Math 110B Final Project Writeup

Date: 09/10/2021

Group Member: Daoyu Li, Jilong Lyu, Shelly Wan, Yue Liu

Abstract:

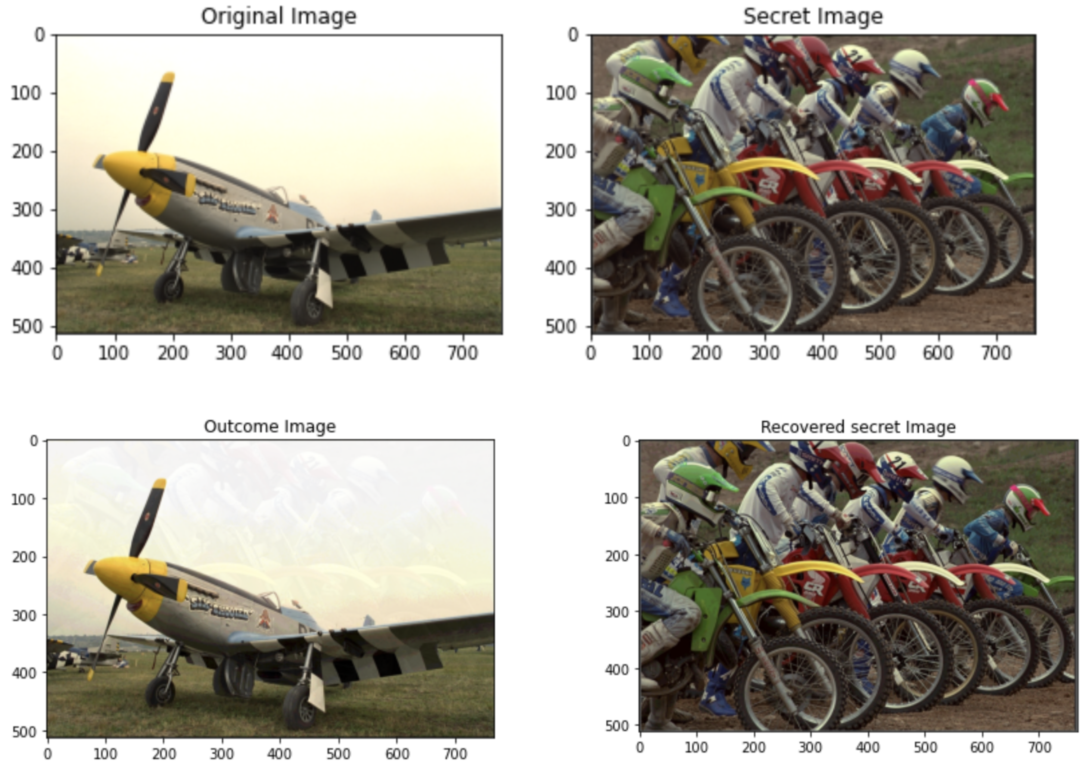
First task is implementing LSB (least significant bits).
The implementation we reference for first task:
https://github.com/kelvins/steganography

Second task is using the neural network to approximate D and E.
The implementation we reference for second task:
https://github.com/fpingham/DeepSteg/blob/master/DeepSteganography.ipynb

First Task: Implement LSB

The LSB implementation is to pick up two images from the data set. One is called "Original Image" and the other one is "Secret Image", and then set them up to the same size. The algorithm gets the pixel map of the two images and then merges the pixels of the secret image to the pixels of the original image and converts it to an integer tuple after checking the pixel map position is valid, which is "Outcome Image". The "Recovered secret Image" is the un-merged image, created by extracting the last 4 bits of the hidden image and then concatenating 4 zero bits to convert it to an integer tuple. The performance of our code is good because the secret image is successfully hidden and from the outcome image, we could see some characteristic features of the secret image. By running this code, because of the random choice, the output will vary the very time. Here we just exhibit one possible output.

Result of LSP:



Second Task: Neural Network

We used the neural network to approximate *D* and *E* and we construct the function according to the objective function in Mathematical aspects:

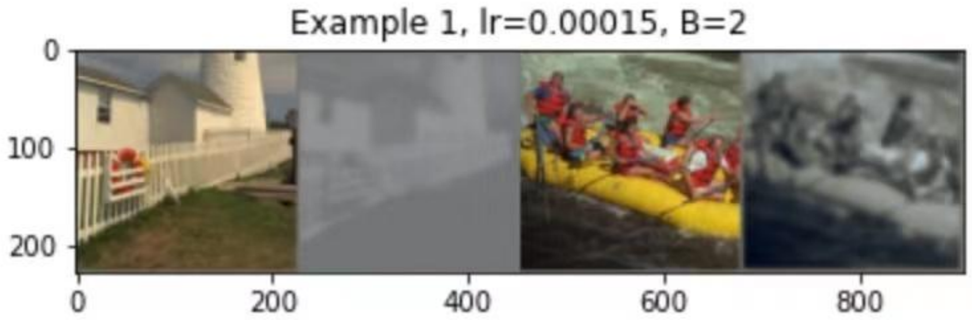
$$\min_{E,D} ||x - E(x, y)|| + \gamma ||y - D(z)||$$

We change the learning rate from 0.0001 to 0.00015 and we let γ=2, we also change the weight decay from 0 to 0.1. The issue we meet in the second task :

We used 180 pictures and three epochs in the trained model, but from the graphs we get from the code, we find that there have no big differences between those epochs. Moreover, from the final graph, we find that the result of the output is not as good as we expect (it is grey).

We consider the reason lead to this issue is the amount of the sample and the epochs are not enough.

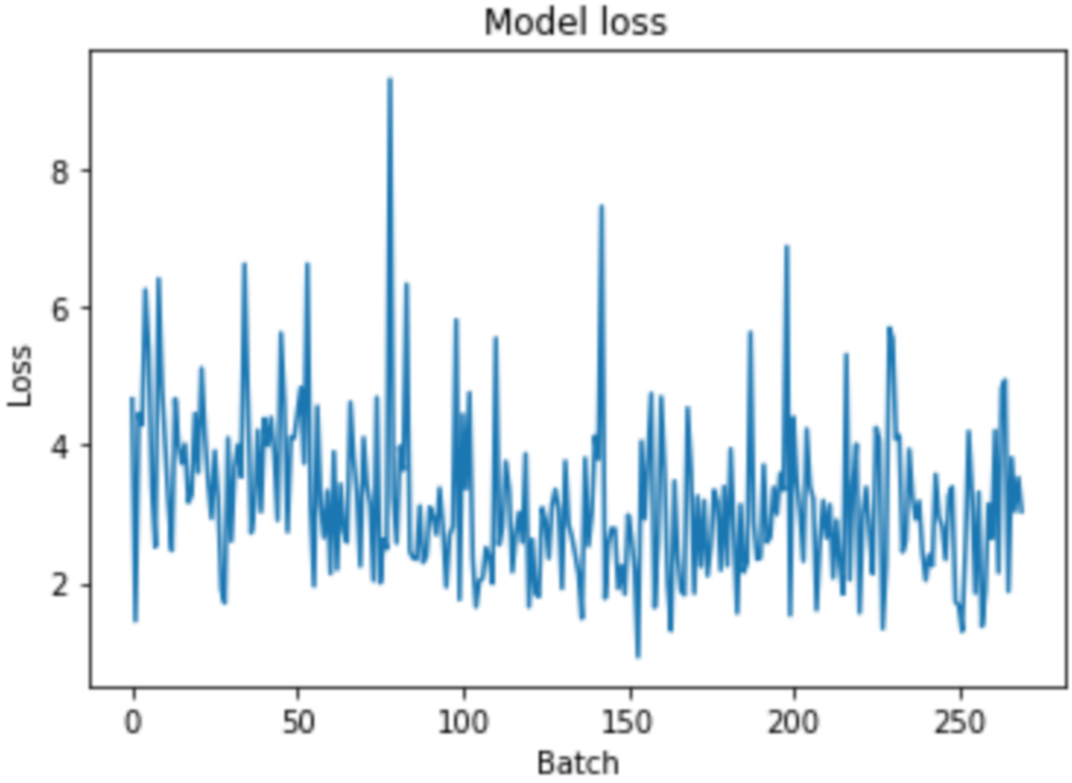
In order to solve the issue, we try to use two pictures and let epoch=100, we find that the loss has distinct decreasing at 50.

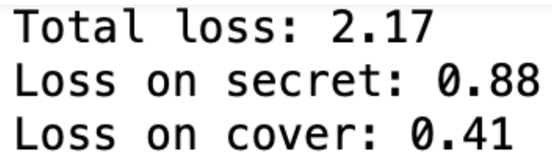


The speed of this method is slower than before and we can use some methods like “Cuda” to accelerate.

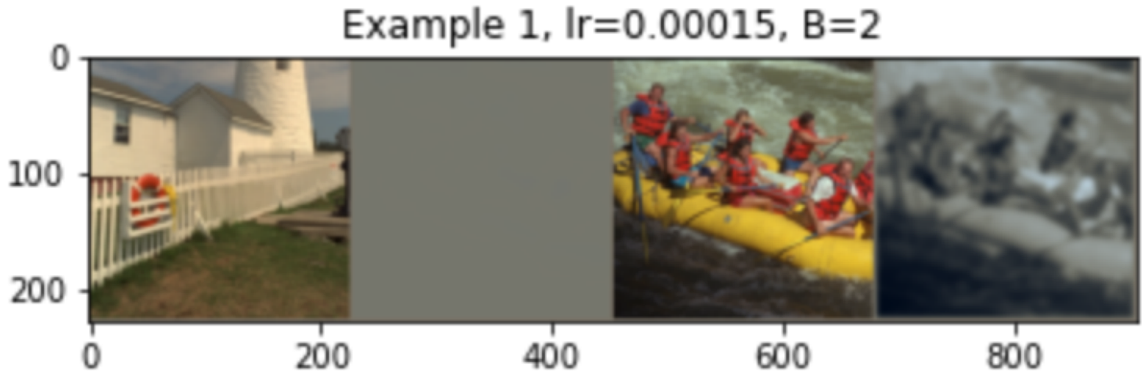
We also tried other methods to solve the problem, but due to the version update, environment configuration, and other problems, we failed to find other new methods effectively.

