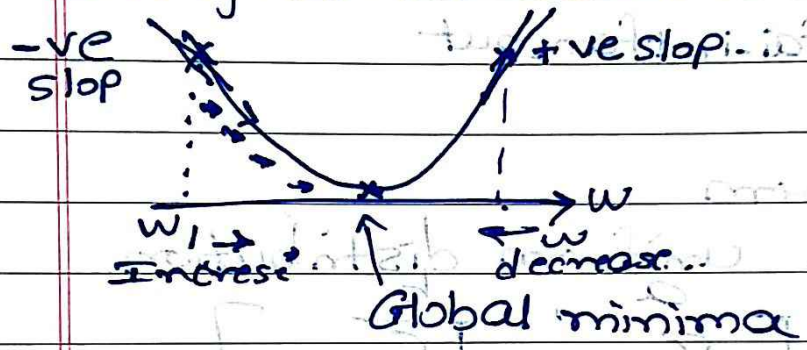
* Stochastic Gradient Descent

In backprpogation weight updation formula is

$$w_{new} = w_{old} - \eta \frac{\partial L}{\partial w_{old}} \quad \text{--- (1)}$$

In gradient descent



In gr

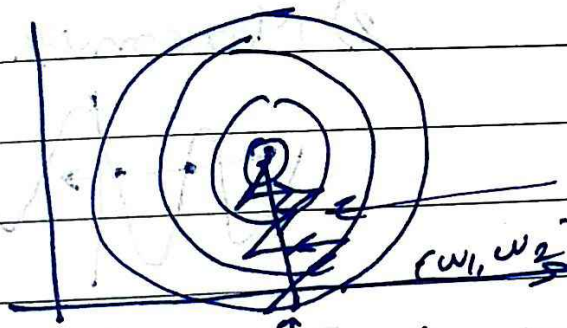
1) In equ^{n} (1) for weight updation consider all traing example for one epoch & update weight by backprpogation. This is a gradient descent algorithm.

2) Where as in stochastic gradient descent consider one training example in one epoch & update the weight.
This is SGD or minibatch SGD

In many ANN & CNN ~~minibath~~ minibatch SGD used.

Because when we have million of record so to load this much record it take lot computation time

In order to prevent this mini batch SGD used.



zig zag by minibatch SGD.

↑ Gradient Descent (But memory required time more)

↑ progress movement is noisy.

* So in mini batch SGD we use k datapoint as minibatch zigzag occurs so to avoide that momentum used.

$$L(w) = \sum_{i=1}^{K} (y - \hat{y})^2$$

K = Batch of training example

$$w_{new} = w_{old} - \eta \times \frac{\partial L}{\partial w_{old}}$$

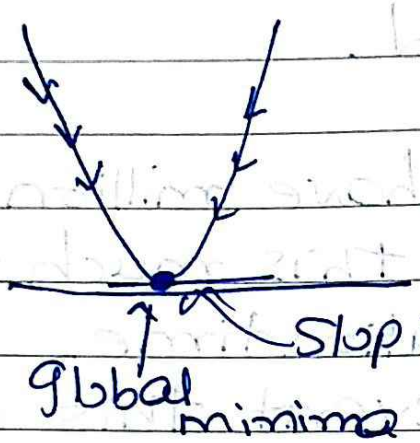
$$\left[\frac{\partial L}{\partial w_{old}}\right] \approx \left[\frac{\partial L}{\partial w_{old}}\right]_{GD}$$

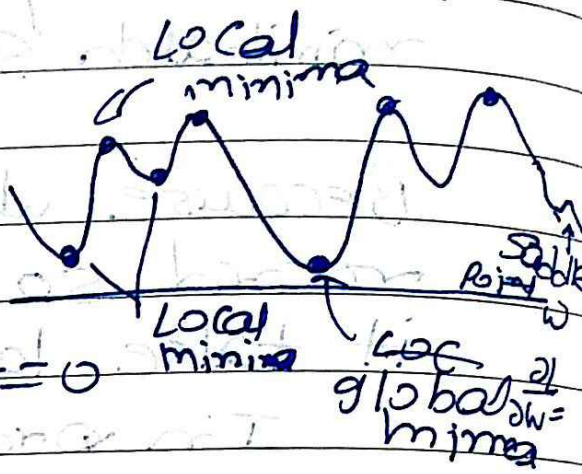
↑ population

(Mini Batch SGD → Sample)

