CSCE 580: Introduction to AI

Lecture 16 Machine Learning – NN, DL

PROF. BIPLAV SRIVASTAVA, AI INSTITUTE

10TH OCT, 2024

Carolinian Creed: "I will practice personal and academic integrity."

Credits: **Copyrights of all material reused acknowledged**

Organization of Lecture 16

- Introduction Segment
  - Recap of Lecture 16
- Main Segment
  - Neural Networks
  - Deep Learning
  - Trust Issues
  - Adversarial Attacks
- Concluding Segment
  - Course Project Discussion
  - About Next Lecture – Lecture 17
  - Ask me anything

Introduction Section

Announcement – Nobel Prize 2024 (Physics) !

https://www.nobelprize.org/prizes/physics/2024/press-release/

Machine Learning fundamentals inspired by Physics

Announcement – Nobel Prize 2024 (Chemistry) !

https://www.nobelprize.org/prizes/chemistry/2024/press-release/

David Baker
University of Washington, Seattle, WA, USA
“for computational protein design”

and the other half jointly to

Demis Hassabis
Google DeepMind, London, UK
John M. Jumper
Google DeepMind, London, UK
“for protein structure prediction”

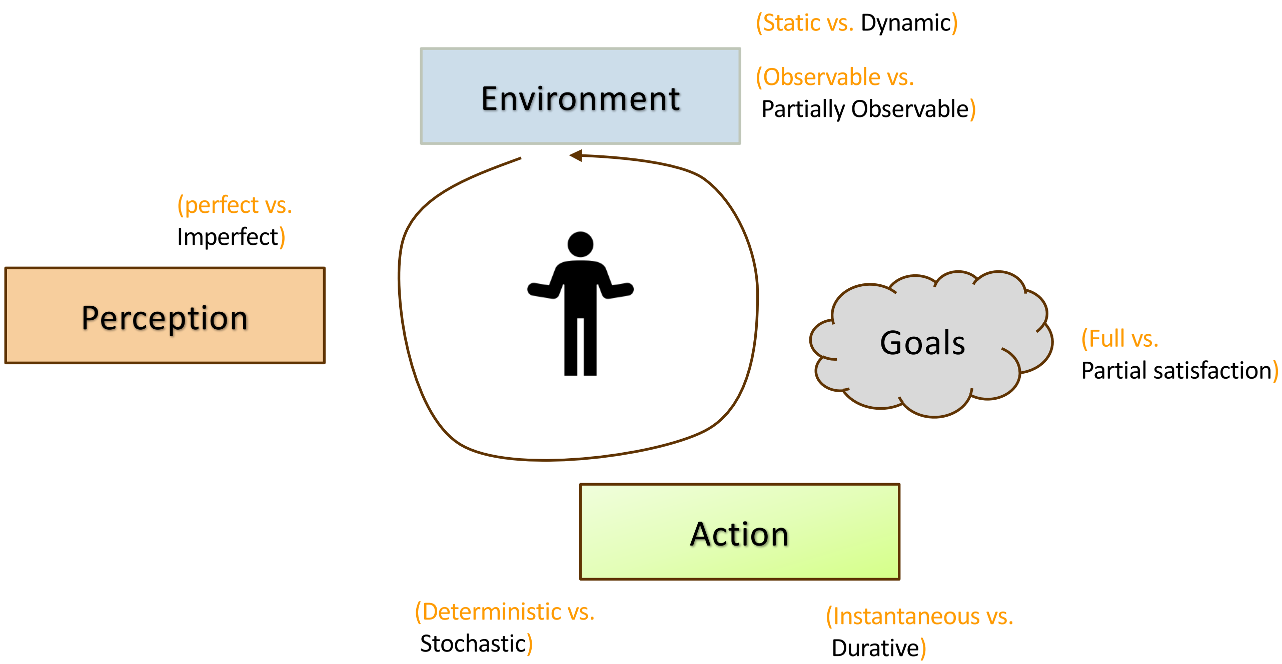
Machine Learning to advance Chemistry (and Biology) – Alpha Fold

