CSCI3230 Fundamentals of Artificial Intelligence
Neural Network Project
The Street View House Numbers Prediction
Due date: 23:59:59 (GMT +08:00), 25st December, 2016

1. Project Specification

Deep neural network have become a hot research topic in machine learning in recently years. Compare to other methods, deep learning have showed its advantages in handling large amount of data and achieving better results. Based on the structure of the network and the neurons, there are many types of deep neural networks. Among them, the deep Convolutional neural network is extremely popular in the area of image processing.

Meanwhile, SVHN is a real-world image dataset for developing machine learning and object recognition algorithms with minimal requirement on data preprocessing and formatting. It can be seen as similar in flavor to MNIST (e.g., the images are of small cropped digits), but incorporates an order of magnitude more labeled data (over 600,000 digit images) and comes from a significantly harder, unsolved, real world problem (recognizing digits and numbers in natural scene images). SVHN is obtained from house numbers in Google Street View images.

In this project, you are asked to develop a deep neural network to classify the images in SVHN using TensorFlow. TensorFlow is an open source software library for numerical computation using data flow graphs. It has many convenient APIs for implementing the deep neural network, which makes the implementing convenient.

2. Dataset

There are two formats in SVHN. To make it convenient, we use the second format for SVHN, which is similar to MNIST. All digits have been resized to a fixed resolution of 32-by-32 pixels. The original character bounding boxes are extended in the appropriate dimension to become square windows, so that resizing them to 32-by-32 pixels does not introduce aspect ratio distortions. Nevertheless this preprocessing introduces some distracting digits to the sides of the digit of interest. Loading the .mat files creates 2 variables: “X” (uppercase)which is a 4-D matrix containing the images, and “y”(lower case) which is a vector of class labels. To access the images, X(:,:,:,i) gives the i-th 32-by-32 RGB image, with class label y(i).

- 10 classes, 1 for each digit. Digit '1' has label 1, '9' has label 9 and '0' has label 10.
- 73257 digits for training, 26032 digits for testing, and 531131 additional, somewhat less difficult samples, to use as extra training data

3. Submission requirements

1. The only supported language for this project is python2.7(we don’t accept python3.x program.)
2. Submission list
    1. Source file for training. Name it as “train.py”
    2. Source file for recovering your network model, Name it as “test.py”
    3. TensorFlow generated files which stores the value of your variables

4. Any other files that helps your programs to work, such like preprocessing files, format converting files.

3. Submission Package:

Put everything in the submission list above into a folder, name the folder as your student id, zip the folder WITHOUT encryption. Submit the zip file into our submission system.

Do not add any images of the SVHN dataset in your package, because we will prepare the data for you.

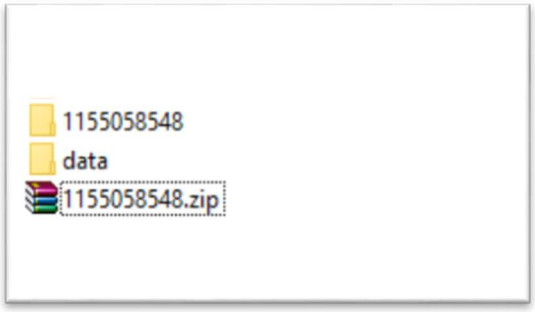
4. Dataset arrangement

The submission system has prepared the images of SVHN in a folder named "data" , it contains two files:

1. train.mat, which is renamed from the train_32x32.mat in SVHN homepage
2. test_images.mat, which contains 1000 random images(without labels) from the test_32x32.mat in SVHN homepage, the format keeps unchanged.

After your submission, we will unzip your package, put your sid named folder alongside the data folder.

For example, there is a student whose sid is 1155058548, after his submission, the files in the submission system is organized like this:



5. Grading policy

We grade your project based on your correctness and your accuracy.

For the correctness, we will run your "train.py" file, which is expected to read the "train.mat" file in data folder and train a deep network. We will check to see if there is any compiling error or run time error. If there is no error for a randomly selected time range, you all get all your correctness points, otherwise your will lost all the correctness points.

For the accuracy, we will run your "test.py" file, which is expected to read the "test_images.mat" file in the data folder. You should try to predict the labels for the 1000 randomly selected testing images, and output your predictions into a file named "labels.txt". The "labels.txt" file is arranged as follows: it contains 1000 lines, and each line is a digit representing the label for the corresponding sample (we use the original representation of SVHN dataset, which means that '10' represents 0). The accuracy score is the percentage of correct prediction that you have made. Your accuracy points is your relative performance compared to other students.

Correctness and Accuracy contributes equally to your final score, that is,

Final score= 0.5 * correctness points + 0.5 * accuracy points.

6. Important Points

To make this project fair and meaningful, there are some other points you **MUST** follows:

1. The size of your “test.py” should not exceed 4k Bytes (to prevent you remember the whole dataset in your test.py)
2. The maximum number of lines in your “test.py” that do not in your “train.py” is 20 (to make sure that “test.py” is only used to recover the trained network)
3. The time limits to run your “test.py” is 20s
4. Plagiarism will be SERIOUSLY punished (ZERO mark plus reporting to department)
5. Late submission will NOT be entertained according to our submission system settings

7. Late Submission

No. of Days Late	Marks Deduction
1	10%
2	30%
3	60%
4 or above	100%