CSCE 790: Neural Networks and Their Applications

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

Dr. Vignesh Narayanan

August 29, 2023

UNIVERSITY OF
South Carolina

Assessments - Start Early!

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%

Homeworks/Projects

- In reference to this course, you are encouraged to discuss with your peers while working on assignments and projects, but do not copy the work of others, either manually or electronically, under these conditions. Write/type your solutions in your own words.
- Always cite your sources!

Homeworks/Projects

- Project Report: Writing tips
- Reading a research paper:Tips
- **Homework Problem 1**
  - Classification of handwritten digits
  - Compilation of prominent results with MNIST

The Big Picture - Model Representation (Recap)

Basic structure for parametric models:

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

Examples: Fourier series, Splines, Neural networks (NN), Wavelets, Kernels, etc.

NN as approximators: $\hat{f}(x) = \theta\phi(\beta x + \gamma)$ approximates $f(x) = \hat{f}(x) + \varepsilon(x)$

$$\gamma = \text{bias}, \quad \beta, \theta = \text{weights},$$

$$\phi(\cdot) = \text{activation function}, \text{ and } \varepsilon(\cdot) = \text{reconstruction/residual/approximation error}$$

Artificial Neural Networks as Parametric Models

- Many parametric methods are available - Kernel- RKHS (Reproducing kernel HS), Wavelets, Splines, Spectral and Pseudo-spectral methods..
- What makes NN better? Is it better?

Definition

Sets are collection of (elements) mathematical objects such as numbers, symbols, points in space, lines, other geometrical shapes, variables, or even other sets

- Set with no element is called a *null set*
- *Set with just one element is called a* singleton

Example

- $A = \{1, 2, 3\}$ (Finite set)
- $B = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$ (Infinite set - Countable)
- $C = \{x : x \in (-1, 1)\}$ (Infinite set - Uncountable)

Operations on set (Attention - Notations and Symbols)

Definition

Given sets A, B

(i) Union - $A \cup B = \{x : x \in A \text{ or } x \in B\}$

(ii) Intersection - $A \cap B = \{x : x \in A \text{ and } x \in B\}$

(iii) Complement - $A \setminus B = \{x : x \in A \text{ and } x \notin B\}$

(iv) Cartesian product - $A \times B = \{(x, y) : x \in A \text{ and } y \in B\}$ – and its elements are called *ordered pairs*.

Relation (Pay Attention to Notations and Symbols)

Definition

Given sets A, B a binary relation $\mathcal{R}$ over sets A and B is a subset of $A \times B$, i.e., relation from A to $B \subset A \times B$.

- We know that $A \times B = \{(x, y) : x \in A \text{ and } y \in B\}$
- The statement $\forall x \in A, \forall y \in B, (x, y) \in \mathcal{R}$ should be read as follows
- "for all x in set A and y in set B, x is related to y through $\mathcal{R}$", or "x is $\mathcal{R}$-related to y"

Definition

- Domain: $\mathcal{D}(\mathcal{R}) = \{x \in A : \exists y \in B : (x, y) \in \mathcal{R}\}$
- Co-domain or Range: $\mathcal{C}(\mathcal{R}) = \{y \in B : \exists x \in A : (x, y) \in \mathcal{R}\}$

Function

Definition

*Function is a relation

Definition

Let A and B be sets. A function f from set A to set B is a set of ordered pairs $(x, f(x))$, where $x \in A$ and $f(x) \in B$. Here A is called the domain of the function and B is called the codomain (or range) of the function.

