Image Forgery Detection using Neural Network

Hoang Ba Cong

February 2020

1 Introduction

In this report, I will describe and explain the formula and implementation for classifying new handwriting data from MNIST. There are 6 parts consisted in this report.

2 Presenting problem

Initially, I will read the data which are abundance of JPEG images and the labels showing whether an image is tampered or not. Then, my task is to implement a multi-layers neural network to identify tampered images from test data.

3 Proposed method

To solve the problem of Image Forgery Detection, a method with 2 steps is proposed in the following:

- Transform each image to vectors consisting of 300 elements using YCrCb converter and Daubechies Wavelet transforms.
- Then, a fully neural network will be used for classifying whether an image is tempered.

4 Feature extraction

First, each RGB patch will be converted into YCrCb channel, then i will use Daubechies Wavelet transform for each YCrCb patch. Wavedec2 from pywt would be used to implement the transform. Consequently, 150 matrices (3 channels 5 transforms 10 result matrices) will be created from each image. In each matrix, its mean, standard variance, and sum are calculated. However, due to the insignificance of 150 elements of Y patch, there will be 300 elements of each image remaining of Cr and Cb patch in order to reduce the complexity.

5 Architecture of Neural Network

Basically, there are total 6 layers, including input, 4 hidden layers and output. The first layer and the last layer are also input layer and output, which have numbers of neurons corresponding to number of the dimension of a feature vector and labels vector. The following layers are hidden layers consequently consisting 300 neurons, 200 neurons, 150 neurons and 100 neurons. I will use Relu activation function with each hidden layers due to its efficient and rapidity to compute than other activation functions. For the output layers, softmax activation function is generally good choice for classification tasks.

6 Implementation

For the feature extraction, I use Wavedec2 to implement Wavelet Transform and for each of result matrix, numpy is commonly used to calculate its mean, standard variance and sum. For the classification, Tensorflow is a very popular deep learning framework released by, and neural network will be built with this library. There should be one thing we have to notice is dropout value. Thus, in order to avoid overfitting in this problem, the dropout value was set to 0,6. Besides, because of the complexity of neural network, I used batch gradient descent to reduce the computation time with the batch size about 20 of 6000 input data. The implementation of training phase lasted not for so long and it was kind of efficient with the converge of loss cost.