Project Part 3: Classification Using Neural Networks and Deep Learning (SVHN Dataset)

Due Jul 4, 2022 by 11:59pm **Points** 15
Submitting a file upload **Available** until Jul 5, 2022 at 11:59pm

This assignment was locked Jul 5, 2022 at 11:59pm.

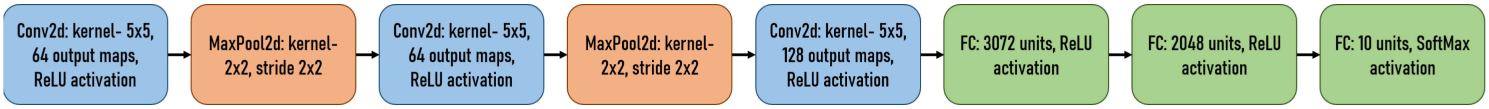
Project Overview:

In this part, we will build and train a small convolutional neural network for classification task. The dataset is the SVHN which consists of printed digits cropped from the house plate pictures belonging to 10 classes. There are a total of 73,257 training samples and 26,032 testing samples. The input image resolution is 32x32 and consists of 3 (RGB) channels. Find more about the dataset **here (http://ufldl.stanford.edu/housenumbers/)** . For this project, download the SVHN dataset here (**train data (https://asu.instructure.com/courses/117787/files/47766945?wrap=1)** and **test data (https://asu.instructure.com/courses/117787/files/47766944?wrap=1)**).

(1) train_32x32.mat consists of training images and their labels.

(2) test_32x32.mat consists of training images and their labels.

The CNN architecture and parameter settings are as follows:


Conv2d: kernel- 5x5, 64 output maps, ReLU activation
MaxPool2d: kernel- 2x2, stride 2x2
Conv2d: kernel- 5x5, 64 output maps, ReLU activation
MaxPool2d: kernel- 2x2, stride 2x2
Conv2d: kernel- 5x5, 128 output maps, ReLU activation
FC: 3072 units, ReLU activation
FC: 2048 units, ReLU activation
FC: 10 units, SoftMax activation

(1) The first hidden layer is a convolutional layer, with 64 output feature maps. The convolution kernels are of 5x5 in size. Use stride 1 for convolution. The activation is ReLU.
(2) The convolutional layer is followed by a max pooling layer. The pooling is 2x2 with stride 2.
(3) After max pooling, the layer is connected to the next convolutional layer, with 64 output feature maps. The convolution kernels are of 5x5 in size. Use stride 1 for convolution. The activation is ReLU.
(4) The second convolutional layer is followed by a max pooling layer. The pooling is 2x2 with stride 2.

(5) After max pooling, the layer is connected to another convolutional layer, with 128 output feature maps. The convolution kernels are of 5x5 in size. Use stride 1 for convolution. The activation is ReLU.
(6) After convolutional layer, there is fully connected layer with 3072 nodes and ReLU activation function.
(7) The fully connected layer is followed by another fully connected layer with 2048 nodes and ReLU activation function, then connected to the last fully connected layer with 10 output nodes (corresponding to the 10 classes). Use the SoftMax activation for the last layer.

Note: For all the convolutional layers, use padding='same' which means that zero padding will be applied to the input to get the output of same size as that of the input.

Train the network using SGD (Stochastic Gradient Descent) optimizer with a learning rate of your choice for 20 epochs (you can experiment with number of epochs as well). You can start with 0.01 learning rate and experiment with it. Use training set to train the network and test it on the testing set.

You are required to plot the training error and the testing error as a function of the epochs.

Steps:

(1) Read the data. You can load the data using the following commands:

```
from scipy.io as io
trX = io.loadmat('train_32x32.mat')[‘X’]
trY = io.loadmat('train_32x32.mat')[‘y’]
tsX = io.loadmat('test_32x32.mat')[‘X’]
tsY = io.loadmat('test_32x32.mat')[‘y’]
```

(2) Normalize the data to bring it in the range of 0-1. Encode the labels using one-hot vector encoding.
(3) Build the model as per the architecture mentioned above.
(4) Train the model using SGD optimizer on training set and test on the testing set.
(5) Plot the training and testing curves as a function of epochs and report the final classification accuracy on testing set.

Algorithm:

Convolutional Neural Network

Resources:

SVHN dataset, Google CoLab

Workspace:

Google CoLab (see file intro_to_colab.docx for more details)

Software and Libraries:

Google CoLab, Keras, PyTorch, TensorFlow

Language(s):

Python

Getting Started:

Read **intro_to_colab.docx (https://asu.instructure.com/courses/117787/files/47766943?wrap=1)** carefully. For more details about Colab, please go to **https://colab.research.google.com/notebooks/welcome.ipynb (https://colab.research.google.com/notebooks/welcome.ipynb)** .

For getting started with Keras – **https://keras.io/getting_started/ (https://keras.io/getting_started/)**

Required Tasks:

1. Follow the steps to build and train the given CNN on SVHN dataset.

2. Plot the training and testing curves as a function of epochs. Report the final test accuracy.

3. Submit a brief report summarizing the above results, along with your code.

What to Submit and Due Dates

1. Code: please add comment properly to explain what you do.

2. A report including the plots for the learning/testing errors and the final classification accuracy.

3. Please submit the code and the report as separate files on Canvas. **Do not zip them.**