

Rahul Gupta
April 8, 2017

PROJECT PROPOSAL

Handwritten Number Recognition:
Deep Learning using ConvNets

Contents

1.	DOMAIN BACKGROUND.....	3
2.	PROBLEM STATEMENT.....	3
3.	DATASETS AND INPUTS	3
4.	SOLUTION STATEMENT	4
5.	BENCHMARK MODEL	5
6.	EVALUATION METRICS.....	5
7.	PROJECT DESIGN	6

1. Domain Background

— the field of research where the project is derived

Computer vision is a field of computer science that works on enabling computers to see, identify and process images in the same way that human vision does, and then provide appropriate output.

Computer vision is closely linked with artificial intelligence, as the computer must interpret what it sees, and then perform appropriate analysis or act accordingly and provide useful results based on the observation.

2. Problem Statement

— a problem being investigated for which a solution will be defined

Computer vision is like imparting human intelligence and instincts to a computer. In reality though, it is a difficult task to enable computers to recognize images of different objects. The difficulty of visual pattern recognition becomes apparent if you attempt to write a computer program to recognize handwritten digits. What seems easy when we do it ourselves suddenly becomes extremely difficult.

3. Datasets and Inputs

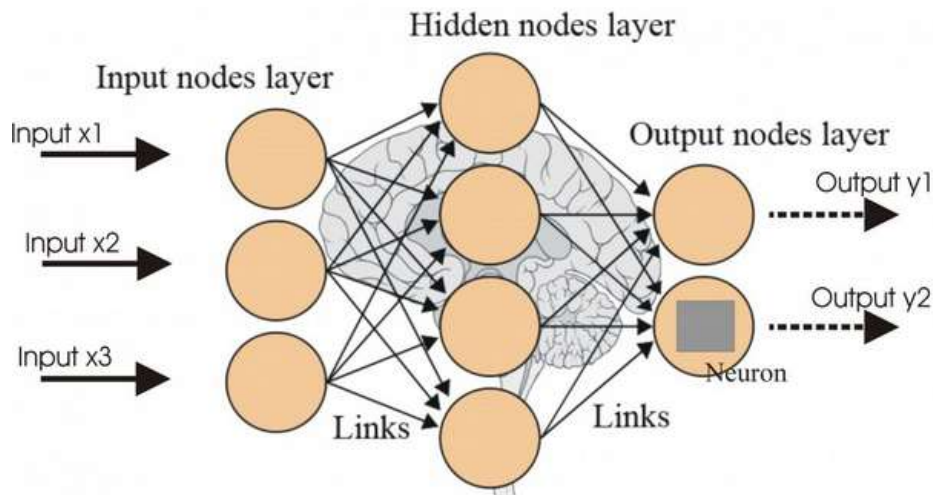
— data or inputs being used for the problem

MNIST is a modified subset of two datasets collected by the U.S. National Institute of Standards and Technology. It contains 70,000 labeled images of handwritten digits. The MNIST database of handwritten digits, has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image. Each image is 28 pixels in height and 28 pixels in width, 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.

4. Solution Statement

— a the solution proposed for the problem given

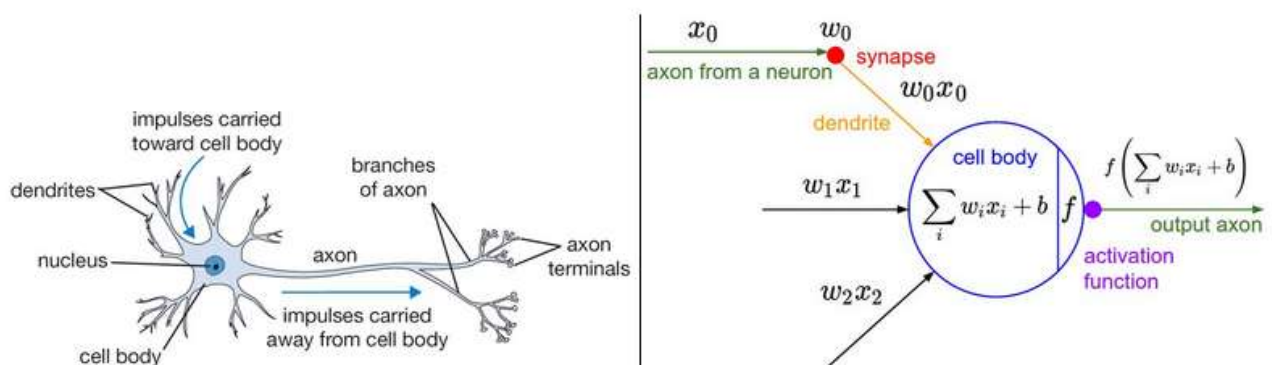
Neural networks and deep learning currently provide the best solutions to many problems in image recognition, speech recognition, and natural language processing.