Recap of Lecture 15

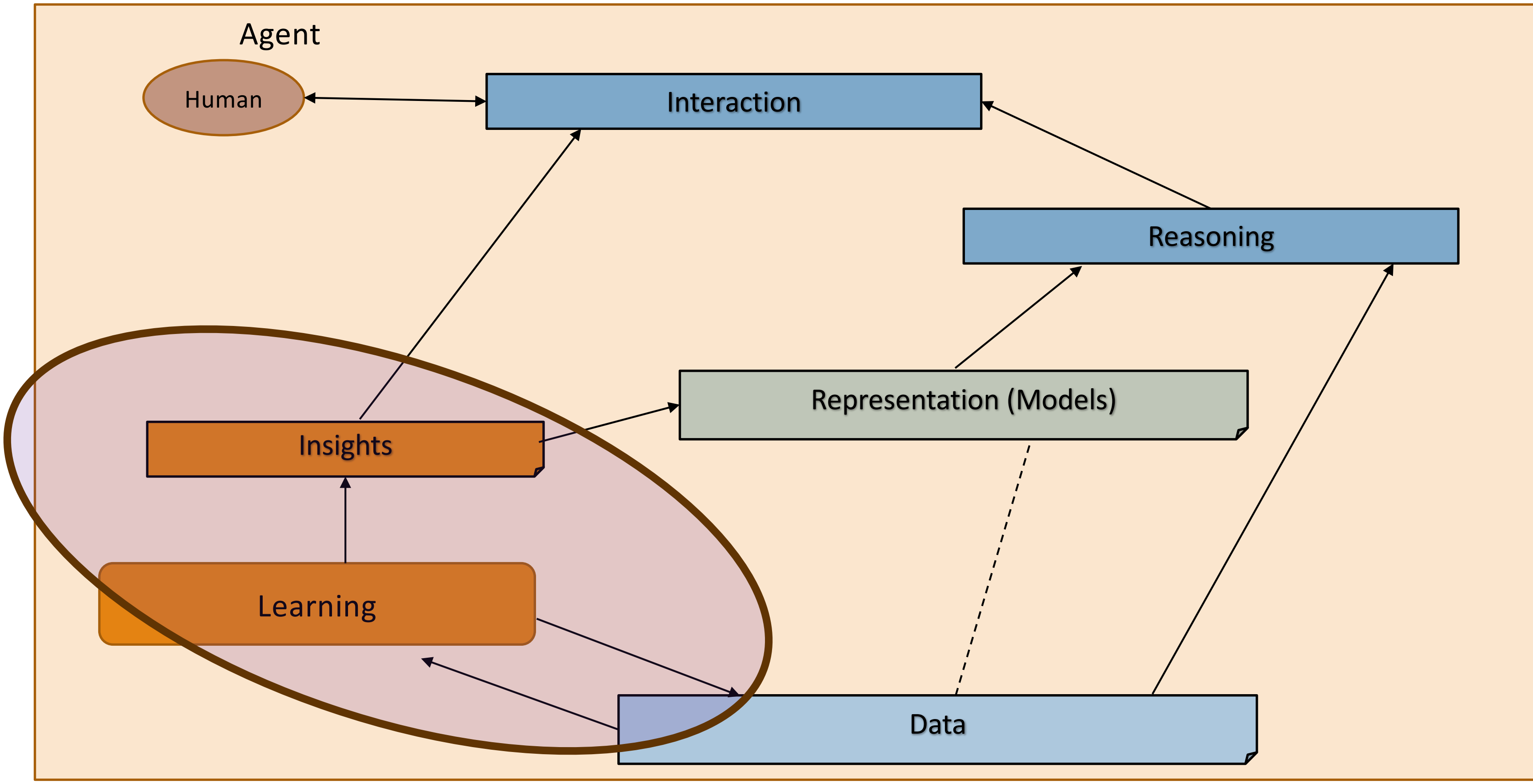
- Topic discussed
  - Student presentations (28)
  - Covered search, fairness, music, deep learning, multi-agent decisions, data science

- Those who did not submit presentation on time
  - Credit will be given to report

