researchai

sampath kovvali

July 7, 2023

Notation

This section provides a concise reference describing the notation used throughout this document.

Numbers and Arrays

a	A scalar (integer or real)
$\boldsymbol{a}$	A vector usually written as a column vector
$\max(\boldsymbol{a})$	max of $\boldsymbol{a}$, output is a scalar
$\boldsymbol{A}$	A matrix
A	A tensor
$\boldsymbol{I}_n$	Identity matrix with n rows and n columns
$\boldsymbol{I}$	Identity matrix with dimensionality implied by context
$\boldsymbol{e}^{(i)}$	Standard basis vector $[0, \dots, 0, 1, 0, \dots, 0]$ with a 1 at position i
$\text{diag}(\boldsymbol{a})$	A square, diagonal matrix with diagonal entries given by $\boldsymbol{a}$
a	A scalar random variable
$\mathbf{a}$	A vector-valued random variable
$\mathbf{A}$	A matrix-valued random variable

Sets and Graphs

$\mathbb{A}$	A set
$\mathbb{R}$	The set of real numbers
$\{0, 1\}$	The set containing 0 and 1
$\{0, 1, \ldots, n\}$	The set of all integers between 0 and n
$[a, b]$	The real interval including a and b
$(a, b]$	The real interval excluding a but including b
$\mathbb{A} \backslash \mathbb{B}$	Set subtraction, i.e., the set containing the elements of $\mathbb{A}$ that are not in $\mathbb{B}$
$\mathcal{G}$	A graph

Indexing

a_i	Element i of vector $\boldsymbol{a}$, with indexing starting at 1
a_{-i}	All elements of vector $\boldsymbol{a}$ except for element i
$A_{i,j}$	Element i, j of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{i,:}$	Row i of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{:,i}$	Column i of matrix $\boldsymbol{A}$
$\mathsf{A}_{i,j,k}$	Element (i, j, k) of a 3-D tensor $\mathbf{A}$
$\mathbf{A}_{:,:,i}$	2-D slice of a 3-D tensor
a_i	Element i of the random vector $\mathbf{a}$

Linear Algebra Operations

$\boldsymbol{A}^\top$	Transpose of matrix $\boldsymbol{A}$
$\boldsymbol{A}^+$	Moore-Penrose pseudo inverse of $\boldsymbol{A}$
$\boldsymbol{A} \odot \boldsymbol{B}$	Element-wise (Hadamard) product of $\boldsymbol{A}$ and $\boldsymbol{B}$
$\det(\boldsymbol{A})$	Determinant of $\boldsymbol{A}$
$\boldsymbol{A}\boldsymbol{x}$	matrix vector product
$\boldsymbol{A}\boldsymbol{B}$	matrix product

Calculus

$\frac{dy}{dx}$	Derivative of y with respect to x
$\frac{\partial y}{\partial x}$	Partial derivative of y with respect to x
$\nabla_{\boldsymbol{x}} y$	Gradient of y with respect to $\boldsymbol{x}$
$\nabla_{\boldsymbol{X}} y$	Matrix derivatives of y with respect to $\boldsymbol{X}$
$\nabla_{\mathbf{X}} y$	Tensor containing derivatives of y with respect to $\mathbf{X}$
$\frac{\partial f}{\partial \boldsymbol{x}}$	Jacobian matrix $\boldsymbol{J} \in \mathbb{R}^{m \times n}$ of $f : \mathbb{R}^n \to \mathbb{R}^m$
$\nabla^2_{\boldsymbol{x}} f(\boldsymbol{x})$ or $\boldsymbol{H}(f)(\boldsymbol{x})$	The Hessian matrix of f at input point $\boldsymbol{x}$
$\int f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral over the entire domain of $\boldsymbol{x}$
$\int_{\mathbb{S}} f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral with respect to $\boldsymbol{x}$ over the set $\mathbb{S}$

Probability and Information Theory

$\mathrm{a} \perp \mathrm{b}$	The random variables a and b are independent
$\mathrm{a} \perp \mathrm{b} \mid \mathrm{c}$	They are conditionally independent given c
$P(\mathrm{a})$	A probability distribution over a discrete variable
$p(\mathrm{a})$	A probability distribution over a continuous variable, or over a variable whose type has not been specified
$\mathrm{a} \sim P$	Random variable a has distribution P
$\mathbb{E}_{\mathrm{x} \sim P}[f(x)]$ or $\mathbb{E}f(x)$	Expectation of $f(x)$ with respect to $P(\mathrm{x})$
$\mathrm{Var}(f(x))$	Variance of $f(x)$ under $P(\mathrm{x})$
$\mathrm{Cov}(f(x), g(x))$	Covariance of $f(x)$ and $g(x)$ under $P(\mathrm{x})$
$H(\mathrm{x})$	Shannon entropy of the random variable x
$D_{\mathrm{KL}}(P \Vert Q)$	Kullback-Leibler divergence of P and Q
$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution over $\boldsymbol{x}$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$

Functions

$f : \mathbb{A} \to \mathbb{B}$	The function f with domain $\mathbb{A}$ and range $\mathbb{B}$
$f \circ g$	Composition of the functions f and g
$f(\boldsymbol{x}; \boldsymbol{\theta})$	A function of $\boldsymbol{x}$ parametrized by $\boldsymbol{\theta}$. (Sometimes we write $f(\boldsymbol{x})$ and omit the argument $\boldsymbol{\theta}$ to lighten notation)
$\log x$	Natural logarithm of x
$\sigma(x)$	Logistic sigmoid, $\dfrac{1}{1 + \exp(-x)}$
$\zeta(x)$	Softplus, $\log(1 + \exp(x))$
$\Vert \boldsymbol{x} \Vert_p$	L^p norm of $\boldsymbol{x}$
$\Vert \boldsymbol{x} \Vert$	L^2 norm of $\boldsymbol{x}$
x^+	Positive part of x, i.e., $\max(0, x)$