Basic structure of an Artificial Neural Network (ANN)
(Source: FutureHumanEvolution.com)

A neural network is a biologically-inspired programming paradigm which enables a computer to learn from observational data. This artificial intelligence technique mimics the operation of the human brain and comprises of densely interconnected computer processors working simultaneously. The key feature of neural networks is that they are programmed to 'learn' by sifting data repeatedly, looking for relationships to build mathematical models, and automatically correcting these models to refine them continuously

Deep learning is a powerful set of techniques for learning and implementing the neural networks.



Drawing of a biological neuron (left) and its mathematical model (right)
(Source: stanford.edu)

5. Benchmark Model

— *some simple or historical model or result to compare*

There are multiple historical models are available in public domain to compare results and provide benchmark. Following are few of them:

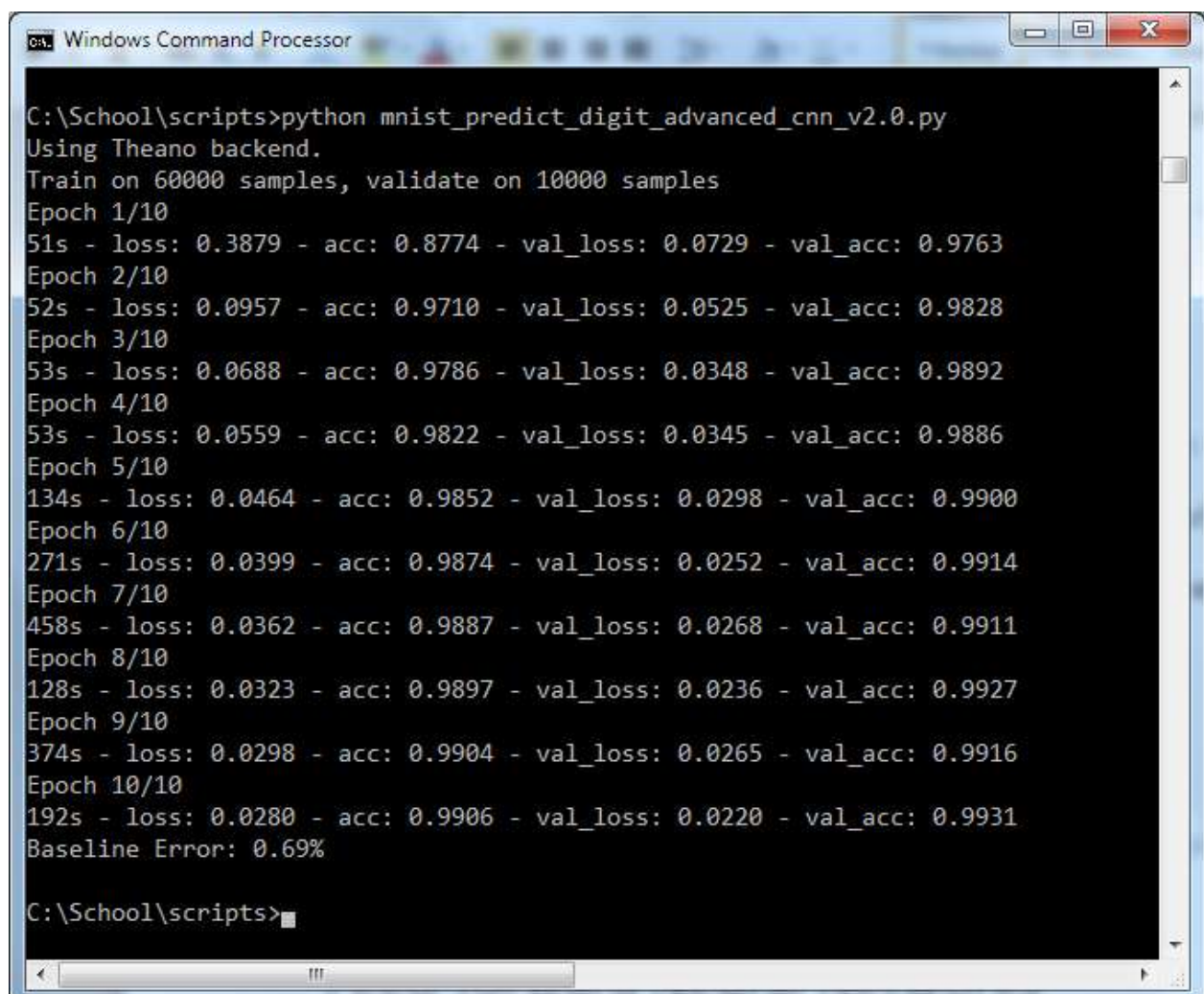
Classifier	Description	Preprocessing	Test Error	Reference
Linear classifiers	linear classifier (1-layer NN)	none	12%	LeCun et al. 1998
Nearest Neighbors	K-nearest-neighbors, Euclidean (L2)	none	3.09%	Kenneth Wilder, U. Chicago
Non Linear Classifiers	1000 RBF + linear classifier	none	3.6	LeCun et al. 1998
SVMs	SVM, Gaussian Kernel	none	1.4%	
Neural Nets	3-layer NN, 500+300 HU, softmax, cross entropy, weight decay	none	1.53%	Hinton, unpublished, 2005

