CS6476 Computer Vision 2017 Fall Final Project

Classification and Detection with Convolutional Neural Networks

Ran Chen
ranchen@gatech.edu

Description and Objective

Detection and Identification / Classification

The objective of this project is to build a robust digit detection and identification application, cable of detection of location and sequence identification for digits in house numbers, given any images that contain a sequence of digits using convolutional neural networks (CNN). The design is as follows, once fed an image, the application creates a sliding window, and for each window the CNN will be applied to determine whether there is a digit in this window, and if yes, identify the digit. This process needs to be applied for every sliding window on the entire image. This implementation, when using a single net, requires that the net is able to identify 11 classes, including 0-9 digits and an additional non-digit class.

Neural Net Training

The CNN net need to be trained on a large labeled house number image database. Here the data format 2 from SVHN database (Stanford) was chosen because each single digit is clearly cut out as 32 x 32 patches and labeled.[1] Since the dataset only have 10 labeled classes (digits 0-9), large number of images of the 11^{th} class (non-digit) will need to be added to obtain appropriate training dataset. In order to add non-digit class sample images to our training process, GMCP Geolocalization dataset (University of Central Florida) consisting over 60,000 Google Streetview images were used.[2]

Method and Algorithm

Dataset Preparation

Due to large size of dataset and limited computer memory resources, the SVHN format 2 datasets were downloaded and converted from .mat file to HDF5 file using Fuel[3] library for python. The data flow can be conveniently generated from the file and feed to CNN training code. Non-digit sample images were generated by randomly choosing an image from GMCP Geolocalization dataset, and randomly cut a 32 x 32 patch out. 20,000, 40,000, and 80,000 non-digit samples were added to SVHN dataset and tested on different nets separately. The SVHN training set contains more than 70,000 samples, and since the sliding window method will likely produce more windows contains no digit, it is desirable to have more non-digit training samples than any of the digit classes. Based on training, validation, and testing statistics, the dataset with 40,000 non-digit samples was chose to be used with the digit detection and classification pipeline.

CNN

Three different approaches were tested for CNN net: (1) a VGG16[4] net with modified input and fully connected top layers trained from scratch; (2) a VGG16[4] net with modified input and fully connected top layers, loaded with pre-trained convolutional layer weights, re-trained weights for fully connected layers; (3) a custom net consisting 7 convolutional and 3 fully connected layers. All trainings were done on SVHN dataset plus 40,000 non-digit samples obtained from GMCP dataset, using stochastic gradient descent optimizer. Figure 1 shows the training and validation statistics for all models tested in this project. As shown, both custom net and VGG16 can obtain >0.95 training and >0.90 validation accuracy, when trained from scratch. Although VGG16 loaded with pre-trained weights did not perform well, after fine tuning it also reached similar level of performance. VGG16 was original designed for general purpose object classification with 1000 output labels, it was trained using huge ImageNet datasets, it performed exceptionally well for more complicated classifications, which requires larger amount of training data due to its deep convolutional layer structure.[4] For the purpose of digits detection, a deep net like VGG16 will likely slow down both training and classification speed, while provide little boost in performance. Some preliminary comparison on digit sequence identification on real world images showed that, while providing similar accuracy, the custom net is faster. Thus the custom net was selected to be applied to the detection/classification pipeline. The training samples were preprocessed before fed to the net. The preprocessing steps include sample wise pixel value centering to reduce the in-sample variance[5], and image augmentation, such as random rotation.