Definition (Basic notations)

- set of real numbers $\mathbb{R}$
- set of natural numbers $\mathbb{N}$
- set of integers $\mathbb{Z}$

Finite, Countable, and Uncountable Sets - Relevance - Examples

Example (Finite Domain and Range)

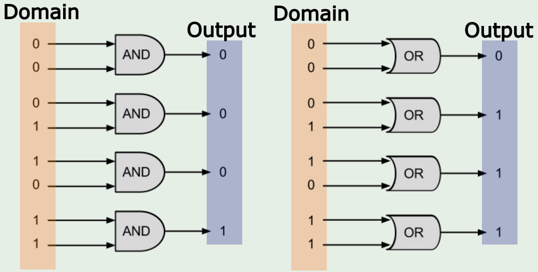

Figure: Logic functions

Example (Infinite Domain and Range)

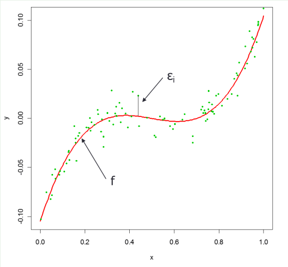

Figure: Continuous function in a closed interval

Example - What is the domain?

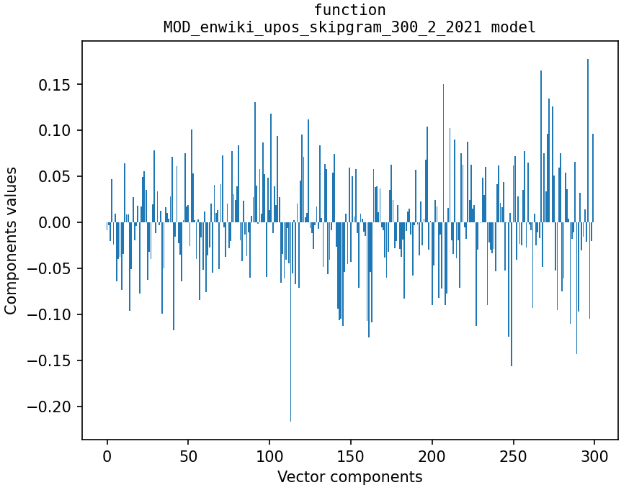

Figure: Word embedding - word2vec representation of the word 'function'

- Word2vec - Converts words to vectors
- "function" $\rightarrow x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{300} \end{pmatrix} \in \mathbb{R}^{300}$ (?)

McCulloch-Pitts Model

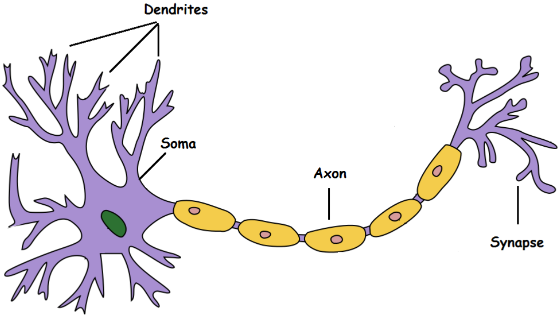

Figure: Simplified single nerve cell (Wiki)

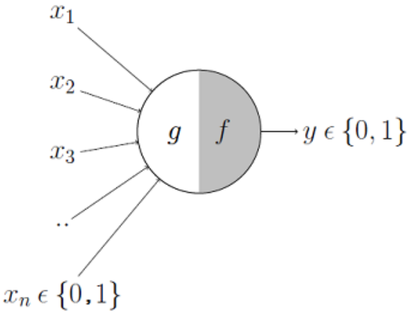

Figure: Model of single neuron - McCulloch-Pitts (Wiki)

Check this page

Stochastic Model of a Neuron

- In an analytically tractable approach, the activation function of McCulloch-Pitts model is given a probabilistic interpretation
- Specifically, the output can only take two states (e.g., -1 or 1)
- The decision for neuron to fire (switch its state from 'ON' and 'OFF') is probabilistic.
- Let x denote the state of the neuron and $P(v)$ denote the probability of firing, where v is the induced local field potential (LFP) of the neuron
- Then $x = \begin{cases} +1 & \text{with probability} \quad P(v) \\ -1 & \text{with probability} \quad 1 - P(v) \end{cases}$
- Example: $P(v) = \frac{1}{1+\exp^{-v/\tau}}$, where τ is an uncertain component controlling the 'noise-level'

Mathematical Model of an Artificial Neuron (Perceptron)

