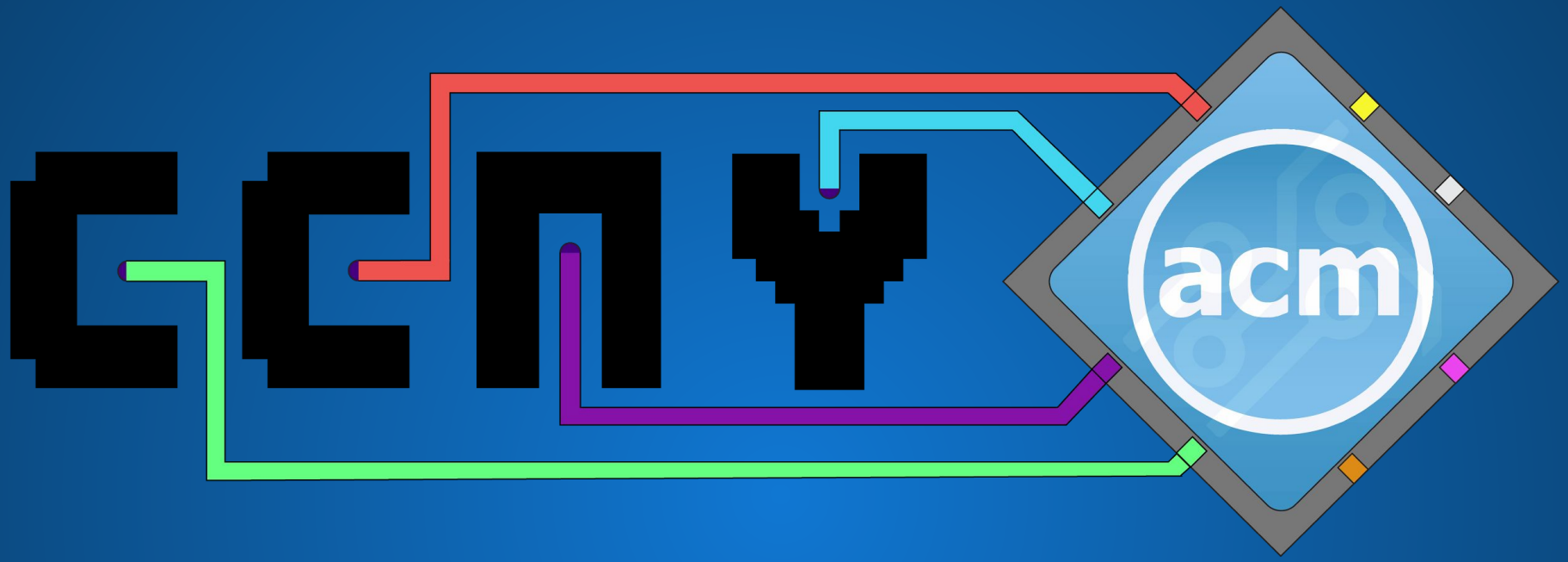


Introduction to Machine Learning Workshop

Framing Question

“How can we build computer systems that automatically improve with experience, and what are the fundamental laws that govern all learning processes?” - Tom Mitchell, Machine Learning Department Head, CMU

What is Machine Learning?

$$p(B \mid A) = \frac{p(A \mid B)\, p(B)}{p(A)}$$

- A branch of artificial intelligence which aims to create systems that learn from data/experience rather than explicit programming
- “ A computer program is said to learn from from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.” - Tom Mitchell, ML Department Head, CMU

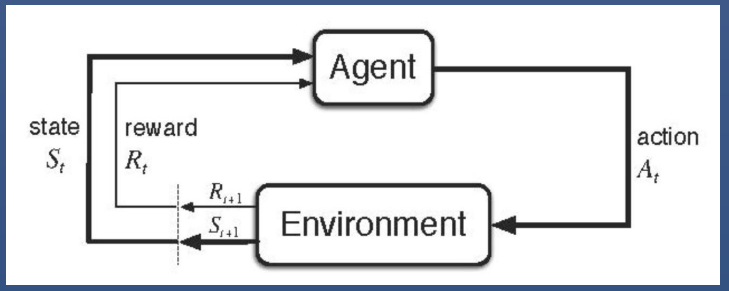
Categories of Machine Learning

- Supervised Learning:
  - Learning from both the features and associated labels of a dataset and developing an estimation of $p(y|x)$
- Unsupervised Learning:
  - Learning the probability distribution or useful properties of the structure of a dataset using only the features
- Reinforcement Learning:
  - Learning from interaction with an environment through actions and corresponding rewards

