The Deep Mathematics Behind A.I. and Deep Learning

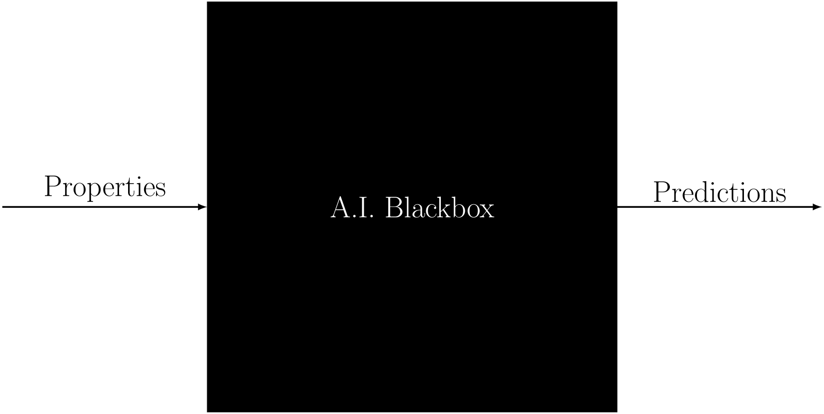
A mathematical generalisation and important theorems in contemporary A.I. and deep learning problems.

Sepehr Saryazdi

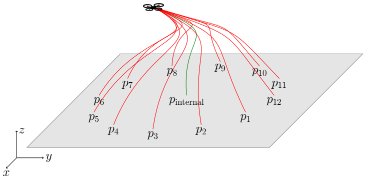
University of Sydney

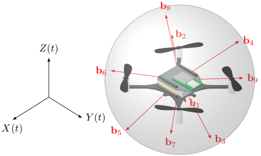
February 25, 2023

Why learn the math behind AI?



Who am I?


p_1
p_2
p_3
p_4
p_5
p_6
p_7
p_8
p_9
p_{10}
p_{11}
p_{12}
p_{internal}
x
y
z


$Z(t)$
$Y(t)$
$X(t)$
$\mathbf{b}_1$
$\mathbf{b}_2$
$\mathbf{b}_3$
$\mathbf{b}_4$
$\mathbf{b}_5$
$\mathbf{b}_6$
$\mathbf{b}_7$
$\mathbf{b}_8$
$\mathbf{b}_9$

Overview

The Face-Space Problem

Logical Boundaries

Fuzzy Logic

ROC/Confusion Matrix Adjustment

A.I. Model Representations

- Let P be a topological space, representing the model's parameters (i.e. $P = \mathbb{R}^n$).
- Let X, Y be topological spaces, and let $C(X, Y)$ be the space of continuous functions from X to Y.
- Let a map that takes a parameter to A.I. models for $X \to Y$ be denoted by $\mathcal{M}$.

A.I. Model Representations

Every A.I. model may be viewed as a function $\mathcal{M}$ that maps parameters to continuous functions from $X \to Y$.

$$\mathcal{M} : P \to C(X, Y)$$

A.I. Model Representations

A.I. Model Representations

Every A.I. model may be viewed as a function $\mathcal{M}$ that maps parameters to continuous functions from $X \to Y$.

$$\mathcal{M} : P \to C(X, Y)$$

Example 1: $\mathbb{R} \to \mathbb{R}$ Linear Model

$$P = \mathbb{R}^2, (a, b) \in P, X = Y = \mathbb{R}$$

$$(\mathcal{M}(a, b))(x) := a + bx$$

A.I. Model Representations: $\mathbb{R} \to \mathbb{R}$ Linear Model $(\mathcal{M}(a, b))(x) := a + bx$

A.I. Model Representations: $\mathbb{R} \to \mathbb{R}$ Quadratic Model $(\mathcal{M}(a, b, c))(x) := a + bx + cx^2$

A.I. Model Representations: $\mathbb{R} \to \mathbb{R}^2 \to \mathbb{R}$ Neural Network Model

Example 3: $\mathbb{R} \to \mathbb{R}^2 \to \mathbb{R}$ Neural Network Model

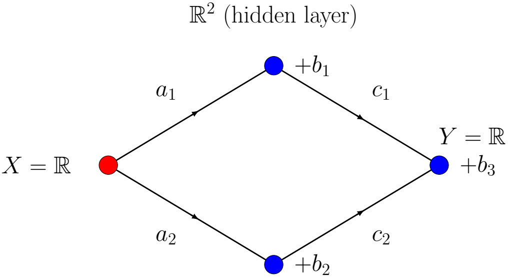
$$P = \mathbb{R}^7, (a_1, a_1, b_1, b_2, b_3, c_1, c_2) \in P, X = Y = \mathbb{R}$$

$$(\mathcal{M}(a_1, a_1, b_1, b_2, b_3, c_1, c_2))(x)$$

$$:= \left(\mathrm{ReLU}\left(\begin{pmatrix} a_1 & a_2 \end{pmatrix} x + \begin{pmatrix} b_1 & b_2 \end{pmatrix}\right)\right) \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} + b_3$$

$$\mathrm{ReLU}(x_1, x_2) := \begin{pmatrix} \max(x_1, 0) & \max(x_2, 0) \end{pmatrix}$$

A.I. Model Representations: $\mathbb{R} \to \mathbb{R}^2 \to \mathbb{R}$ Neural Network Model



Generalised Loss Functions

- Let P, X, Y be topological spaces and $\mathcal{M} : P \to C(X, Y)$ a function.
- Let $d : C(X, Y) \times C(X, Y) \to \mathbb{R}_{\geq 0}$ be a metric.
- Let L_f for some $f \in C(X, Y)$ be named a "loss function".

