Individual lab report #7

Feb, 15, 2017

Xiaoyang Liu

Team F Rescue Rangers

Juncheng Zhang (Henry)

Karthik Ramachandran

Sumit Saxena

Individual Progress

- Developed a Feedforward Neural Network in MATLAB.
- Test on the revised RGB-based human detection algorithms.

Developed a Feedforward Neural Network in MATLAB

1) Initialization

In the initialization process, I used the normalized initialization. The reason why I choose to use it is listed as below

- The standard initialization that we have used

$$W_{ij} \sim U\left[-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}\right],$$

gives rise to variance with the following property:

$$nVar[W] = \frac{1}{3}$$

where n is the layer number. **This will cause the variance of the back-propagated gradient to be dependent on the layer (and decreasing)**.

- The normalization factor may therefore be important when initializing deep networks because of the multiplicative effect through layers, and we suggest the following initialization procedure to approximately satisfy our objectives of maintaining activation variances and back-propagated gradients variance as one moves up or down the network. We call it the normalized initialization:

$$W \sim U\left[-\frac{\sqrt{6}}{\sqrt{n_j + n_{j+1}}}, \frac{\sqrt{6}}{\sqrt{n_j + n_{j+1}}}\right]$$

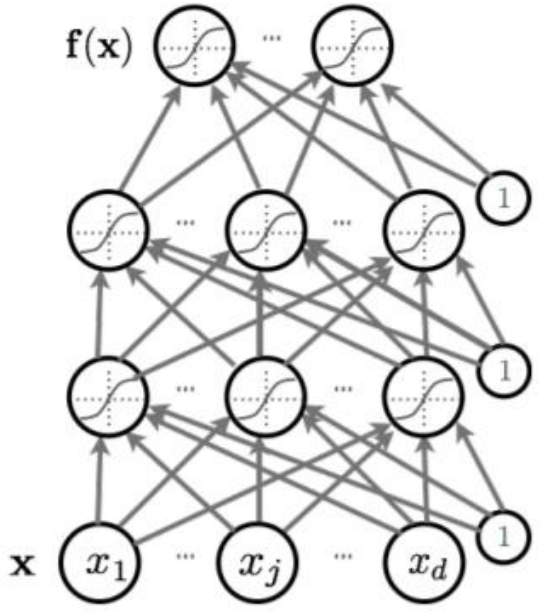

Figure.1 The Feedforward Neural Network

2) Using SGD for training the backpropagation

In terms of number of epochs, SGD is faster to be trained if we don't need to go through all samples in the sample set, otherwise the training time is almost the same. In terms of number of iterations, SGD is faster to be trained, because in each iteration, only one sample should be used to update the parameters in SGD compared to BGD which needs to use all samples in one iteration.

3) Using ReLu for training the backpropagation

With the extracted features, I plan to build a basic feedforward neural network next. I did some research on the activation functions, and finally choose to use ReLU.

The reason is that there are two major benefits of ReLUs. First of all, it has a sparsity and a reduced likelihood of vanishing gradient. The definition of a ReLU is h=max(0,a)h=max(0,a) where a=Wx+ba=Wx+b. So,

- The reduced likelihood of the gradient to vanish. This arises when a>0a>0. In this regime, the gradient has a constant value. In contrast, the gradient of sigmoid becomes increasingly small as the absolute value of x increases. The constant gradient of ReLU results in faster learning.
- Sparsity arises when a⩽0a⩽0. The more such units that exist in a layer the sparser the resulting representation. Sigmoid on the other hand are always likely to generate

some non-zero value resulting in dense representations. Sparse representations seem to be more beneficial than dense representations.

