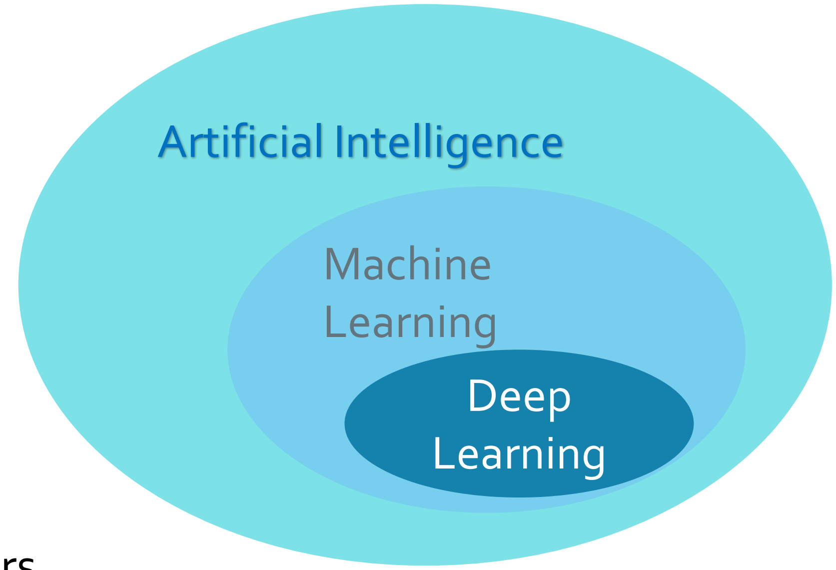
INTRODUCTION TO DEEP LEARNING

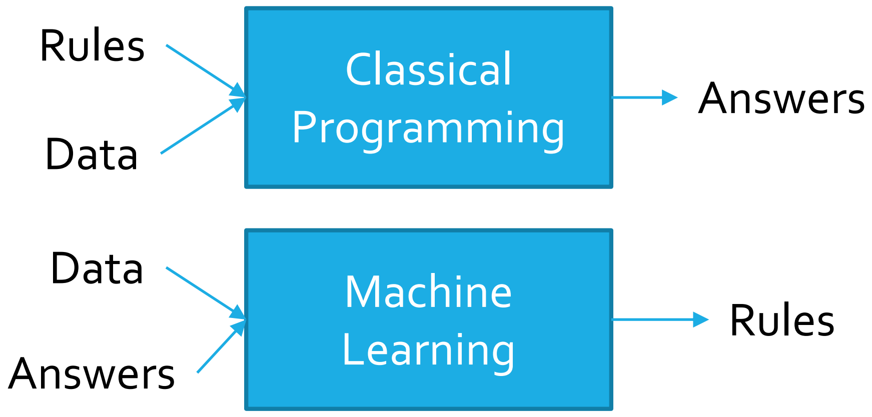
Vishal Gaurav,

PhD Student, Shibata Lab

Deep Learning

- What is AI?
- Symbolic AI
- Machine learning



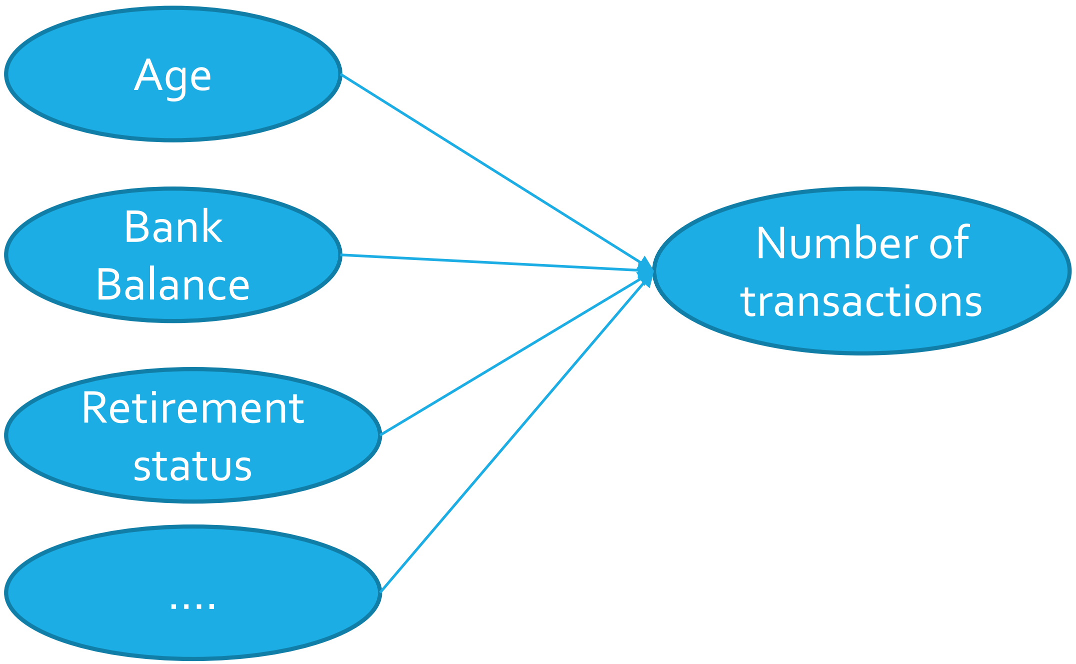


Learning representation from data

- For ML
  - Input data points
  - Examples of the expected output
  - A way to measure weather the algorithm is doing a good job
- DL is a mathematical framework for learning representation from data

Introduction

- Imagine you work for a bank
- Need to predict how many transaction each customer will make next year



