C- LoRA

Compression - Low Rank Adaptation

Aiffel Online 5th Research

3L 맹선재, 박혜원, 양주영, 황인준

Table of contents

01

Part.1

- Preliminary
- What is C-LoRA?
- What is WDS?
- Toy Project_WDS

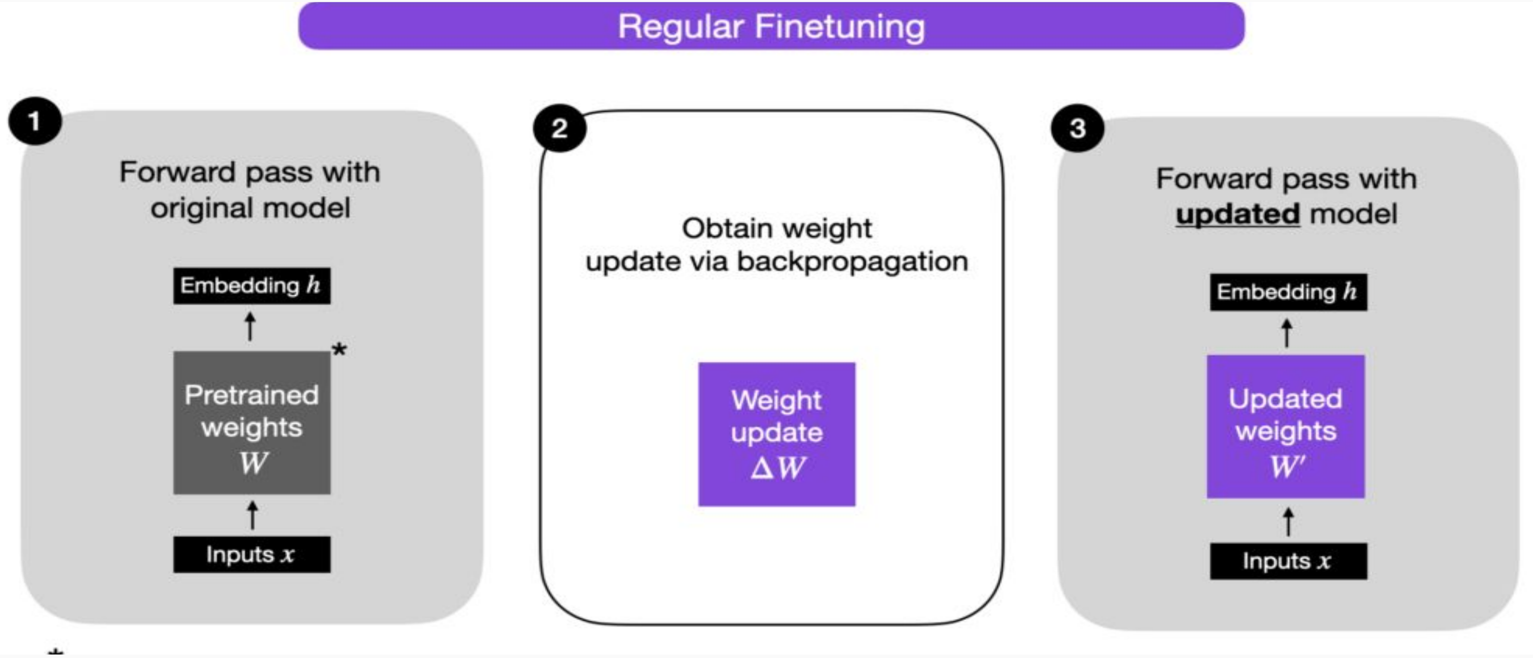
Part.1 **Preliminary - LoRA**

LoRA (Low-Rank Adaptation) :

