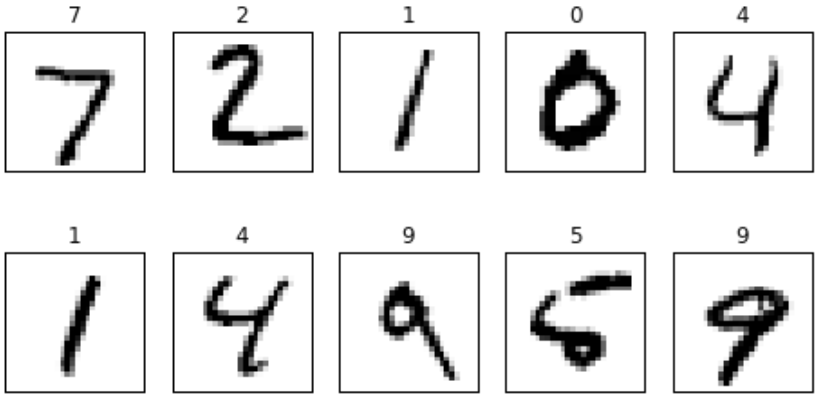
Intelligent Systems: Reasoning and Recognition

James L. Crowley and Nachwa Aboubakr

MOSIG 1 Winter Semester 2018-2019
Exercise 6 7 March 2019

Recognizing Digits using Neural Networks



The MNIST dataset is commonly used for benchmarking purpose in machine learning research. It contains images of handwritten digits from 0 to 9. Each image is 28 x 28 pixels, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255, inclusive. The dataset contains 70.000 images; a sample is shown below. Dataset files are split into train and test files. Training files should be used for designing (while training) a model, while testing files are used for testing the obtained model (network).

Given these gray-scale MNIST images, design a neural network model that recognizes hand-drawn digits; the model can be consist of convolution layers, fully connected layers, auto-encoder, or a combination of them. The model can has weights pre-trained on a different dataset or trained from scratch.

- use Keras with tensorflow backend libraries to help you in completing this exercise.
- In building the model use data augmentation techniques to acquire more data examples for training and thus probably better generalizing network.
- choose derivable loss function that evaluates the model. (e.g. cross_entropy)
- choose a suitable optimizer (e.g. gradient descent)
- compare the effect of hyper parameters in your model (batch size, kernel sizes, # of epochs, etc.)
- compute and report error rate as performance metric on test images.

Recommended actions:

Design an artificial neural network model for recognizing digits.
Design an artificial neural network model trained on MNIST dataset that is able to predict new testing examples correctly. Show some failure cases and explain them.
Design an artificial neural network model trained on MNIST dataset that is able to predict new testing examples correctly. The use data augmentation techniques. Showing some failure cases and explain them.
Show and compare the network performance for different hyper parameters values.
Compare the performance between different optimizers and loss functions.
Try different artificial network architectures. Report differences in performance.
impress us!