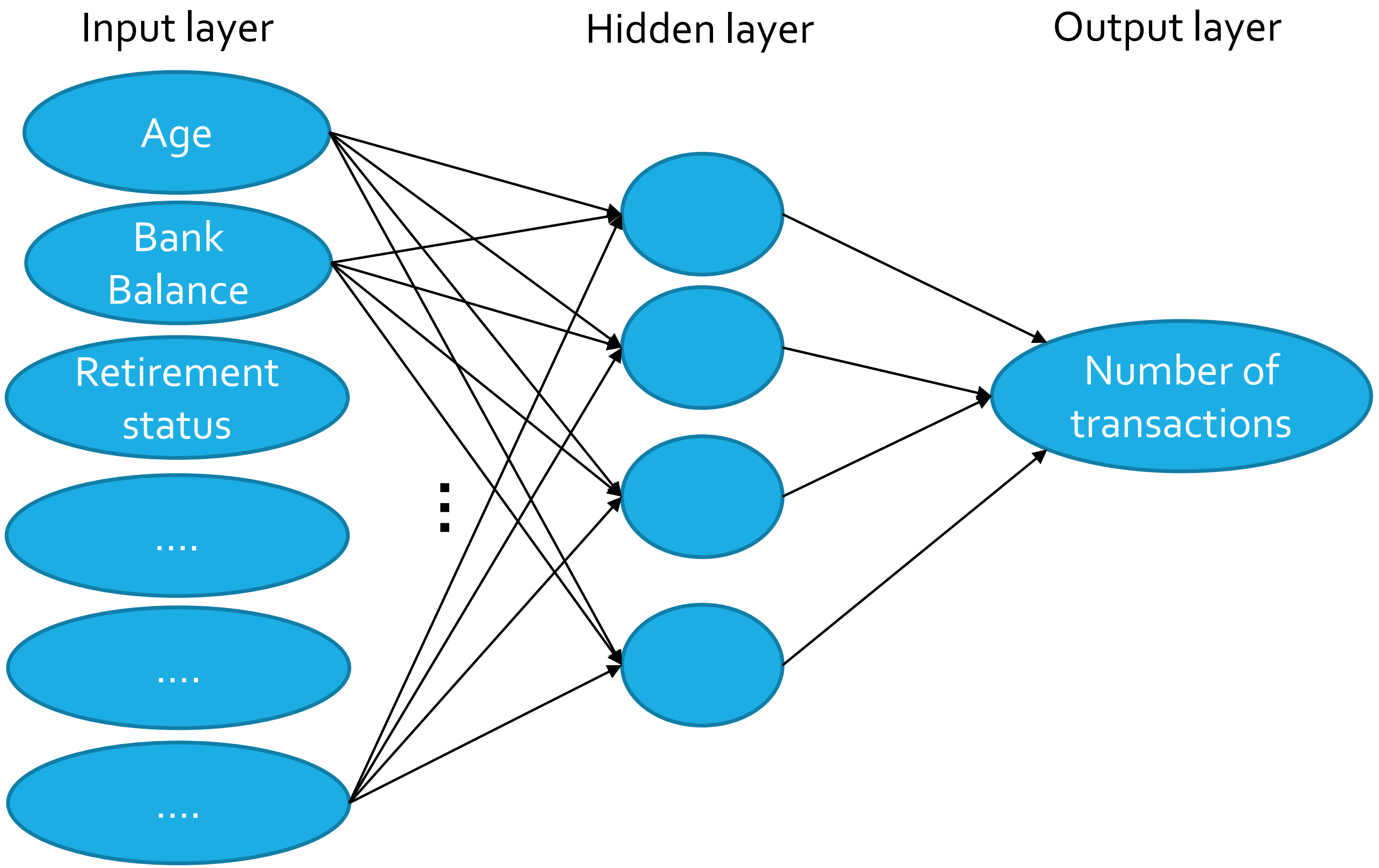
Interaction

- Neural Networks account for interactions really well
- Deep learning uses especially powerful neural networks
- Application
  - Text
  - Images
  - Videos
  - Audio
  - Source code

