Neural networks

Training neural networks - optimization
OPTIMIZATION

Topics: local optimum, global optimum, plateau

• Notes on the optimization problem
  ‣ there isn’t a single global optimum (non-convex optimization)
    - we can permute the hidden units (with their connections) and get the same function
    - we say that the hidden unit parameters are not identifiable

  ‣ Optimization can get stuck in local minimum or plateaus
Topics: local optimum, global optimum, plateau

Neural network training demo
(by Andrej Karpathy)

http://cs.stanford.edu/~karpathy/svmjs/demo/demonn.html
GRADIENT DESCENT

**Topics:** convergence conditions, decrease constant

- Stochastic gradient descent will converge if
  \[ \sum_{t=1}^{\infty} \alpha_t = \infty \]
  \[ \sum_{t=1}^{\infty} \alpha_t^2 < \infty \]
  where \( \alpha_t \) is the learning rate of the \( t^{\text{th}} \) update

- Decreasing strategies: (\( \delta \) is the decrease constant)
  \[ \alpha_t = \frac{\alpha}{1+\delta t} \]
  \[ \alpha_t = \frac{\alpha}{t^\delta} \quad (0.5 < \delta \leq 1) \]

- Better to use a fixed learning rate for the first few updates
GRADIENT DESCENT

Topics: mini-batch, momentum

• Can update based on a mini-batch of example (instead of 1 example):
  ‣ the gradient is the average regularized loss for that mini-batch
  ‣ can give a more accurate estimate of the risk gradient
  ‣ can leverage matrix/matrix operations, which are more efficient

• Can use an exponential average of previous gradients:

\[ \overline{\nabla_\theta}^{(t)} = \nabla_\theta l(f(x^{(t)}), y^{(t)}) + \beta \overline{\nabla_\theta}^{(t-1)} \]

  ‣ can get through plateaus more quickly, by “gaining momentum”
**Topics:** Newton’s method

- If we locally approximate the loss through Taylor expansion:

\[
l(f(x; \theta), y) \approx l(f(x; \theta^{(t)}), y) + \nabla_{\theta} l(f(x; \theta^{(t)}), y)^\top (\theta - \theta^{(t)}) \\
+ 0.5(\theta - \theta^{(t)})^\top \left( \nabla_{\theta}^2 l(f(x; \theta^{(t)}), y) \right) (\theta - \theta^{(t)})
\]

- We could minimize that approximation, by solving:

\[
0 = \nabla_{\theta} l(f(x; \theta^{(t)}), y) + \left( \nabla_{\theta}^2 l(f(x; \theta^{(t)}), y) \right) (\theta - \theta^{(t)})
\]
Topics: Newton’s method

• We can show that the minimum is:

\[ \theta^{(t+1)} = \theta^{(t)} - \left( \nabla^2_\theta l(f(x; \theta^{(t)}), y) \right)^{-1} \left( \nabla_\theta l(f(x; \theta^{(t)}), y) \right) \]

• Only practical if:
  ‣ few parameters (so we can invert Hessian)
  ‣ locally convex (so the Hessian is invertible)

• See recommended readings for more on optimization of neural networks