Machine Learning Basics: Stochastic Gradient Descent

Sargur N. Srihari
srihari@cedar.buffalo.edu

Topics

1. Learning Algorithms
2. Capacity, Overfitting and Underfitting
3. Hyperparameters and Validation Sets
4. Estimators, Bias and Variance
5. Maximum Likelihood Estimation
6. Bayesian Statistics
7. Supervised Learning Algorithms
8. Unsupervised Learning Algorithms
9. Stochastic Gradient Descent
10. Building a Machine Learning Algorithm
11. Challenges Motivating Deep Learning

Stochastic Gradient Descent (SGD)

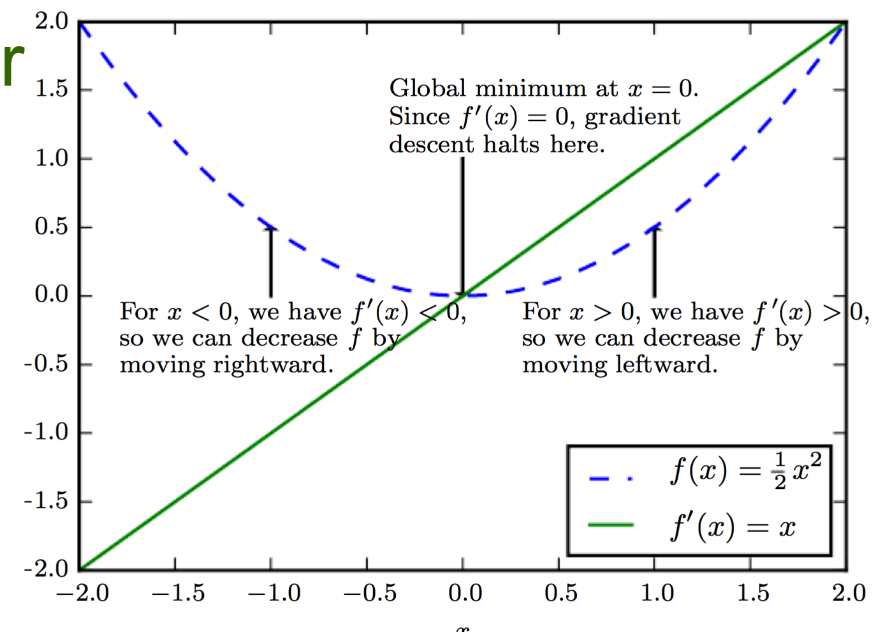
- Nearly all deep learning is powered by one very important algorithm: SGD
- SGD is an extension of the gradient descent algorithm
- A recurring problem in machine learning:
  - large training sets are necessary for good generalization
  - but large training sets are also computationally expensive

Method of Gradient Descent

- Criterion $f(\boldsymbol{x})$ is minimized by moving from the current solution in direction of the negative of gradient
- Steepest descent proposes a new point

$$\boldsymbol{x}' = \boldsymbol{x} - \varepsilon \nabla_{\boldsymbol{x}} f(\boldsymbol{x})$$

  – where ε is the learning r to a small constant.



Cost function is sum over samples

- Criterion in machine learning is a cost function
- Cost function often decomposes as a sum of per sample loss function
  - E.g., Negative conditional log-likelihood of training data is

$$\boxed{J(\theta) = E_{\boldsymbol{x},y\sim\hat{p}_{\text{data}}} L(\boldsymbol{x},y,\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} L\left(\boldsymbol{x}^{(i)},y^{(i)},\boldsymbol{\theta}\right)}$$

where m is the no. of samples and

L is the per-example loss $L(\boldsymbol{x},y,\boldsymbol{\theta}) = -\log p(y|\boldsymbol{x};\boldsymbol{\theta})$

Gradient is also sum over samples

- For these additive cost functions, gradient descent requires computing

$$\nabla_\theta J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \nabla_\theta L\left(\boldsymbol{x}^{(i)}, y^{(i)}, \boldsymbol{\theta}\right)$$

- Computational cost of this operation is $O(m)$
- As training set size grows to billions, time taken for single gradient step becomes prohibitively long

Insight of SGD

- Insight: Gradient descent is an expectation
  - Expectation may be approximated using small set of samples
- In each step of SGD we can sample a *minibatch* of examples $B = \{x^{(1)},..,x^{(m')}\}$
  - drawn uniformly from the training set
  - Minibatch size m' is typically chosen to be small: 1 to a hundred
    - Crucially m' is held fixed even if sample set is in billions
    - We may fit a training set with billions of examples using updates computed on only a hundred examples

SGD Estimate on minibatch

- Estimate of gradient is formed as

$$\boldsymbol{g} = \frac{1}{m'} \nabla_\theta \sum_{i=1}^{m'} L\left(\boldsymbol{x}^{(i)}, y^{(i)}, \boldsymbol{\theta}\right)$$

  – using examples from minibatch B

- SGD then follows the estimated gradient downhill

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \varepsilon \boldsymbol{g}$$

  – where ε is the learning rate

How good is SGD?

- In the past gradient descent was regarded as slow and unreliable
- Application of gradient descent to non-convex optimization problems was regarded as unprincipled
- SGD is not guaranteed to arrive at even a local minumum in reasonable time
- But it often finds a very low value of the cost function quickly enough

SGD and Training Set Size

- SGD is the main way to train large linear models on very large data sets
- For a fixed model size, the cost per SGD update does not depend on the training set size m
- As $m \rightarrow \infty$ model will eventually converge to its best possible test error before SGD has sampled every example in the training set
- Asymptotic cost of training a model with SGD is $O(1)$ as a function of m

Deep Learning vs SVM

- Prior to advent of DL main way to learn nonlinear models was to use the kernel trick in combination with a linear model
  - Many kernel learning algorithms require constructing an m x m matrix $G_{i,j}=k(x^{(i)},x^{(j)})$
    - Constructing this matrix is $O(m^2)$
- Growth of Interest in Deep Learning
  - In the beginning because it performed better on medium sized data sets (thousands of examples)
  - Additional interest in industry because it provided a scalable way of training nonlinear models on large datasets