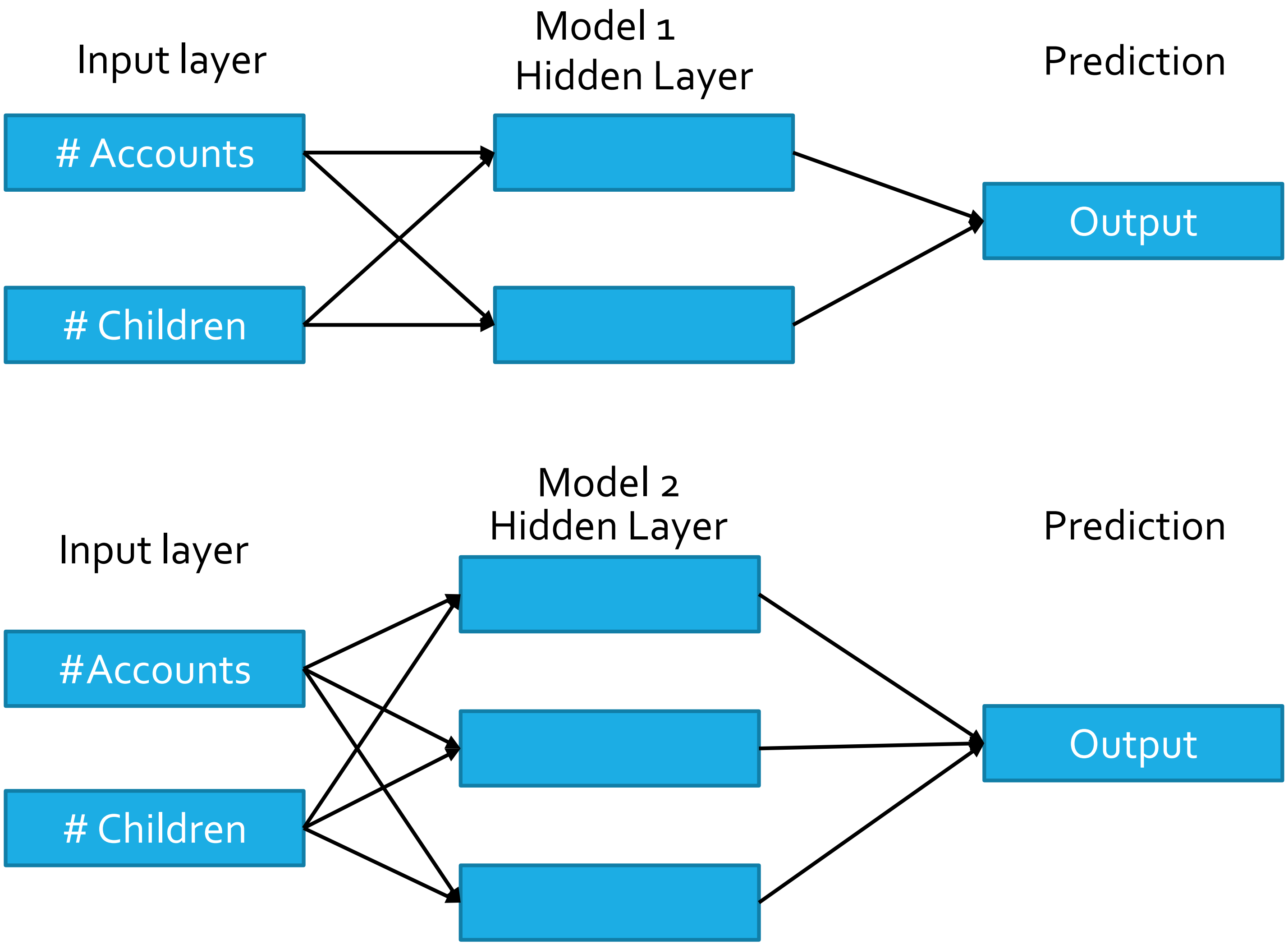
Course structure

- First we focus on conceptual knowledge
  - Debug and tune deep learning models on conventional prediction problems
  - Lay the foundation for progressing towards modern applications

Deep learning models capture interactions



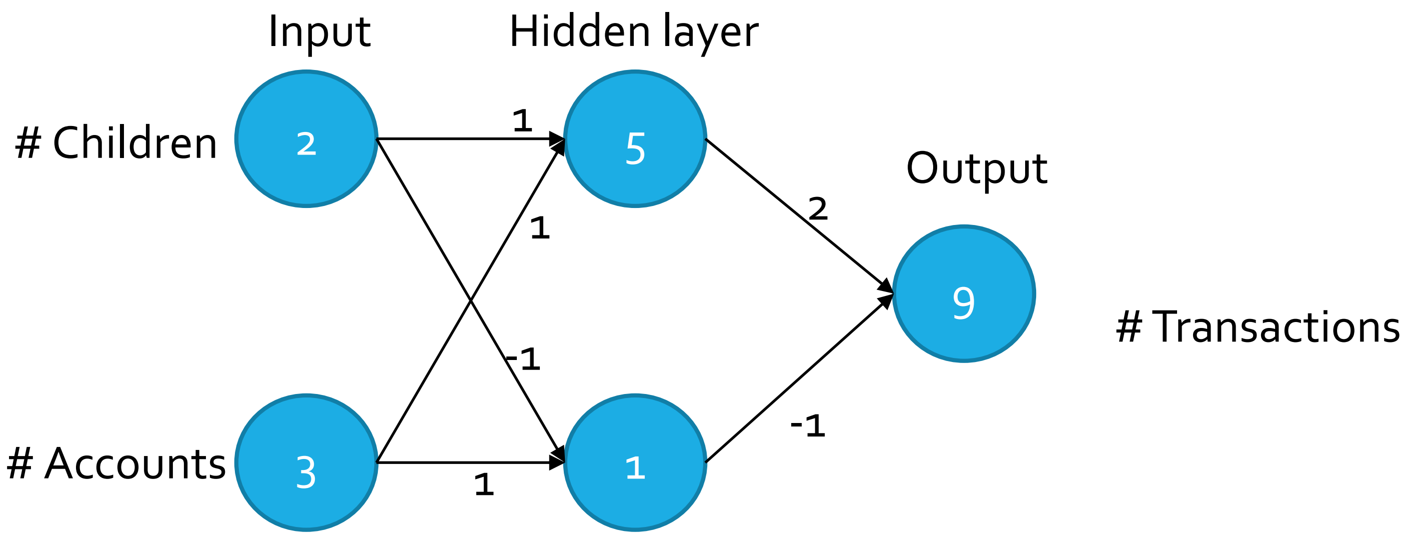
Quiz?



