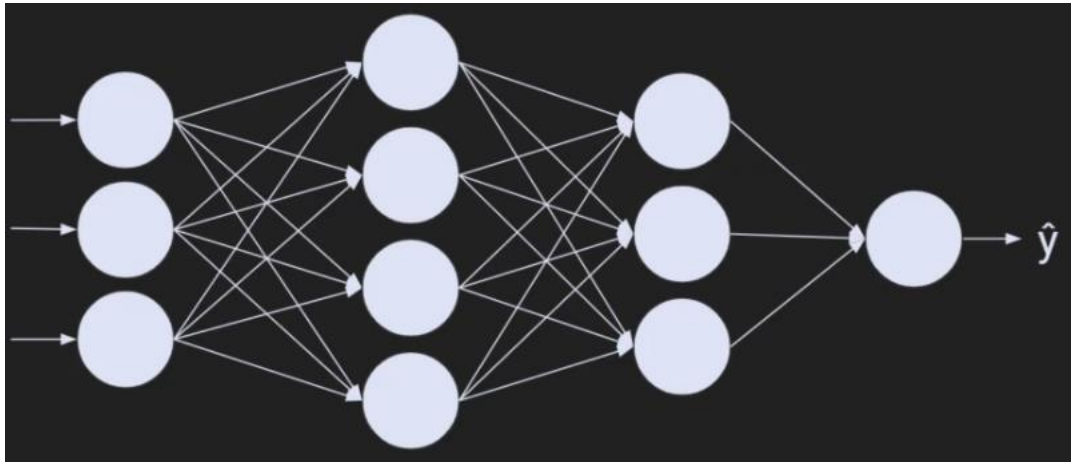
Deep Dive into FFN

Farid Afzali, Ph.D., P.Eng.

FFN

- All samples are processed the same
- All features are treated the same
- Weights remain numerically stable
- Z transform shifts and stretches, but doesn’t change the shape
- Min_max scaling is common for images and uniform-data
- Z_scoring is common for data that are normally distributed.



MNIST Dataset

MNIST database

12 languages

Article Talk Read Edit View history Tools

From Wikipedia, the free encyclopedia

The **MNIST database** (*Modified National Institute of Standards and Technology database*[1]) is a large database of handwritten digits that is commonly used for training various image processing systems.[2][3] The database is also widely used for training and testing in the field of machine learning.[4][5] It was created by "re-mixing" the samples from NIST's original datasets.[6] The creators felt that since NIST's training dataset was taken from American Census Bureau employees, while the testing dataset was taken from American high school students, it was not well-suited for machine learning experiments.[7] Furthermore, the black and white images from NIST were normalized to fit into a 28x28 pixel bounding box and anti-aliased, which introduced grayscale levels.[7]

Sample images from MNIST test dataset

The MNIST database contains 60,000 training images and 10,000 testing images.[8] Half of the training set and half of the test set were taken from NIST's training dataset, while the other half of the training set and the other half of the test set were taken from NIST's testing dataset.[9] The original creators of the database keep a list of some of the methods tested on it.[7] In their original paper, they use a support-vector machine to get an error rate of 0.8%.[10]