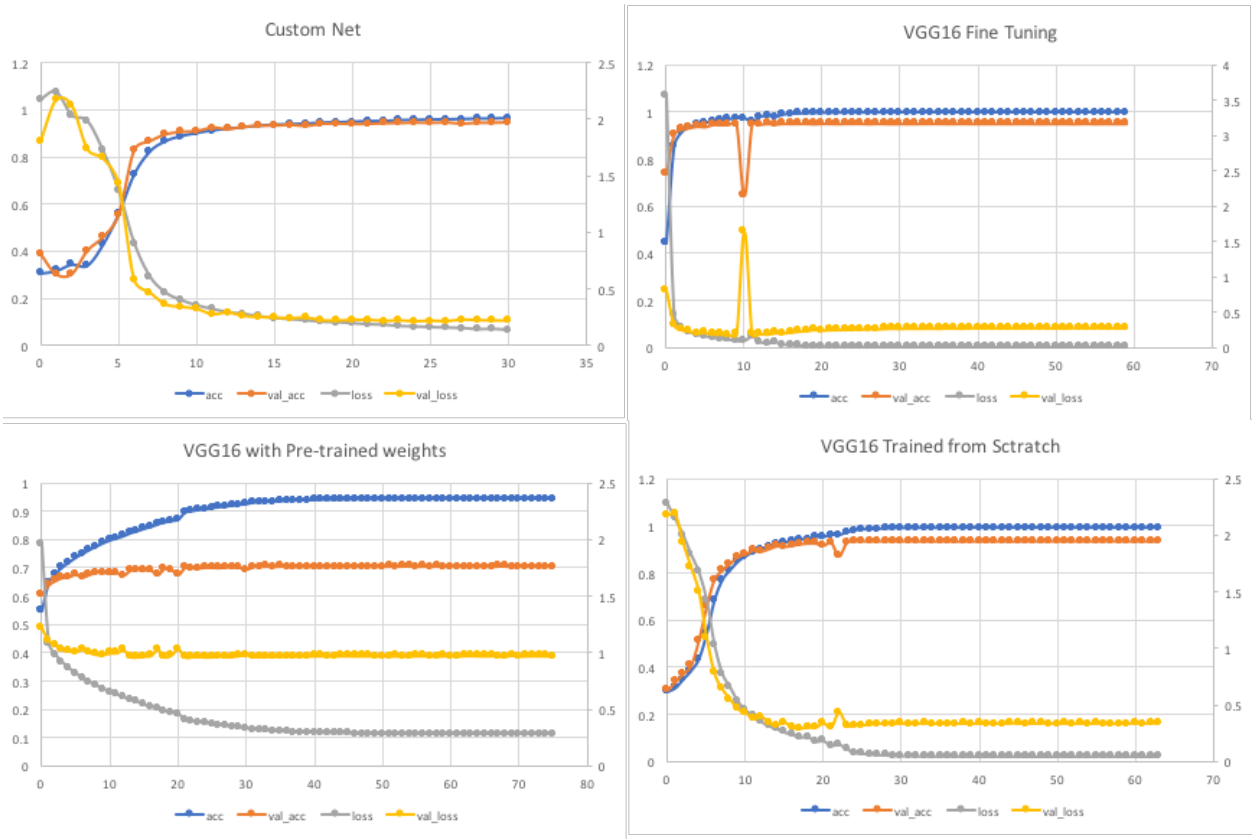

Figure 1 Training statistics for different net tested in this project.

Pipeline

The application pipeline consists the following steps:

(1) Image Pyramid

The program reads in an image, preprocess it, then resize the images with scales ranging from 0.5 to 1.5 and create a pyramid of rescaled images. The preprocessing step includes image wise pixel value centering.

(2) Sliding Window

The program then creates sliding windows on all images in the pyramid, makes 32 x 32 small patches and feeds them to the CNN classifier.

(3) CNN classification

Apply neural net to the cutout, obtain predictions. The net produces a vector of 11 probabilities corresponding to 11 classes (including non-digit). A threshold on the probabilities (typically 0.9999 in this project) to remove results with low confidence. The classification of non-digit class with high confidence level is also discarded
discard window positions with non-digit prediction. The window position, window size (rescaled from different pyramid levels back to original image), and confidence level (probability) of the remaining detections are kept.

(4) Result Collection and Cleaning.

All detected window positions with their corresponding predictions, probabilities, and rescaled window sizes, are fed to a non-maximum suppression algorithm, to remove overlapping detection windows while retain the detection with highest confidence level.

(5) Output Result

The results are displayed on the original images using a bounding box corresponding to the detection window, along with the classification results.

Results and Discussion

The digit detection and classification detection pipeline shows certain level of robustness against rotation as shown in Figure 2: the digits are still accurately located and identified. This is due to the fact that image augmentation including random rotation was used during model training, as a result the net was able to identify rotated digits.

Video result linked here: https://youtu.be/e7xge9eWlSY (backup link: https://1drv.ms/v/s!AlUiFxjxy5SQhP5YHjsNqNVM0qX7QQ)

Figure 2 Rotation invariance.

The classifier was also able to handle mildly noisy images, Figure 3 shows the comparison between the original image and the same image with added Gaussian noise. The net was able to identify the digits in the noisy images possibly because many noisy images were included in the training dataset.

Figure 3 Handling noise.

During testing of the pipeline on many house number images, several failures occurred mostly in cases where the digits are very close to each other as in the “335”case shown in Figure 4, improper rescaling as in “6307” case, or too many distracting features as in “1103” case. Most of these failures can be addressed by further optimize the pipeline instead of the net itself. The accuracy of the CNN classifier is usually not the bottleneck, however it is possible to confuse the net with a lot of distract features, such as corners that look like a “7”, round objects that look like a “0”, ect. This could be addressed by specifically adding image patches containing these

confusing features to the training set as non-digit class to produce a more robust model agains these features. Another shortcoming of the model trained using the dataset described above is that, the non-digit samples were all cut from the GMCP database, where all images were collected from a few downtown areas in the United States (including Manhattan and Pittsburgh), while the images we tried to apply our net on were mostly from residential areas, and the scene could be very different. This can be improved by incorporating non-digit samples from a residential are Google Streetview image database. In addition, if the purpose is to extract digits from a street address, then samples containing letters should also be included as non-digit samples in the training dataset, otherwise some of the letters can be easily confused with digits.

Figure 4 Failures.

One additional improvements that could be made is that, instead of a single digit classifier, a digit string classifier can be used instead. The output should include the identified digit sequence, location sequence, length information etc., as Goodfellow et al. demonstrated[6], this approach can produce very robust and accurate results. While identification accuracy could be further improved by incorporating a deep residual learning method, which deepens the network 8 times, and uses an ensemble net for prediction.[7]

Presentation

Presentation video: https://youtu.be/EE6ZivHLLik
Backup Link: https://1drv.ms/v/s!AlUiFxjxy5SQhP5ZNkaCdf66iw5NRg

References

[1] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, “Reading Digits in Natural Images with Unsupervised Feature Learning,” 2011.

[2] A. R. Zamir and M. Shah, “Image Geo-Localization Based on MultipleNearest Neighbor Feature Matching UsingGeneralized Graphs,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 8, pp. 1546–1558, Aug. 2014.

[3] “Welcome to Fuel’s documentation! — Fuel 0.2.0 documentation.” [Online]. Available: http://fuel.readthedocs.io/en/latest/index.html. [Accessed: 03-Dec-2017].

[4] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *ArXiv14091556 Cs*, Sep. 2014.

[5] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” *ArXiv150203167 Cs*, Feb. 2015.

[6] I. J. Goodfellow, Y. Bulatov, J. Ibarz, S. Arnoud, and V. Shet, “Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks,” *ArXiv13126082 Cs*, Dec. 2013.

[7] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” Dec. 2015.