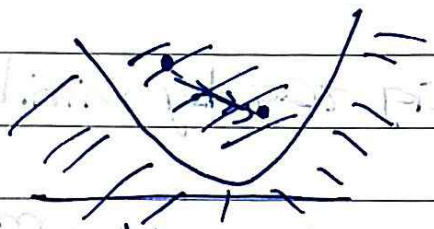
So to avoide computation time use SGD

Global minima Vs Local minima



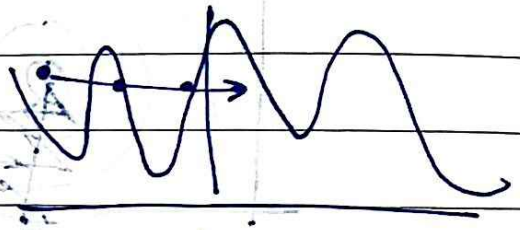


1) Convex function



Normachine learning.

2) Nonconvex function



Deep Learning

- In deep learning as so many layer & neuron are present due to which nonconvex function having local & global minima.

$$w_{new} = w_{old} - \eta * \frac{\partial L}{\partial w_{old}}$$

$$= w_{old} - \left[r v_{t-1} + \eta \frac{\partial L}{\partial w_{old}} \right]$$

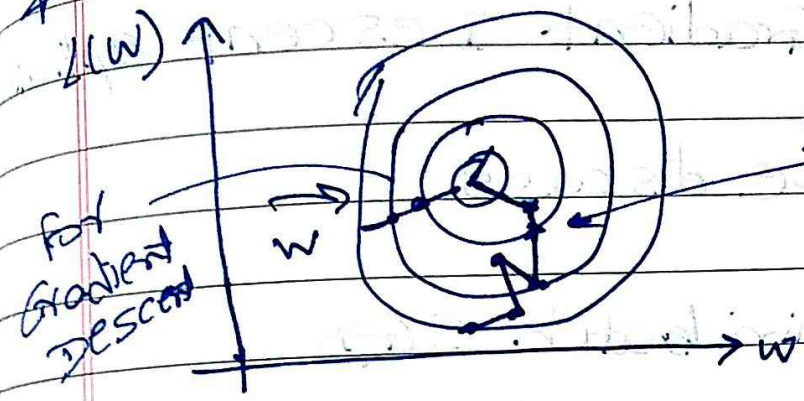
$$v_{t-1} = r \times \left[\frac{\partial L}{\partial old}\right]_t + r^2 \left[\frac{\partial L}{\partial old}\right]_{t-2}^{t} \cdots$$

Data point at different time interval.

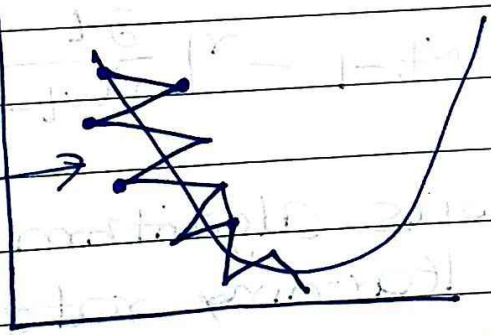
$r = 0 \to 1$

classmate
Date
Page

* SGD With Momentum



1) In minibatch SGD we take k data point (Some times noise gets added in data)



Due to Noise in SGD weight are updated in zig zag manner

Because this following problem occur.