4) Configuration settings

- **Learning rate 0.001**

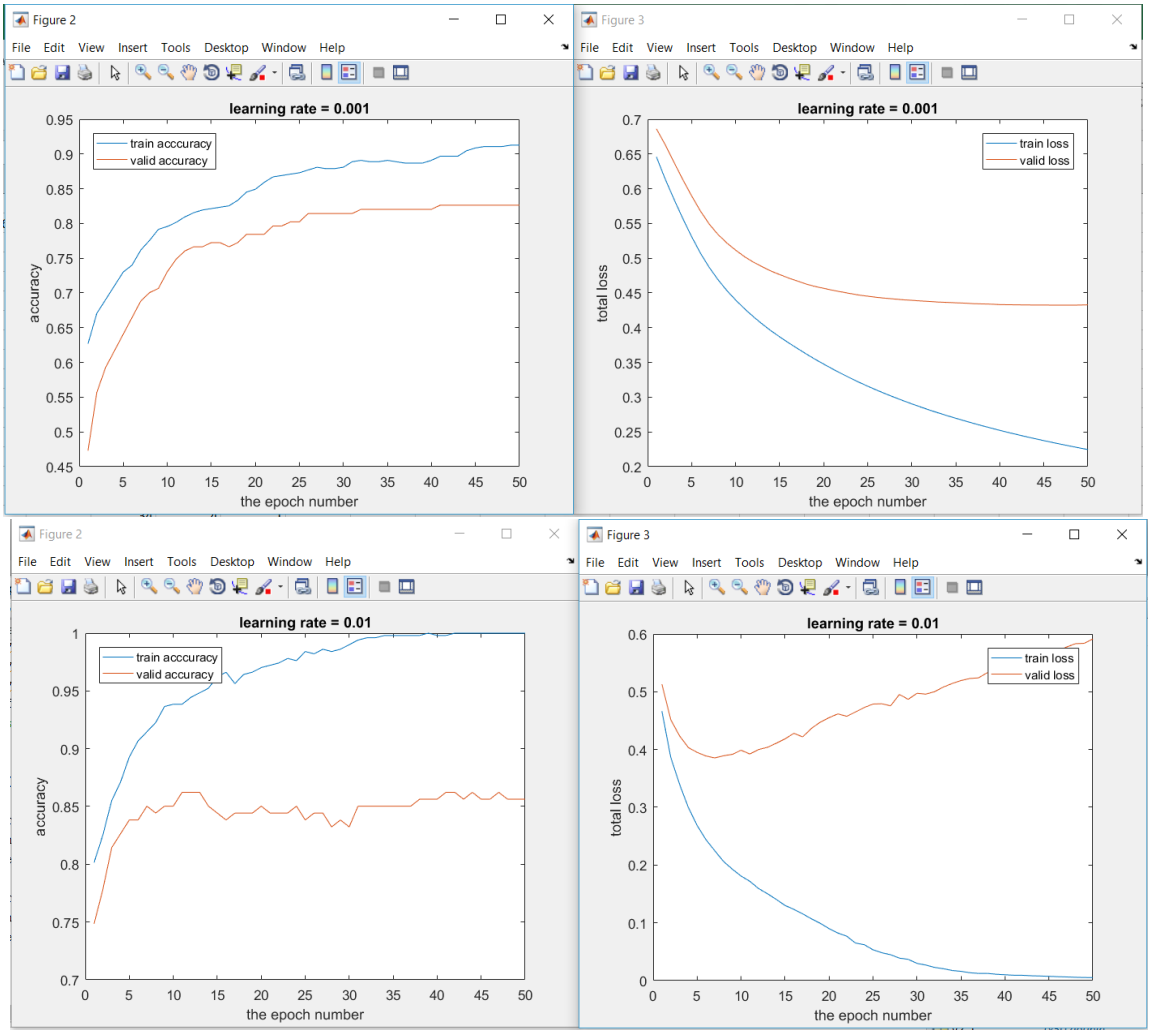

Figure.2 The confusion matrix of test images before and after the modification

As you can see in figure.2, with the same number of epoch time, the training loss with a learning rate of 0.001 is decreasing continuously. However even if the training accuracy corresponding with a learning rate of the is 0.01 is extremely high(nearly 100%), the loss of its validation set is increasing very fast after 10 epochs.

- **Layers: [288 200 150 2]**
- **Training epochs: 50**

Test on the revised RGB-based human detection algorithms.

1) Test on FNN vs SVM with **cropped human candidates (40*24 pixels)**
    - HOG feature extraction
        - Each image has 288 features
    - Dataset
        - Test set = 229 images
        - Training set = 504 images
        - Validation set = 167 images
    - Result is shown in Table 2

Table.2 Accuracy comparison on HOG + FNN and HOG + SVM algorithm

	SVM	**FNN**
Overall Accuracy	*90.4%*	*84.6%*

2) Test on HOG + FNN vs HOG + SVM **with the whole image (960*540/ 2560*1440 pixels)**