Intelligent Agent Model



Relationship Between Main AI Topics



Where We Are in the Course

CSCE 580/ 581 – In This Course

- Week 1: Introduction, Aim: Chatbot / Intelligence Agent
- Weeks 2-3: Data: Formats, Representation and the Trust Problem
- Week 4-5: Search, Heuristics - Decision Making
- Week 6: Constraints, Optimization – Decision Making
- Week 7: Classical Machine Learning – Decision Making, Explanation
- Week 8: Machine Learning - Classification
- Week 9: Machine Learning - Classification – Trust Issues and Mitigation Methods
- Topic 10: Learning neural network, deep learning, Adversarial attacks
- Week 11: Large Language Models – Representation, Issues
- Topic 12: Markov Decision Processes, Hidden Markov models - Decision making
- Topic 13: Planning, Reinforcement Learning – Sequential decision making
- Week 14: AI for Real World: Tools, Emerging Standards and Laws; Safe AI/ Chatbots

Main Section

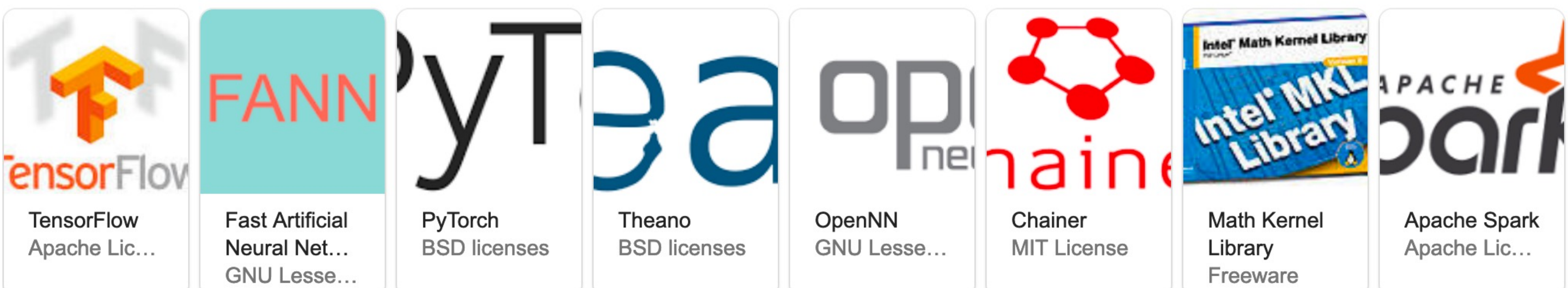
Credit: Retrieved from internet

Machine Learning – Insights from Data

- Descriptive analysis
  - Describe a past phenomenon
  - **Methods**: classification (feedback from label), clustering, dimensionality reduction, anomaly detection, neural methods, reinforcement learning (feedback from hint/ reward)
- Predictive analysis
  - Predict about a new situation
  - **Methods**: time-series, neural networks
- Prescriptive analysis
  - What an agent should do
  - **Methods**: simulation, reinforcement learning, reasoning

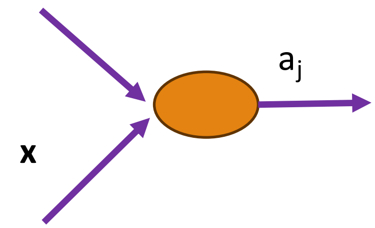
- New areas
  - Counterfactual analysis
  - Causal Inferencing
  - Scenario planning
  - Representation learning

Neural Network Methods



Node (Unit) of a NN

- Notations and meanings
  - a_j: output of a unit j
  - $w_{i,j}$: weight of link from unit i to unit j
  - $a_j = g_j\ (\Sigma\ w_{i,j}\ a_i\ )$, where g_j is a nonlinear activation function
- $a_j = g_j\ (\mathbf{w}^T \mathbf{x})$, where **w** is vector of weights leading into unit j and **x** is the inputs to unit j