1) For Convergence it take more time to reach to global minima because batch
2) For the same we need to reduce noise

So to rdce noise use

* Expontially moving Average
ie Momentum

* Adgard Optimizer *
(Adaptive Gradient Descent Optimizer

Previously we discuss:

1) GD, SGD, Minibatch SGD

2) Consider weight updation

$$w_{new} = w_{old} - \eta \frac{\partial L}{\partial w_{old}}$$

$$w_t = w_{t-1} - \eta \frac{\partial L}{\partial w_{t-1}}$$

3) In previous algorithm we use same weights & learning rate.
But in Adgard can we change different learning rate for different iteration number.

4) We consider different learning rate because of
1) Dense feature (&
2) Sparse feature ← BoW (many are ze
&some one)

1) In dense most of the value are of ~~feature~~
non zero

2) In sparse most of the value are of featur

are zero.
So in Adgard due to dense & sparse we use different learning rate using on different iteration number.

$$\therefore W_t = W_{t-1} - \eta_t \times \frac{\partial L}{\partial W_{t-1}}$$ t' No of iteration

$$\eta'_t = \frac{\eta}{\sqrt{\alpha_t + \varepsilon}}$$ η = intial learning rate.

ε — small the number.

$$\alpha_t = \sum_{i=1}^{t} \left(\frac{\partial L}{\partial W_{t-1}}\right)^2 \qquad \eta'_t = \frac{\eta}{\sqrt{\alpha_t + \varepsilon}}$$

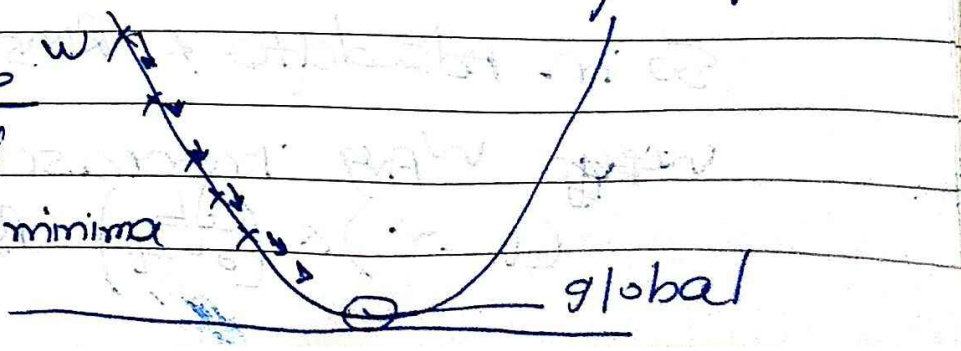
As α_t increases η'_t decreases as iteration goes on increasing, due to which weight decreasing slowly.

So adgard use different different learning rate by which we can update weight & optimize global minima

Limitation of Adgard ← As sometime iteration ↑es. α ↑es

Some time α_t become very high number

It will handle dense & sparse feature & converges for global minima



* Adadelta & RMSprop optimizer

In Adgard

$$w_t = w_{t-1} - \eta' \frac{\partial L}{\partial_{t-1}}$$

$$\eta'_t = \frac{\eta}{\sqrt{\alpha_t + \epsilon}} \qquad \alpha_t = \sum_{i=1}^{t} \left(\frac{\partial L}{\partial w_i}\right)^2$$

So here α_t is high due to which η'_t becomes very small & weight updation not happen.

So in Adadelta

$$\therefore \eta'_t = \frac{\eta}{\sqrt{w_{avg} + \epsilon}}$$

ϵ ← small +ve No. 0.05

weighted avg.

Exponential avg. $w_{Avg_t} = \gamma w_{Avg_{t-1}} + (1-\gamma) \times \left(\frac{\partial L}{\partial w_t}\right)^2$

$\gamma = 0.95$ selected

$$\alpha_t = \sum_{i=1}^{t} \left(\frac{\partial L}{\partial w_i}\right)^2 \leftarrow \text{Adgard}$$

So in Adadelta & RMS. we restricted w_{Avg} increase by adding $(1-\gamma) \times \left(\frac{\partial L}{\partial w_t}\right)$ By slecting $\gamma = 0.95$

$\therefore 1-\gamma = 0.05$

so by restricting this learning rate also changes slowly. & converges take less time.