Vectorized Gradients

Let’s compute the backward step for a node that computes $x^T A x$, 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 $\frac{\partial f}{\partial A}$?
   $\frac{\partial f}{\partial A} = \frac{\partial f}{\partial c} xx^T$

3. What is $\frac{\partial f}{\partial x}$?
   Use the Product Rule:
   $$\frac{\partial f}{\partial x_k} = \frac{\partial f}{\partial c} \left( \sum_{j=1}^{m} A_{kj} x_{j} + \sum_{i=1}^{m} A_{ik} x_{i} \right) = \frac{\partial f}{\partial c} (A_{kj} x_{j} + A_{ik} x_{i})$$

4. What is $\frac{\partial f}{\partial c}$?
   $\frac{\partial f}{\partial c} = \frac{\partial f}{\partial c} (A + A^T)x$
2 Neural Networks Potpourri

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

- Training error
- Validation/held-out error
- Test error

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

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

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

- Sigmoid
- Tanh
- ReLU

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

- Increasing the learning rate
- Increasing the batch size
- 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?

- 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.
- 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
- Newton’s method