Major Types of NNs

Spiking NN

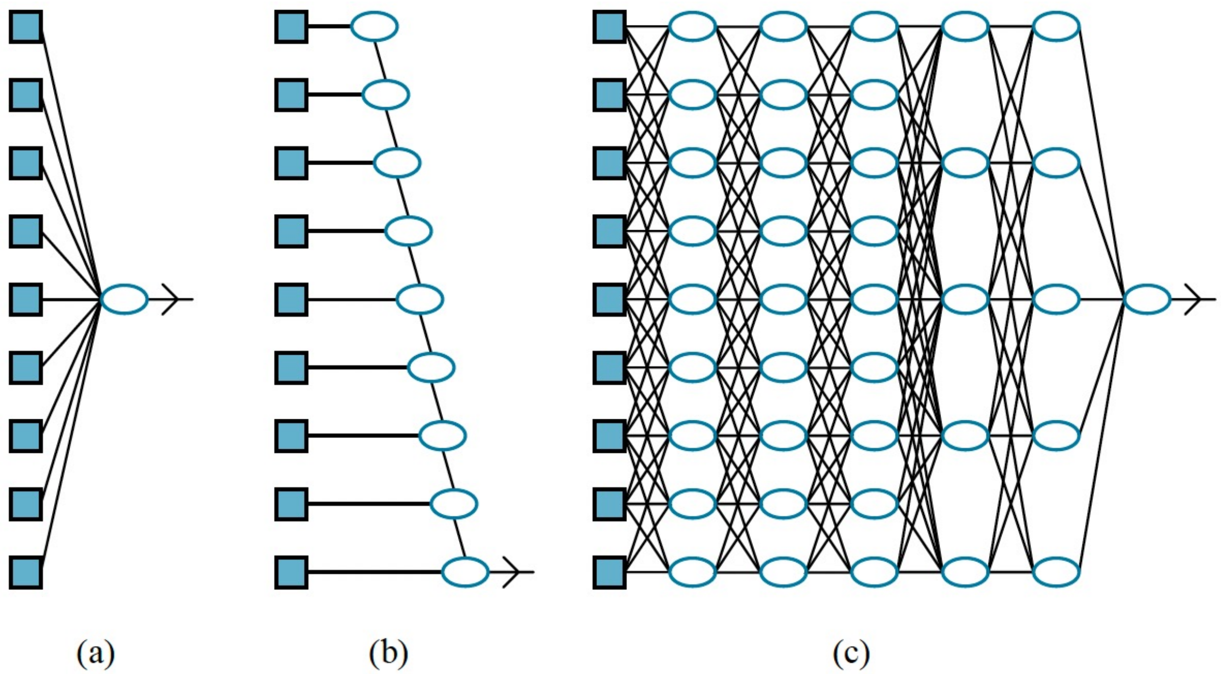
- The idea is that neurons do not transmit information at each propagation cycle only when a membrane potential—an intrinsic quality of the neuron related to its membrane electrical charge—reaches a specific value, called the threshold.
- When the membrane potential reaches the threshold, the neuron fires, and generates a signal that travels to other neurons which, in turn, increase or decrease their potentials in response to this signal.
- A neuron model that fires at the moment of threshold crossing is also called a spiking neuron model.[3]

Artificial NN

- Artificial Neurons are highly abstracted out
  - no modeling of action potentials or spikes
  - no accounting of temporality or spatiality and they are stateless
- Artificial Neurons are simple non-linear activation functions
  - e.g. ReLU, Sigmoid, etc.
- Real-valued output of artificial neurons is analogous to the firing rate of spiking neurons

Credit: https://en.wikipedia.org/wiki/Spiking_neural_network

Model Depth and Learning Ability



(a) A shallow model, such as linear regression, has short computation paths between inputs and output. (b) A decision list network has some long paths for some possible input values, but most paths are short. (c) A deep learning network has longer computation paths, allowing each variable to interact with all the others.

Adapted from:
Russell & Norvig, AI: A Modern Approach

NN Background

- Neural networks were first proposed in 1944 by Warren McCullough and Walter Pitts,
- The first trainable neural network, the Perceptron, was demonstrated by the Cornell University psychologist Frank Rosenblatt in 1957.

