Name:

CMSC 838B & 498Z: Differentiable Programming

Tues/Thur 12:30pm – 1:45pm

http://www.cs.umd.edu/class/fall2021/cmsc838b

Ming C. Lin

IRB 5162

lin@cs.umd.edu

http://www.cs.umd.edu/~lin

Office Hours: After Class or By Appointment

M. C. Lin

1

Backpropagation

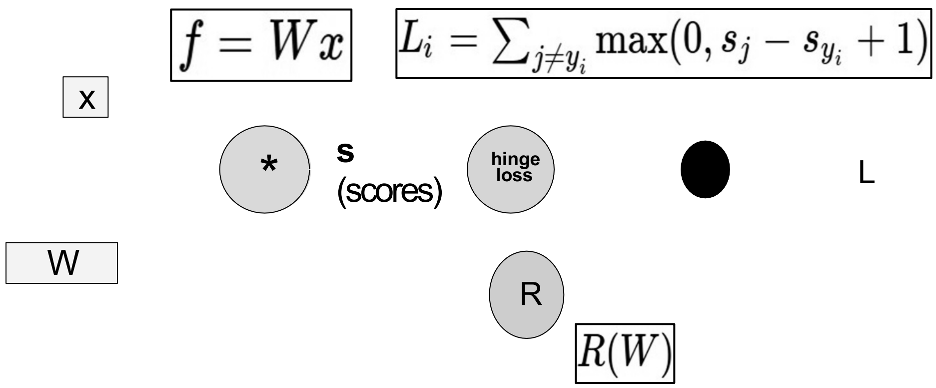
- Widely used for training feed-forward neural networks and generalized for ANNs & functions
- It computes the gradient of the loss function w.r.t. the weights of the network for a single I/O example and does it very efficiently
- Its efficiency makes it possible for training multilayer networks & updating weights to minimize losses
- Computing the gradient of the loss function w.r.t. each weight by the chain rule, computing the gradient one layer at a time, iterating backward from the last layer to avoid redundant calculations of intermediate terms in the chain rule

M. C. Lin

2

Name:

Computational Graphs



M. C. Lin

3

Computational Graphs

- Every operation in the computational graph given its inputs can immediately compute two things:
  - its output value
  - the *local* gradient of its inputs w.r.t its output value

- The chain rule tells us literally that each operation should take its local gradients and multiply them by the gradient that *flows* backwards into it

  → *This is backpropagation!!!*

M. C. Lin

4

Name:

Unintuitive Effects of Backprop: Multiplication

- Consider multiplication op: $f(a, b) = a \times b$
- The gradients are clearly $\partial f/\partial b = a$ and $\partial f/\partial a = b$.
  - in a computational graph these would be local gradients w.r.t inputs
- If a is large and b is tiny, then gradient assigned to b will be large, and the gradient to a would be small
- This has implications: e.g. linear classifiers ($w^T x_i$) where you perform many multiplications
  - the magnitude of the gradient is directly proportional to the magnitude of the data
  - multiply x_i by 1000, and the gradients also increase by 1000
  - if you don't lower the learning rate to compensate your model might not learn
- ***Need to always pay attention to data normalization***

M. C. Lin

5

Unintuitive Effects of Backprop: *vanishing gradients of the sigmoid*

- Popular to use sigmoids (or tanh) in hidden layers...

Gradient of $\sigma(x) = \sigma(x)(1 - \sigma(x))$

- As part of a larger network where this is local gradient, if x is large (+ve or -ve), then all gradients backwards from this point will be zero due to multiplication of chain rule
  - Why might x be large?
- Maximum gradient is achieved when $x = 0$ ($\sigma(x) = 0.5$, dx =0.25). i.e. the maximum gradient that can flow out of a sigmoid will be a quarter of input gradient
  - What's the implication of this in a deep network with sigmoid activations?

M. C. Lin

6

Name:

Working Examples

See the attached worksheet

M. C. Lin

7