CSCE 790: Neural Networks and Their Applications

AIISC and Dept. Computer Science and Engineering
Email: vignar@sc.edu

Dr. Vignesh Narayanan

August 24, 2023

UNIVERSITY OF
South Carolina

Office Hours

Email: vignar@sc.edu	Course Website: https://blackboard.sc.edu
	Teaching Assistant: N/A
	Office Hours: TR 2:00 pm – 3:00 pm, or by appointment

Textbooks and References

1. Martin T. Hagan, Howard B. Demuth and Mark Beale, Neural Network Design, PWS Publishing Company, 1995.
2. S. Haykin, Neural Networks: A Comprehensive Foundation, Prentice-Hall, NJ, 1999.
3. Ian Goodfellow, Yoshua Bengio, and Aaron Courville, Deep Learning, The MIT Press, 2016.
4. F.L. Lewis, S. Jagannathan, and A. Yesilderik, Neural Network Control of Robot Manipulators and Nonlinear Systems, Taylor and Francis, UK, 1999.
5. Relevant material (e.g., link to research papers, supplementary book references) will be provided to the students via Blackboard.

Textbooks and References - Free Online Resources

1. Neural Network Design - Read and practice
2. Deeplearning - My lectures
3. Understanding Deep Learning - Reading
4. Neural network control of robots and nonlinear systems - Robotic applications
5. Neural Networks and Deep Learning - Reading

Assessments

Your overall final course letter grade will be determined by your grades on the following assessments.

Homework Assignment	15%
Presentation	15%
Midterm Exam (Take home)	15%
Final Project	55%

Bonus Points*

Summary of Assessments

- **Homework Assignments:** You will be required to turn in assignments/reports on time. They will typically involve reading research papers, reporting critiques, design, development, and implementation of codes on MATLAB/Python.
- **Presentation:** You will be required to prepare an in-class presentation (30 minutes each). This will be done either independently or in groups. The grading will be based on how well you present the motivation for the project/research you are presenting, problem definition, ideas, techniques, and limitations of the work.
- **Exam:** There will be a Take home exam around November.
- **Final Project:** For the final project, topics shall be decided after discussing with the instructor. You will be expected to turn in a project report by the end of the course ($\approx$ end of November 2023), which will contribute 55% toward your final grade.
- **Bonus Points:** You can get bonus points by solving extra problems in homework assignments.

Course Overview

- This course covers background, mechanisms, and techniques used to build learning algorithms using artificial neural networks.
- Topics include basics of neural network topologies (e.g., multi-layer perceptron, Linear-in-the-parameter neural nets, Hopfield and generalized recurrent neural networks), neural network learning paradigms and rules (e.g., backpropagation), and applications of neural networks for classification, pattern recognition, and function approximation.
- We will also discuss (time-permitting) neural network control applications, basics of dynamical systems, and recent developments in artificial neural networks based on research papers.

Course Overview

- Network architectures
  1. Multi-layer perceptron
  2. Recurrent neural networks, ...
- Neural networks in learning paradigms
  1. Supervised learning - Classification and Regression
  2. Unsupervised learning - Self-organized maps
  3. Reinforcement learning - Control, Estimation and Identification
- Learning/Training Rules
  1. Delta rule
  2. Back propagation, ...
- Theoretical tools
  1. Analysis
  2. Optimization
  3. Linear algebra
  4. Differential (and/or Difference) equations
- Applications

  ⋮

AI-ML - Nomenclature

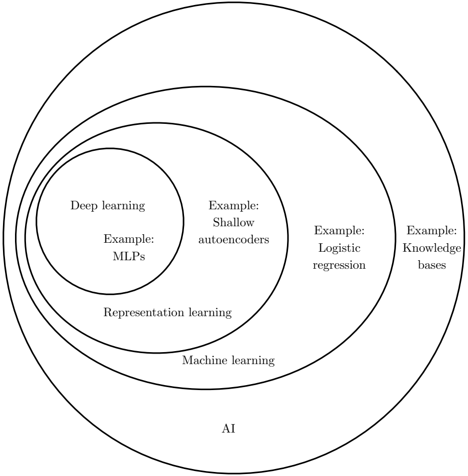


Figure: Neural networks → Deep learning (Deep Learning, 2016)