Fine-tuning시, low-rank의 행렬에 대해서만 Parameter update

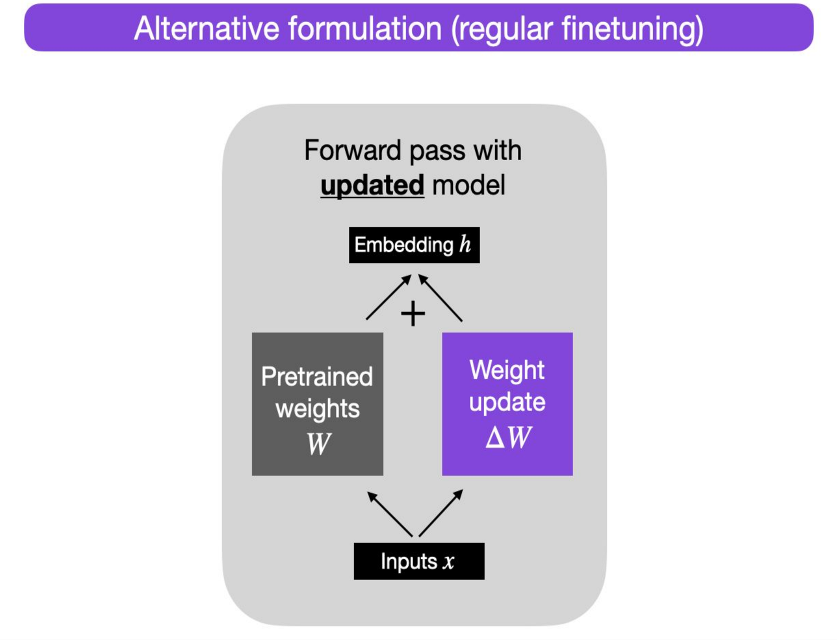
Part.1 Preliminary - LoRA

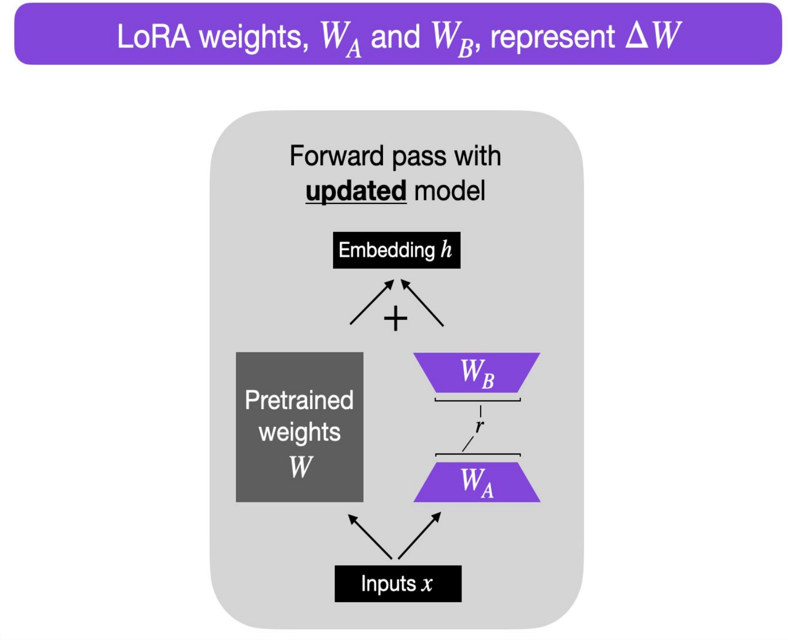
Pretrained weight (W) **전체**를 fine-tuning



Part.1 **Preliminary - LoRA**

병렬적으로 추가한 adapter의 **가중치 AB만** update   ∴ PEFT (parameter-efficient-fine-tuning)