Result of NN:

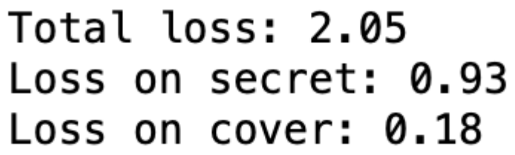




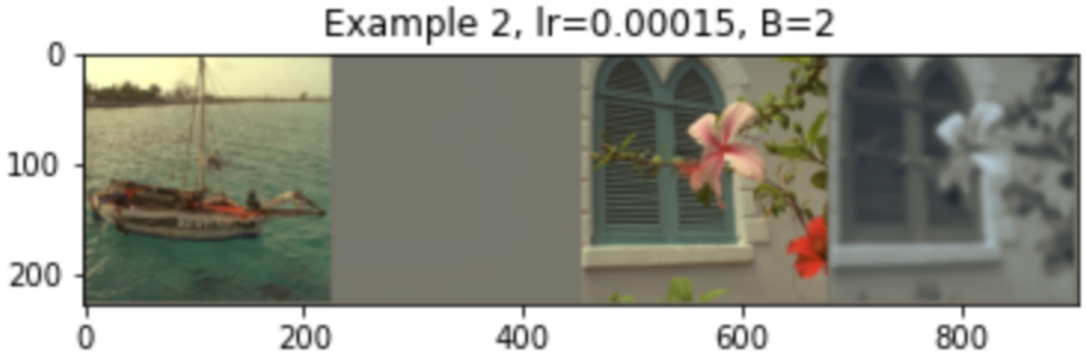
```
Total loss: 2.17
Loss on secret: 0.88
Loss on cover: 0.41
```



Example 1, lr=0.00015, B=2

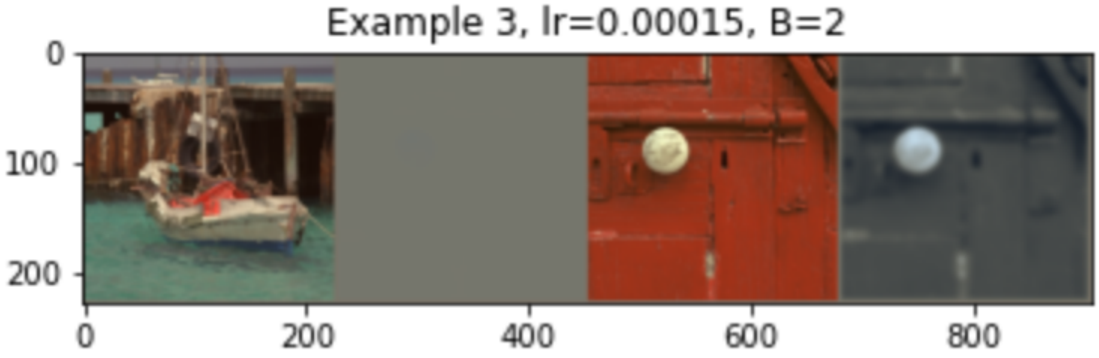


```
Total loss: 2.05
Loss on secret: 0.93
Loss on cover: 0.18
```

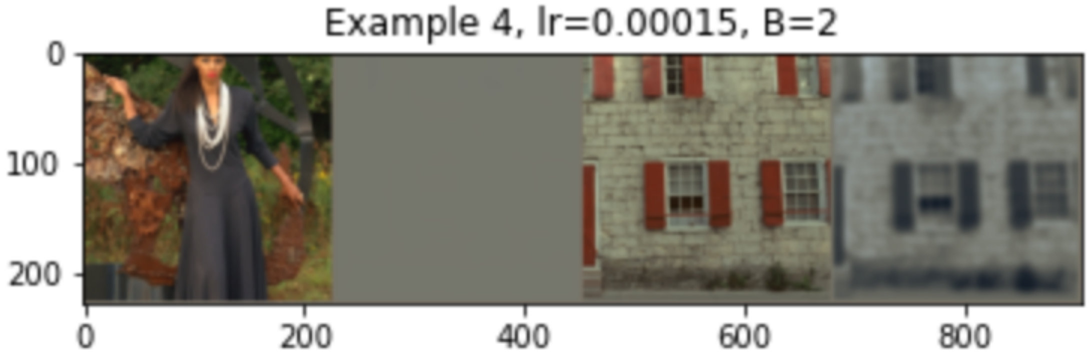


Example 2, lr=0.00015, B=2

```
Total loss: 2.30
Loss on secret: 0.76
Loss on cover: 0.79
```



```
Total loss: 2.34
Loss on secret: 1.08
Loss on cover: 0.17
```



```
Average loss on test set: 1.95
```

Finally, we can find that our results are reasonable and we completed task 1 and task 2 correctly.