Deep Learning através das abstrações

Treinando Redes Neurais

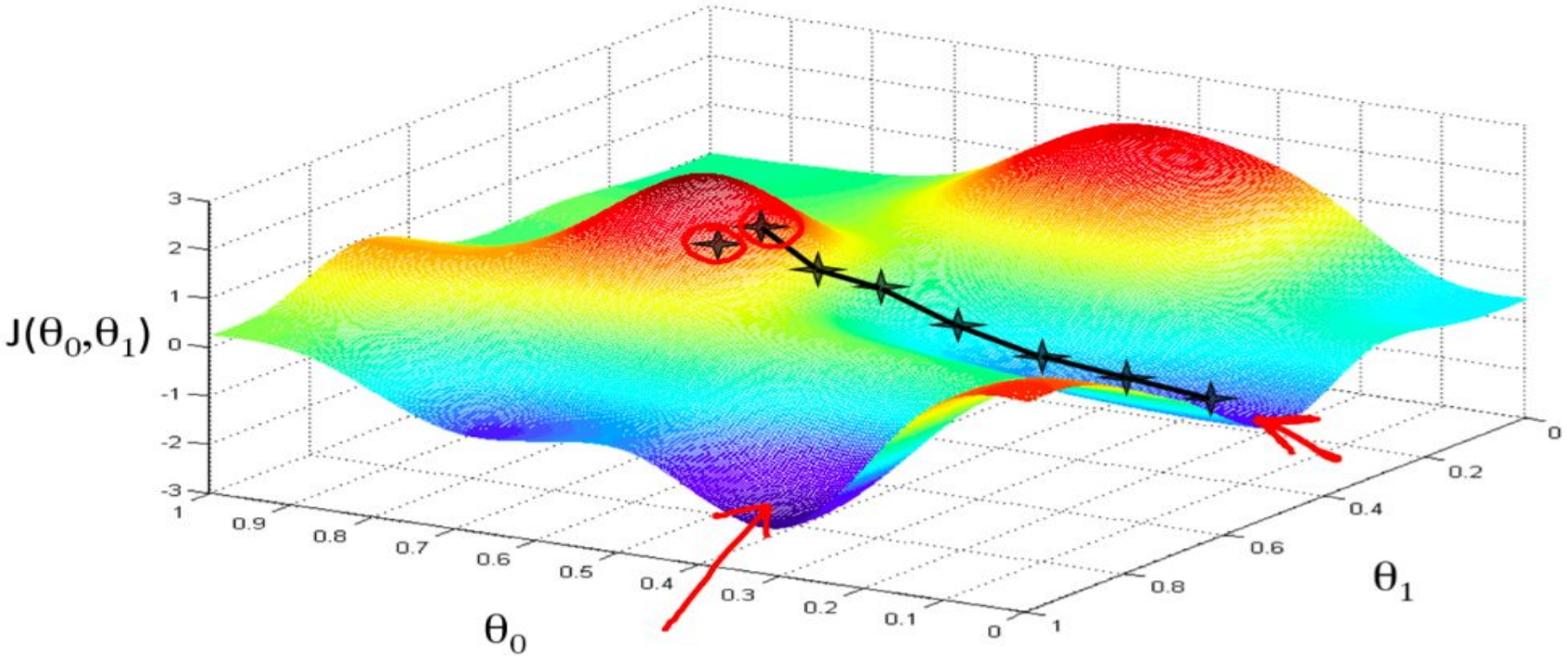
Recapitulando...

- Camadas são conjuntos de perceptrons
- Conjunto de camadas forma uma rede neural
- Podemos tratar o treinamento de uma rede neural como um problema de otimização .

Mas o que realmente garatimos com isso?

- Mínimos locais
- Conjunto de dados limitados
- Overfitting
- Conjunto de treino, de teste e de validação