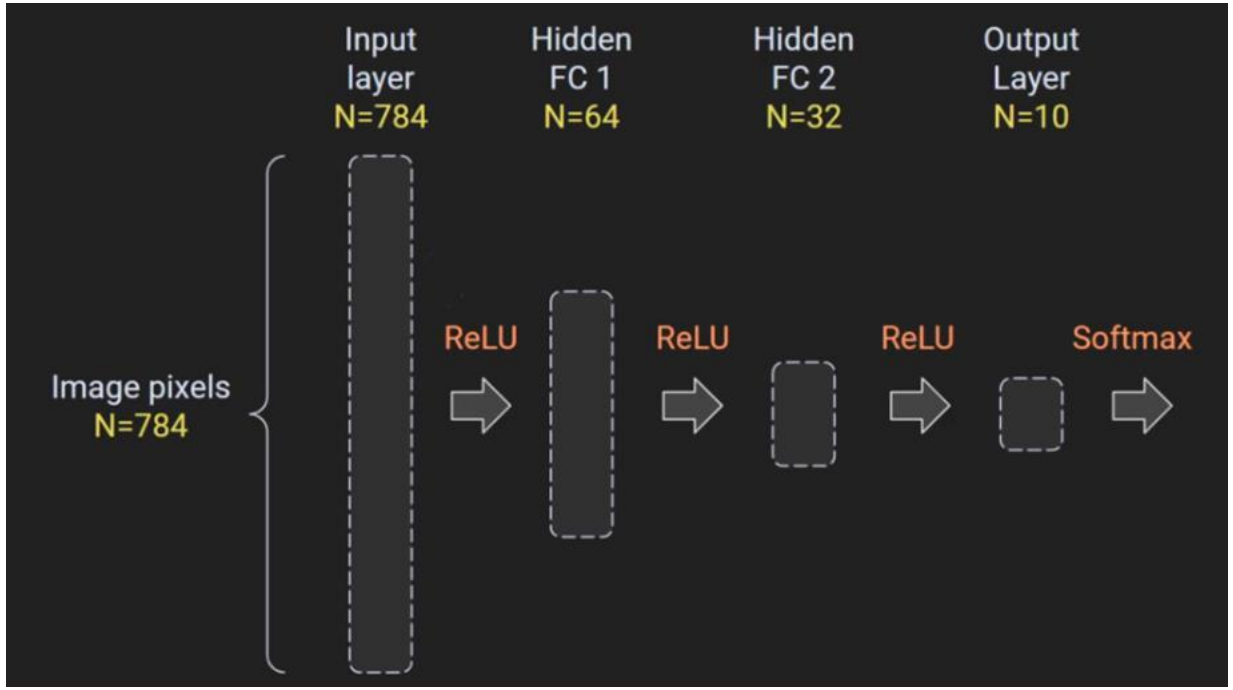
Extended MNIST (EMNIST) is a newer dataset developed and released by NIST to be the (final) successor to MNIST.[11][12] MNIST included images only of handwritten digits. EMNIST includes all the images from NIST Special Database 19, which is a large database of handwritten uppercase and lower case letters as well as digits.[13][14] The images in EMNIST were converted into the same 28x28 pixel format, by the same process, as were the MNIST images. Accordingly, tools which work with the older, smaller, MNIST dataset will likely work unmodified with EMNIST.

MNIST Dataset Different Classifiers

Type	Classifier	Distortion	Preprocessing	Error rate (%)
Linear classifier	Pairwise linear classifier	None	Deskewing	7.6[10]
Decision stream with Extremely randomized trees	Single model (depth > 400 levels)	None	None	2.7[28]
K-Nearest Neighbors	K-NN with rigid transformations	None	None	0.96[29]
K-Nearest Neighbors	K-NN with non-linear deformation (P2DHMDM)	None	Shiftable edges	0.52[30]
Boosted Stumps	Product of stumps on Haar features	None	Haar features	0.87[31]
Non-linear classifier	40 PCA + quadratic classifier	None	None	3.3[10]
Random Forest	Fast Unified Random Forests for Survival, Regression, and Classification (RF-SRC)[32]	None	Simple statistical pixel importance	2.8[33]
Support-vector machine (SVM)	Virtual SVM, deg-9 poly, 2-pixel jittered	None	Deskewing	0.56[34]
Neural network	2-layer 784-800-10	None	None	1.6[35]
Neural network	2-layer 784-800-10	Elastic distortions	None	0.7[35]
Deep neural network (DNN)	6-layer 784-2500-2000-1500-1000-500-10	Elastic distortions	None	0.35[36]
Convolutional neural network (CNN)	6-layer 784-40-80-500-1000-2000-10	None	Expansion of the training data	0.31[37]
Convolutional neural network	6-layer 784-50-100-500-1000-10-10	None	Expansion of the training data	0.27[38]
Convolutional neural network (CNN)	13-layer 64-128(5x)-256(3x)-512-2048-256-256-10	None	None	0.25[22]
Convolutional neural network	Committee of 35 CNNs, 1-20-P-40-P-150-10	Elastic distortions	Width normalizations	0.23[17]
Convolutional neural network	Committee of 5 CNNs, 6-layer 784-50-100-500-1000-10-10	None	Expansion of the training data	0.21[24][25]
Random Multimodel Deep Learning (RMDL)	10 NN-10 RNN - 10 CNN	None	None	0.18[27]
Convolutional neural network	Committee of 20 CNNS with Squeeze-and-Excitation Networks[39]	None	Data augmentation	0.17[40]
Convolutional neural network	Ensemble of 3 CNNs with varying kernel sizes	None	Data augmentation consisting of rotation and translation	0.09[41]

Deep Learning metaparameters

- See a simple FFN model that reaches the accuracy of 90%
- The chance performance is 10%.
- We will learn about log-softmax and see the advantage over the regular softmax.
- Number of units per layer