(Source: github.com)

6. Evaluation Metrics

— *functional representations for how the solution can be measured*

Execution returns and displays the the loss value & metrics values for the model in test mode.



```
C:\School\scripts>python mnist_predict_digit_advanced_cnn_v2.0.py
Using Theano backend.
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
51s - loss: 0.3879 - acc: 0.8774 - val_loss: 0.0729 - val_acc: 0.9763
Epoch 2/10
52s - loss: 0.0957 - acc: 0.9710 - val_loss: 0.0525 - val_acc: 0.9828
Epoch 3/10
53s - loss: 0.0688 - acc: 0.9786 - val_loss: 0.0348 - val_acc: 0.9892
Epoch 4/10
53s - loss: 0.0559 - acc: 0.9822 - val_loss: 0.0345 - val_acc: 0.9886
Epoch 5/10
134s - loss: 0.0464 - acc: 0.9852 - val_loss: 0.0298 - val_acc: 0.9900
Epoch 6/10
271s - loss: 0.0399 - acc: 0.9874 - val_loss: 0.0252 - val_acc: 0.9914
Epoch 7/10
458s - loss: 0.0362 - acc: 0.9887 - val_loss: 0.0268 - val_acc: 0.9911
Epoch 8/10
128s - loss: 0.0323 - acc: 0.9897 - val_loss: 0.0236 - val_acc: 0.9927
Epoch 9/10
374s - loss: 0.0298 - acc: 0.9904 - val_loss: 0.0265 - val_acc: 0.9916
Epoch 10/10
192s - loss: 0.0280 - acc: 0.9906 - val_loss: 0.0220 - val_acc: 0.9931
Baseline Error: 0.69%

C:\School\scripts>
```