Details and Credits:

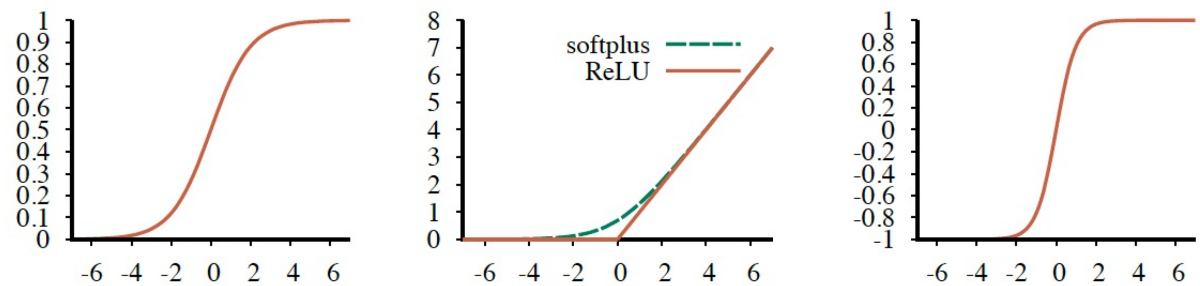
a) https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414

b) Sarker I. H. (2021). Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *SN computer science*, *2*(6), 420. https://doi.org/10.1007/s42979-021-00815-1

Popular Activation Functions

- Logistics or sigmoid function: $\sigma(x) = 1/(1 + e^{-x})$
- ReLU (rectified linear unit): max (0, x)
- Softplus function: $\log(1 + e^{x})$
  - Smooth version of ReLU
- $\tanh(x) = (e^{2x} - 1) / (e^{2x} + 1)$
  - Scaled and shifter version of sigmoid; $\tanh(x) = 2\sigma(2x) - 1$

a) the logistic or sigmoid function
b) the ReLU function and the softplus function
c) the tanh function.



(a) (b) (c)

Adapted from:
Russell & Norvig, AI: A Modern Approach

Note: All activation functions are non-linear

Loss functions

- Mean squared error

$$MSE = \frac{1}{n}\sum_{j=1}^{n}\left[f(X_{j\cdot}) - y_j\right]^2$$

- Categorical Cross Entropy

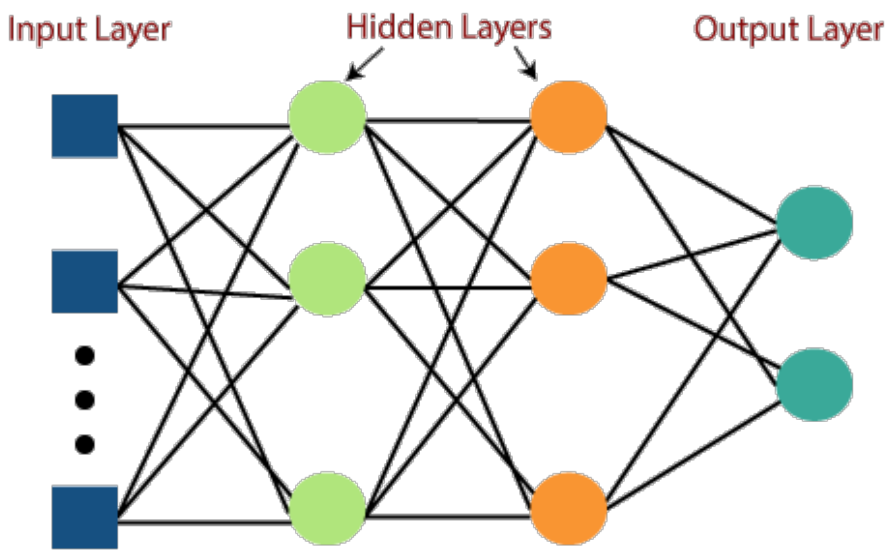
$$\text{Cost} = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k}[y_{ij}\log(\hat{y}_{ij})]$$

k is classes,
y = actual value
Ŷ = prediction

- More loss functions:

https://www.analyticsvidhya.com/blog/2022/06/understanding-loss-function-in-deep-learning/

NN – Multi Layer Perceptron



Content and Image Courtesy:
https://github.com/Thanasis1101/MLP-from-scratch

(Stochastic) Gradient Descent

Gradient Descent

w ← any point in the parameter space

While not converged do:

For each w_i in **w** do:

$w_i \leftarrow w_i - \alpha \ (\partial / \partial w_i) \ \text{Loss}(\mathbf{w})$

Calculate the gradient of the loss function with respect to the weights along the gradient direction to reduce the loss.

