image that only a single 8X8 block was used for analysis, and further it can be seen that there is a much difference in the values of the coefficients; white represents very high values and black very low, so there is a high variance of the data here.

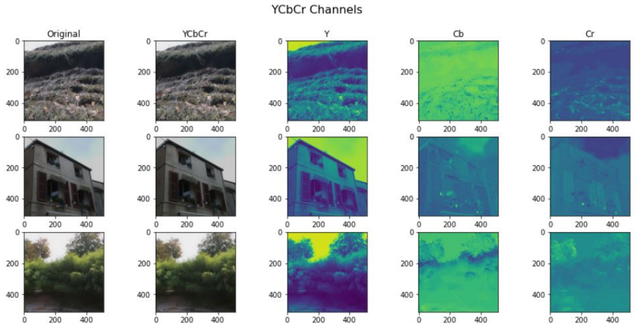


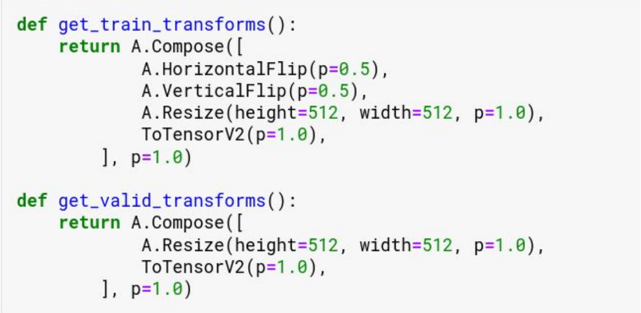
Fig. 1. Original Image and its YCbCr channels

B. Data Preprocessing and Augmentation

Now we will process the insights that we have gathered by EDA. As we have found that the chances of data being hidden in spatial domain are more than of it being in frequency domain.

First we performed some simple augmentations;

Augmentation: It is done to reduce the chances of overfitting of a model. Here images are cloned but they are also rotated or flipped so that the size of training data can be increased and as a result there will be less chances of the model being overfitted.



```
def get_train_transforms():
    return A.Compose([
            A.HorizontalFlip(p=0.5),
            A.VerticalFlip(p=0.5),
            A.Resize(height=512, width=512, p=1.0),
            ToTensorV2(p=1.0),
        ], p=1.0)

def get_valid_transforms():
    return A.Compose([
            A.Resize(height=512, width=512, p=1.0),
            ToTensorV2(p=1.0),
        ], p=1.0)
```

Fig. 2. Simple Augmentation

After Augmentation, the next step is preprocessing the data. Data preprocessing is basically processing the data so that it can be feeded properly to the neural network. It includes converting image channels from BGR to RGB. Further the insights that we have gathered from EDA that spatial domain has high potential of having hidden data so the spatial domain data of the images will be extracted from the images i.e decompressing the original image to its YCbCr components.

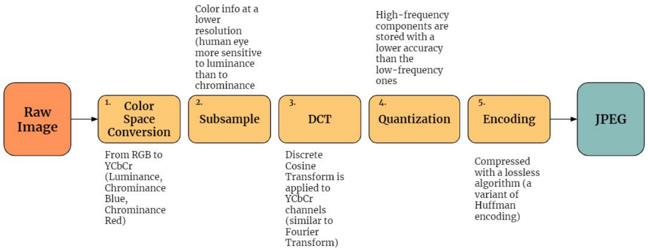

Raw Image
1. Color Space Conversion
2. Subsample
3. DCT
4. Quantization
5. Encoding
JPEG
Color info at a lower resolution (human eye more sensitive to luminance than to chrominance
High-frequency components are stored with a lower accuracy than the low-frequency ones
From RGB to YCbCr (Luminance, Chrominance Blue, Chrominance Red)
Discrete Cosine Transform is applied to YCbCr channels (similar to Fourier Transform)
Compressed with a lossless algorithm (a variant of Huffman encoding)

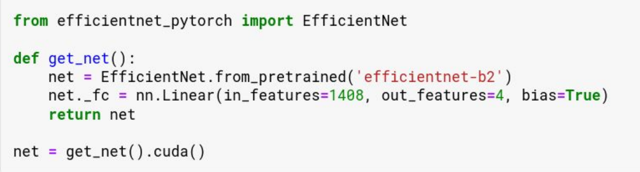
Fig. 3. JPEG Compression

Further Image normalization will be done. Here the pixel values of images are divided by 255 so that they will remain in the range 0-1 so that it can further be fed to the network without any error.

C. Training and Inference

Now as the data has been preprocessed it is ready to be further passed to the model. So we first fit the data to the model and then we train it.