Softmax log-softmax

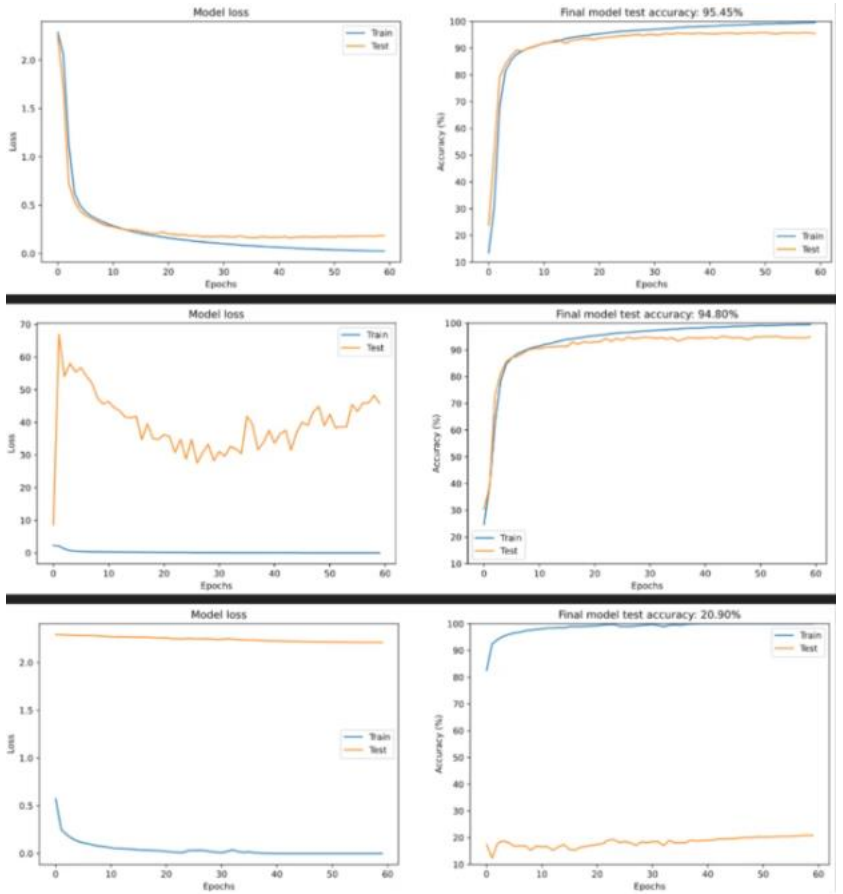
- Log-softmax stretches out the penalties for incorrect guesses which can improve the category separability and increase numerical stability
- Lin-softmax works fine on problems with small number of categories or when categories easily recognizable.

