**Optimizers**

An optimizer is a function or an algorithm that adjusts the attributes of the neural network, such as weights and learning rates.

While training the deep learning optimizers model, modify each epoch’s weights and minimize the loss function.

* **Epoch** – The number of times the algorithm runs on the whole training dataset.
* **Sample** – A single row of a dataset.
* **Batch** – It denotes the number of samples to be taken to for updating the model parameters.
* **Learning rate** – It is a parameter that provides the model a scale of how much model weights should be updated.
* **Cost Function/Loss Function** – A cost function is used to calculate the cost, which is the difference between the predicted value and the actual value.
* **Weights/ Bias** – The learnable parameters in a model that controls the signal between two neurons.

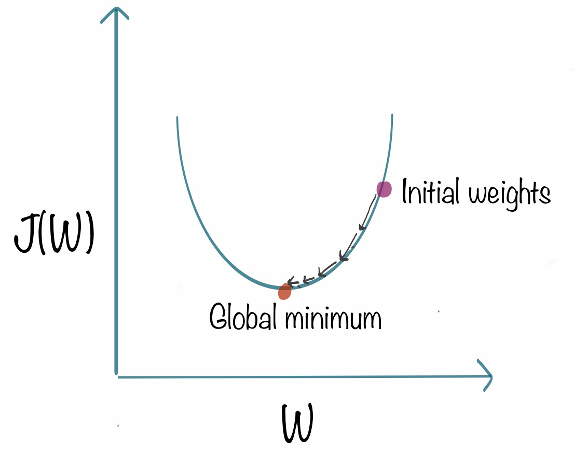
## Gradient Descent

This optimization algorithm uses calculus to modify the values consistently and to achieve the local minimum.

Gradient Descent Deep Learning Optimizer formula

Gradient descent works as follows:

1. It starts with some coefficients, sees their cost, and searches for cost value lesser than what it is now.
2. It moves towards the lower weight and updates the value of the coefficients.
3. The process repeats until the local minimum is reached. A local minimum is a point beyond which it can not proceed.



## Stochastic Gradient

The term stochastic means randomness on which the algorithm is based upon. In stochastic gradient descent, instead of taking the whole dataset for each iteration, we randomly select the batches of data. That means we only take a few samples from the dataset.

Stochastic Gradient Descent Deep Learning Optimizer formula

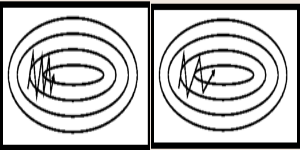
The procedure is first to select the initial parameters w and learning rate n. Then randomly shuffle the data at each iteration to reach an approximate minimum.

## Stochastic Gradient Descent With Momentum

Stochastic gradient descent takes a much more noisy path than the gradient descent algorithm.

Due to this reason, it requires a more significant number of iterations to reach the optimal minimum, and hence computation time is very slow.

To overcome the problem, we use stochastic gradient descent with a momentum algorithm.



In the above image, the left part shows the convergence graph of the stochastic gradient descent algorithm.

At the same time, the right side shows SGD with momentum. From the image, you can compare the path chosen by both algorithms and realize that using momentum helps reach convergence in less time.

## Mini Batch Gradient Descent

In this variant of gradient descent, instead of taking all the training data, only a subset of the dataset is used for calculating the loss function.

Since we are using a batch of data instead of taking the whole dataset, fewer iterations are needed.

That is why the mini-batch gradient descent algorithm is faster than both stochastic gradient descent and batch gradient descent algorithms.

## Adagrad (Adaptive Gradient Descent)

it uses different learning rates for each iteration.

The change in learning rate depends upon the difference in the parameters during training.

The more the parameters get changed, the more minor the learning rate changes.

The Adagrad algorithm uses the below formula to update the weights.

Adagrad (Adaptive Gradient Descent) Deep Learning Optimizer 1

Adagrad (Adaptive Gradient Descent) Deep Learning Optimizer 2

Here the alpha(t) denotes the different learning rates at each iteration, n is a constant, and E is a small positive to avoid division by 0.

### **Adam Optimizer**

Adaptive Moment Estimation combines the power of RMSProp (root-mean-square prop) and momentum-based GD.

In Adam optimizers, the power of momentum GD to hold the history of updates and the adaptive learning rate provided by RMSProp makes Adam optimizer a powerful method.

Adma Deep Learning Optimizer

The above formula represents the working of adam optimizer. Here B1 and B2 represent the decay rate of the average of the gradients.