The neural network that we have chosen is efficientNet-B2 because of its good accuracy.



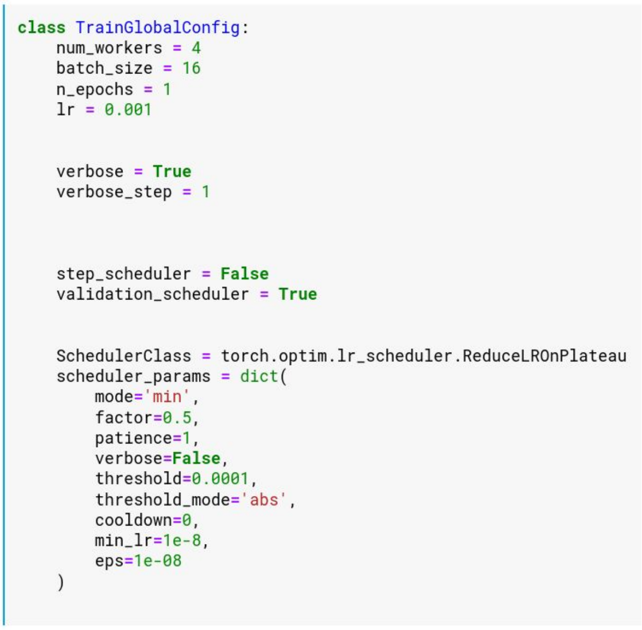
```
from efficientnet_pytorch import EfficientNet

def get_net():
    net = EfficientNet.from_pretrained('efficientnet-b2')
    net._fc = nn.Linear(in_features=1408, out_features=4, bias=True)
    return net

net = get_net().cuda()
```

Fig. 4. Neural Network

Further hyperparameters are chosen as per it suited for the best performance.



```
class TrainGlobalConfig:
    num_workers = 4
    batch_size = 16
    n_epochs = 1
    lr = 0.001


    verbose = True
    verbose_step = 1



    step_scheduler = False
    validation_scheduler = True


    SchedulerClass = torch.optim.lr_scheduler.ReduceLROnPlateau
    scheduler_params = dict(
        mode='min',
        factor=0.5,
        patience=1,
        verbose=False,
        threshold=0.0001,
        threshold_mode='abs',
        cooldown=0,
        min_lr=1e-8,
        eps=1e-08
    )
```

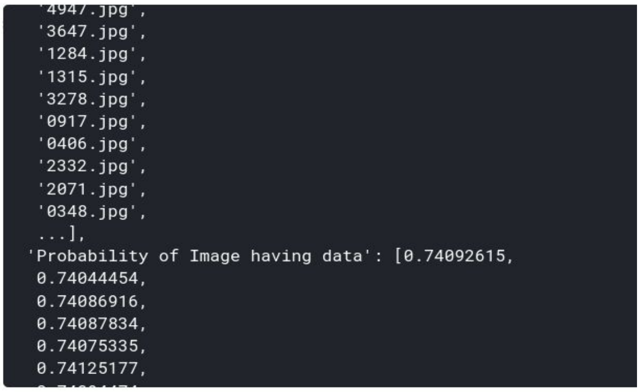
Fig. 5. Hyperparameters and Parameters

The dataset contained normal images and three different types of images that are encoded with hidden data using different algorithms which we have already mentioned above.

So, for training and testing purposes we divided the dataset in 8:2 ratio; where 80% images are used for training and 20% are used for testing how better the model performs.