- Test Accuracy Rate: 99.31%
- Test Error Rate: 0.69%

7. Project Design

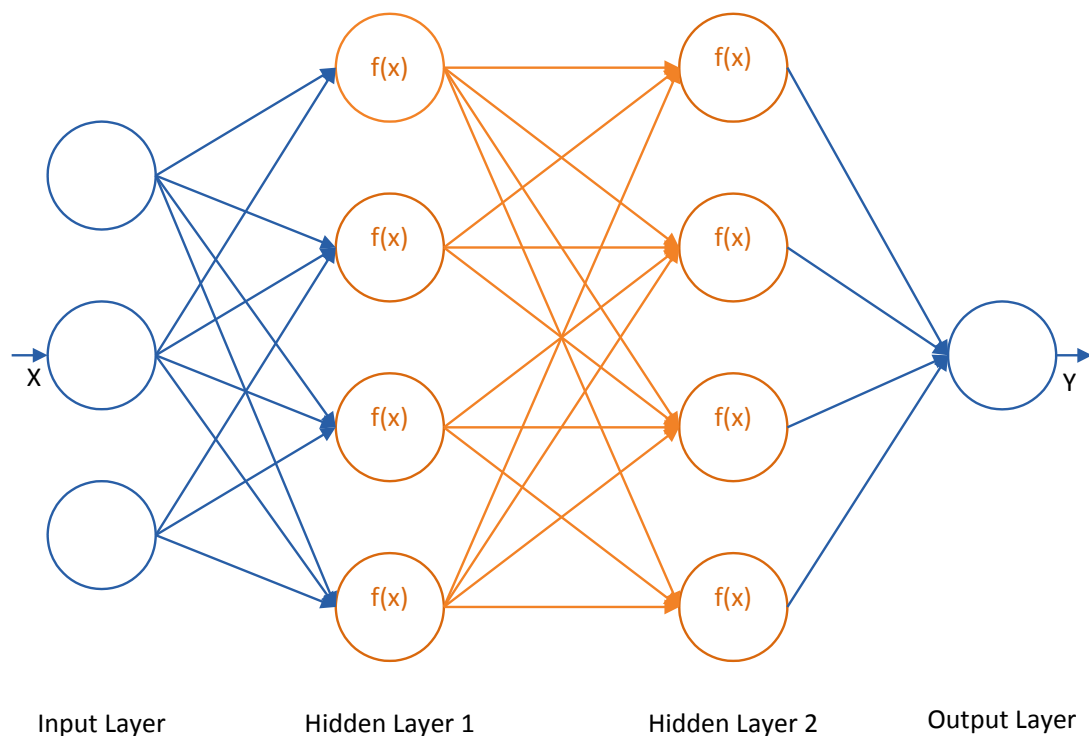
— *how the solution will be developed and results obtained*

Feed-forward Neural Networks

Feed-forward Neural Networks are the simple form of Artificial Neural Networks.

These networks have 3 types of layers: Input layer, hidden layer and output layer. In these networks, data moves from the input layer through the hidden nodes (if any) and to the output nodes.

Below is an example of a fully-connected feed-forward neural network with 2 hidden layers. "Fully-connected" means that each node is connected to all the nodes in the next layer.



Convolutional Neural Networks

Convolutional neural networks also referred as ConvNets or CNN are a special type of feed-forward networks. These models are designed to emulate the behavior of a visual cortex. CNNs perform very well on visual recognition tasks. CNNs have special layers called convolutional layers and pooling layers that allow the network to encode certain images properties. We will be using ConvNets approach in order to recognize the handwritten digit images provided in MNIST dataset.

The neural network topology can be summarized as follows:

1. The first hidden layer is a convolutional layer called a Convolution2D. The layer has 32 feature maps, which with the size of 5×5 and a rectifier activation function. This is the input layer, expecting images with the structure outline of [pixels][width][height].
2. Pooling layer that takes the max called MaxPooling2D. It is configured with a pool size of 2×2.
3. Convolutional layer with 16 feature maps of size 3×3.
4. Pooling layer taking the max over 2*2 patches.
5. A regularization layer using dropout called Dropout. It is configured to randomly exclude 20% of neurons in the layer in order to reduce over fitting.
6. Next layer converts the 2D matrix data to a vector called Flatten. It allows the output to be processed by standard fully connected layers.
7. Fully connected layer with 128 neurons and rectifier activation function.
8. Fully connected layer with 50 neurons and rectifier activation.
9. The output layer has 10 neurons for the 10 classes and a softmax activation function to output probability-like predictions for each class.