Examples of Unsupervised Learning

- Clustering
  - ex: K-Means Clustering
- Visualizational and Dimensionality Reduction
  - ex: Principal Component Analysis (PCA)
  - ex: t-distributed Stochastic Neighbor Embedding (t-SNE)
- Neural Networks
  - ex: Generative Adversarial Networks
  - ex: Deep Belief Networks
- Many other interesting techniques

Reinforcement Learning



- Aim to learn control policies for agents interacting with unknown environments
  - ex: Playing Video Games, Self Driving Cars
- Such environments are often formalized as Markov Decision Processes (MDPs) that are defined by the tuple (*S, A, P, R*)
  - An agent at each time step is in some state $s_t \in S$ and takes some action $a_t \in A$ which determines some reward $r_t \sim R(s_t,a_t)$ and next state $s_{t+1} \sim P(s_t,a_t)$

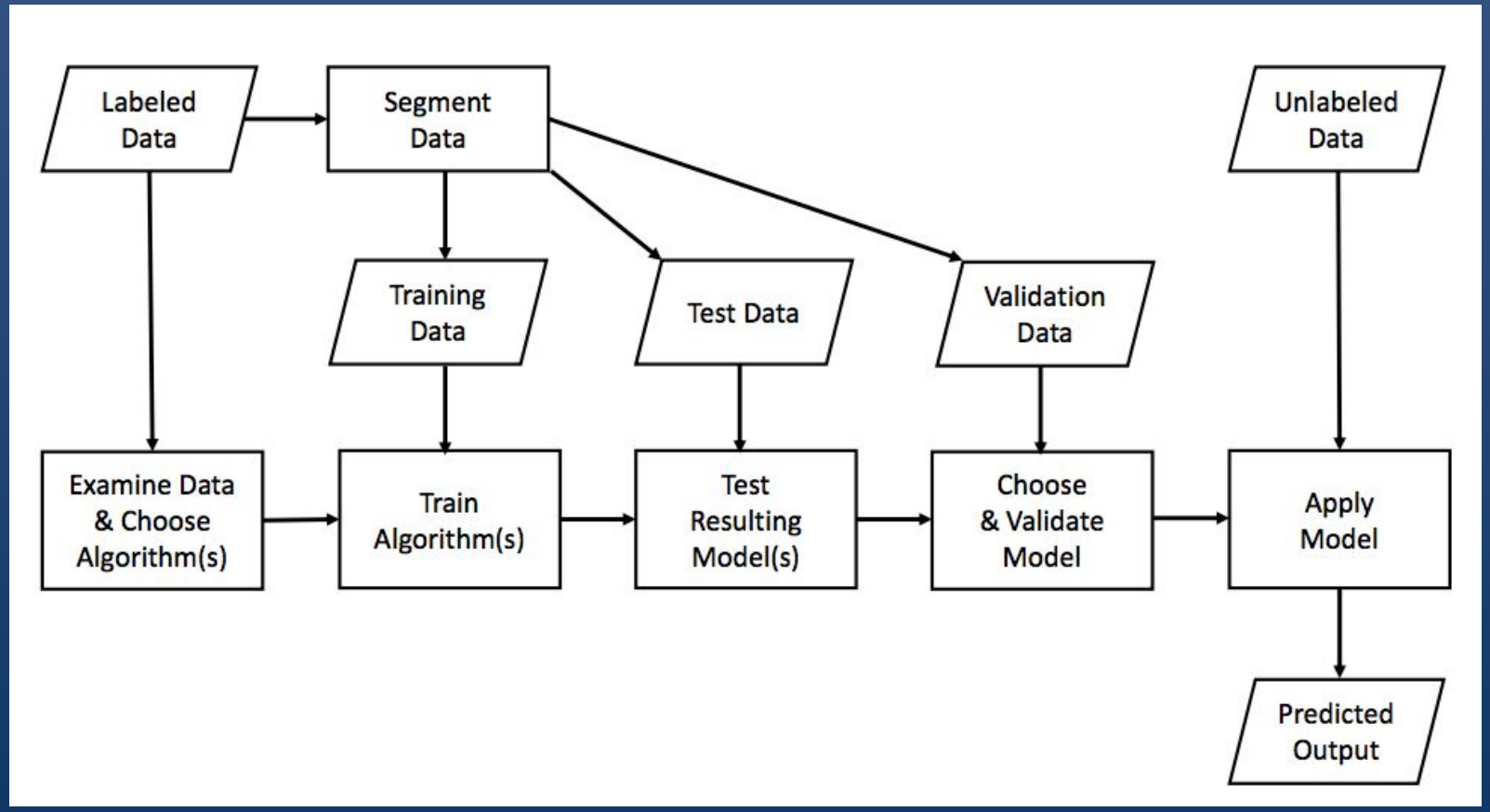
Examples of Reinforcement Learning

- Policy Based:
  - Learn an optimal mapping from each state to its optimal action
  - ex: Policy Iteration
  - ex: Policy Gradient
- Value Based:
  - Learn to approximate the value of each action at each state
  - ex: Q-learning
  - ex: Deep Q-Network (DQN) and Deep Recurrent Q-Networks (DRQN)

Supervised Learning

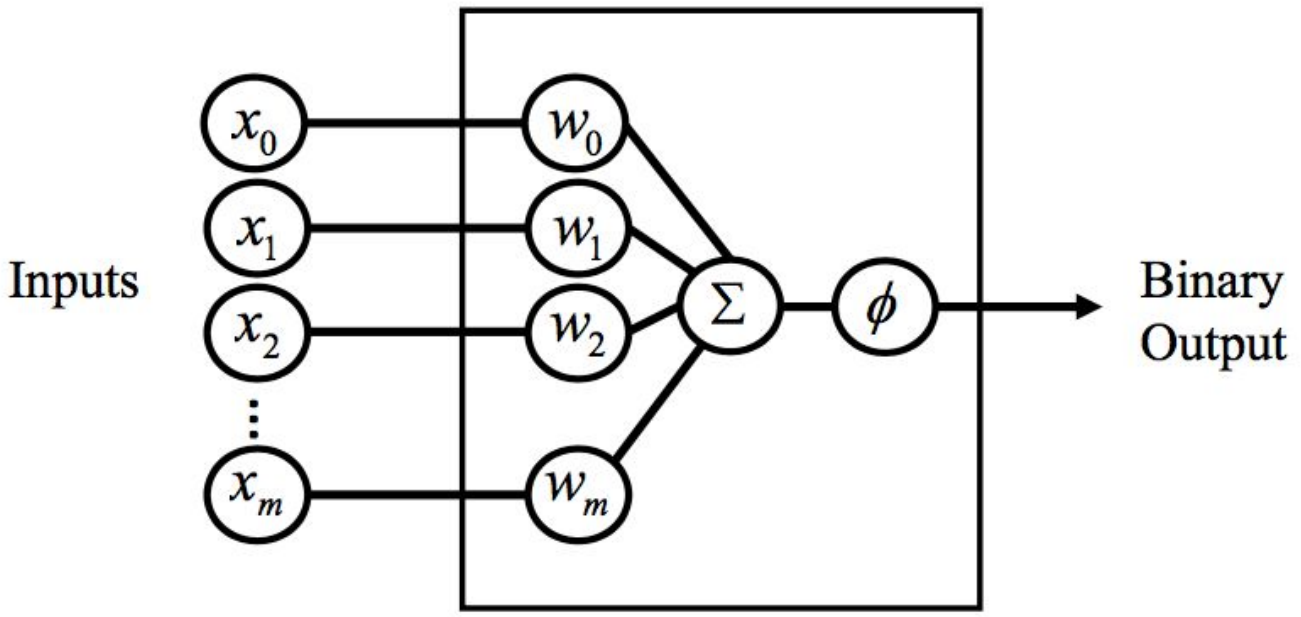
- Learning to approximate some unknown function *f : X -> Y* using features X and their corresponding labels Y
- Classification:
  - Identifying appropriate categories for a given input example
  - ex: Image Classification
- Regression:
  - Predicting some continuous quantity
  - ex: Predicting Home Prices

Supervised Learning Process Flow



Professor Erik Grimmelmann, CCNY

Artificial Neuron (Perceptron)