Part.1

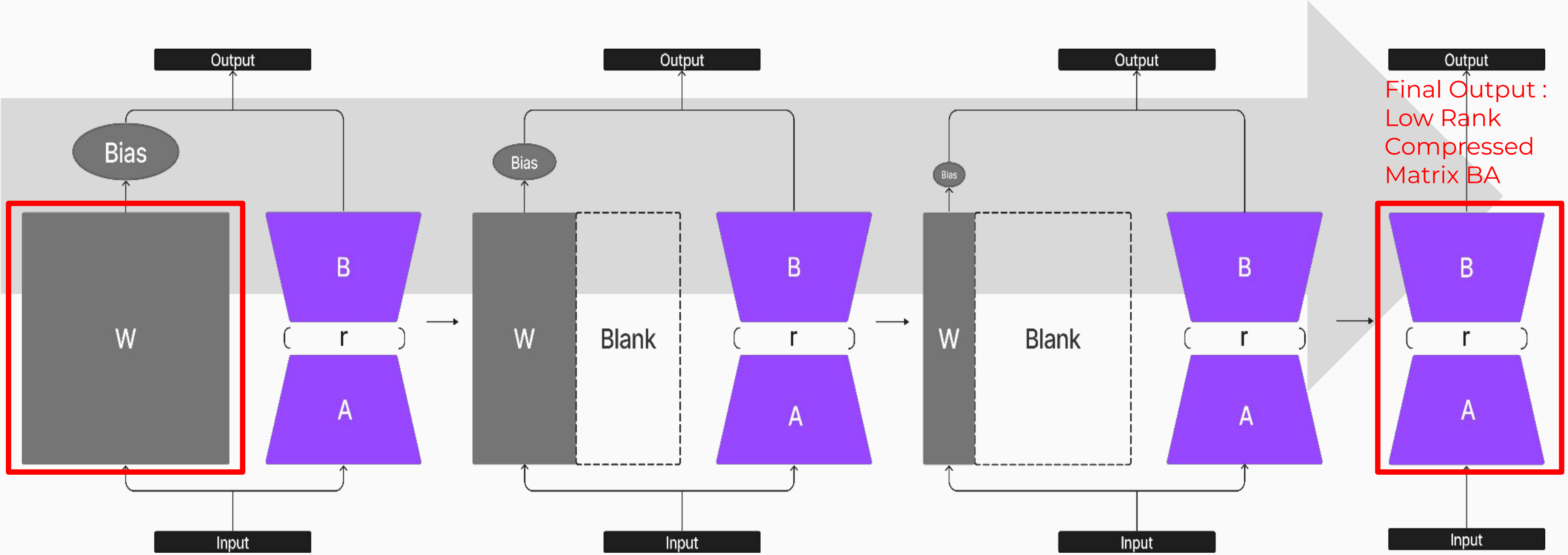
Based on LoRA, we propose a new *model compression* framework.

C-LoRA (Compression-LoRA)

Part.1 **What is WDS?**_Weight Decay Scheduling

Adapter처럼 W + BA 형태로 학습시킴과 동시에, W를 점진적으로 decay하여,

학습이 진행됨에 따라 BA에 잃어버린 W 정보 학습



Part.1 **Toy Project_ WDS**

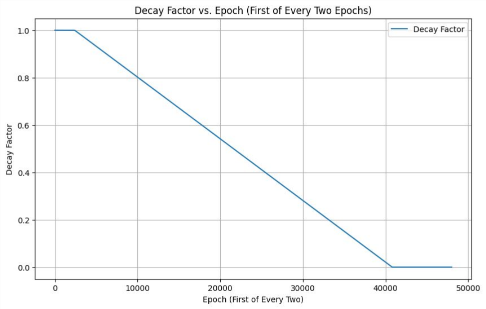
Model structure: 3 dense layers with units of (256, 128, 10)

Dataset: Fashion MNIST(28*28, 70000 images, 10 categories)

Compression target layers: dense layers with units of (256, 128)