AI-ML - Components

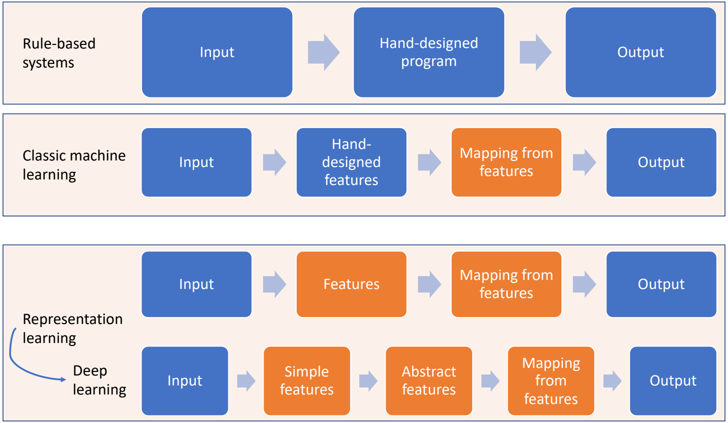


Figure: Components of various AI disciplines

Data representation - "Feature selection"

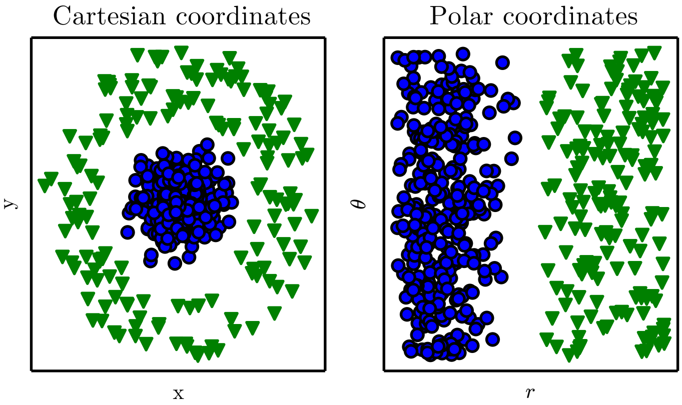


Figure: Representation of data (Deep Learning, 2016)

Background on Neural Networks

- The study of artificial neural networks (ANNs) is motivated by the recognition that the human brain computes in an entirely different way from the conventional digital computers
- Brain is a complex, nonlinear, and parallel information processing system.
- It can organize the neurons such that complex tasks are accomplished much faster than digital computers in an energy efficient manner
- At birth, brain has great modular structure and has the ability to build its own rules through experience.
- A developing neuron is synonymous to plastic brain (plasticity enables adapatation)
- ANNs are designed to model the way in which the brain performs a particular task

Biological Neural Networks

- ANNs are (over) simplified models inspired by biological neural networks
- Human brain consists of $\approx 10^{11}$ highly interconnected neurons ($\approx 10^4$ connections/element)
- One may view the human nervous system as a $3-$stage system composed of
- Stimulus $\rightarrow$ Receptors$\leftrightarrow$Neural net $\leftrightarrow$ Effectors $\rightarrow$ Response
- Signal conversions: Electrical $\rightarrow$ Chemical $\rightarrow$ Electrical

Biological Neurons - Quantity

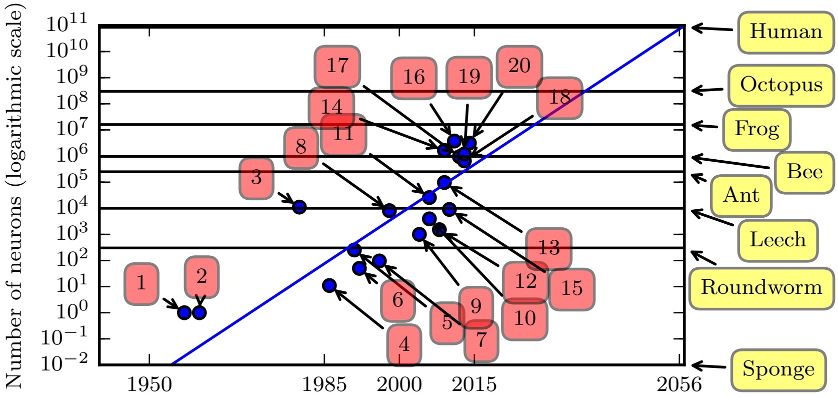


Figure: Growing scale of neural networks (DL, 2016)

