Practical 2.2

Neural Networks – An overview on Torch's nn package

Overview (I)

- Jacobian and Hessian partial derivative formulation
- Logistic unit with `nn` package, forward step-by-step
  - `nn.Linear(in, out)`
    - `{…}` operator
    - `nn.Linear().weight` and `nn.Linear().bias`
    - `nn.Linear().gradWeight` and `nn.Linear().gradBias`
    - `nn.Module():zeroGradParameters()`
  - `nn.Sigmoid()`
  - `nn.Module():forward()`

Overview (II)

- Logistic unit with `nn` package, back-propagation step-by-step
  - `nn.MSECriterion()` loss function
    - `nn.MSECriterion().sizeAverage`
    - `nn.MSECriterion():forward(input, target)`
    - `nn.MSECriterion():updateGradInput(input, target)`
  - `nn.Module():updateGradInput(input, gradOutput)`
  - `nn.Linear():accGradParameters(input, gradOutput)`

Overview (III)

- Logistic unit with `nn` package, using `nn.Sequential()`
  - `nn.Sequential():add()`
  - `nn.Sequential():forward(input)`
  - `nn.Criterion():forward(input, target)`
  - `nn.Criterion():backward(input, target)`
  - `nn.Sequential():get(layer)`
  - `nn.Sequential():zeroAccParameters()`
  - `nn.Sequential():backward(input, gradCriterion)`
  - `nn.Sequential():updateParameters(etha)`

Overview (IV)

- Training a generic network
  - Stochastic Gradient Descent (SGD)
  - (Mini-) Batch Gradient Descent (BGD)
- Training with `nn.StochasticGradient(net, loss)`
  - `nn.StochasticGradient():train(dataset)`
- Regression with a 3-layer neural network
  - github.com/Atcold/torch-Machine-learning-with-Torch

Notation (I) (Jacobian formulation)

$$\frac{\partial y}{\partial \underline{x}} = \begin{bmatrix} \frac{\partial y}{\partial x_1} & \cdots & \frac{\partial y}{\partial x_n} \end{bmatrix}$$

$$\frac{d \underline{y}}{d x} = \begin{bmatrix} \frac{d y_1}{d x} \\ \vdots \\ \frac{d y_m}{d x} \end{bmatrix}$$

$$\frac{\partial \underline{y}}{\partial \underline{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial y_m}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix} = J(\underline{y})$$

$$\frac{\partial y}{\partial X} \in \mathbb{R}^{n \times m}$$

$$[X] \in \mathbb{R}^{m \times n}$$

Wiki → MATRIX CALCULUS

Notation (II)

$$v) \quad \underbrace{\frac{\partial E}{\partial \boldsymbol{\Theta}^{(\ell)}}}_{\mathbb{R}^{s_{\ell+1} \times (s_\ell + 1)}} = \underbrace{\hat{\mathbf{a}}^{(\ell)}}_{s_\ell + 1} \overbrace{(\overbrace{\boldsymbol{\delta}^{(\ell+1)}}^{s_{\ell+1}})^\top}^{1 \times s_{\ell+1}} \in \mathbb{R}^{(s_\ell + 1) \times s_{\ell+1}}$$

HESSIAN FORMULATION ← DENOMINATOR layout

$$\Theta \to \underbrace{\Theta} + \eta \underbrace{\frac{\partial E}{\partial \Theta}}$$

$$\frac{\partial y}{\partial X} \in \mathbb{R}^{m \times n}$$

$$X \in \mathbb{R}^{m \times n}$$

$$(\quad)^\top$$ JACOBIAN FORMULATION

NUMERATOR layout