Handwriting recognition:

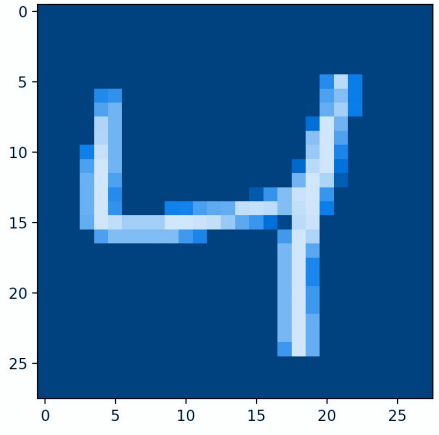
First steps into deep learning

Eliana Garcia

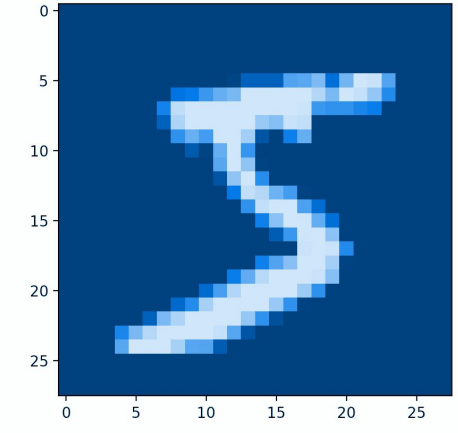
Aim of the project

Handwriting recognition

- Apply **neural networks** and **deep learning** to do **image recognition**
- Compare two deep neural network (**DNN**) architectures: **“flat”** relational data (vectorized the 2D matrix) and **“convolutional”** approach (2D matrix)
- **Extra challenge:** only 1000 sets from the 60,000 (due to limitation in computational power)



- Data set: The *MNIST Handwritten Digit Classification Challenge*
  - MNIST is a modified subset collected by the U.S. National Institute of Standards and Technology. It contains 60,000 labeled images of handwritten digits. Each image is 28 pixels by 28 pixels



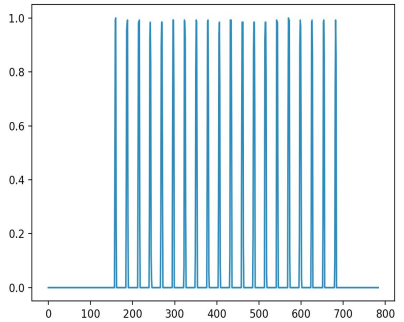
Hypothesis

Handwriting recognition

- Classification performance is at least of 80% for distinguishing among 10 different handwriting digits (0-9).
- The “flat” DNN’s accuracy is comparable to the “convolutional” DNN.

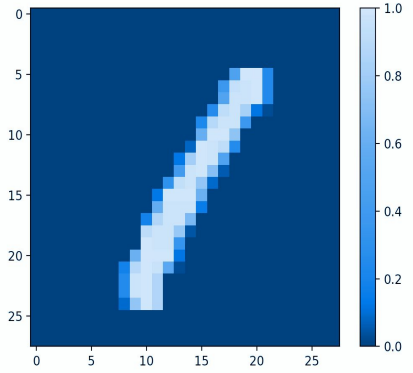
Handwriting: number 1 in 1D plot

“Flat” DNN



Handwriting: number 1 in 2D image

“Convolutional” DNN



Data evaluation process

Handwriting recognition

Import libraries: matplotlib, numpy, keras, tensorflow

Download the data; select training, test data sets and label and transform either as vector or 2D

Create the model;add the input layer, add hidden layers; add output layer and compile

