Deep Learning

Lecture 4

Fernando Perez-Cruz
based on Thomas Hofmann lectures

Swiss Data Science Center
ETH Zurich and EPFL – datascience.ch

October 15th, 2018

Overview

1. Backpropagation

Section 1

Backpropagation

Gradient Descent

Learning in neural networks = gradient-based optimization (with very few exceptions)

- gradient of objective with regard to parameters θ

$$\nabla_\theta \mathcal{R} = \left(\frac{\partial \mathcal{R}}{\partial \theta_1}, \ldots, \frac{\partial \mathcal{R}}{\partial \theta_d} \right)^\top$$

- steepest descent and stochastic gradient decent

$$\theta(t+1) \leftarrow \theta(t) - \eta \, \nabla_\theta \, \mathcal{R}(\mathcal{S})$$

  - here $t = 0, 1, 2, \ldots$ is an iteration index
  - $\mathcal{S}$ = all training data $\Longrightarrow$ steepest descent
  - $\mathcal{S}$ = mini batch of data $\Longrightarrow$ SGD

Gradient Computation via Backpropagation

Computational challenge: how to compute $\nabla_\theta \mathcal{R}$?

Exploit compositional structure of network = backpropagation

Basic steps:

1. perform a forward pass (for given training input $\mathbf{x}$) to compute activations for all units
2. compute gradient of $\mathcal{R}$ w.r.t. output layer activations (for given target $\mathbf{y}$)
3. iteratively propagate activation gradient information from outputs to inputs
4. compute local gradients of activations w.r.t. weights

Backpropagation in Plain English

- How do changes in the output layer activities change the objective?
  - depends on choice of objective
- How does the activity of a parent unit influence the activity of each of its child units (in DAG)?
  - layer structure $\Longrightarrow$ concurrently between subsequent layers
- Propagate influence information through reverse DAG
  - details are implied by chain rule of differentiation
- What is the effect of a change of an incoming weight on the activity of a unit?
  - can only change activities (given $\mathbf{x}$) by modifying weights

Chain Rule

Compositionality of functions $\Longrightarrow$ use of chain rule

- chain rule

$$(f \circ g)' = (f' \circ g) \cdot g'$$

- or – equivalently – with formal variables

$$\frac{d(f \circ g)}{dx}\Big|_{x=x_0} = \frac{df}{dz}\Big|_{z=g(x_0)} \cdot \frac{dg}{dx}\Big|_{x=x_0}$$

Jacobi Matrix

Vector-valued function (map) $F : \mathbb{R}^n \to \mathbb{R}^m$

- each component function has gradient $\nabla F_i \in \mathbb{R}^n$, $i \in [1:m]$
- collect all gradients (as rows) into Jacobi matrix

$$\mathbf{J}_F := \begin{pmatrix} \nabla^\top F_1 \\ \nabla^\top F_2 \\ \cdots \\ \nabla^\top F_m \end{pmatrix} = \begin{pmatrix} \frac{\partial F_1}{\partial x_1} & \frac{\partial F_1}{\partial x_2} & \cdots & \frac{\partial F_1}{\partial x_n} \\ \frac{\partial F_2}{\partial x_1} & \frac{\partial F_2}{\partial x_2} & \cdots & \frac{\partial F_2}{\partial x_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial F_m}{\partial x_1} & \frac{\partial F_m}{\partial x_2} & \cdots & \frac{\partial F_m}{\partial x_n} \end{pmatrix} \in \mathbb{R}^{m \times n}$$

- derivative of each output with regard to each input, i.e. $(\mathbf{J}_F)_{ij} = \partial F_i / \partial x_j$

Jacobi Matrix Chain Rule

Vector-valued functions $G : \mathbb{R}^n \to \mathbb{R}^q$, $H : \mathbb{R}^q \to \mathbb{R}^m$, $F := H \circ G$

- componentwise rule

$$\frac{\partial F_i}{\partial x_j}\Big|_{\mathbf{x}=\mathbf{x}_0} = \frac{\partial (H \circ G)_i}{\partial x_j}\Big|_{\mathbf{x}=\mathbf{x}_0} = \sum_{k=1}^{q} \frac{\partial H_i}{\partial z_k}\Big|_{\mathbf{z}=G(\mathbf{x}_0)} \cdot \frac{\partial G_k}{\partial x_j}\Big|_{\mathbf{x}=\mathbf{x}_0}$$

<u>Lemma</u>: Jacobi matrix chain rule

$$\mathbf{J}_{H \circ G}\Big|_{\mathbf{x}=\mathbf{x}_0} = \mathbf{J}_H\Big|_{\mathbf{z}=G(\mathbf{x}_0)} \cdot \mathbf{J}_G\Big|_{\mathbf{x}=\mathbf{x}_0}$$

Function Composition

Special case: composition of a map with a function

$$G : \mathbb{R}^n \to \mathbb{R}^m, \quad h : \mathbb{R}^m \to \mathbb{R}, \quad h \circ G : \mathbb{R}^n \to \mathbb{R}$$

Use more intuitive variable notation

$$\mathbb{R}^n \ni \mathbf{x} \stackrel{G}{\mapsto} \mathbf{y} \stackrel{h}{\mapsto} z \in \mathbb{R}$$

Then

$$\nabla_{\mathbf{x}}^\top z = \nabla_{\mathbf{y}}^\top z \ \cdot \ \mathbf{J}_G, \quad \frac{\partial z}{\partial x_i} = \sum_j \frac{\partial z}{\partial y_j} \frac{\partial y_j}{\partial x_i}$$

Warning: Notation!

