16 July Day 3

23 November 2022 06:58

GD Disadvantages.

1. weights updation takes time.
2. computationally expensive
3. take time to convergence.

GD Advantages.

1. optimal solution is guaranteed.
2. smooth curvature.

Epoch

Whenever entire data set is being pass to the neural network is called epoch.

1000 → NN → $\hat{y}$ → Loss | output 1 Epoch

10 Epochs

1000 → NN → $\hat{y}$ → Loss [high]

⋮

1000 [10th] → NN → $\hat{y}$ → Loss [Low]

Iteration.

[1000] — batch size: 100 × 10 (no. of batches) = 1000

[bais] ↓ $w_{new} = w_{old} - \alpha \frac{\partial L}{\partial w_{old}}$

1st batch

[100] → NN → $\hat{y}$ → Loss ← [WB] Iteration 1

updated WB → NN → $\hat{y}$ → Loss ← [WB] Iteration 2

2nd batch [100]

Iteration 10

1000	1000	1000
1 Epoch	10 Epoch	Epoch 1
[1000]	1000 × [10] NN	Iteration 10 (→ 100, → 10)

Stochastic Gradient Descent

└ Random.

	x1	x2	x3	x4	Yact
2	*	*	*	*	*
	*	*	*	*	*
1	*	*	*	*	*
	*	*	*	*	*

GD Entire data set

SGD 1 Record at a time which is randomly picked.

1 Record → NN → $\hat{y}$ → Loss → update [WB]

updated WB 2nd Record → NN → $\hat{y}$ → Loss ← [WB]

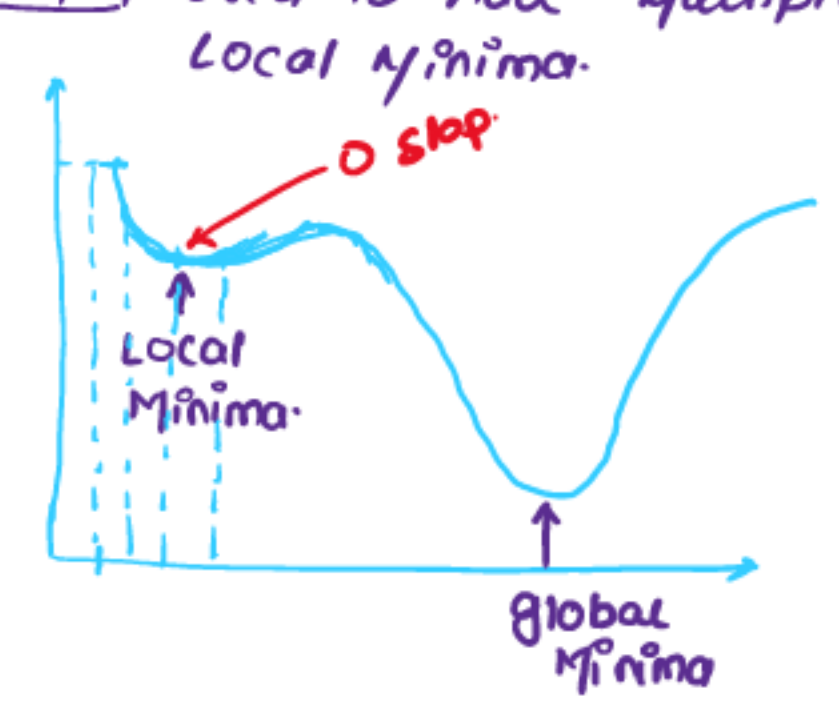
⋮

Entire Dataset

Epoch 1, Iteration 1000, WB = 1000 times.

×10 1000×10 1000×10

[SGD] used to move multiple Local Minima.



