# Day 78 Intro. to Optimizers By: Loga Aswin

## Optimizer in Machine Learning:

- Optimizers are algorithms used to minimize or maximize the loss function by updating the model's parameters during the training process.
- They play a crucial role in the efficiency and effectiveness of training neural networks.

## Gradient Descent:

Fundamental optimization algorithm used to minimize the loss function by adjusting parameters in the direction of steepest descent of the gradient.

Variants include:

- **Batch Gradient Descent**: Computes the gradient of the entire dataset.
- **Stochastic Gradient Descent (SGD):** Computes the gradient for each sample separately, updating the parameters after each iteration.
- **Mini-batch Gradient Descent**: Computes the gradient for a subset of the dataset, striking a balance between batch and stochastic approaches.

### Momentum Optimizer:

- Enhances SGD by adding a fraction of the previous update to the current update, smoothing out fluctuations in the optimization process.

- Helps accelerate convergence, especially in areas with high curvature.

**Adaptive Learning Rate Optimizers:**

**AdaGrad (Adaptive Gradient Algorithm):**

- Adjusts learning rates for each parameter based on the historical gradients.
- Scales down the learning rate for frequently occurring features, providing a smaller update for more frequently updated weights.

**RMSprop (Root Mean Square Propagation):**

- Addresses the diminishing learning rate problem in AdaGrad by using a moving average of squared gradients.
- Divides the learning rate by a running average of the magnitudes of recent gradients.

**AdaDelta:**

- Improves upon RMSprop by addressing its aggressive, monotonically decreasing learning rate.
- Eliminates the need for a learning rate hyperparameter by using a running average of parameter updates.

**Adam (Adaptive Moment Estimation):**

- Combines the advantages of both RMSprop and momentum optimization.
- Uses adaptive learning rates for each parameter and maintains momentum.

**Choosing an Optimizer:**

- The choice of optimizer depends on factors such as the nature of the problem, dataset size, computational resources, and network architecture.
- Experimentation is often necessary to determine which optimizer works best for a particular scenario.

**Hyperparameter Tuning:**

- Adjusting hyperparameters like learning rate, momentum, decay rates, etc., can significantly impact the performance of optimizers.
- Grid search or random search can help identify the optimal set of hyperparameters for a given problem.