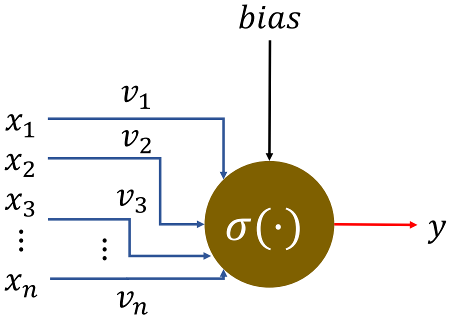

Figure: Model of single neuron - Perceptron (Rosenblatt, 1959)

- $\{v_1, v_2, \dots, v_n\}$ - Dendritic weights
- *bias* - Firing threshold
- $\sigma(\cdot)$ - Nonlinear activation
- $\{x_1, x_2, \dots, x_n\}$ and y - Inputs and Output
- $y = \sigma\left(\sum_{j=1}^{n}(v_j x_j) + bias\right)$

- Positive weight correspond to excitatory synapse and negative weight correspond to inhibitory synapse

Mathematical Model of an Artificial Neuron- Simplified Notation

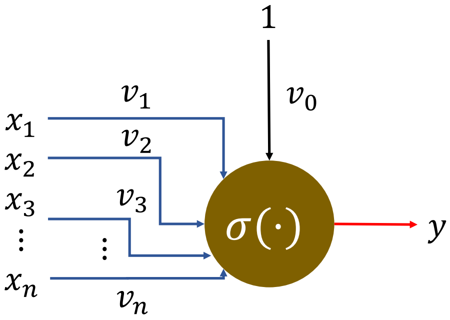

Figure: Model of single neuron - Compact notation

- $\{v_1, v_2, \ldots, v_n\}$ - Dendritic weights
- v_0 - Firing threshold and $x_0 = 1$
- $\sigma(\cdot)$ - Nonlinear activation
- $\{x_1, x_2, \ldots, x_n\}$ and y - Inputs and Output
- $y = \sigma\left(\sum_{j=0}^{n} v_j x_j\right)$

Compact Notation

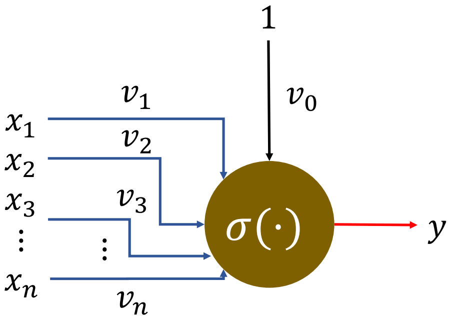

Figure: Model of single neuron - Compact notation

- Vector notations

$$x = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{pmatrix} \in \mathbb{R}^{(n+1)\times 1}, \quad v = \begin{pmatrix} v_0 \\ v_1 \\ \vdots \\ v_n \end{pmatrix} \in \mathbb{R}^{(n+1)}$$

$$y = \sigma(v'x),$$

- where $v' = (v_0, v_1, \ldots, v_n) \in \mathbb{R}^{1\times(n+1)}$

Role of Threshold Functions/ Bias

- The use of bias v_0 has the effect of applying an *affine transformation* to the linear weighted combination of inputs

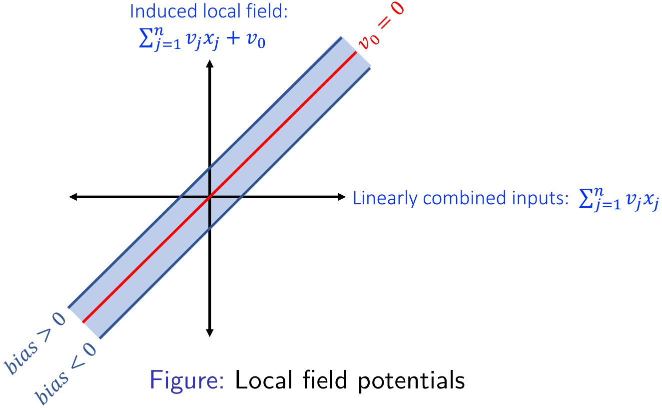

Figure: Local field potentials

- Depending on whether the bias is positive or negative, the relationship between the induced *local field* or *activation potential* of a neuron and the linearly combined output is modified