Stochastic Gradient Descent (SGD)

Randomly select a small number of training examples at each step

Sources:

- https://en.wikipedia.org/wiki/Stochastic_gradient_descent
- Russell & Norvig, AI: A Modern Approach, Chapter 19

Logistic Regression in a Slide

Function estimate (linear)
W: weight, b: bias

$$f(X_j) = X_j W + b$$

Update Weight

$$W^* = W - \eta \frac{dL}{dW}$$

Error Term (mean squared error)

$$MSE = \frac{1}{n} \sum_{j=1}^{n} \left[f(X_{j\cdot}) - y_j \right]^2$$

Common Code Pattern

```
y = tf.matmul(x, W) + b
loss = tf.reduce_mean(tf.square(y - y_label))
```

NN Concepts

- **Epoch**: The number of times the learning algorithm will iterate over the entire dataset
- **Batch**: how many samples are processed before updating the model's internal parameters.
  - **Batch Gradient Descent**: Batch Size = Size of Training Set
  - **Stochastic Gradient Descent**: Batch Size = 1
  - **Mini-Batch Gradient Descent**: 1 < Batch Size < Size of Training Set

Credit: https://rentry.org/llm-training

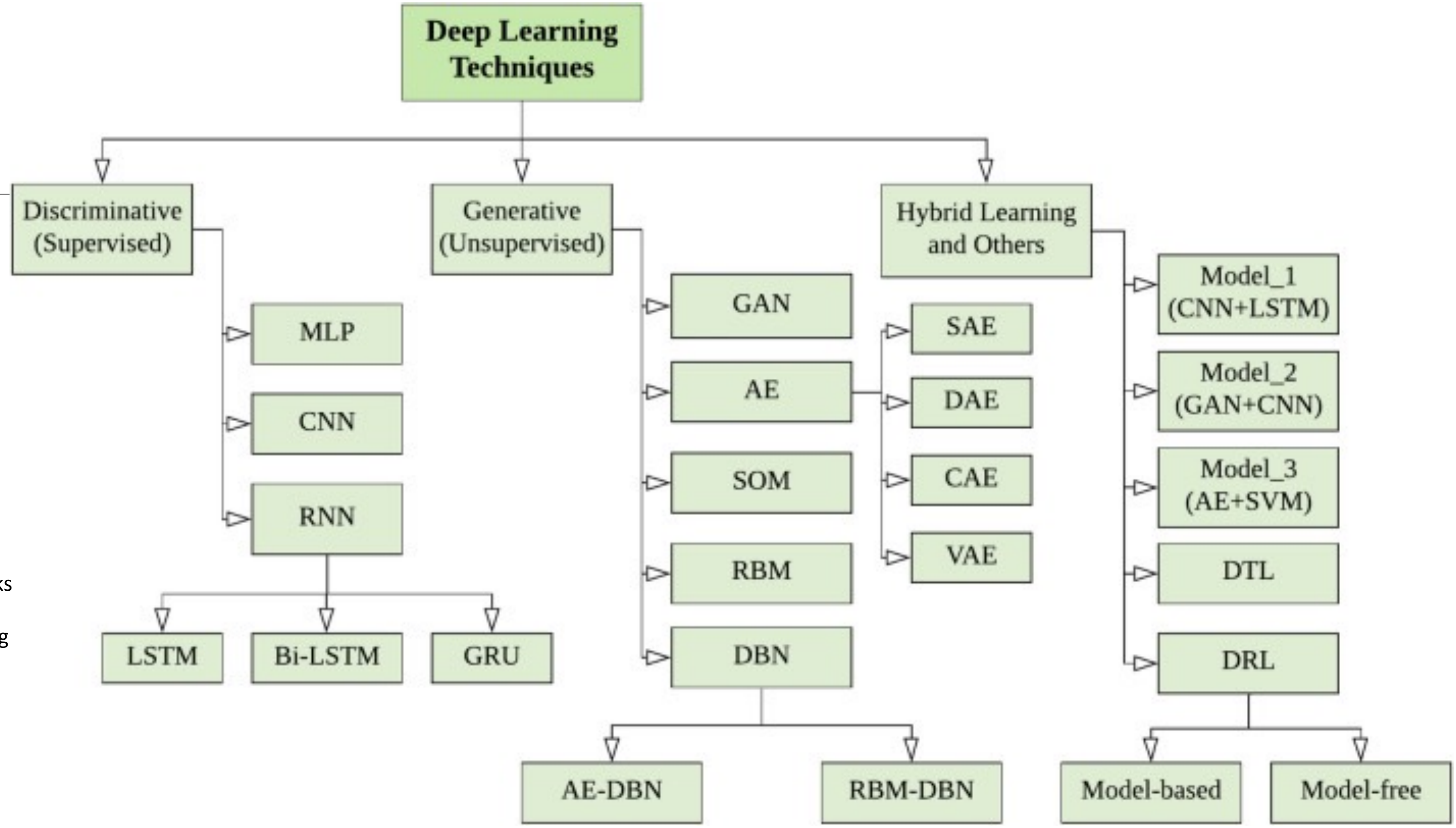
Universal Approximation Theorem

- A network with just two layers of computation units, first nonlinear, and the second linear, <u>can approximate any continuous function to an arbitrary degree of accuracy.</u>
- **Why**: a sufficiently large network can implement a lookup table for continuous functions
  - Nonlinear layer is the key

Sources:

- https://en.wikipedia.org/wiki/Universal_approximation_theorem
- Russell & Norvig, AI: A Modern Approach, Chapter 21

Deep NN Types



A taxonomy of DL techniques, broadly divided into three major categories (i) deep networks for supervised or discriminative learning, (ii) deep networks for unsupervised or generative learning, and (ii) deep networks for hybrid learning and relevant others

Details and Credits:

