Hands-on-SGD

Release 2022

Paul Rodriguez

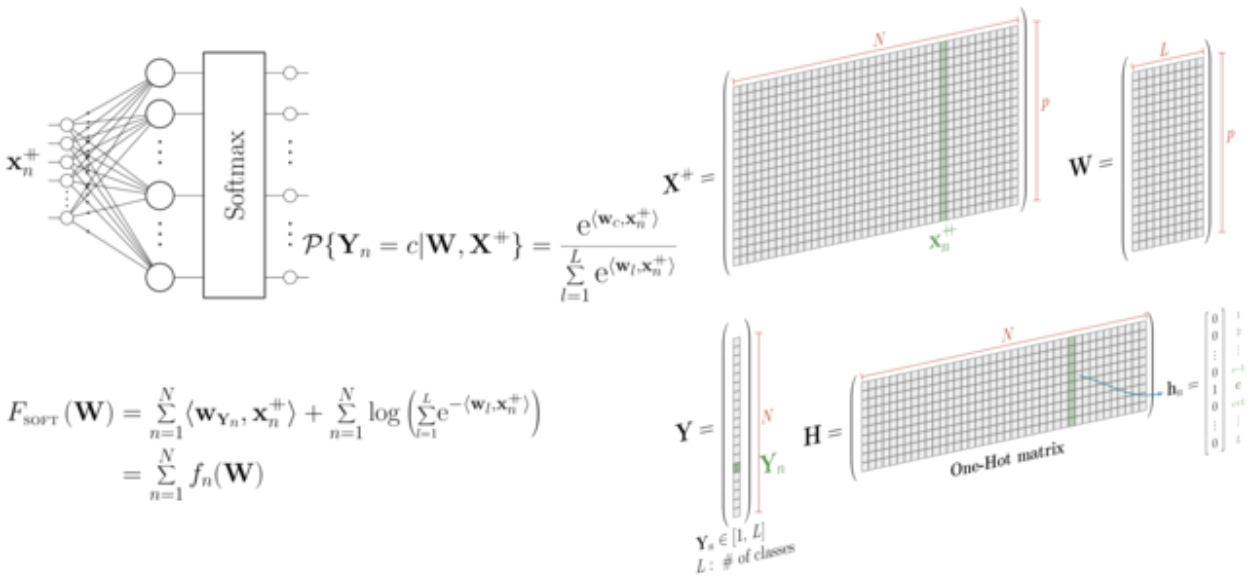
Jun 09, 2023

COURSE DESCRIPTION

CHAPTER

ONE

EDUCATION SHORT COURSE



Important:

- This documentation is related to SC-1 Short Courses @ ICASSP'23.
- Last update: June 9th, 2023.
- Lecture slides are intended to be observed in **presentation mode** (full screen).
- Partially supported by the Army Research Office (ARO) under Grant # W911NF-22-1-0296.

CHAPTER

TWO

COURSE DESCRIPTION

Gradient descent (GD) is a well-known first order optimization method, which uses the gradient of the loss function, along with a step-size (or learning rate), to iteratively update the solution. When the loss (cost) function is dependent on datasets with large cardinality, such in cases typically associated with deep learning (DL), GD becomes impractical.

In this scenario, stochastic GD (SGD), which uses a noisy gradient approximation (computed over a random fraction of the dataset), has become crucial. There exits several variants/improvements over the "vanilla" SGD, such RMSprop, Adagrad, Adadelta, Adam, Nadam, etc., which are usually given as black-boxes by most of DL's libraries (TensorFlow, PyTorch, MXNet, etc.).

The primary objective of this course is to combined the essential theoretical aspects related to SGD and variants, along with hands on experience to program in Python, from scratch (i.e. not based on DL's libraries such as TensorFlow, PyTorch, MXNet) the SGD along with the RMSprop, Adagrad, Adadelta, Adam and Nadam algorithms and to test their performance using the MNIST and CIFAR-10 datasets for shallow networks (consisting of up to two ReLU layers and a Softmax as the last layer).

Lecture 1

- Introduction
- – Bayes' theorem.
  – MAP (maximum a posteriori).
  – Linear regression.
  – Logistic and softmax regression.
  – Gradient descent (GD) and stochastic GD.
- – The MNIST dataset.
  – Data preparation.
  – GD implementation; simple (quadratic) test.
  – SGD implementation; multiclass regression for the MNIST dataset.

Lecture 2

- – Adaptive step-sizes
  – Momentum.
  – Nesterov acceleration.
  – Anderson acceleration.
  – ∗ Accelerated GD implementation.
    ∗ Comparisons w.r.t. GD.
- – Momentum (SGD-MTM)
  – Nesterov (SGD-NTRV)
  – SG Clipping (SGC)
  – Adagrad
  – Adadelta
  – RMSprop
  – Adam
  – AdaMax
  – Nadam
  – AdaBelief
  – SGD variants' taxonomy
  – ∗ SGD variants implementation.
    ∗ Multiclass regression for the MNIST dataset.

Lecture 3

- – Introduction.
  – Linear vs. non-linear.
  – Activation functions.
  – ∗ Impact of adding one, random value, ReLU hidden layer.
    ∗ Classification of the MNIST dataset.
    ∗ Classification of the CIFAR dataset.
- – Introduction.
  – The backpropagation (BP) algorithm.
  – SGD and BP working together.
  – ∗ BP and SGD along with one ReLU hidden layer.
    ∗ Classification of the CIFAR dataset.

Lecture 4

- – BP and SGD along with two hidden layers / SGD variants.
  – Classification of the CIFAR dataset.
- – Introduction.
  – Convolutional layer
  – Other layers: maxpool, dropout, dense, etc.
  – DL libraries: TensorFlow, PyTorch, MXNet.
  – * Using TensorFlow (TF).
    * Performance comparison (w.r.t. Previously developed code).
    * Implementing your own solver in TF.
    * Using simple DL networks.

2.1 Session 1

2.1.1 Lecture slides: Here

2.1.2 GD – Hands-on: exercises and solution.

2.1.3 SGD – Hands-on: exercises and solution.

2.2 Session 2

2.2.1 Lecture slides: Part A, Part B

2.2.2 Accelerated GD – Hands-on: exercises and solution.

2.2.3 SGD variants – Hands-on: exercises and solution.

2.3 Session 3

2.3.1 Lecture slides: Hands-on: Part A, Part B

2.3.2 Hidden layer, ELM – Hands-on: exercises and solution.

2.3.3 Backpropagation – Hands-on: exercises and solution.

2.4 Session 4

2.4.1 Lecture slides: Here

2.4.2 TensorFlow own variant – Hands-on: exercises and solution.