Biological Neurons - Connectivity

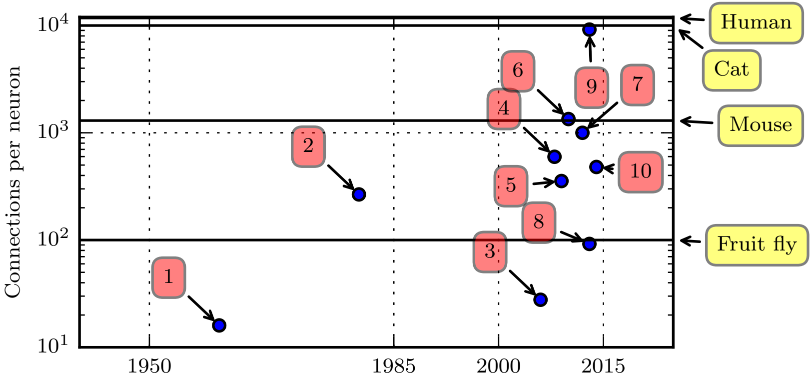


Figure: Neurons' connections (DL, 2016)

History

- Background work for ANNs occurred in late 19^{th} and early 20^{th} centuries.
- This was an interdisciplinary effort involving researchers working in physics, psychology, and neurophysiology. Example: Generalized theory of learning, visual processing, and conditioning - Hermann Vol Hemholtz, Ernst Mach, and Ivan Pavlov.
- Modern view of NNs began in $1940's$ with the work of Warren McCulloch and Walter Pitts.
- They showed that ANNs could, in principle, compute any arithmetic or logical function.
- Later, Donald Hebb, demonstrated that the classical conditioning is due to properties of individual neurons.

History

- First practical application of ANNs in late 1950$'s$ with Frank Rosenblatt through the invention of the Perceptron network and an associated learning rule.
- It was discovered that the perceptron network can only be used to solve limited class of problems (related to pattern recognition)
- At the same time, Bernard Widrow and Ted Hoff developed new learning algorithms for adaptive linear networks.
- This too suffered similar limitations (recorded by Marvin Minsky and Seymour Papert)
- For about a decade, ANNs were not as popular as before - (one more reason- lack of digital processors)

ANNs Resurgence

- Work by John Hopfield - Use of statistical mechanics to explain operation of a certain class of recurrent neural networks (RNNs).
- Influential work by David Rumelhart, James McClelland, and contemporaries - Backpropagation algorithm for training multilayer perceptron networks.
- We now have many NN architectures
  1. MLP, CNN, DNN, GNN, DQNs, Transformers. . ..
  2. RNN, LSTM, RCN, . . .
- Note: These are simplified models of biological neurons
- We still do not have a complete understanding of brain and neural mechanisms.

Historic Trends

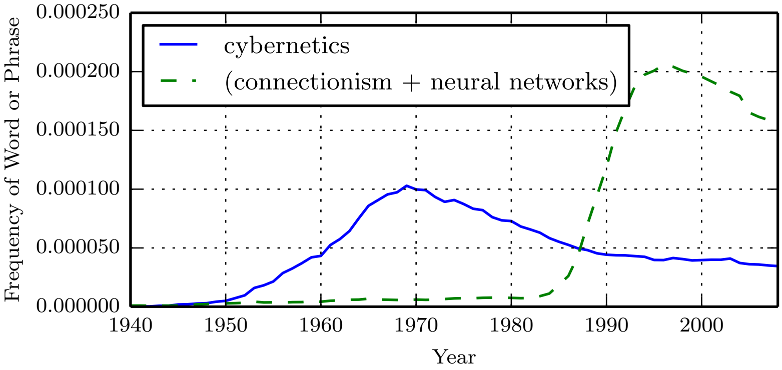


Figure: Evolution of NNs (DL, 2016)