[slope 0]

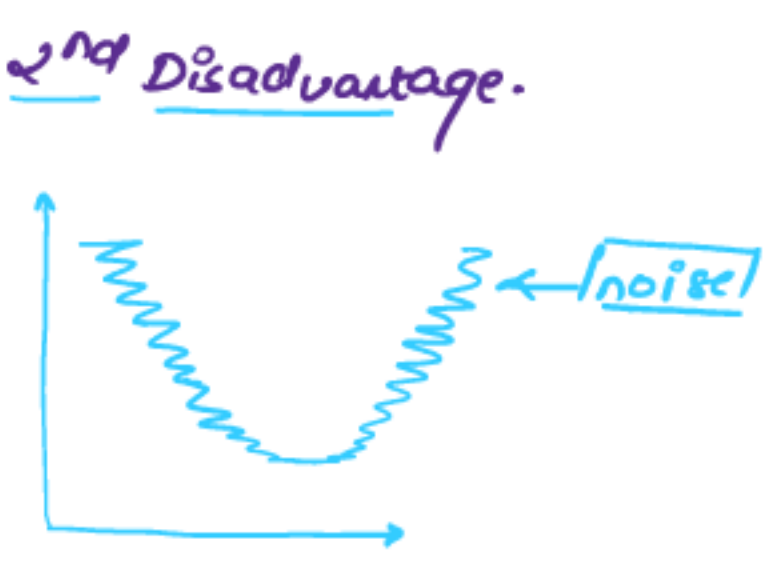
$w_{new} = w_{old} - \alpha \frac{\partial L}{\partial w_{old}}$

1. Record 1 good Loss ↓
2. Record 2 good Loss ↓
3. Record 3 good Loss ↓
4. Record 4 bad Loss ↑

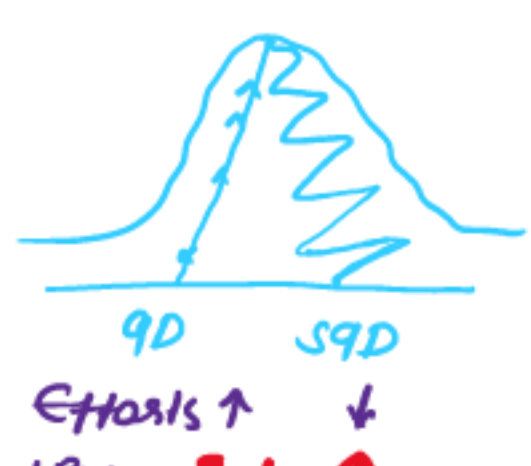
1. ES
2. w_{new} = old wt

slope reduce 0

2nd Disadvantage.



[Loss ↑ Loss ↓] ← [w ↑ w ↓]



	GD	SGD
Epochs	↑	↓
time	↓	↑
noise	minimal	high

as one sample from dataset is chosen randomly path taken to reach global minima is usualy noisier than typical gradient Descent

Mini batch stochastic gradient Descent

gd ⇒ Entire Dataset

sgd ⇒ 1 by 1

MBSGD ⇒ Batch by batch

Data set 1000

200 200 200 200 200

1 Epoch

5 Iteration

stochgsd (1 Record) [RP]

MB → [Batch] → (100) Read time

100 100

SGD: 1000 → Batch size 1, Batch no. 1000

200 1 Batch → NN → $\hat{y}$ → Loss ← [WB]

updated WB → NN → $\hat{y}$ → Loss ← [WB]

200

blood	blood	blood
[1000]	[1]	[200]
↑ strong	[↑↓]	[↑↓]

noise ↓ [MBSGD]

1. Here we have solved the problem of noise to some extent from SGD.
2. Batch sizes are often tuned to computational architecture. Such as power of [2]

