```
# autogenerated pyup.io config file
# see https://pyup.io/docs/configuration/ for all available options

# configure dependency pinning globally
# default: True
# allowed: True, False
pin: False

# update schedule
# default: empty
# allowed: "every day", "every week", ..
schedule: every day

language: python
python:
 - "3.7"
script:
   - python CNN_Keras/CNN_MNIST.py
```

Footer

Handwritten Digit Recognition using Machine Learning and Deep Learning

Note : Since, Git LFS on github.com does not currently support pushing LFS objects to public forks - Full pledged repository is moved to GitLab. Avalable On Banee Ishaque K / Handwritten-Digit-Recognition-using-Deep-Learning. This repository is keep for tracking updates from upstream branch, anyway updates from all repositories will be synced as soon as possible.

Ready-to-Code codebeat badge

Published Paper

IJARCET-VOL-6-ISSUE-7-990-997

Requirements

Python 3.5 +

Scikit-Learn (latest version)

Numpy (+ mkl for Windows)

Matplotlib

Usage

1. Download the four MNIST dataset files from this link:

http://yann.lecun.com/exdb/mnist/

2. Unzip and place the files in the dataset folder inside the MNIST_Dataset_Loader folder under each ML Algorithm folder i.e :

KNN

|_ MNIST_Dataset_Loader

  |_ dataset

    |_ train-images-idx3-ubyte

    |_ train-labels-idx1-ubyte

    |_ t10k-images-idx3-ubyte

    |_ t10k-labels-idx1-ubyte

Do this for SVM and RFC folders and you should be good to go.

3. To run the code, navigate to one of the directories for which you want to run the code using command prompt:

cd 1. K Nearest Neighbors/

and then run the file "knn.py" as follows:

python knn.py

or

python3 knn.py

This will run the code and all the print statements will be logged into the "summary.log" file.

NOTE: If you want to see the output to print on the Command prompt, just comment out line 16, 17, 18, 106 and 107 and hence you will get all the prints on the screen.

Alternatively, you can also use PyCharm to run the code and run the ".py" file in there.

Repeat the steps for SVM and RFC code.

4. To run the CNN code, you don't need to provide in the MNIST dataset as it'll be downloaded automatically.

Just run the file as :

python CNN_MNIST.py

or

python3 CNN_MNIST.py

and it should run fine.

5. If you want to save the CNN model weights after training, run the code with the following arguments:

python CNN_MNIST.py --save_model 1 --save_weights cnn_weights.hdf5

or

python3 CNN_MNIST.py --save_model 1 --save_weights cnn_weights.hdf5

and it should save the model weights in the same directory.

6. To load the saved model weights and avoid the training time again, use the following command:

python CNN_MNIST.py --load_model 1 --save_weights cnn_weights.hdf5

or

python3 CNN_MNIST.py --load_model 1 --save_weights cnn_weights.hdf5

and it should load the model and show the Outputs.

Accuracy using Machine Learning Algorithms:

i) K Nearest Neighbors: 96.67%

ii) SVM: 97.91%

iii) Random Forest Classifier: 96.82%

Accuracy using Deep Neural Networks:

i) Three Layer Convolutional Neural Network using Tensorflow: 99.70%

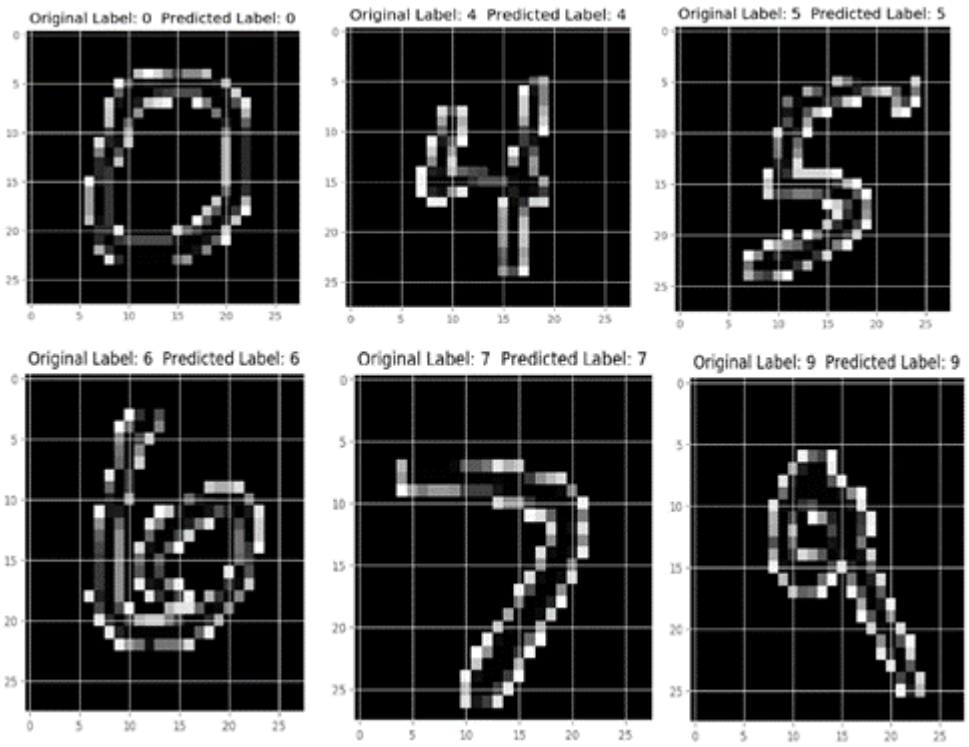
ii) Three Layer Convolutional Neural Network using Keras and Theano: 98.75%

All code written in Python 3.5. Code executed on Intel Xeon Processor / AWS EC2 Server.

Video Link:

https://www.youtube.com/watch?v=7kpYpmw5FfE

Test Images Classification Output:


Original Label: 0 Predicted Label: 0
Original Label: 4 Predicted Label: 4
Original Label: 5 Predicted Label: 5
Original Label: 6 Predicted Label: 6
Original Label: 7 Predicted Label: 7
Original Label: 9 Predicted Label: 9