Input layer
Model 1
Hidden Layer
Prediction
Accounts
Children
Output
Input layer
Model 2
Hidden Layer
Prediction
#Accounts
Children
Output

Forward Propagation

- Bank transaction example
- Only using #children and # Accounts

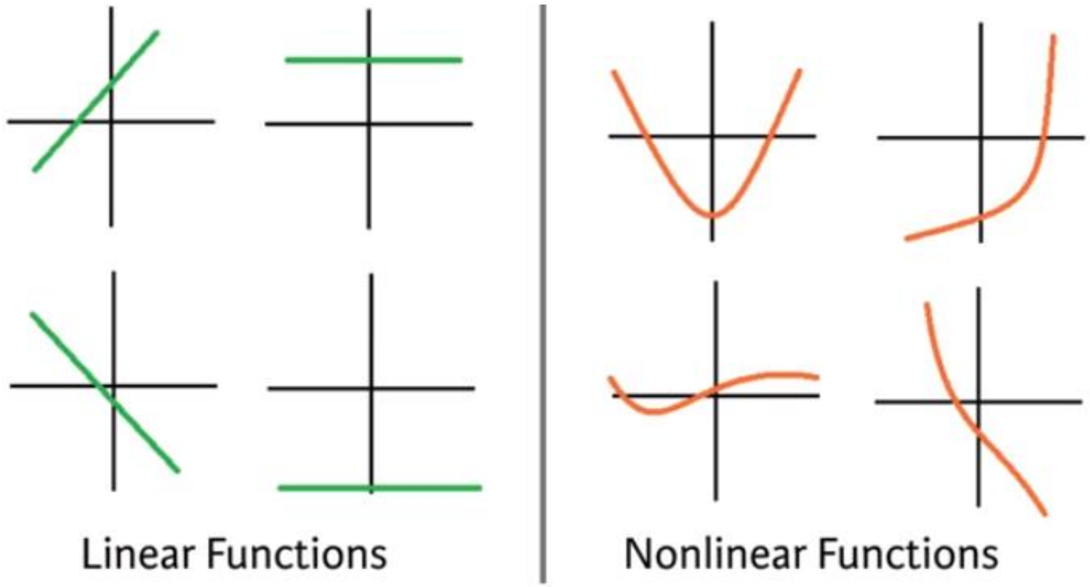


Forward Propagation

- Multiply-add process
- Dot product
- Forward propagation for one data point at a time
- Output is the prediction for that data point