- x_i input
- w_i weight
- Σ sum of $w_i x_i = z$
- ϕ activation function

$$x_0 = 1$$

$$w_0 = -\lambda$$

Applying Perceptron to NAND and XOR

A	B	out	x_0	x_1	x_2	$\Phi(z)$	z	$z = \boldsymbol{w}^T\boldsymbol{x}$	w_0	w_1	w_2
0	0	1									
0	1	1									
1	0	1									
1	1	0									

A	B	out	x_0	x_1	x_2	$\Phi(z)$	z	$z = \boldsymbol{w}^T\boldsymbol{x}$	w_0	w_1	w_2
0	0	0									
0	1	1									
1	0	1									
1	1	0									

Applying Perceptron to NAND and XOR

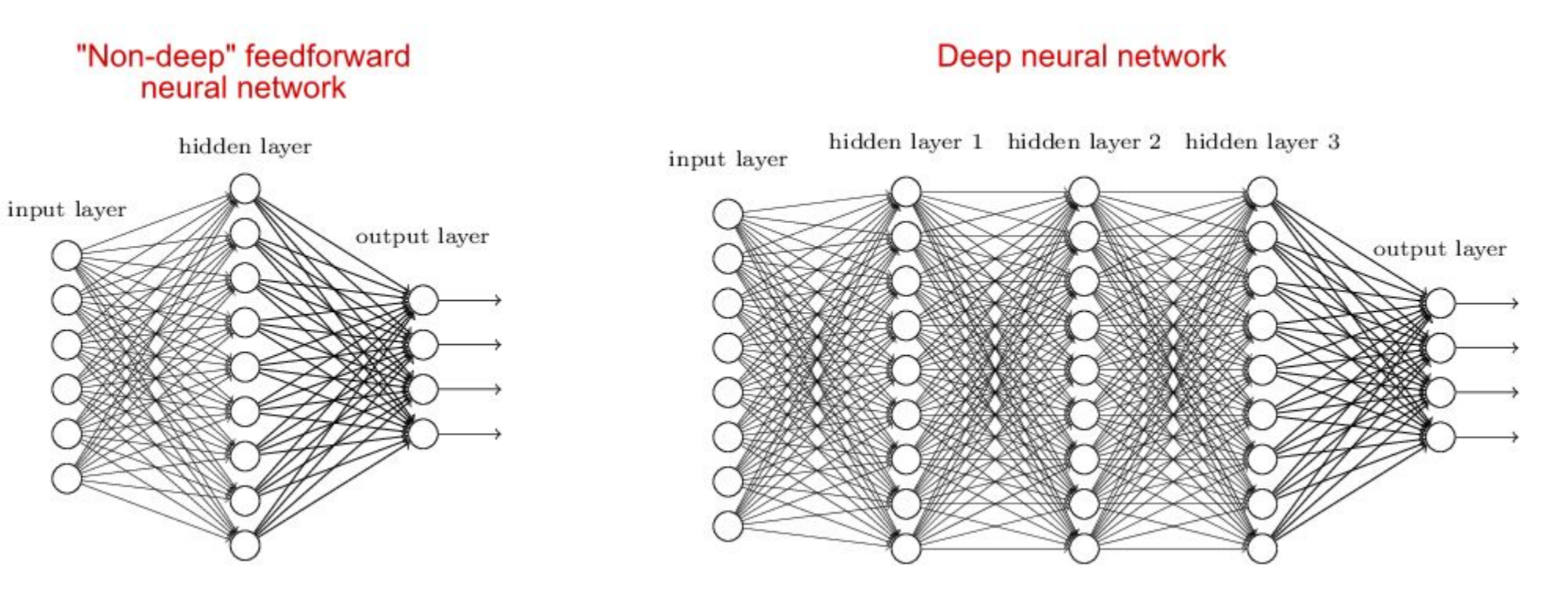
A	B	Out	x_0	x_1	x_2	$\Phi(z)$	z	$z = w^T x$	w_0	w_1	w_2
0	0	1	1	0	0	1	≥ 0	w_0	3	-2	-2
0	1	1	1	0	1	1	≥ 0	w_0+w_2			
1	0	1	1	1	0	1	≥ 0	w_0+w_1			
1	1	0	1	1	1	0	< 0	$w_0+w_1+w_2$			