a) https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414

b) Sarker I. H. (2021). Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *SN computer science*, *2*(6), 420. https://doi.org/10.1007/s42979-021-00815-1

Datasets

- In keras, https://keras.io/api/datasets/
  - boston_housing
  - cifar10 module, cifar100, fashion_mnist, mnist
  - imdb module
  - reuters module

- In TF, https://www.tensorflow.org/datasets/catalog/overview#all_datasets

Keras Walkthrough

- Package: https://keras.io/about/
- Example model:
  - Sequential: https://keras.io/guides/sequential_model/
- Many examples: classification, image, text, audio
  - https://keras.io/examples/
- Future Keras: https://keras.io/keras_core/
  - Keras Core - run Keras workflows on top of TensorFlow, JAX, and PyTorch; preview of Keras 3.0

Code Examples With Keras and TF

1. Classification – diabetes

2. Try code
- Play with hyper-parameters
- Look at keras features used

Code location:
https://github.com/biplav-s/course-ai-tai-f23/tree/main/sample-code/Class19-To-21-DL

Discussion

- Impact of network structure:
  - Nodes / layer:
  - Layers:
  - Inter-connection structure:
- Impact of hyper-parameters:
  - Epochs:
  - Batch size:

Code Examples With Keras and TF

1. Classification – diabetes

2. Prediction/ representation learning – autoencoder

Code location:
https://github.com/biplav-s/course-ai-tai-f23/tree/main/sample-code/Class19-To-21-DL

Discussion

- Impact of network structure:
  - Nodes / layer:
  - Layers:
  - Inter-connection structure:
- Impact of hyper-parameters:
  - Epochs:
  - Batch size:

Code Examples With Keras and TF

1. Classification – diabetes
2. Prediction/ representation learning – autoencoder
3. **Classification – MNIST**

Code location:
https://github.com/biplav-s/course-ai-tai-f23/tree/main/sample-code/Class19-To-21-DL