We have a lot of indices!

- index of a layer: put as a superscript
- index of a dimension of a vector: put as a subscript
- shorthand for layer activations

$$\mathbf{x}^l := (F^l \circ \cdots \circ F^1)(\mathbf{x}) \in \mathbb{R}^{m_l}$$

$$x_i^l \in \mathbb{R} : \text{ activation of } i\text{-th unit in layer } l$$

- index of a data point, omitted where possible, rectangular brackets $(\mathbf{x}[i], \mathbf{y}[i])$

Deep Function Compositions

Composition of multiple maps with a final cost function

$$F = F^L \circ \cdots \circ F^1 : \mathbb{R}^n \to \mathbb{R}^m$$

$$\mathbf{x} = \mathbf{x}^0 \overset{F^1}{\mapsto} \mathbf{x}^1 \overset{F^2}{\mapsto} \mathbf{x}^2 \mapsto \cdots \overset{F^L}{\mapsto} \mathbf{x}^L = \mathbf{y} \overset{\mathcal{R}}{\mapsto} \mathcal{R}(\theta; \mathbf{y})$$

Proposition: Activity Backpropagation

$$\mathbf{e}^L := \nabla_{\mathbf{y}}^\top \mathcal{R}, \quad \mathbf{e}^l := \nabla_{\mathbf{x}^l}^\top \mathcal{R} = \mathbf{e}^L \cdot \mathbf{J}_{F^L} \cdots \mathbf{J}_{F^{l+1}} = \mathbf{e}^{l+1} \cdot \mathbf{J}_{F^{l+1}}$$

Compute activity gradients in backward order via successive multiplication with Jacobians. Backpropagation of error terms $\mathbf{e}^l$.

Linear network in reversed direction with "activities" $\mathbf{e}^l$.

Jacobian Matrix: Ridge Functions

How does a Jacobian matrix for a ridge function look like?

$$\mathbf{x}^l = F^l(\mathbf{x}^{l-1}) = \sigma\left(\mathbf{W}^l \mathbf{x}^{l-1} + \mathbf{b}^l\right)$$

Hence (assuming differentiability of σ):

$$\frac{\partial x_i^l}{\partial x_j^{l-1}} = \sigma'\left(\langle \mathbf{w}_i^l, \mathbf{x}^{l-1}\rangle + b_i^l\right) w_{ij}^l =: \tilde{w}_{ij}^l$$

and thus simply (!)

$$\mathbf{J}_{F^l} = \tilde{\mathbf{W}}^l$$

- for ReLU $\tilde{w}_{ij}^l \in \{0, w_{ij}^l\} \implies \tilde{\mathbf{W}}^l =$ sparsified weight matrix

Loss Function (Negative) Gradients

Quadratic loss

$$-\nabla_{\mathbf{y}} \mathcal{R}(\mathbf{x}, \mathbf{y}^*) = -\nabla_{\mathbf{y}} \frac{1}{2} \|\mathbf{y}^* - \mathbf{y}\|^2 = \mathbf{y}^* - \mathbf{y}$$

Multivariate logistic loss

$$-\frac{\partial \mathcal{R}(\mathbf{x}, y^*)}{\partial z_y} = \frac{\partial}{\partial z_y} \left[z_{y^*} - \log \sum_i \exp[z_i] \right]$$

$$= \delta_{yy^*} - \frac{\exp[z_y]}{\sum_i \exp[z_i]} = \delta_{yy^*} - p(y|\mathbf{x})$$

From Activations to Weights

How can we get from gradients w.r.t. activations to gradients w.r.t. weights? Easily!

Need to apply chain rule one more time - locally:

$$\frac{\partial \mathcal{R}}{\partial w_{ij}^l} = \frac{\partial \mathcal{R}}{\partial x_i^l} \cdot \frac{\partial x_i^l}{\partial w_{ij}^l} = \underbrace{\frac{\partial \mathcal{R}}{\partial x_i^l}}_{\text{backprop}} \cdot \underbrace{\sigma'\left(\langle \mathbf{w}_i^l, \mathbf{x}^{l-1}\rangle + b_i^l\right)}_{\text{sensitivity of } i\text{-th unit}} \cdot \underbrace{x_j^{l-1}}_{\substack{j\text{-th unit} \\ \text{activity}}}$$

$$\frac{\partial \mathcal{R}}{\partial b_i^l} = \frac{\partial \mathcal{R}}{\partial x_i^l} \cdot \frac{\partial x_i^l}{\partial b_i^l} = \frac{\partial \mathcal{R}}{\partial x_i^l} \cdot \sigma'\left(\langle \mathbf{w}_i^l, \mathbf{x}^{l-1}\rangle + b_i^l\right) \cdot 1$$

- each weight/bias influences exactly one unit
- can "reshape" gradient into matrix/tensor form

Specialized Programming Languages: Theano

Symbolic representation of mathematical expressions.

Access to full computational graph (stability, optimization).

Symbolic differentiation.

Bergstra, James, et al. "Theano: A CPU and GPU math compiler in Python." Proc. 9th Python in Science Conf. 2010.

Bibliography

- I. Goodfellow, Y. Bengio and A. Courville, (2016). “Deep Learning”. MIT Press (Section 6.5).
- D. Mackay, (2003). “ Information Theory, Inference and Learning Algorithms”. Cambridge University Press (Chapters 39 and 44).