Datasets and Distributions

$\mathbb{X}$	A set of training examples
$\boldsymbol{x}^{(i)}$	The i-th example (input) from a dataset
$y^{(i)}$ or $\boldsymbol{y}^{(i)}$	The target associated with $\boldsymbol{x}^{(i)}$ for supervised learning
$\boldsymbol{X}$	The $m \times n$ matrix with input example $\boldsymbol{x}^{(i)}$ in row $\boldsymbol{X}_{i,:}$

Introduction

The motivation behind writing this book is to understand the math used in **AI** algorithms like **Linear Regression**, **KNN**, to all types of **Neural Networks** during my research. This book contains the math with diagrams where ever necessary to understand the core principles of algorithms. How this book differs from other AI books is by introducing the algorithms just not standalone math.

Disclaimer: This book does not include computational algorithms like Python code. It only includes the math required to write code. And certainly not a math book. Think like a reference book whenever you need to code an algorithm.

Short history of neural networks

Throughout history, people have used various methods to perform mathematical calculations, ranging from using their hands and sticks to mechanical integrators in the 18th century, which were specifically designed to solve integration problems. With the introduction of electricity to human society, many began to power their calculators using this new resource. One example is the Bombe, a device designed by Alan Turing during World War II for cracking German codes.

In 1943, Warren McCulloch and Walter Pitts developed a neural network model inspired by biological neurons, known as the McCulloch-Pitts neural network. This marked the beginning of research into teaching machines to learn from data, which has since become a highly demanded field due to the increasing computational requirements of modern society.

Neural Networks

The working principles and architectures of various **NN** are inspired from many papers.

1 Layers

All types of layers for any type of networks.

1.1 Dense

These are the full connected, where one layer's output is input to the next layer's during **forward pass** and the reverse when using **backpropagration algorithm**.

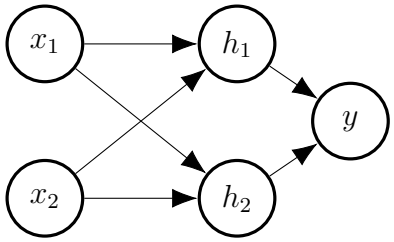

Figure 1: Simple feed forward network of one **Dense** layer with 2 inputs $x1$, $x2$ and two hidden units $h1$, $h2$ and one output y.

For the hidden layer if the inputs be $\boldsymbol{x}$, weights be $\boldsymbol{W}$ where $\boldsymbol{W}_{:,0}$ the connections from $x1$ to $\boldsymbol{h}$ and so on, biases $\boldsymbol{b}$. Then output $\boldsymbol{o}$ is given by

$$\boldsymbol{o} = \boldsymbol{x}^\top \boldsymbol{W} \tag{1}$$

similarly, this applies to the output layer to obtain y.

Batched data

If the inputs are to be passes as a batch of data $\boldsymbol{X}$, then the output $\boldsymbol{O}$ for one layer is computed as

$$\boldsymbol{O} = \boldsymbol{X}\boldsymbol{W} \tag{2}$$

2 Activation functions

All types of activation's for any layers.

2.1 RelU

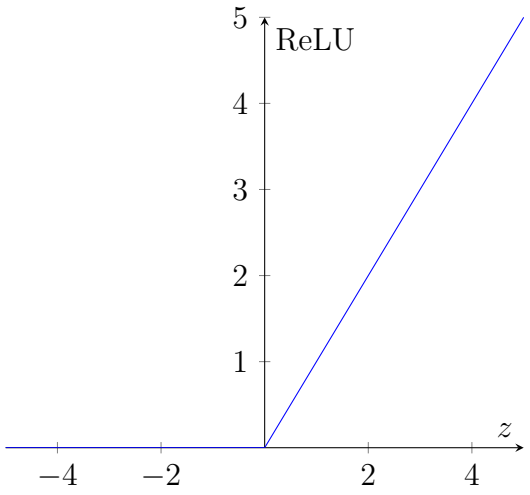

Figure 2: ReLU function

After forward pass we can get activated output $\boldsymbol{O}$ by

$$\text{ReLU}(z) = \max\{0, z\} \tag{3}$$

2.2 Softmax

It is usually used in the output layer.

$$\text{softmax}(\boldsymbol{x})_i = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{4}$$

2.2.1 Numerical stability

For overflow prevention $\boldsymbol{x} = \boldsymbol{x} - \max(\boldsymbol{x})$.

- All exponentiated values will be between 0 and 1.

- This prevents overflow errors (but we are still prone to underflows)
- At least one of the exponentiated values is 1 i.e. at least one value is guarenteed not to underflow
- Our denominator will always be ≥ 1, preventing division by zero errors.
- We have at least one non-zero numerator, so softmax can't result in a zero vector

[1]

3 Cost functions

3.1 Cross Entropy

Cross-entropy is a measure from the field of information theory that calculates the difference between two probability distributions. It is commonly used in machine learning as a loss function.

$$H(\boldsymbol{x}, \boldsymbol{y}) = -\sum_{j} P(x_j) \log Q(y_j) \tag{5}$$

3.1.1 Numerical stability

Cross-entropy results in **inf** because of $\log(0)$ so, we need to clip inputs from both sides by a small positive value like 1×10^{-7}.

References

[1] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016, p. 78. URL: http://www.deeplearningbook.org.