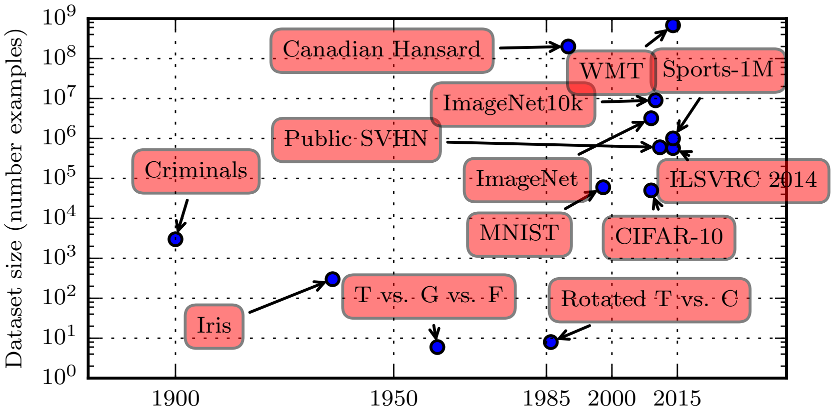


Figure: Evolution of datasets (DL, 2016)

Example

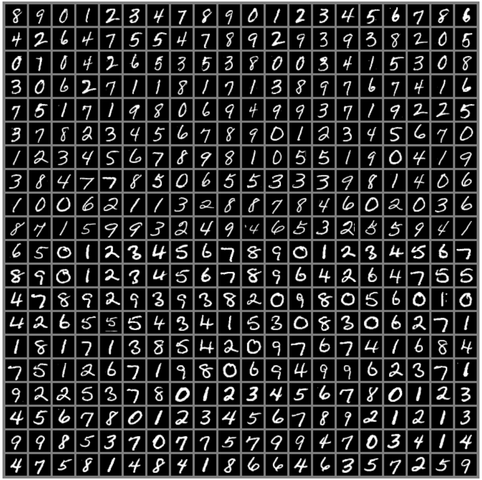


Figure: Handwriting recognition - MNIST (DL, 2016)

Example: Evolution of performance

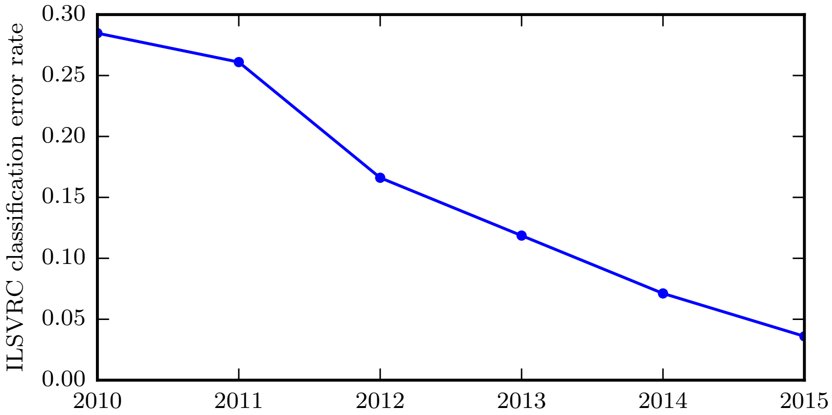


Figure: Performance of NN over the years (DL, 2016)

Why ANNs'

- NNs derive its computational power through:
  1. Its massively distributed parallel structure
  2. Its ability to learn and generalize
- Generalization - NNs producing reasonable outputs for inputs not encountered during training (or learning)
- Characteristics:
  1. Nonlinearity
  2. Input-output mapping
  3. Adaptivity
  4. Evidential response
  5. Universality and modularity
  6. Neurobiological analogy

What is an ANN?

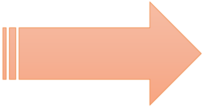
Definition (Haykin)

An ANN is a massively parallel distributed processor made of simple process units, which has a natural propensity for storing experiential knowledge and making it available for use. It resembles brain in two respects:

- Knowledge is acquired by the NN from its environment through learning process
- Inter-neuron connection strengths, known as synaptic weights, are used to store the acquired knowledge.