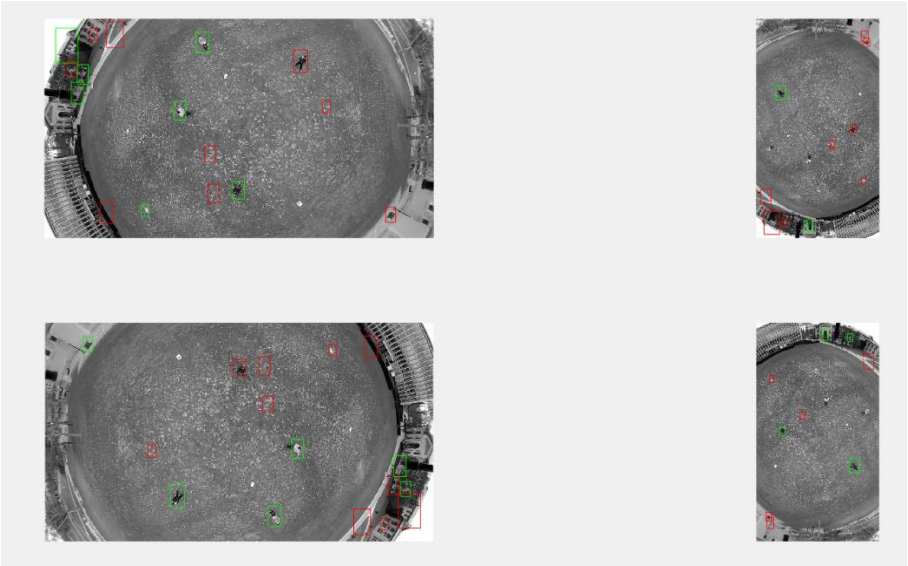

Figure.3 HOG + FNN test on self-taken aerial images

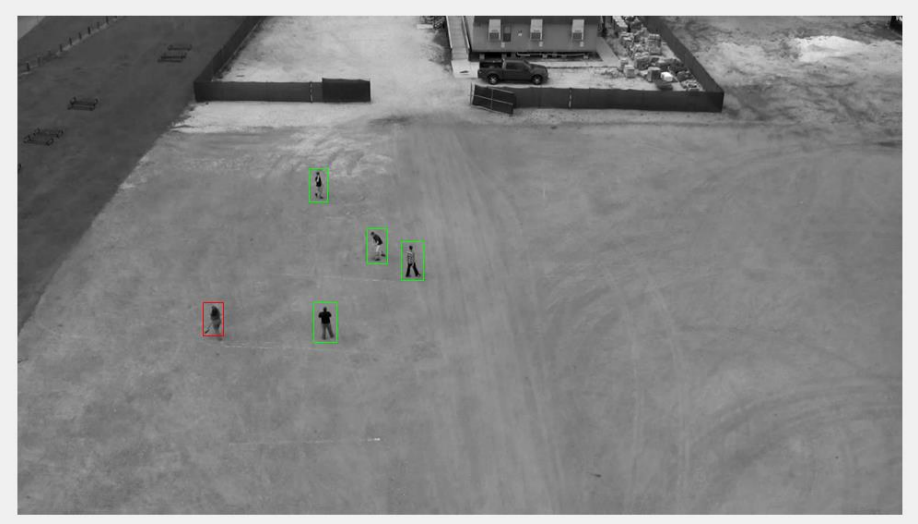

Figure.4 HOG + FNN test on online aerial images

Table.2 Accuracy comparison on HOG + FNN and HOG + SVM algorithm

Image category	**HOG+SVM**	**HOG+FNN**
Online images (20)	*65.6%*	*66.7%*
Self-takes images (20)	*61.5%*	*63.9%*

Challenge

In terms of the idea and implementation of FNN is not that difficult, because we have done something similar before. Also the dataset has been well prepared due to the effort we made in the last semester. However, one of the biggest obstacles that we faced will be the lack of training and testing data. We could get our own thermal and RGB camera next week. Also, because we change our strategy into practices on our own drone, hopefully this challenge will be overcome.

Team Work

In this week, Juncheng (Henry) devoted most of his time exploring thermal images feature extraction and segmentation. The result turns out that Gaussian Blur before Edge detection is

useful, and we may stick to the HOG + FNN method on thermal images as well. Karthik worked on improving the performance on sound detection by using the new microphone. Sumit mainly focused on implementing FNN on Tensorflow and find the possibility to extend it into CNN in the future.

Future Work

Before the next progress review, I will try to integrate RGB and thermal algorithm together and try to figure out the possibility that we can cluster objects by using some basic algorithms. In this way we might be eliminate some false positives. Juncheng will explore more on tracking on thermal images. Sumit will develop CNN and improve bright object detection. He also will get GPS location of signature from image. Karthik will work on CNN for thermal and rgb and maybe fuse sound with the other two.