$$\text{lin-softmax} = \frac{e^{z_i}}{\sum e^{z}}$$

$$\text{log-softmax} = \log\left(\frac{e^{z_i}}{\sum e^{z}}\right)$$

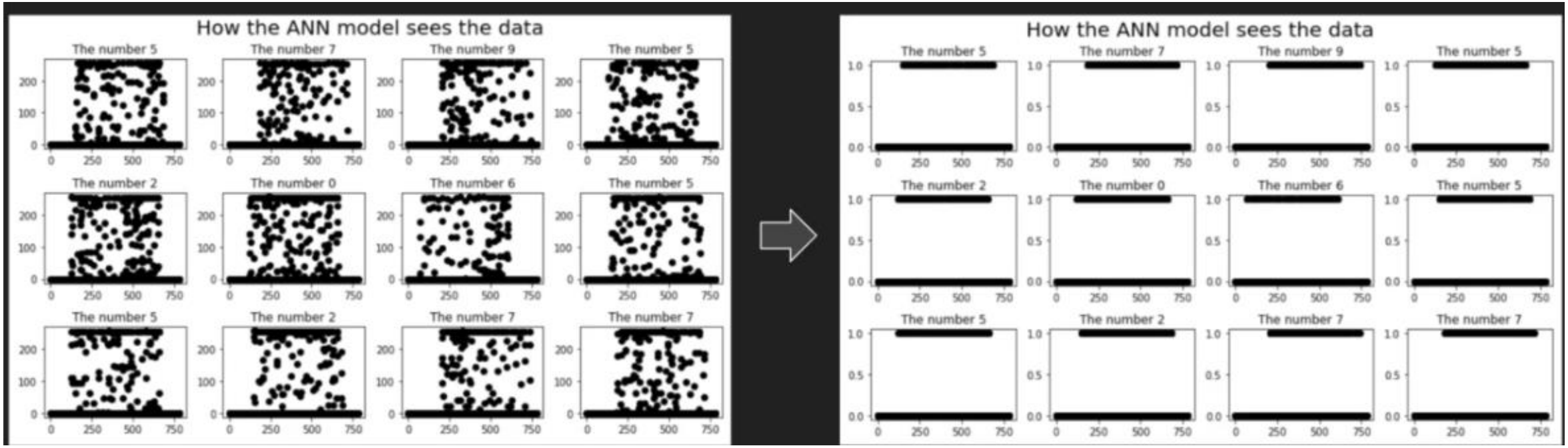
Loss vs accuracy

- Loss scale dependent
- DL models find patterns in the data; numerical range may not be important
- Losses show whether the mode is still learning; accuracy is the metric you really care about



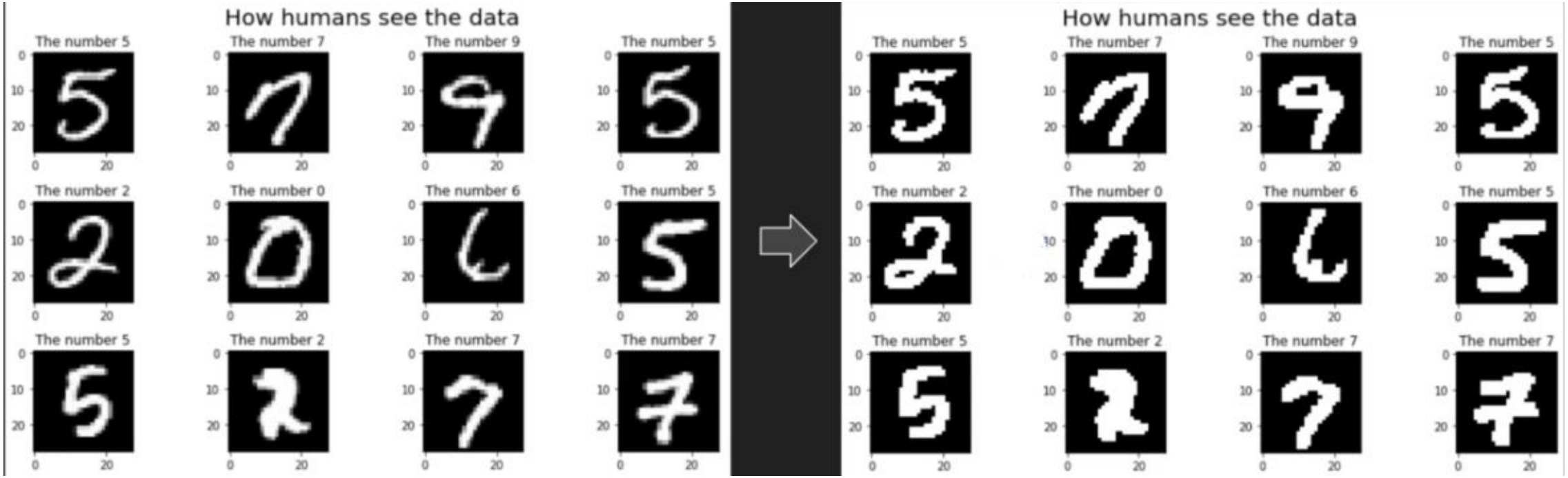
Challenge#1

- Does FNN model needs a range of pixel values?
- Let's binarized the pixel values and see how the model is developed?