4 8 16 32 64 512 →

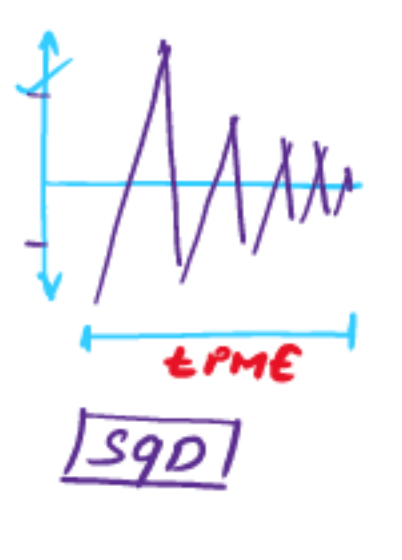
[512] ← batch size

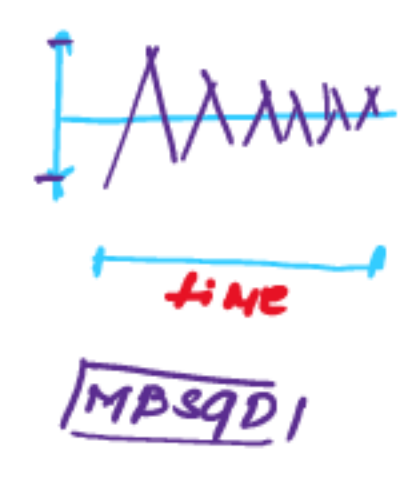
Depending upon batch size the updates can be made less noisy.

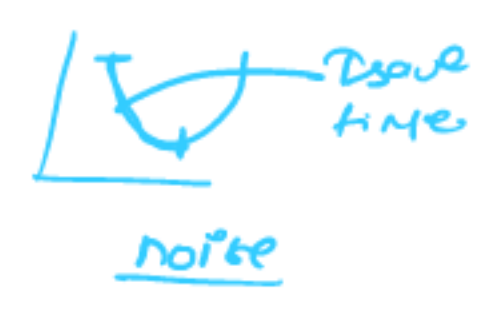
batch [↑] noise [↓]

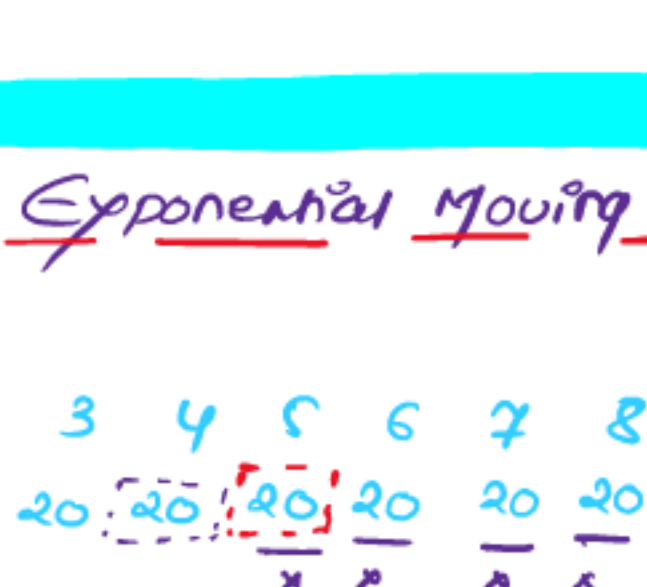
no. of batch 1 → (GD)

1000 → no. b 1, batch 1000 ← MBSGD ⇓ [GD]









Exponential Moving Avg

1	2	3	4	5	6	7	8	9	10	11
20	20	20	20	20	20	20	20	40	40	[P]

↑ Today

→ MA 33.3

→ Exp 27.2

Moving avg

8	9	10	11
20	40	40	[P] [33.33] ✓

(3) sliding window

$\frac{20+20+20}{3} = 60/3 = [20]$

$= \frac{60}{3} = [20]$

$= \frac{100}{3} = 33.33$

$\frac{120}{5} = 24$

[EMA]

P = B × previous day + (1−B) × Today. (B → 0.8)

= 0.8 × 24 + (1−0.8) × 40

= 0.8 × 24 + 0.2 × 40

= 19.2 + 8

[P = 27.2] 11th

$= \boxed{\frac{1}{1-B}}$

$= \frac{1}{1-0.8} = \frac{1}{0.2}$

[5 days]

$\frac{1}{1-B}$ = 10 days = 3 days

0.9, 0.7

1 yes [365] ←

$\frac{1}{1-0.99}$ ✓

Exponential smoothening is basically used in time related problems.

stock prices prediction

climate forecasting.