Nestors Accelerated Gradient Descent.

Nesterov Accelerated Gradient (NAG), which is commonly used to enhance training speed and stability in neural networks. Building on our previous discussion of momentum-based optimization, this article will explore how NAG works, its advantages, and how to implement it in deep learning libraries like Keras.

What is an Optimizer?

An **optimizer** is an algorithm used in machine learning to adjust a model's parameters, such as weights, in order to minimize the loss function and improve model accuracy. Optimization is crucial because it allows the model to learn and generalize better from data.

Common optimizers include **gradient descent variants** like:

1. **Batch Gradient Descent**: Uses all data points to compute the gradient and update weights. This can be accurate but slow.
2. **Stochastic Gradient Descent (SGD)**: Updates weights based on each individual data point, making it faster but less stable.
3. **Mini-Batch Gradient Descent**: Combines the best of both worlds by updating weights based on small, random batches of data points.

Nesterov's Accelerated Gradient (NAG) is a momentum-based optimization technique that improves upon standard gradient descent by using a "look-ahead" gradient, which calculates the gradient at a future position, leading to faster and smoother convergence. Here's a more detailed explanation:

- **What it is:**

NAG is a refinement of the standard momentum method, designed to further smooth and speed up convergence in optimization algorithms like training deep neural networks.

- **How it works:**
- **Look-ahead gradient:** NAG calculates the gradient at a future position (the position the parameters will be after the momentum update) instead of the current position, as in standard momentum.
- **Momentum:** NAG uses a momentum term to accumulate past gradients, helping the optimization process move more efficiently towards the minimum of the loss function.
- **Two Steps:** NAG consists of two steps: a gradient descent step, followed by a "momentum stage" that refines the update direction.
- **Why it's beneficial:**
- **Faster convergence:** The look-ahead gradient helps the optimization process to move more directly towards the optimal solution, leading to faster convergence.
- **Smoother convergence:** By using a momentum term and a look-ahead gradient, NAG can help to reduce oscillations and improve the stability of the optimization process.
- **In relation to other optimizers:**
- **Gradient Descent:** NAG is an improvement over standard gradient descent, which simply follows the negative gradient at the current position.
- **Momentum:** NAG builds upon the momentum method by using a look-ahead gradient, which further enhances the convergence speed and stability.

- **Adam:** Adam is another popular optimizer that combines momentum and adaptive learning rates, while NAG focuses on improving the momentum aspect.