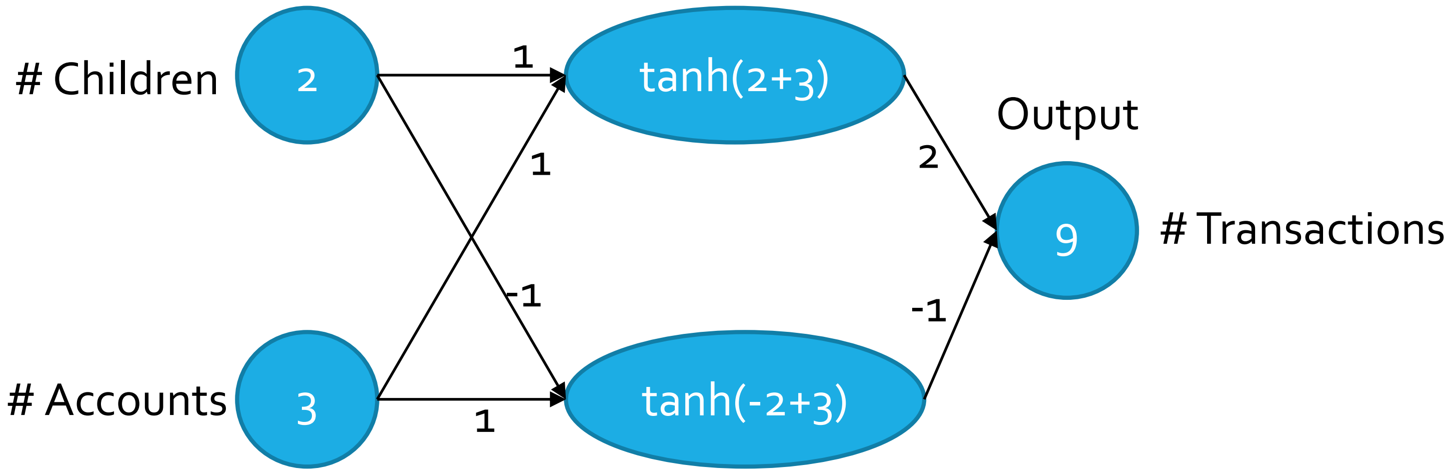
Activation Functions

- Linear vs Non-linear



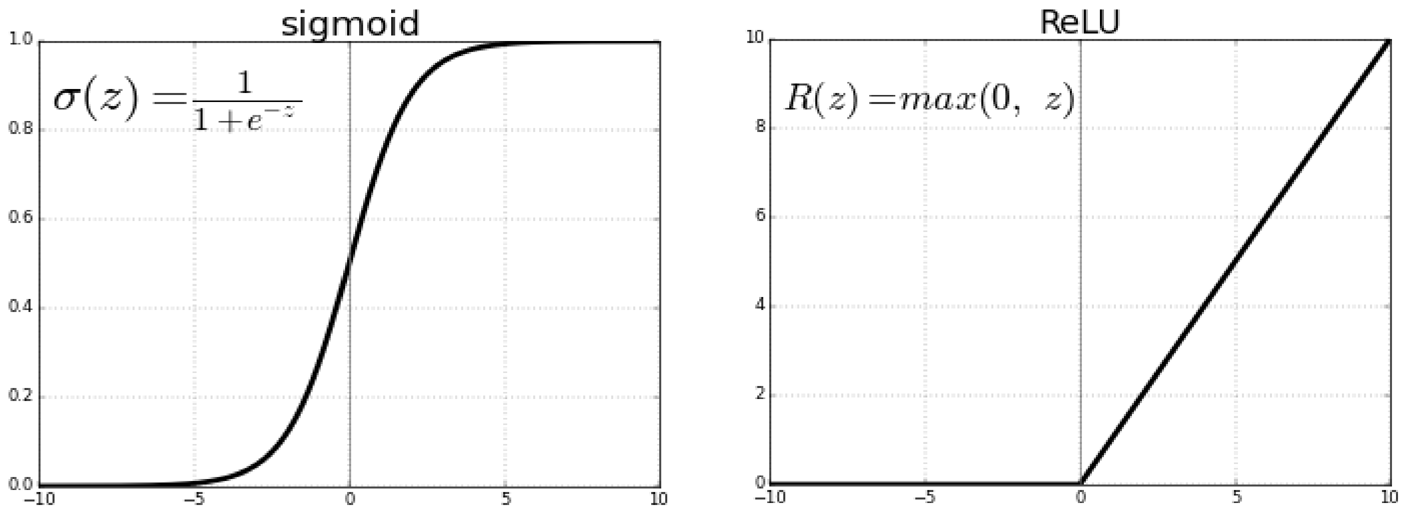
Activation function

