AdaGrad

AdaGrad (Adaptive Gradient Algorithm) is an optimization algorithm used in machine learning to address the issue of diminishing gradients, particularly in problems with sparse features. It aims to improve the convergence speed and performance compared to standard gradient descent.

How AdaGrad Works:

1. **Initialization:**
    - Initialize the learning rate (`lr`) and an accumulator (`h`) for each parameter (weight) in the model. The accumulator is typically filled with zeros.

2. **Gradient Calculation:**
    - During each training step, calculate the gradient of the loss function with respect to each parameter.

3. **Accumulator Update:**
    - For each parameter:
        - Square the current gradient.
        - Add this squared gradient value to the corresponding accumulator (`h`).
        - Apply a square root operation (element-wise) to the accumulator to avoid exploding values.
        - Optionally, add a small epsilon value (`ε`) to the square root to prevent division by zero.

$$H_t = H_(t-1) + g_t^2$$

4. **Parameter Update:**
    - Divide the learning rate by the square root of the updated accumulator (element-wise).
    - Multiply this scaled learning rate with the current gradient to get the update value.
    - Subtract the update value from the current parameter to update the weight.

$$\theta_t = \theta_(t - 1) - lr/(sqrt(H_t + \varepsilon)) * g_t$$

Benefits of AdaGrad:

- **Adapts to Sparse Features:** By accumulating squared gradients, AdaGrad gives more weight to updates for parameters with historically small gradients (often corresponding to sparse features). This can help address the diminishing gradients issue in such cases.
- **Faster Convergence:** In problems with sparse features, AdaGrad can often converge faster than standard gradient descent.

Drawbacks of AdaGrad:

- **Accumulating Gradients:** The accumulator keeps growing over time, which can eventually lead to learning rate decay for all parameters, potentially hindering further improvements.
- **Hyperparameter Tuning:** Setting the learning rate and the optional epsilon value can be crucial for optimal performance.