Activation Functions

- The activation function $\sigma(\cdot)$ is selected on a case-by-case basis
- The role of the activation function is to model the behavior of the nerve cell, where there is no o/p below a certain value of the argument of $\sigma(\cdot)$ and it takes a specific magnitude above the value of the argument.
- A general class of monotonically nondecreasing function taking on bounded values at $-\infty$ to ∞ is the sigmoid functions.
- Typically, the normalized amplitude range of the output of a neuron is written as the closed unit interval (e.g., $[0, 1]$, $[-1, 1]$).

Examples of Activation Function

Example (Threshold Function - Heaviside Function)

Let $\alpha = \sum_{j=1}^{n} v_j x_j + v_0$. The threshold function can be defined as $\sigma(\alpha) = \begin{cases} 1 & \text{if} \quad \alpha > 0 \\ 0 & \text{if} \quad \alpha \leq 0 \end{cases}$

Example (Piecewise Linear Function)

$$\sigma(\alpha) = \begin{cases} 1 & \text{if} \quad \alpha \geq \frac{1}{2} \\ \alpha & \text{if} \quad \frac{1}{2} > \alpha > \frac{-1}{2} \\ 0 & \text{if} \quad \alpha \leq \frac{-1}{2} \end{cases}$$

Example (Sigmoid Function)

- $\sigma(\alpha) = \frac{1}{1+\exp^{-\beta\alpha}}$
- β determines slope
- slope at origin is $\beta/4$ and as $\beta \to \infty$, sigmoid $\to$ threshold

Multilayer Feedforward Neural Networks – Neurodynamics - Rosenblatt, 1962

- "...there has been a failure to comprehend the difference in motivation between the perceptron program and the various engineering projects concerned with automatic pattern recognition, "artificial intelligence", and advanced computers. For this writer, the perceptron program is not primarily concerned with the invention of devices for "artificial intelligence", but rather with investigating the physical structures and neurodynamic principles which underlie "natural intelligence". A perceptron is first: and foremost a brain model, not an invention for pattern recognition...."

Principles of Neurodynamics - Perceptrons and the Theory of Brain Mechanisms

Feedforward Neural Networks

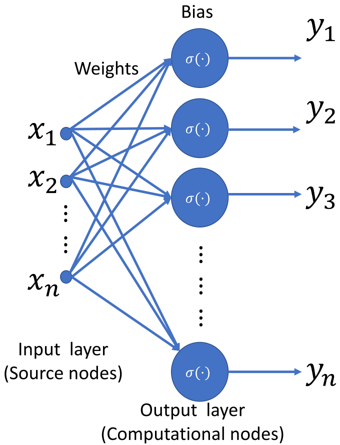

Figure: Single layer feed-forward network

- Neurons are organized in layers
- Simplest form: i/p layer of source nodes that projects on to output layer of neurons (computational node)
- Acyclic or feedforward - because the 'direction' is non-reversible
- This is called *single-layer network* (referring to the o/p layer of computational neurons)

Multilayer Feedforward Neural Networks

- There are networks with one or more hidden layers whose computational nodes are correspondingly hidden layer neurons
- By adding one or more layers, the network is enabled to 'extract higher order statistics' and make abstractions