Generalised Loss Functions

Define the loss function $L_f : P \to \mathbb{R}_{\geq 0}$ to satisfy

$$L_f(p) := d(f, \mathcal{M}(p))$$

Generalised Loss Functions

Example: $L^2(X, \mathbb{R}_{\geq 0})$ Loss Function

$$L_f(p) = \int_X ||f(x) - (\mathcal{M}(p))(x)||^2 \mathrm{d}x$$

Loss Surfaces: $f(x) := \sin x$, $(M(a,b))(x) := a + bx$, L^2-Loss Function

Loss Surfaces: Optimal Parameter Search

- Let $(p_n)_{n\in\mathbb{N}} \subseteq P$ be a sequence of parameters.

Optimal Parameter Search

An optimal parameter search is an algorithm where for any $p_0 \in P$, it returns a sequence $(p_n)_{n\in\mathbb{N}} \subseteq P$ such that

$$L_f(p_n) \xrightarrow[n\to\infty]{} \inf_{p\in P} L_f(p)$$

- Note: Such a sequence need not have a unique limit, as P and $\mathcal{M}$ may over-cover the goal function f.

Loss Surfaces: Gradient Descent

Loss Surfaces: Gradient Descent

- Let $\gamma \in \mathbb{R}_{>0}$ be named the "learning rate".
- Let ∇L_f denote the gradient function of L_f.

Gradient Descent Optimal Parameter Search Algorithm

Assuming $p_0 \in P$ is given, construct the sequence $(p_n)_{n \in \mathbb{N}}$ with recursion given by

$$p_{n+1} = p_n - \gamma(\nabla L_f)(p)$$

Loss Functions: Finite Sampling of $f : X \to Y$

- In practice, we can only sample $N \in \mathbb{N}$ finitely many points $(x_i, y_i) \in X \times Y$ with $f(x_i) = y_i$.
- This collapses the L^2 loss function to a finite sum

Finite Sampling of $L^2(X, \mathbb{R}_{\geq 0})$ Loss Function (MSE)

$$L_f(p) \approx \frac{1}{N} \sum_{i=1}^{N} ||f(x_i) - (\mathcal{M}(p))(x_i)||^2$$

- As $N \to \infty$ and assuming the sampling is distributed uniformly over X, then this will approach the true L^2 loss function.

Kolmogorov–Arnold Representation Theorem

- This is a solution to the famous 13th Hilbert Problem.
- Colloquially, this theorem says that the "only true continuous multivariable function is the sum".

KA Representation Theorem

Let $f : \mathbb{R}^n \to \mathbb{R}^m$ with $f(\mathbf{x}) := f(x_1, ..., x_n)$ be a continuous function. Then there exists univariate functions $\Phi_q : \mathbb{R} \to \mathbb{R}^m, \phi_{q,p} : \mathbb{R} \to \mathbb{R}$ such that:

$$f(\mathbf{x}) = \sum_{q=0}^{2n} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right)$$

Consequences of KA Representation Theorem

- From this theorem, feature engineering was born.
- This theorem allows us to transform each data column independently before combining them with sums in an A.I. algorithm.

x_1	x_2	x_3
10	99	85
67	13	8
82	48	89
$\vdots$	$\vdots$	$\vdots$
7	76	25

$\longrightarrow$

$0.5x_1 + 2x_2 - x_3$
118
51.5
48
$\vdots$
130.5

Universal Approximation Theorem

- This says that neural networks can be used to approximate any continuous function if X is closed and bounded.

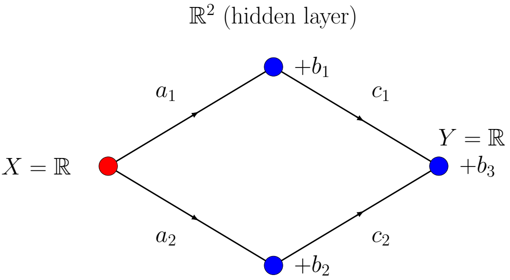
Universal Approximation Theorem

If given $f : X \subseteq \mathbb{R}^n \to \mathbb{R}^m$ and a valid non-polynomial function $\sigma : \mathbb{R} \to \mathbb{R}$, then $L_f(p) := ||f - \mathcal{M}(p)||_\infty$ can be made arbitrarily small where

$$(\mathcal{M}(p))(x) := C_p\left(\sigma \circ \left(A_p(x) + b\right)\right)$$

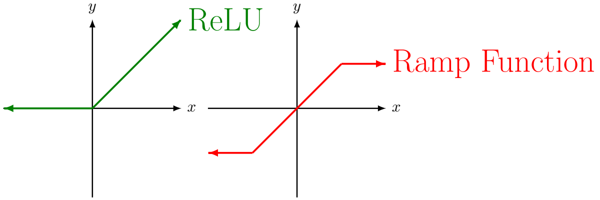
$$p \in P := \mathbb{R}^{k(1+m+n)}, A_p \in \mathbb{R}^{k \times n}, C_p \in \mathbb{R}^{m \times k}, b \in \mathbb{R}^k$$

Universal Approximation Theorem



Universal Approximation Theorem: How It Works

- The choice of $\sigma : \mathbb{R} \to \mathbb{R}$ was really the crucial part of the entire theorem.
- This is because σ acts as a tool for approximating the 'ramp' function and this can then approximate any continuous function.



Mercer's Theorem

Representer Theorem

Representer Theorem