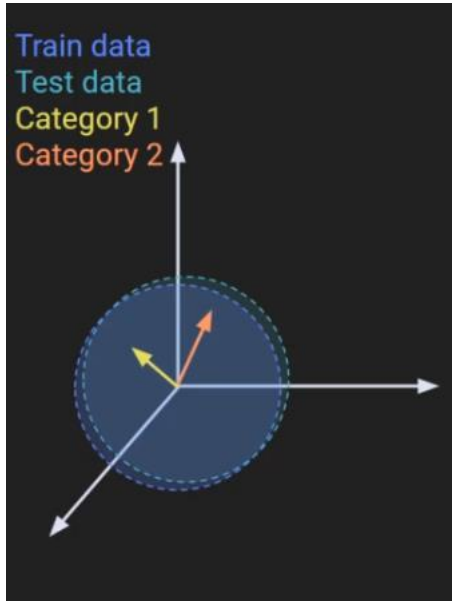


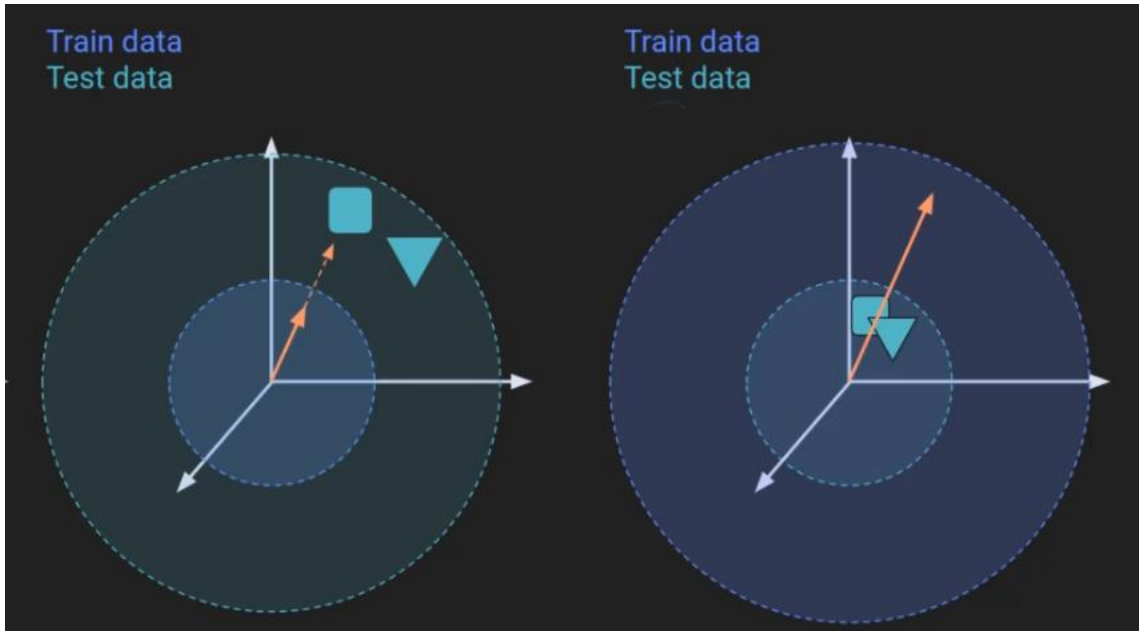
Challenge#1

- Does FNN model needs a range of pixel values?
- Let's binarized the pixel values and see how the model is developed?



Normalization





Loss vs accuracy

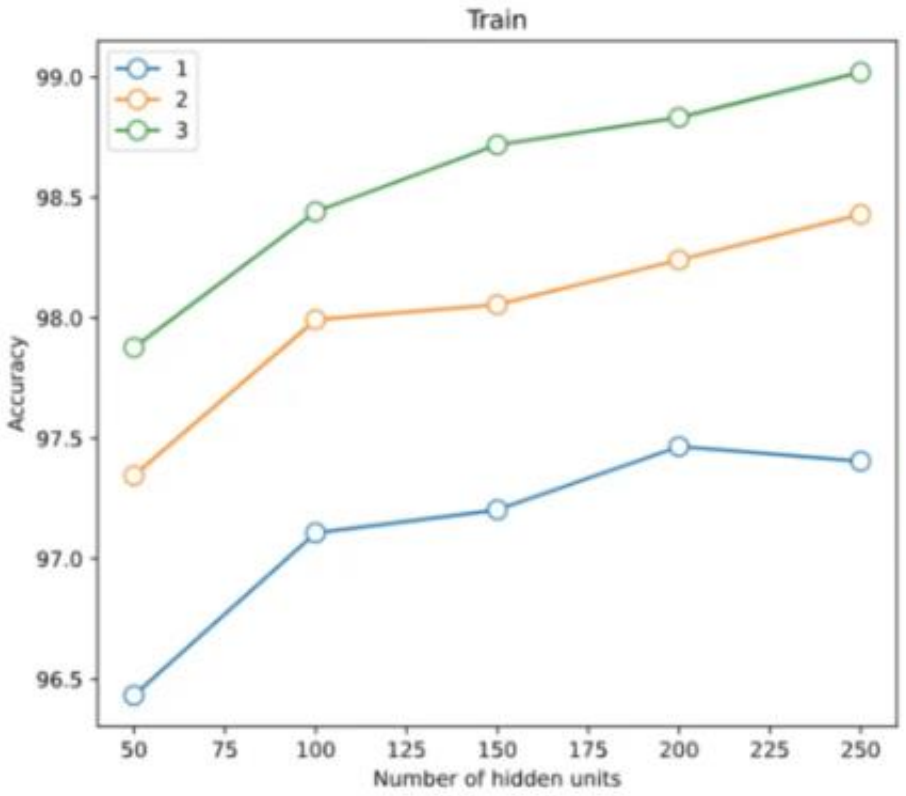
- Alwayne normalize the data
- Normalize the train and test data using the same normalization factor
- Normalization also helps prevent overflow and underflow in weights and gradients.
- Some datasets and models may perform well without normalization, do not assume they will, just do the normalization

