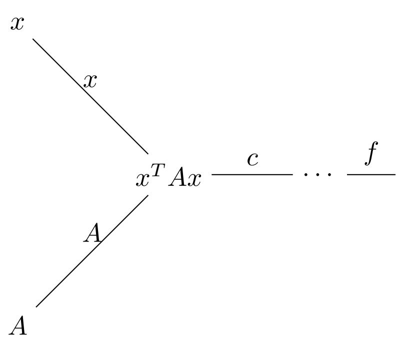
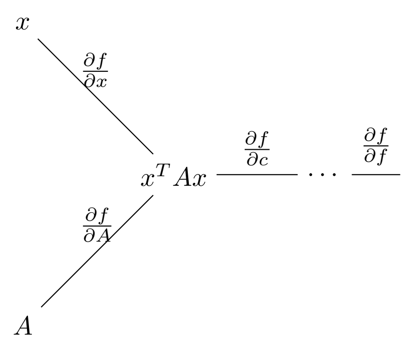
CS 188
Summer 2023

Discussion 7B Solutions

1 Vectorized Gradients

Let's compute the backward step for a node that computes $x^T Ax$, where x is a vector with m values, and A is a matrix with shape $m \times m$. Thus, $c = \sum_{i=1}^{m} x_i \sum_{j=1}^{m} A_{ij} x_j = \sum_{i=1}^{m} \sum_{j=1}^{m} A_{ij} x_i x_j = \sum_{j=1}^{m} x_j \sum_{i=1}^{m} A_{ij} x_i$.





1. What is $\frac{\partial f}{\partial A_{ij}}$?

   $\frac{\partial f}{\partial A_{ij}} = \frac{\partial f}{\partial c} \frac{\partial c}{\partial A_{ij}} = \frac{\partial f}{\partial c} x_i x_j$

2. What is $\frac{\partial f}{\partial A}$?

   $\frac{\partial f}{\partial A} = \frac{\partial f}{\partial c} x x^T$

3. What is $\frac{\partial f}{\partial x_k}$?

   Use the Product Rule:

$$\frac{\partial f}{\partial x_k} = \frac{\partial f}{\partial c} \frac{\partial c}{\partial x_k} = \frac{\partial f}{\partial c}\left(\frac{d}{dx_k} \sum_{i=1}^{m} \sum_{j=1}^{m} A_{ij} x_i x_j\right) = \frac{\partial f}{\partial c}\left(\left(\frac{d}{dx_k} x_k\right) \sum_{j=1}^{m} A_{kj} x_j + \left(\frac{d}{dx_k} x_k\right) \sum_{i=1}^{m} A_{ik} x_i\right)$$

$$\frac{\partial f}{\partial x_k} = \frac{\partial f}{\partial c}\left(\sum_{j=1}^{m} A_{kj} x_j + \sum_{i=1}^{m} A^T_{ki} x_i\right) = \frac{\partial f}{\partial c}\left(\sum_{j=1}^{m} (A_{kj} + A^T_{kj}) x_j\right)$$

4. What is $\frac{\partial f}{\partial x}$?

   $\frac{\partial f}{\partial x} = \frac{\partial f}{\partial c}(A + A^T)x$

2 Neural Networks Potpourri

(a) Which of these are likely to increase during overfitting?

- [ ] Training error
- [x] Validation/held-out error
- [x] Test error

(b) Which of these techniques may be used to prevent overfitting?

- [x] Early stopping
- [x] L1/L2 regularization
- [x] Dropout
- [x] Laplace smoothing
- [x] Model interpolation
- [x] Using additional training data

(c) Which of these activation functions are differentiable everywhere?

- [x] Sigmoid
- [x] Tanh
- [ ] ReLU

(d) Which of these may speed up the convergence of mini-batch gradient descent (in terms of wall-clock time)?

- [x] Increasing the learning rate
- [x] Increasing the batch size
- [x] Decreasing the learning rate
- [ ] Decreasing the batch size

(e) Which of these methods updates the model parameters least often (in terms of wall-clock time)?

- ● Batch gradient descent
- ○ Stochastic gradient descent
- ○ Mini-batch gradient descent

(f) Which of these statements is true?

- [x] Momentum can speed up the convergence of vanilla SGD.
- [ ] Using momentum requires storing a list of previously computed gradients.
- [ ] Learning rates should typically increase over the timecourse of training.
- [x] Parameters can each have their own learning rate.

(g) Which of these optimization approaches requires computing second derivatives?

- [ ] Stochastic gradient descent
- [ ] Nesterov accelerated gradient
- [ ] Adam
- [x] Newton's method