- The process used to acquire and store knowledge is called a learning algorithm.
- This approach is not entirely new but it closely resembles linear adaptive filter theory (LAFT).
- This deviates from the LAFT in its ability to change/modify the ANN topology as needed during learning, which is motivated by the human brain, where neurons die and new synaptic connections grow.

The Big Picture - Typical Learning Objective

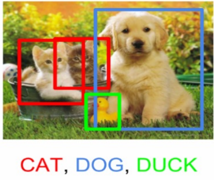


Data

Actionable Insights

What generates data and how much?

Complex systems like **smart environments, fleet of vehicles, machining center, process control, smart grid and health care applications** generate large quantities of data.



Source of images: www.interaction-design.org

The Big Picture - Machine Learning Framework - Functional Viewpoint

$$y = f(x)$$

Output $=$ **Prediction-function(Input)**

- **Training:** given a training set of input-output examples $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_N, y_N)\}$, learn/estimate the prediction function f by minimizing the prediction error on the training set
- **Testing:** apply f to a never-before-seen test example x and output the predicted value $\hat{y} = f(x)$...

The Big Picture - ML Terminologies

- Input X: feature, predictor, independent variable
- Output Y: label, response, dependent variable

Data types:

- Quantitative: Measurements or counts, recorded as numerical values (e.g. height, temperature, etc.)
- Qualitative: group or categories
  - Ordinal: possesses a natural ordering (e.g. shirt sizes)
  - Nominal: just name the categories (e.g. marital status, gender, etc.)

The Big Picture - Learning Process/Components

- Data
- Feature selection
- Model selection
- Training
- Evaluation

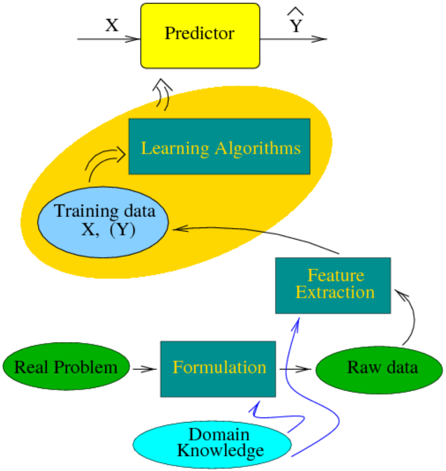


Figure: Components of learning

The Big Picture - How do we estimate f?

- First, we assume that we have observed a set of training data.

$$\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_N, y_N)\}$$

- Second, select a 'model' and use the training data and identify the learning paradigm to estimate f.
- Parametric methods: This reduces the learning problem of estimating the target function f down to a problem of estimating a set of parameters.

The Big Picture - Parametric Methods

- Let $x_i = (x_i^{(1)}, x_i^{(2)}, x_i^{(3)}, x_i^{(4)})'$, $i = 1, 2, 3, \ldots, N$.
- Given $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_N, y_N)\}$
  - Step 1: Make some assumptions about the functional form of f. The most common example is a linear model:

$$f(x) = \theta_1 x^{(1)} + \theta_2 x^{(2)} + \theta_3 x^{(3)} + \theta_4 x^{(4)}$$

  - What is a good candidate model?
  - How many features to feed to the model?
  - Step 2: We use the training data to fit the model (i.e. estimate f....the unknown parameters).
  - The most common approach for estimating the parameters in a linear model is via ordinary least squares (OLS) linear regression.

The Big Picture - Model Representation

Basic structure: Observed data mapped to regressor space, regressor space mapped to output space through a nonlinear map (forming a basis in the output space). Finally, coordinates are determined (Parameter estimation).

$$\hat{f} = \sum_{k=1}^{N} \theta_k \phi_k(x), \quad \phi_k(x) = h(\beta_k(x_k - \gamma_k))$$

β_k scaling, γ_k translation.
Examples: Fourier series, Splines, Neural networks (NN), Wavelets, Kernels, etc.
Design decisions: Choice of regressor, Selecting ϕ, Number of basis functions – N, Finding the values of location and scaling parameters β, γ, finding the coordinate parameter θ
NN as approximators: $\hat{f}(x) = \theta' \phi(\beta' x + \gamma)$ $(+\varepsilon(x) = f(x))$