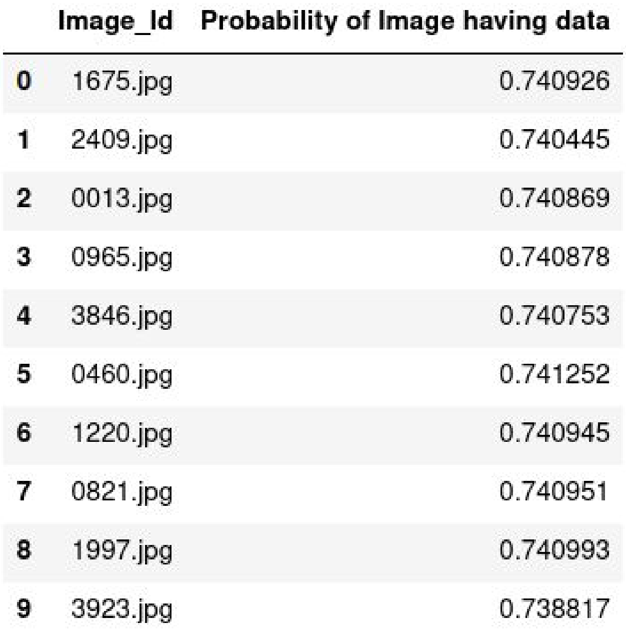
V. Results and Analysis

For each Id (image) in the test set, we provided a score that indicates how likely this image contains hidden data: the higher the score, the more it is assumed that image contains secret data. After training the data, we obtained a list containing the image id and their label. As we can see the predicted probability is high enough to classify an image as having some hidden data.



```
  '4947.jpg',
  '3647.jpg',
  '1284.jpg',
  '1315.jpg',
  '3278.jpg',
  '0917.jpg',
  '0406.jpg',
  '2332.jpg',
  '2071.jpg',
  '0348.jpg',
  ...],
 'Probability of Image having data': [0.74092615,
  0.74044454,
  0.74086916,
  0.74087834,
  0.74075335,
  0.74125177,
```

Fig. 6. List of ids and their predicted labels



	Image_Id	Probability of Image having data
0	1675.jpg	0.740926
1	2409.jpg	0.740445
2	0013.jpg	0.740869
3	0965.jpg	0.740878
4	3846.jpg	0.740753
5	0460.jpg	0.741252
6	1220.jpg	0.740945
7	0821.jpg	0.740951
8	1997.jpg	0.740993
9	3923.jpg	0.738817

Fig. 7. Table of first 10 images and their predicted probability of having data

We divided the dataset into 100 bins and plotted the histogram for different bins of images and the range of probabilities predicted for them. As shown a large number of bins have probabilities in range 0.6 to 0.8.

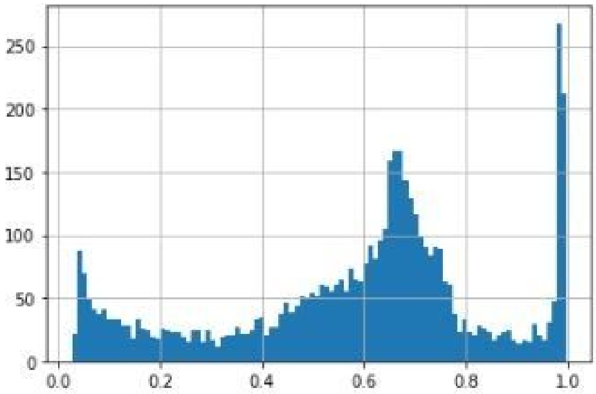


Fig. 8. Histogram of dataset and probabilities

We plotted the ROC curve and chose AUC as the performance metric.We obtained an area of 0.92 under the curve which depicts the high accuracy of the model.

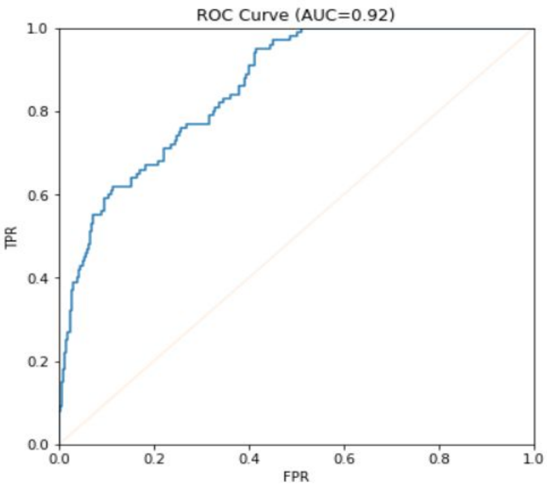


Fig. 9. Resultant ROC curve

VI. Conclusion And future work

In this paper, we have proposed a model which uses a EfficientNet B-2 neural network and trained on a dataset consisting four sets(cover,jmipod,juniverd,uerd) each of 75K images.

It has been demonstrated that this model can beat feature-based methods except in very difficult cases, and is therefore a promising research direction for further performance improvement.

Future works to assist move this inquire about ahead incorporates the taking after:

1. Supplant 4×4 DCT with more successful channel banks or something proportionate.
2. Bring the data caused by pooling (subsampling) back by making the Neural Network phase-ware.
3. Making the Neural Network indeed more profound.
4. Test the proposed Neural Network on other JPEG steganographic calculations

References

[1] https://www.researchgate.net/publication/221632616_A_Methodology_of_Steganalysis_for_Images

[2] https://www.researchgate.net/publication/333226568_Deep_Learning_Applied_to_Steganalysis_of_Digital_Images_A_Systematic_Review

[3] https://dl.acm.org/doi/abs/10.1145/3082031.3083236