Class Activity#1

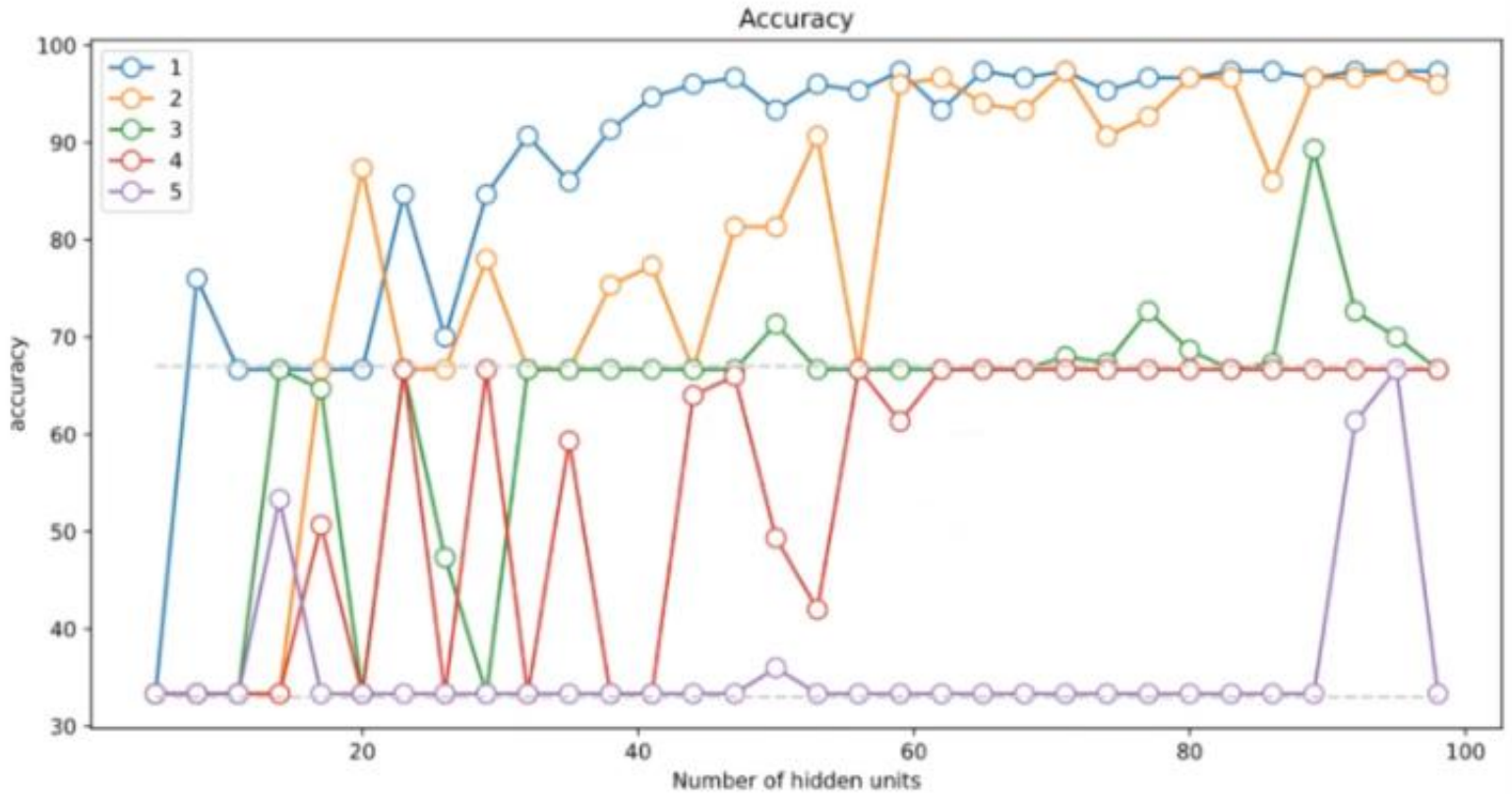
- Start with 02-FFN_FFNonMNIST and ANN-9-Breath vs depth
- Vary the number of hidden layers between 1-3 and number of hidden units between 50-30
- Which models learn the best?