Fit the model with training data set and test the model with test data set

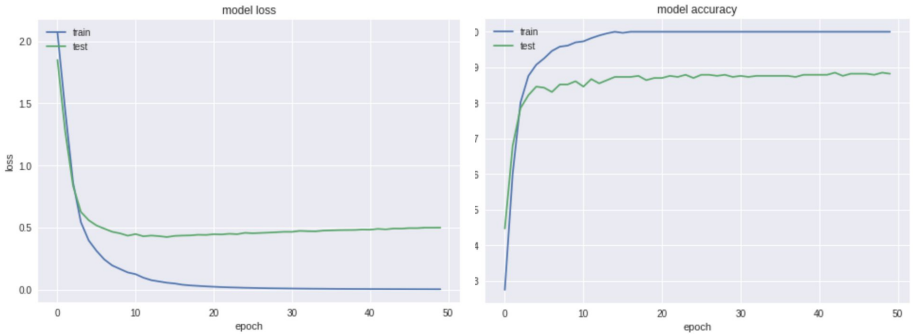
Compare loss and accuracy between models

Handwriting recognition

Results

"Flat" DNN

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 50)                39250
_________________________________________________________________
dense_2 (Dense)              (None, 50)                2550
_________________________________________________________________
dense_3 (Dense)              (None, 10)                510
=================================================================
Total params: 42,310
Trainable params: 42,310
Non-trainable params: 0
_________________________________________________________________
None
```

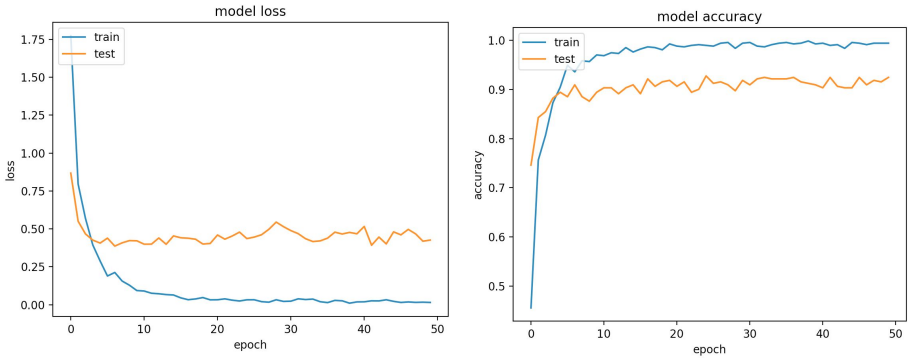


Using atom, terminal

"Convolutional" DNN

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 32, 26, 26)	320
conv2d_2 (Conv2D)	(None, 32, 24, 24)	9248
max_pooling2d_1 (MaxPooling2	(None, 32, 12, 12)	0
dropout_1 (Dropout)	(None, 32, 12, 12)	0
flatten_1 (Flatten)	(None, 4608)	0
dense_1 (Dense)	(None, 128)	589952
dropout_2 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 10)	1290

Total params: 600,810
Trainable params: 600,810
Non-trainable params: 0



Using jupyter

Conclusion

Handwriting recognition

- Classification performance was higher than 80%
- The "flat" DNN's accuracy was comparable to the "convolutional" DNN, but the "convolutional" DNN was slightly superior but required longer training time:
  - **"Convolutional DNN"**: accuracy 0.91, loss 0.48, estimated running time 214.51 seconds (2,7 GHz Intel Core i5)
  - **"Flat" DNN**: accuracy 0.84, loss 0.61, estimated running time 2.66 seconds (2,7 GHz Intel Core i5)

Taken from:
https://ahmedbesbes.com/understanding-deep-convolutional-neural-networks-with-a-practical-use-case-in-tensorflow-and-keras.html

This is me…

It was a lot of fun…. Python, machine learning, deep neural networks, jupyter, atom

Exciting references

Handwriting recognition

- EliteDataScience, *Keras Tutorial;* 07/22/18 accessed; https://elitedatascience.com/keras-tutorial-deep-learning-in-python#step-5
- Dan Becker, *Deep learning in Python;* 12/01/17 accessed; https://www.datacamp.com/courses/deep-learning-in-python
- Keras Team, Trains a simple convnet on the MNIST dataset; 07/22/18 accessed; https://github.com/keras-team/keras/blob/master/examples/mnist_cnn.py
- Michael Nielsen, *Neural Networks and Deep Learning;* 03/21/18 accessed; http://neuralnetworksanddeeplearning.com/index.html