Discussion

- Impact of network structure:
  - Nodes / layer:
  - Layers:
  - Inter-connection structure:
- Impact of hyper-parameters:
  - Epochs:
  - Batch size:

Keras and TensorFlow

- By Example:
  - https://github.com/biplav-s/course-nl-f22/blob/main/sample-code/l11-nn-dl/Basic%20TensorFlow%20and%20Keras.ipynb
  - TensorFlow's NMIST tutorial
  - https://www.tensorflow.org/tutorials/quickstart/beginner

- More examples
  - Number Addition by sequence learning: https://keras.io/examples/nlp/addition_rnn/
  - AutoEncoder: https://machinelearningmastery.com/lstm-autoencoders/

NN/ MLP

- Code examples:
  - https://github.com/biplav-s/course-d2d-ai/blob/main/sample-code/l6-l7-l8-supervised-ml/Supervised-NaiveBayes-GradientBoost-NN-Classification.ipynb
- Scikit Library:
  - MLP: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html

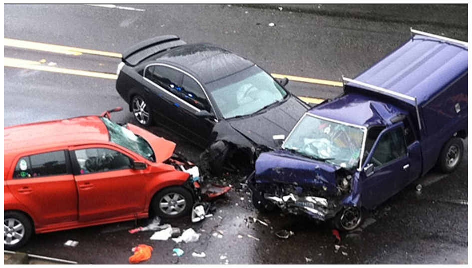
Consideration: Which NN/DL Tool to Use

- See:
  - https://www.simplilearn.com/keras-vs-tensorflow-vs-pytorch-article
  - In theory, keras supports all major ones
    - Pytorch used in academic research more
    - TF used in production systems

Trust: Adversarial Attacks

Example (Gu et al. 2017)

- **ML Application**: Detect and classify street signs in images
- **Poisoning method**: Insert images where a special sticker is added to stop signs and the label changed to speed limit
- **Backdoor**: Adversaries ensure that any stop sign is misclassified simply by placing a sticker on it



Trust: Adversarial Attacks

- Cat and mouse on attacks and defenses
  - Example code: https://github.com/Trusted-AI/adversarial-robustness-toolbox/blob/main/notebooks/adversarial_training_mnist.ipynb
- Tools
  - Adversarial Robustness Toolbox (ART) - Python library for Machine Learning Security, https://github.com/Trusted-AI/adversarial-robustness-toolbox

Trust Issues with NN

- Robustness: can the model give the results in the presence of (input) perturbation? Noise?
- Computation/ footprint: why does the learning take so much compute resources?
- Data: is the data representative? How was the data obtained?
- Explainability: why does the model work?
- Fairness: Is the output fair to user groups?

Resources and Books

- Understanding Deep Learning, https://udlbook.github.io/udlbook/
- Deep Learning, Ian Goodfellow, Yoshua Bengio and Aaron Courville, https://www.deeplearningbook.org/
- AI – A Modern Approach, Russell & Norvig, https://aima.cs.berkeley.edu/
- Websites of libraries – Keras.

Course Project

Discussion: Projects

- New: two projects
  - Project 1: model assignment
  - Project 2: single problem/ llm based solving / fine-tuning/ presenting result

Lecture 16: Summary

- We talked about
  - Neural Networks
  - Deep Learning
  - Adversarial attacks
  - Trust Issues

Concluding Section

About Next Lecture – Lecture 17

Lecture 17: Text, Large Language Models

- Text processing
- Language Models (LMs)
- Large LMs

13	Oct 1 (Tu)	Machine Learning – Classification – Decision Trees, Random Forest, NBC, Gradient Boosting, ML-Text
14	Oct 3 (Th)	ML – Unsupervised / Clustering
15	Oct 8 (Tu)	Student presentations - project
16	Oct 10 (Th)	ML – NN, Deep Learning
17	Oct 15 (Tu)	Processing Natural Languages/ Language Models
	Oct 17 (Th)	
18	Oct 22 (Tu)	Large Language Models (LLMs) / Foundation Models
19	Oct 24 (Th)	Using LLMs – how and when ?