A	B	Out	x_0	x_1	x_2	$\Phi(z)$	z	$z = w^T x$	w_0	w_1	w_2
0	0	0	1	0	0	0	< 0	w_0	{}		
0	1	1	1	0	1	1	≥ 0	w_0+w_2			
1	0	1	1	1	0	1	≥ 0	w_0+w_1			
1	1	0	1	1	1	0	< 0	$w_0+w_1+w_2$			

Deep Learning

- An instance of biological mimicry to try to take advantage of how our own brains learn
- Deep Neural Networks: A stacking of layers of nonlinear transformations which produce progressively higher-level representations of an input and map to an output
- Currently the foundation of the state-of-the-art in a variety of areas such as instance segmentation and reinforcement learning

Neural Networks



Michael Nielsen
http://neuralnetworksanddeeplearning.com/chap5.html

Math of Neural Networks

$$\mathbf{a}^{(1)} = \sigma(\mathbf{W}\mathbf{a}^{(0)} + \mathbf{b})$$

$$\sigma\left(\begin{bmatrix} w_{0,0} & w_{0,1} & \cdots & w_{0,n} \\ w_{1,0} & w_{1,1} & \cdots & w_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{k,0} & w_{k,1} & \cdots & w_{k,n} \end{bmatrix}\begin{bmatrix} a_0^{(0)} \\ a_1^{(0)} \\ \vdots \\ a_n^{(0)} \end{bmatrix} + \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{bmatrix}\right)$$

3Blue1Brown, Youtube

Math of Neural Networks (continued)

- Activation function:
  - Determine when and how intense the neurons “fire” or “activate”
- Loss function:
  - Measure of how bad the predicted output is compared to the actual labels
- Optimization:
  - How the network adjusts its weights to improve performance (i.e. reduce the error on the current inputs)
  - This uses the gradient descent optimization algorithm which finds the minimum of a function (in this case the loss function) by iteratively taking steps proportional to the negative of the gradient

TIME FOR CODE!!!

Code: https://GitHub.com/PooneetThaper/CCNY-ACM-ML-Workshop

Challenges (for whichever version you prefer):

- Add more hidden layers and/or more neurons and test
- Change the activation functions to see if that makes a difference
- Change the loss function and see if it makes a difference

Advanced Challenges (preferably work in the TF version):

- Set up a logarithmic decaying learning rate and modify the network to use that learning rate
- Download and load in the iris dataset and modify the network to make a class prediction on that data

Additional Topics to Explore

- Train-(Validation)-Test Split
- Batching and Epochs
- Overfitting and Regularization (L1, L2, Dropout, Validation)
- Hyperparameter tuning and parameter search (grid search)
- Cross-validation (k-fold)
- Different metrics
  - MSE, RMSE, Cross Entropy, NMI, F-score, etc
- Different optimization tricks
  - LR decay, Adaptive LR, Momentum, Exploration vs Exploitation

Additional Topics (continued)

- Creative advancements on neural networks
  - Convolutional, Recurrent, Generative Adversarial, Deep Q, etc
- Different algorithms
  - Support Vector Machines, Random Forests, Logistic Regression, etc
  - No Free Lunch Theorem (No one algorithm is perfect)
- Machine learning libraries
  - SciKit Learn, TensorFlow, PyTorch, etc
- Applications of machine learning
  - Advertising, recommendation, digital assistants, etc

Additional Resources

- http://www.deeplearningbook.org
- https://medium.com/topic/artificial-intelligence
- https://medium.com/topic/data-science
- https://arxiv.org/archive/cs
- https://github.com/songrotek/Deep-Learning-Papers-Reading-Roadmap
- https://www.youtube.com/channel/UC_x5XG1OV2P6uZZ5FSM9Ttw
- https://www.youtube.com/channel/UCWN3xxRkmTPmbKwht9FuE5A
- https://www.youtube.com/playlist?list=PLZHQObOWTQDNU6R1_67000Dx_ZCJB-3pi
- https://www.youtube.com/user/shiffman
- https://www.deeplearning.ai
- https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-034-artificial-intelligence-fall-2010/
- https://www.youtube.com/playlist?list=PLrAXtmErZgOeiKm4sgNOknGvNjby9efdf
- https://www.udacity.com/courses/machine-learning