Result might be different in various datasets

MNIST: Deeper models are better



Iris: Shallow models are better



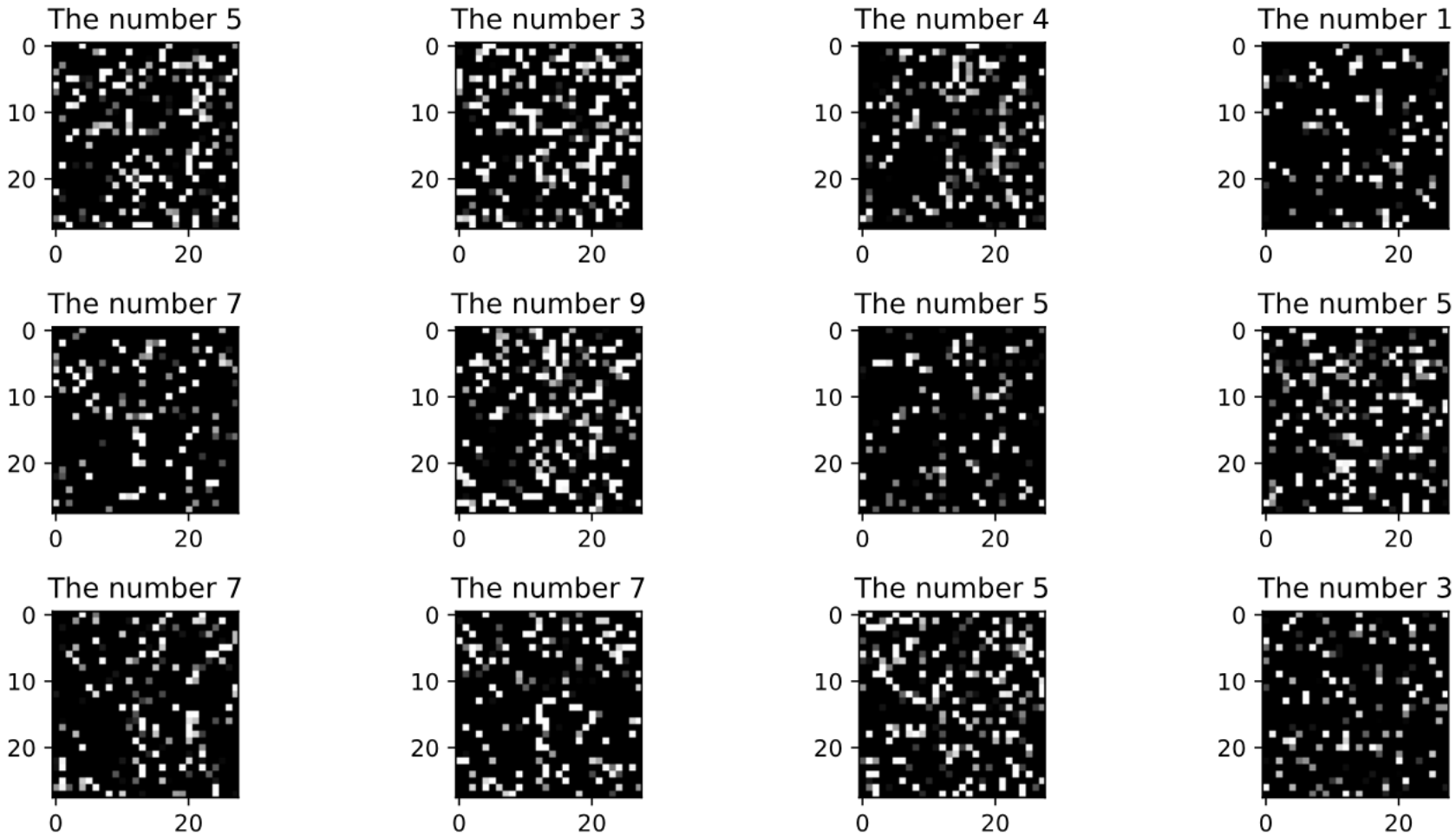
Class Activity- Results

- That is the reason we need to apply experimental scientist for every datasets and examples
- More complex problems do better with deeper network.
- Iris dataset is relatively simple and using multiple layers adds complexity and training time.