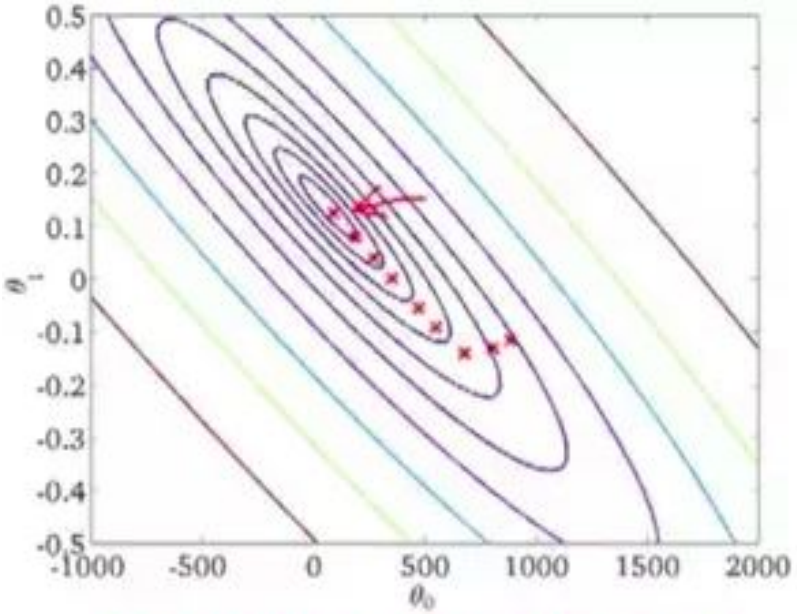
Descida de Gradiente



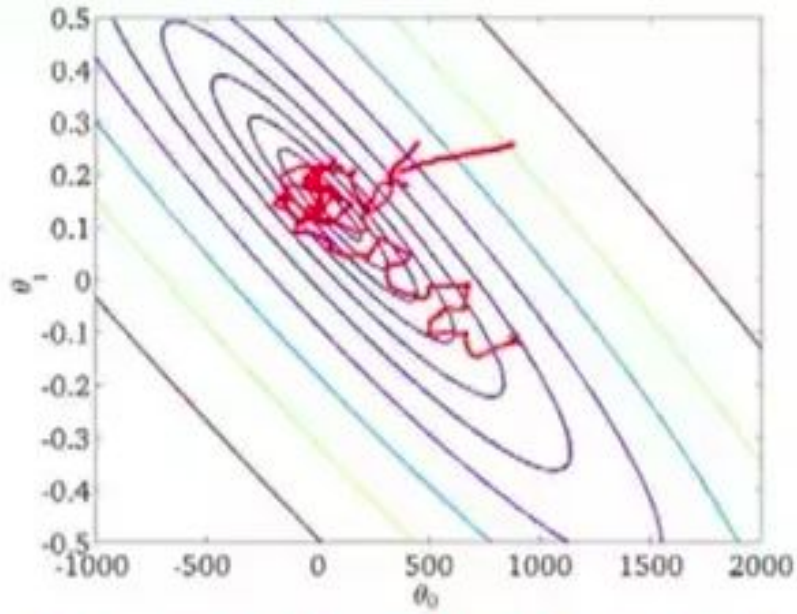
Batch Gradient Descent

- Método clássico
- Funciona para espacos até infinito-dimensionais
- Fica mais lento a medida que se aproxima do mínimo
- Usa todo o training set para treinar
- Demora muito para computar cada passo