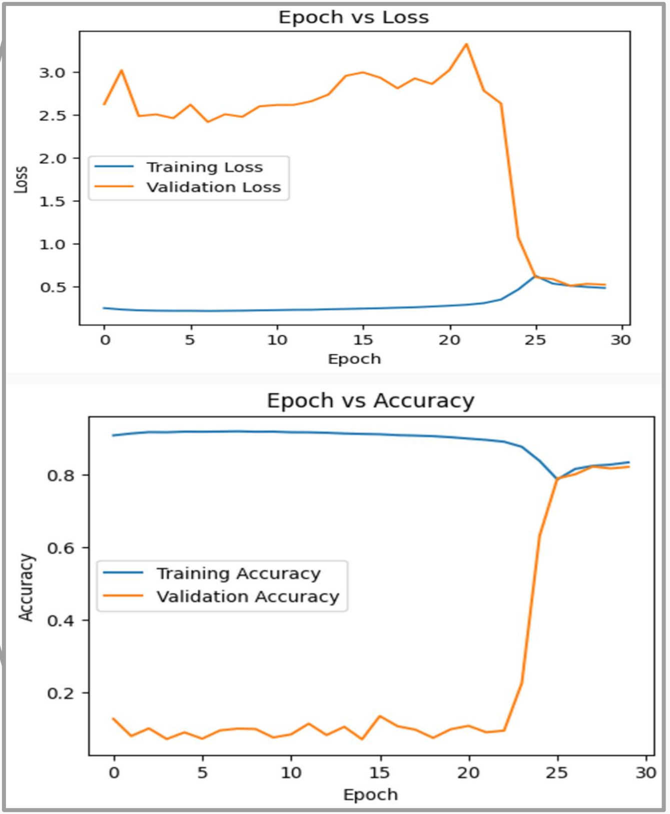
Original val_accuracy: 87.79%

WDS:



Part.1 **Toy Project_ WDS**

Compression Ratio(%)	**Rank**	**Accuracy**(%)
100	-	87.79
9.9	16	82.1**(-5.69%P)**
19.5	32	83.2
38.9	64	82.87



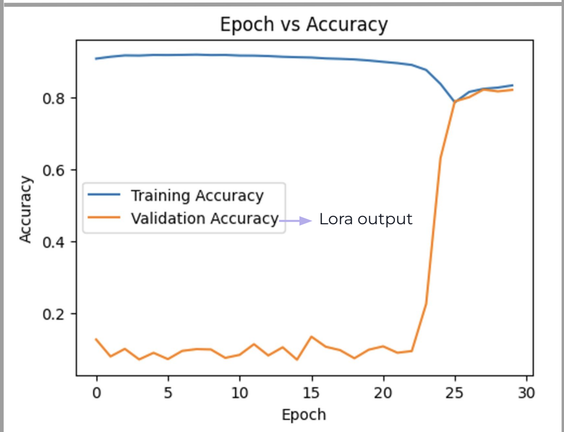
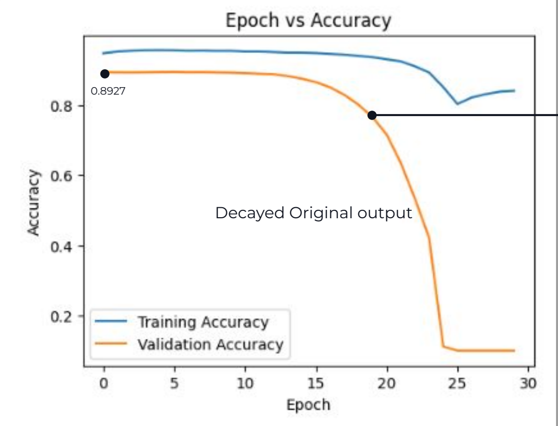
Compression ratio : compressed model params / original total params

02

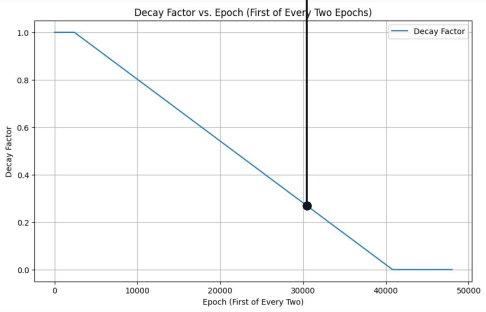
Part.2

- Toy project Issues
- Implementing Noise & Bias

Part.2 **Toy Project_**Issues

<Adapter Acc>	<Decaying original Acc>
	

Epoch : 20
Decay Factor: 0.23
val_acc:
0.8007(-9.2%p)