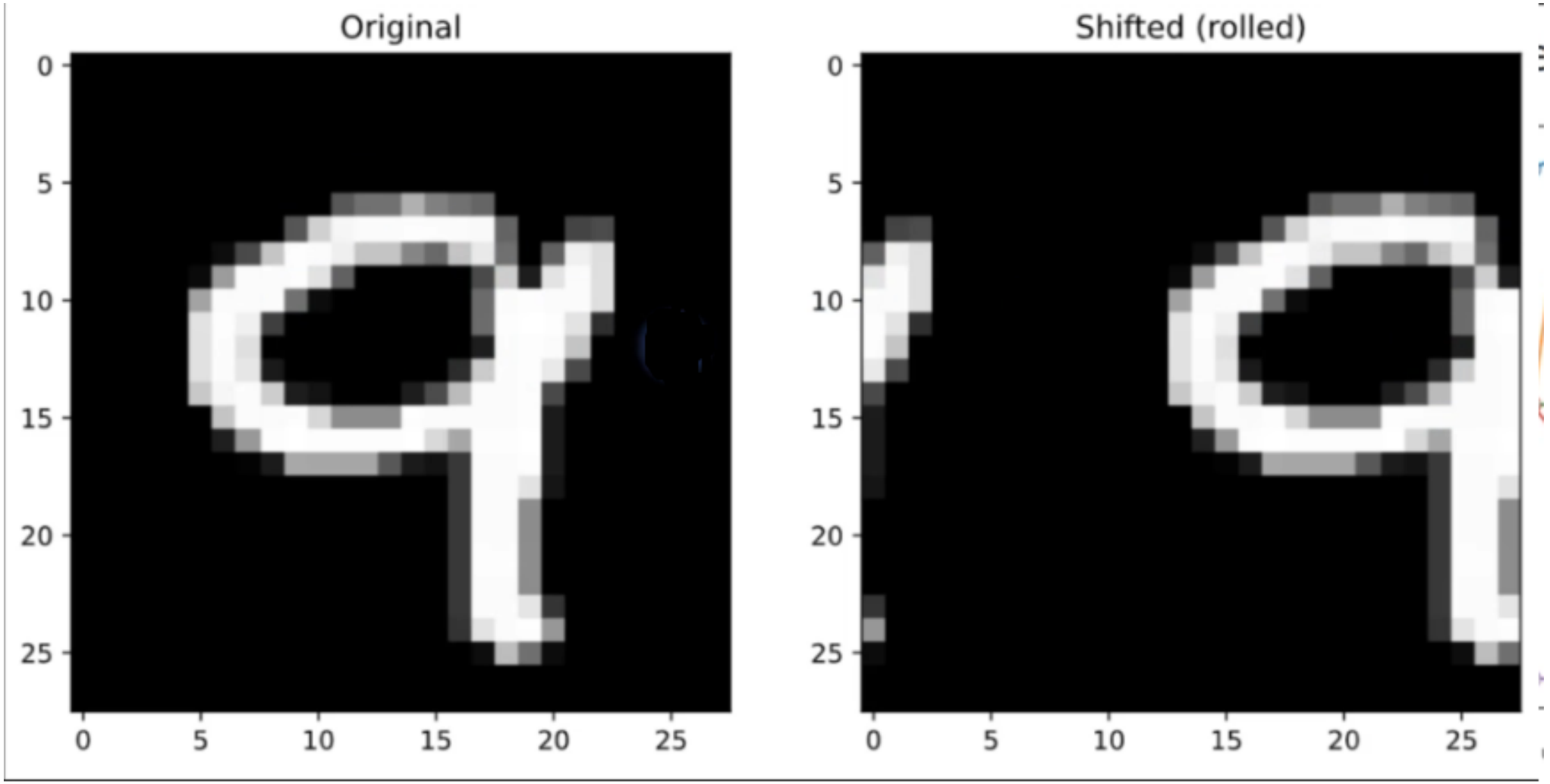
Class Activity#2

- Start with 02-FFN_FFNonMNIST and 14_metaparams_CodeChallengeOptimizers
- Test performance three optimizers(SGD, RMSprop, Adam) and learning rates(0.0001 to 0.1 in 6 log steps)
- Which models learn the best?

Spatial information-FFN



Shifting images-FFN



Shifting images-FFN

- It is obvious that shifting all images (train and test) will not affect categorization performance.
- We are going to see if we leave the training image as-is and shift the test images by a few pixel

Class Activity #3

- Let's assume that we do not have any image of number 7 in our training datasets.
- We are going to figure out how the model is going to predict fot 7 numbers which might be available in the test dataset