Second-Order Optimization Methods
Spring 2020
Today

• Rest of google cloud setup
• Brief overview of
  • Second order optimization
  • Batch normalization
  • CNN visualization
• Reading research papers
• Tensorflow practice: visualizing hw1 models
Second Order Optimization: Key insight

Leverage second-order derivatives (gradient) in addition to first-order derivatives to converge faster to minima
Newton’s method for convex functions

• Iterative update of model parameters like gradient descent

• Key update step

\[ x^{k+1} = x^k - H(x^k)^{-1} \nabla f(x^k) \]

• Compare with gradient descent

\[ x^{k+1} = x^k - \eta^k \nabla f(x^k) \]
In two steps

• Function of single variable
• Function of multiple variables
Derivative at minima

\( f(x) \)

\( \frac{dy}{dx} = 0 \)
Turning Points

• Both *maxima* and *minima* have zero derivative
• Both are turning points
Derivatives of a curve

- Both *maxima* and *minima* are turning points.
- Both *maxima* and *minima* have zero derivative.
• The second derivative $f''(x)$ is –ve at maxima and +ve at minima
Summary

- All locations with zero derivative are critical points
- The second derivative is
  - $\geq 0$ at minima
  - $\leq 0$ at maxima
  - Zero at inflection points
In two steps

• Function of single variable
• Function of multiple variables
Gradient of function with multi-variate inputs

- Consider $f(X) = f(x_1, x_2, ..., x_n)$

- $\nabla f(X) = \begin{bmatrix} \frac{\partial f(X)}{\partial x_1} & \frac{\partial f(X)}{\partial x_2} & \cdots & \frac{\partial f(X)}{\partial x_n} \end{bmatrix}$

Note: Scalar function of multiple variables
The Hessian

• The Hessian of a function $f(x_1, x_2, \ldots, x_n)$:

$$\nabla^2 f(x_1, \ldots, x_n) := 
\begin{bmatrix}
\frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\
\frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\
\vdots & \ddots & \ddots & \ddots \\
\frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2}
\end{bmatrix}$$
Unconstrained minimization of multivariate function

1. Solve for the $X$ where the gradient equation equals to zero

$$\nabla f(X) = 0$$

2. Compute the Hessian Matrix $\nabla^2 f(X)$ at the candidate solution and verify that
   - Hessian is positive definite (eigenvalues positive) $\rightarrow$ to identify local minima
   - Hessian is negative definite (eigenvalues negative) $\rightarrow$ to identify local maxima
Catch

• Closed form solutions not always available

• Instead use an iterative refinement approach
  • (Stochastic) gradient descent makes use of first-order derivatives (gradient)
  • Can we do better with second-order derivatives (Hessian)?
Newton’s method for convex functions

- Iterative update of model parameters like gradient descent

- Key update step

\[ x^{k+1} = x^k - H(x^k)^{-1} \nabla f(x^k) \]

- Compare with gradient descent

\[ x^{k+1} = x^k - \eta^k \nabla f(x^k) \]
Taylor series

The Taylor series of a function $f(x)$ that is infinitely differentiable at the point $a$ is the power series

\[
\begin{align*}
  f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots.
\end{align*}
\]
Taylor series second-order approximation

The Taylor series second-order approximation of a function $f(x)$ that is infinitely differentiable at the point $a$ is

$$f(a) + f'(a)(x - a) + \frac{1}{2} f''(a)(x - a)^2$$
Local minimum of Taylor series second-order approximation

\[ f(a) + f'(a)(x - a) + \frac{1}{2} f''(a)(x - a)^2 \]

\[ x_m = a - \frac{1}{f''(a)} f'(a) \text{ if } f''(a) > 0 \]
Newton’s method approach

Take step to local minima of second-order Taylor approximation of loss function
Example

Murphy, Machine Learning, Fig 8.4
Taylor series second-order approximation for multivariate function

\[ f(a) + \nabla f(a)(x - a) + \frac{1}{2} \nabla f^2(a)(x - a)^2 \]

\[ f(x^k) + \nabla f(x^k) + \frac{1}{2} H(x^k)(x - x^k)^2 \]
Deriving update rule

Local minima of this function

\[ f(x^k) + \nabla f(x^k) + \frac{1}{2} H(x^k)(x - x^k)^2 \]

is

\[ x = x^k - H(x^k)^{-1} \nabla f(x^k) \]
Weakness of Newton’s method (1)

- Appropriate when function is strictly convex
  - Hessian always positive definite

![Diagram showing Newton's method iterations](Murphy, Machine Learning, Fig 8.4)
Weakness of Newton’s method (2)

• Computing inverse Hessian explicitly is too expensive
  • $O(k^3)$ if there are $k$ model parameters: inverting a $k \times k$ matrix
Quasi-Newton methods address weakness

- Iteratively build up approximation to the Hessian

- Popular method for training deep networks
  - Limited memory BFGS (L-BFGS)
  - Will discuss in a later lecture
Acknowledgment

Based in part on material from
• CMU 11-785
• Spring 2019 course
Example

• Minimize

\[ f(x_1, x_2, x_3) = (x_1)^2 + x_1(1 - x_2) - (x_2)^2 - x_2x_3 + (x_3)^2 + x_3 \]

• Gradient

\[ \nabla f = \begin{bmatrix} 2x_1 + 1 - x_2 \\ -x_1 + 2x_2 - x_3 \\ -x_2 + 2x_3 + 1 \end{bmatrix}^T \]
Example

• Set the gradient to null

\[ \nabla f = 0 \Rightarrow \begin{bmatrix} 2x_1 + 1 - x_2 \\ -x_1 + 2x_2 - x_3 \\ -x_2 + 2x_3 + 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \]

• Solving the 3 equations system with 3 unknowns

\[ x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \]
Example

• Compute the Hessian matrix  \( \nabla^2 f = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} \)

• Evaluate the eigenvalues of the Hessian matrix
  \( \lambda_1 = 3.414, \ \lambda_2 = 0.586, \ \lambda_3 = 2 \)

• All the eigenvalues are positive \( \Rightarrow \) the Hessian matrix is positive definite

• This point is a minimum
  \[ x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \]