MACHINE LEARNING PART 31

# Optimization Algorithms

## in machine learning

Optimization algorithms play a crucial role in training machine learning models by iteratively **updating the model parameters to minimize a certain objective function**, often referred to as the loss or cost function.

Here are some key optimization algorithms used in machine learning:

# Gradient Descent

**Basic Idea:**

- *Update model parameters in the opposite direction of the gradient of the loss function with respect to the parameters.*

**Variants:**

- *Stochastic Gradient Descent (SGD): Updates parameters using a single random data point at a time.*
- *Mini-Batch Gradient Descent: Updates parameters using a small batch of randomly selected data points.*

# Momentum

**Idea:**

- *Introduces a momentum term to prevent oscillations and speed up convergence.*

**Update Rule:**

- $v=\beta \cdot v-\alpha \cdot \nabla J(\theta)$
- $\theta=\theta+v$

**Hyperparameter $\beta$:**

- *Controls the contribution of the previous update to the current update.*

# Adagrad

**Idea:**

- *Adjusts the learning rates of each parameter individually based on the historical gradient information.*

**Update Rule:**

- $\theta = \theta - (\alpha / root(G + \epsilon)) \cdot \nabla J(\theta)$

**Accumulated Squared Gradients G:**

- *Keeps track of the sum of the squares of the gradients.*

# RMSprop

**Idea:**

- *Addresses the diminishing learning rates problem of Adagrad by using a moving average of squared gradients.*

**Update Rule:**

- $\theta = \theta - (\alpha / root(E[G] + \epsilon)) \cdot \nabla J(\theta)$

**Exponential Moving Average E[G]:**

  - *A moving average of squared gradients.*

# Adam

**Combines Momentum and RMSprop:**

- *Utilizes both momentum-based updates and adaptive learning rates.*

**Update Rule:**

- $m=\beta 1 \cdot m+(1-\beta 1) \cdot \nabla J(\theta)$
- $v=\beta 2 \cdot v+(1-\beta 2) \cdot (\nabla J(\theta))2$
- $\theta=\theta-(\alpha/root(v+\epsilon)) \cdot m$

**Hyperparameters β1 and β2:**

  - *Control the decay rates of the moment estimates.*

# AdaDelta

**Idea:**

- *Addresses the accumulation of squared gradients issue in RMSprop by using a more sophisticated update rule.*

**Update Rule:**

- $\theta=\theta - (root(E[G]+\epsilon)/ root(E[\Delta\theta]+\epsilon)) \cdot \nabla J(\theta)$

**Exponential Moving Average of Squared Parameter Updates E[Δθ]:**

- *A moving average of squared parameter updates.*

# Nadam

**Combines Nesterov Accelerated Gradient (NAG) with Adam:**

- *Incorporates NAG's momentum term into Adam.*

**Update Rule:**

- *Similar to Adam, but with Nesterov momentum.*

# Adapting Learning Rate Schedules

**Learning Rate Schedulers:**

- *Techniques that dynamically adjust the learning rate during training.*

**Examples:**

- *Step Decay: Reduce the learning rate after a fixed number of epochs.*
- *Exponential Decay: Reduce the learning rate exponentially over time.*