Classification and Detection with Convolutional Neural Networks

CS6476 Computer Vison Final Project (Fall 2019)

Aiping Zheng
azheng39@gatech.edu

1. Introduction

Although recognizing characters on scanned documents and books or handwritings have been extensively studied, recognizing multi-character text in natural image still could be very challenging, and at the same time very important for a range of real world applications, including identification of street view house number from a real street image[1, 7].

To solve this problem on object detection, models of deep Convolutional Neural Network (CNN) were developed, and with the increasing availability of computational resources and large size of training data, huge success have been achieved in the last decade[3- 5].

In this report, the CNN architecture introduced by Goodfellow et al in 2013 [1] was used as a reference to build the method to detect and identify numbers from natural images.

2. Methods

2.1 Data and preprocessing

32x32 RGB images in Format 2 of the Street View House Numbers (SVHN) Dataset [2] were used to train the neural network to detect single digits. SVHN Format 2 has 73257 cropped single digits for training, 26032 cropped single digits for testing.

In the dataset, digit 0 was originally labeled 10, after preprocessing, it was changed to 0. And 10000 images from CIFAR10 was used as negative control (no digit in the image) and labeled 10. So totally there is 11 categories. The images was resized to 48x48 since VGG16 has 5 layers of MaxPool and needs input image to be at least 32x32. And then the input image was normalized to mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225] which was originally calculated for images in ImageNet.

2.2 Models architecture and training

Three models were tested. Model 1 is VGG16 with pre-trained weight, and the final output layer was changed from originally 1000 classes to 11 classes. Model 2 is VGG16 without pre-trained weight and the final output layer was also changed to 11 classes. Model 3 has the same architecture as the one in Goodfellow's paper which was designed for multi-digit number classification using Format 1 of SVHN dataset originally [1], and the output layer were also changed to suit 11 classes, and dropout rate was set to 0.2.

To train the models, categorical cross entropy was used as loss. For multiclass classification,

Loss for each observation was calculated by $-\sum_{c=1}^{M} y_{o,c} \log(p_{o,c})$, in which M is the number of classes, p is the predicted probability that observation o is of class c, and y is the binary indicator if class label c is the correct classification for observation o. Cross-entropy loss increases as the predicted probability departs from the real label.

In this project, Stochastic gradient descent (SGD) optimizer of PyTorch with mini batch , which could result in faster and smoother convergence, was used, $w := w - \eta \sum_{i=1}^{n} \nabla Q_i(w)/n,$, η is learning rate, n is the batch size, Q is the loss function, ω is the weight. In mini-batch SGD, at each iteration , the gradient of the cost function of a small batch of samples was used for weight update instead of the sum of the gradient of the cost function of all the examples. In this project, batch size was set to 128, learning rate was set to 0.001. Batch size and learning rate were selected and tuned based on the learning curve, and there is trade off between training speed/memory size and accuracy.

1% of the images in Format 2 training dataset was used for validation. The training procedure ends when the validation accuracy does not improve from current best validation accuracy in 6 consecutive epochs.

2.3 Overall pipeline

For each input image, maximally stable extremal regions (MSER) of openCV was used to detect possible locations of digits and generate bounding boxes, then the trained CNN model 3 was used to detect digits inside the bounding boxes and generate corresponding softmax outputs. All bounding boxes without digits inside were removed. Some other thresholds were also applied for bounding box selection, including width and height ratio of bounding boxes. Then Non-maximum Suppression (NMS) was used to remove overlapped boxes, and softmax of bounding boxes were used as confidence score. The remaining boxes were clustered, and the cluster with the highest average confidence score were kept. After that, the digits inside the remaining bounding boxes were combined into one number and drawn on the image. The bounding boxes for the whole number were also generated and drawn.

3. Results

3.1 Model performance

The learning curves were shown in figure 1, the performance of all three models are very good. Compared to model 2 which don't have pretrained weights, all three accuracies of model 1, which has pretrained weights, are better (0.9947 vs 0.9872 for training, 0.9519 vs. 0.9363 for validation, 0.9492 vs.0.9202 for testing), so the chosen of initial weights does affect model performance. The best accuracy of both models achieved at epoch 10, after that, although

training accuracy still increase, the validation accuracy and testing accuracy started to decrease, which indicates overfitting.

The performance of model 3, whose architecture was built upon Goodfellow's paper, is better than model 2, and almost as good as model 1 (model 1 vs. 3, 0.9947 vs 0.9889 for training, 0.9519 vs. 0.9495 for validation, 0.9492 vs.0.9345 for testing). And model 3 converged after epoch 6, compared to epoch 10 needed by model 1 and 2. The training of model 1 and 2 both need about 95 min using GPU on CoLab, while model 3 only need about 26 min, which is much faster.