Stochastic Gradient Descent



Batch Gradient Descent

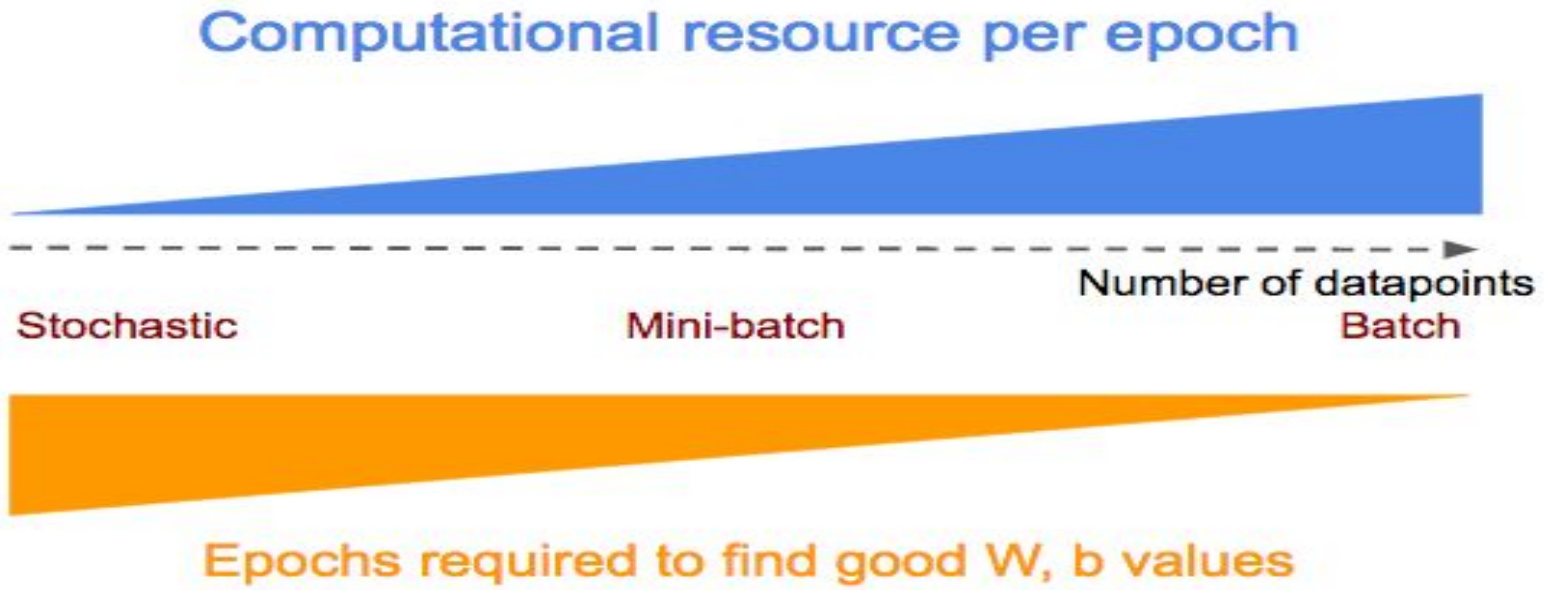


Stochastic Gradient Descent

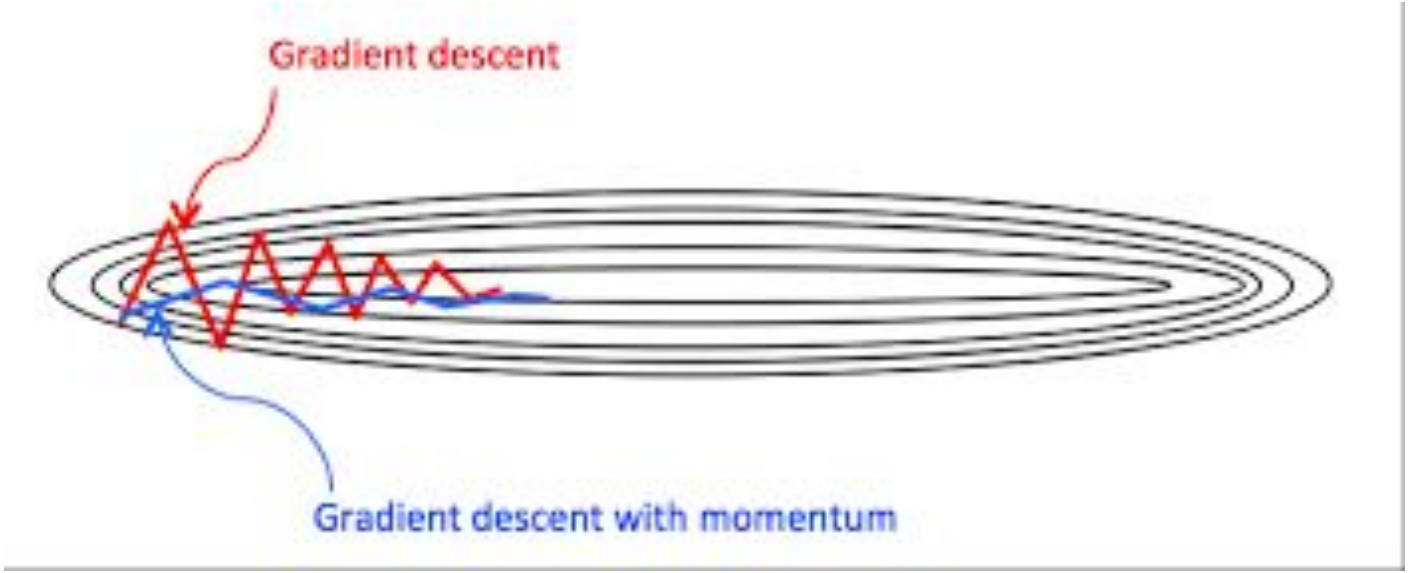
Stochastic Gradient Descent

- Versão normal usa um exemplo por vez
- Geralmente implementa-se com mini-batches
- Computa cada passo bem mais rápido
- Mais "barulhento"
- Base para os métodos modernos

Stochastic Gradient Descent



Momentum



Momentum

- $w = w + \alpha \Delta w - \eta \nabla L$
- Evita mudancas bruscas de direcao
- Maior estabilidade
- Melhor convergência
- Explicacao teórica

Outros métodos modernos

- AdaGrad
- AdaDelta
- RMSProp
- Adam
- Natural Gradient
- Site do Sebastian Ruder

Regularização

- Overfitting
- Bia-Variance Tradeoff
- Limitam nosso espaço de funções possíveis
- Modificam a função de erro

Regularização L1

- Gera soluções esparsas
- Robusta a outliers
- Gera modelos simples e interpretáveis, porém muitas vezes não consegue aprender padrões complexos

$$L(x,y) \equiv \sum_{i=1}^{n} (y_i - h_\theta(x_i))^2 + \lambda \sum_{i=1}^{n} |\theta_i|$$

Regularização L2

- Gera predições melhores quando a saída é uma função de todas as features
- Não é robusta contra outliers
- Consegue aprender padrões complexos

$$L(x, y) \equiv \sum_{i=1}^{n} (y_i - h_\theta(x_i))^2 + \lambda \sum_{i=1}^{n} \theta_i^2$$