- Applied to node inputs to produce node output



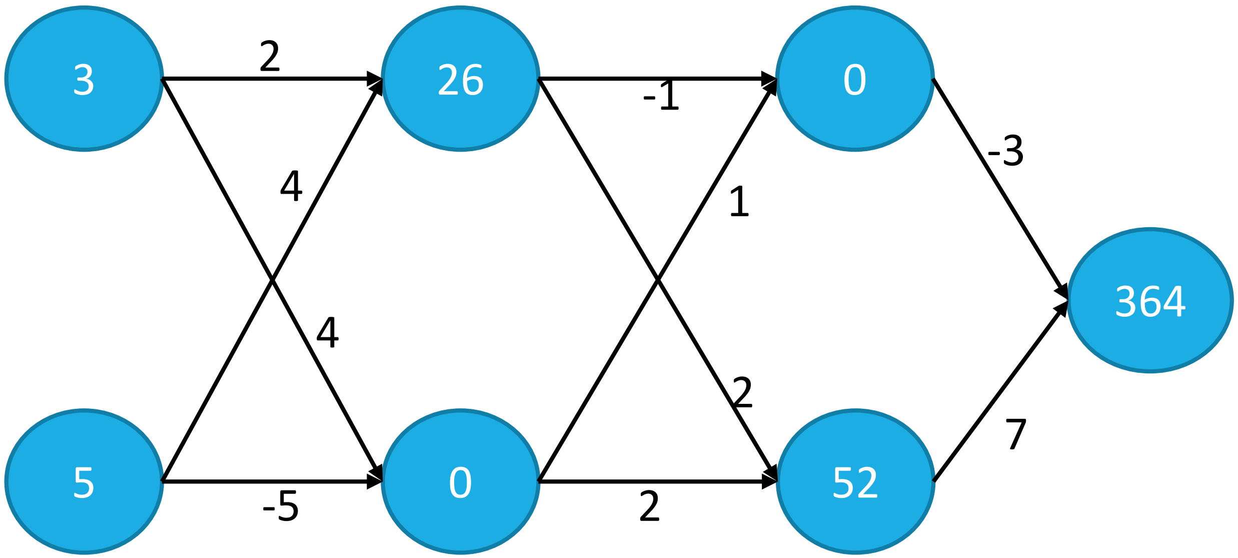
- Eg. Sigmoid, tanh, relu, leakyRelu etc..