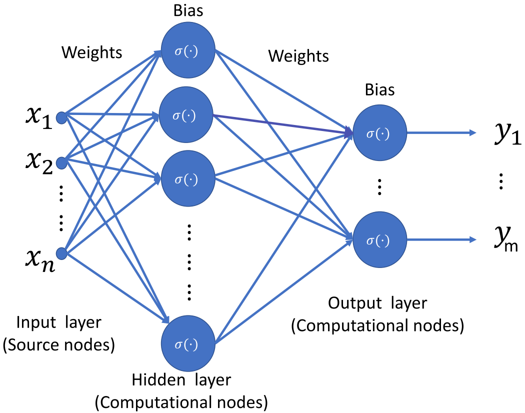

Figure: Multilayer feedforward NN

Two-Layer Feedforward Neural Networks - Output Equation

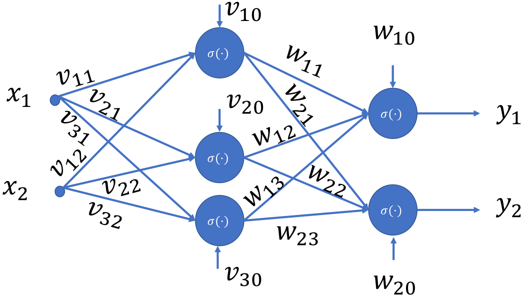

Figure: Two-layer feedforward NN

- Let the output of the hidden layer be $\begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix}$

- $$\begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} = \sigma \begin{pmatrix} v_{10} + v_{11}x_1 + v_{12}x_2 \\ v_{20} + v_{21}x_1 + v_{22}x_2 \\ v_{30} + v_{31}x_1 + v_{32}x_2 \end{pmatrix} = \sigma \left[\begin{pmatrix} v_{10} & v_{11} & v_{12} \\ v_{20} & v_{21} & v_{22} \\ v_{30} & v_{31} & v_{32} \end{pmatrix} \begin{pmatrix} 1 \\ x_1 \\ x_2 \end{pmatrix} \right] = \sigma(Vx)$$

- $z_l = \sigma(\sum_{j=1}^{2} v_{lj}x_j + v_{l0}), \quad l = 1, 2, 3.$

Two-Layer Feedforward Neural Networks - Output Equation

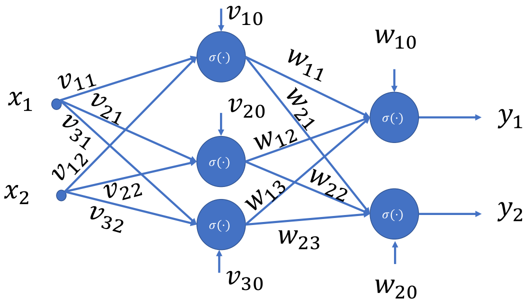

Figure: Two-layer feedforward NN

- Let the output of the NN be $y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$
- $$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \sigma \begin{pmatrix} w_{10} + w_{11} z_1 + w_{12} z_2 + w_{13} z_3 \\ w_{20} + w_{21} z_1 + w_{22} z_2 + w_{23} z_3 \end{pmatrix} = \sigma \left[\begin{pmatrix} w_{10} & w_{11} & w_{12} & w_{13} \\ w_{20} & w_{21} & w_{22} & w_{23} \end{pmatrix} \begin{pmatrix} 1 \\ z_1 \\ z_2 \\ z_3 \end{pmatrix} \right] = \sigma(Wz)$$
- $y_i = \sigma(\sum_{l=1}^{3} w_{il} z_l + w_{i0}), \quad i = 1, 2.$

Linear-in-the-Parameter Networks

- Consider the two-layer NN

$$y = W\phi(vx), \quad \text{output layer activation function is linear.}$$

- If the first layer weights v are predetermined by some apriori technique, then only the second layer weights W and threshold are to be trained
- In this case, we can define $\sigma(x) = \phi(vx)$ so that $y = w\sigma(x)$, where $x \in \mathbb{R}^n$ and $y = \mathbb{R}^m$, $\sigma : \mathbb{R}^n \to \mathbb{R}^L$, L is the number of hidden layer neurons
- $y = W\sigma(x)$ is called *function-link neural network (FLNN, Sadegh, 1993)*
- Here $\sigma(x)$ is allowed to be a general function from $\mathbb{R}^n \to \mathbb{R}^L$ and it is not diagonal.
- RVFL - Random vector functional-link neural network - Stochastic Basis (Igelnik and Pao 1995)