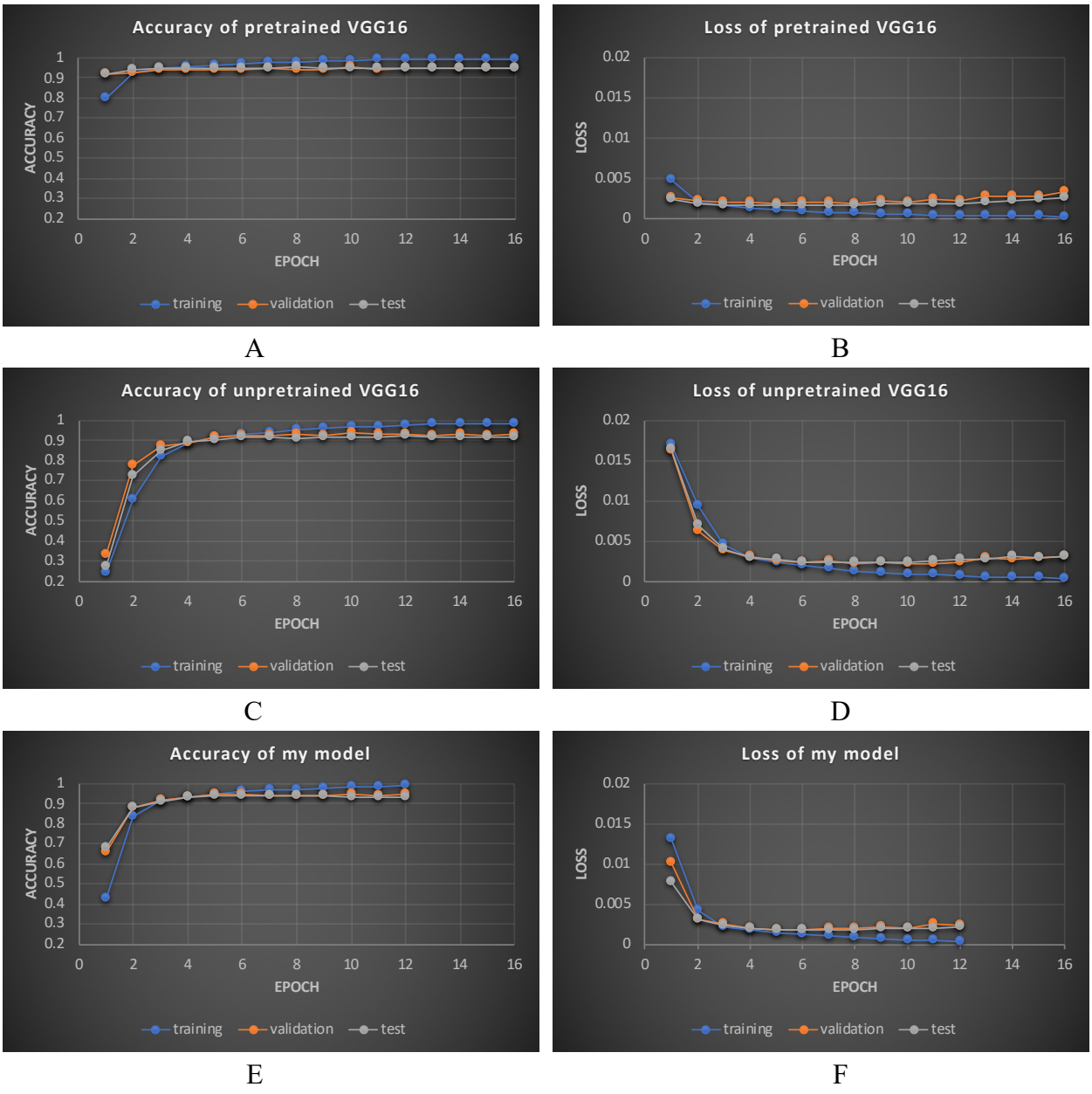

Figure 1. Performance of three CNN models

3.2 Detection on real images

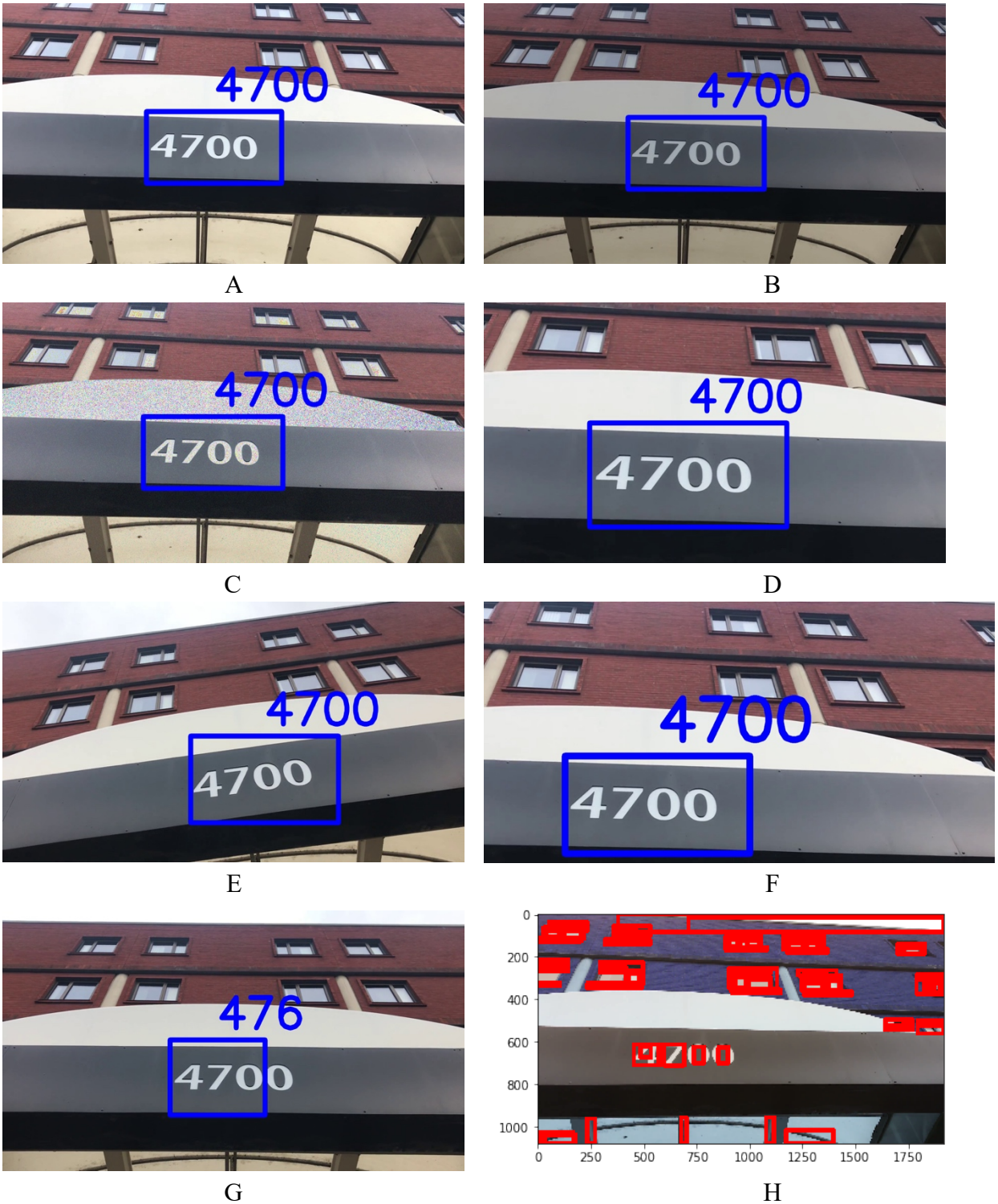

Figure 2. Detection performance on natural street images.

As shown in Figure 2, the final pipeline is able to detect and classify street view house number in images with different lighting (Fig. 2B), different noises (Fig. 2C), different scales (Fig. 2D), different orientations (Fig. 2E) and different locations (Fig. 2F).
One negative result was shown in Fig. 2G, which identify 4700 as 476. This is because MSER did not identify the bounding boxes for two zeros correctly in the first place (Fig. 2H), and the CNN model did not distinguish 6 and 0 correctly.

3.3 Video

Detection: https://youtu.be/07kUGX_Vehc and
https://www.dropbox.com/s/eolccjpveq3w8h8/IMG_2890_with_numbers.mp4?dl=0
Presentation: https://youtu.be/BdNUUHfY5H4 and
https://www.dropbox.com/s/0ky99n1q8rbm67i/presentation_video.mov?dl=0

4. Discussion

4.1 Comparison to state-of-art work

The testing accuracy of our model is 93.45% , a little less than 96% in I. J. Goodfellow et al paper [1] and a little better and 91% in Y. Netzer et al. paper[7], while the accuracy of human performance is 98%.

4.2 Future improvements

The pipeline worked well on selected images, but did not work so well on the video I took from the street. To improve the performance, parameters for MSER need to be further tuned to detect the location of digits. Negative control datasets better than CIFAR10 probably could help remove false identification of existence of digits. Extra training dataset probably could help distinguish similar digits such as 6 and 0.

5. References

1. I. J. Goodfellow et al. “Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks” ICLR 2014 conference.(2014)
2. http://ufldl.stanford.edu/housenumbers/
3. K. Simonyan and A. Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition” ICLR 2014 conference.(2015)
4. M. Jaderberg et al. “Reading Text in the Wild with Convolutional Neural Networks”. *International Journal of Computer Vision*. (2014)
5. M. Jaderberg et al. “Deep Features for Text Spotting”. European Conference on Computer Vision. (2014)
6. Q. Guo et al. “Hybrid CNN-HMM Model for Street View House Number Recognition”, *Asian Conference on Computer Vision*. (2015)
7. Y. Netzer et al. “Reading Digits in Natural Images with Unsupervised Feature Learning” *NIPS 2011* (2011)