ReLU (Rectified Linear Units)



- Defined as the positive part of its argument:
$$f(x) = \max(0, x)$$
- Where *x* is input to neuron
- Introduced by Hahnloser et. Al. in 2000 paper in NATURE.
- The function and its derivative both are monotonic

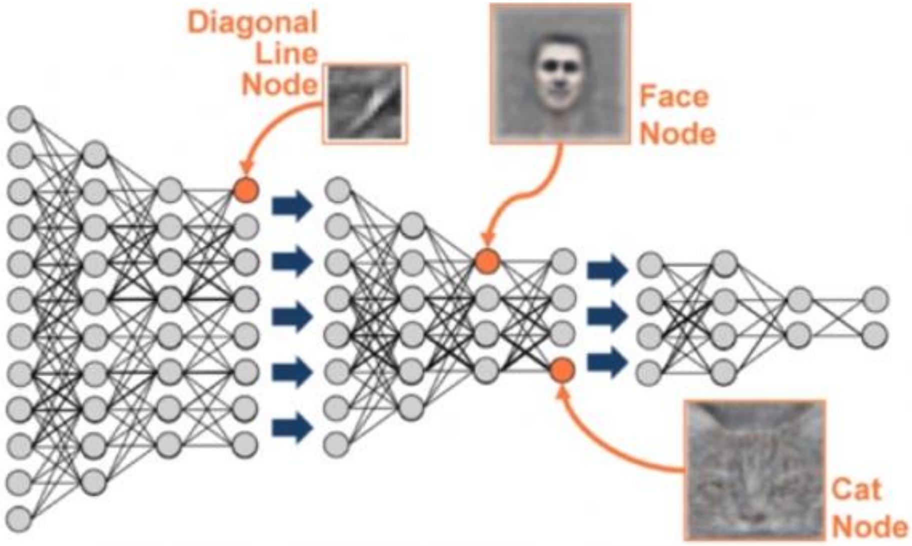
Deeper Networks



Calculated with RELU activation function

Representation learning

- Deep networks internally build representation of patterns in the data
- Partially replace the need for feature engineering
- Subsequent layers build increasingly sophisticated representation of raw data



Deep Learning

- Modeler doesn’t need to specify the interactions
- When you train the model, the neural network gets weights that find the relevant patterns to make better predictions

Back Propagation

- Generative equation

$$y = w^T x + b$$

- Where x is input data
- Y is label/target/ output vector
- W and b are weights and bias

Gradient Decent

- Loss function:

$$L(y_i, \hat{y}_i\ ) = -[y_i \log \hat{y}_i + (1 - y_i) \log \hat{y}_i]$$

- We prefer to use convex loss function
- Cost function: its just average of loss

$$J(W, b) = -\frac{1}{m}\sum_{1}^{m}[y_i \log \hat{y}_i + (1 - y_i) \log \hat{y}_i]$$

Gradient Decent

- Parameter Update:

$$W = W - \alpha \frac{\delta J}{\delta W}$$

$$b = b - \alpha \frac{\delta J}{\delta b}$$

- Where α is learning rate.

Assignments

- Implement CNN classification for MNIST dataset. You can either use Keras or tensorflow or Pytorch
- Visualize the activation output of each layer.