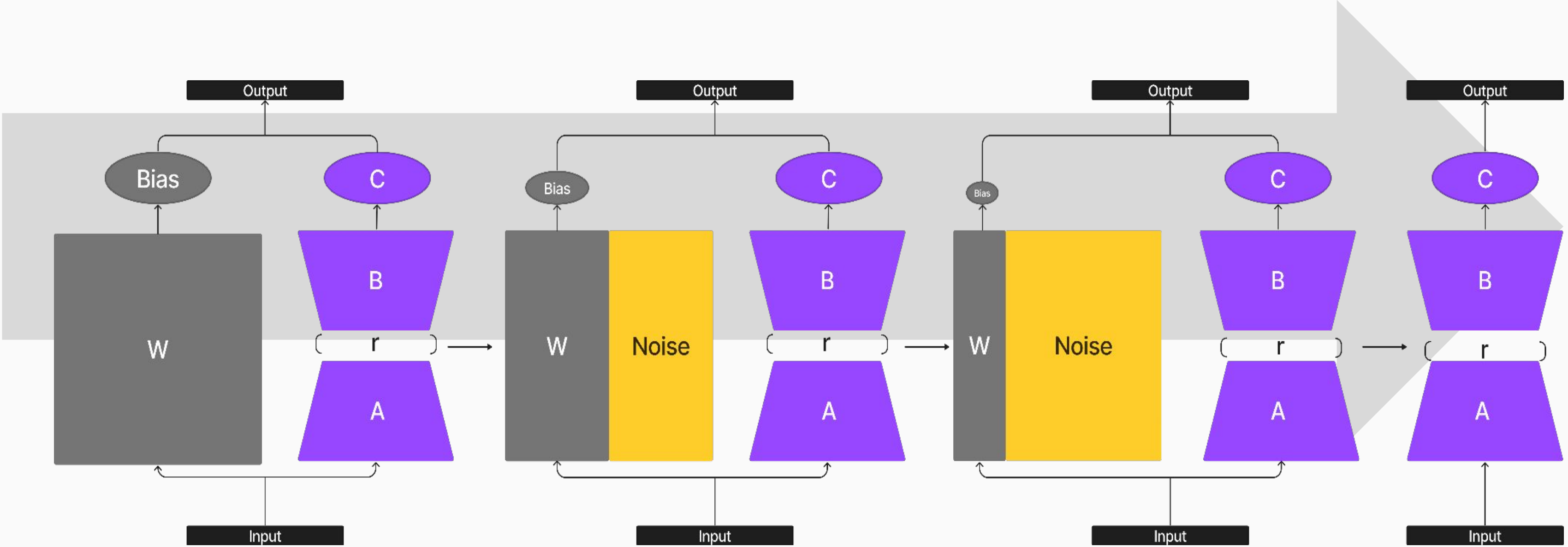


decay factor 가 0.23으로 작아졌음에도 불구하고, decayed W 만으로 추론했을 때, 9% 정도의 성능 저하만 있다는 것을 확인

=> 단순히 스칼라 값만 곱하는 것이, W의 학습 정보를 decay 시키지 못한다는 것을 추정함.

Part.2 **Implementing Noise & Bias**

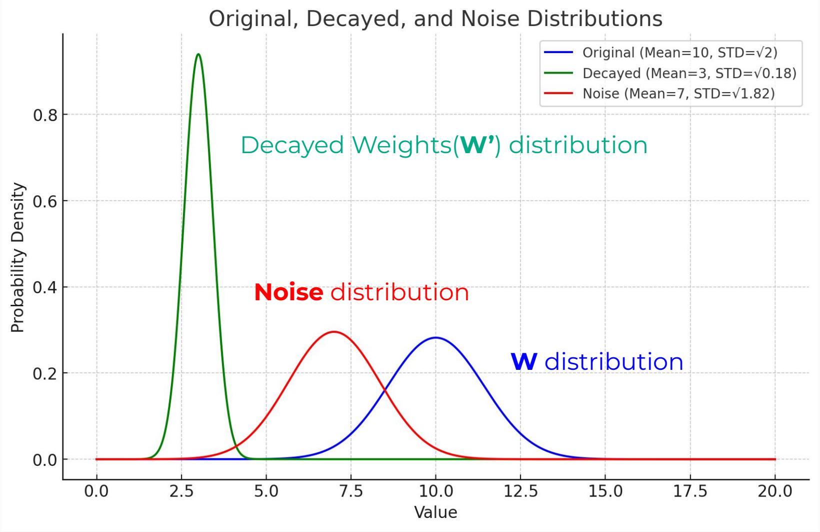
Noise와 **Bias**를 추가하여 W의 학습 정보를 공격적으로 decay.

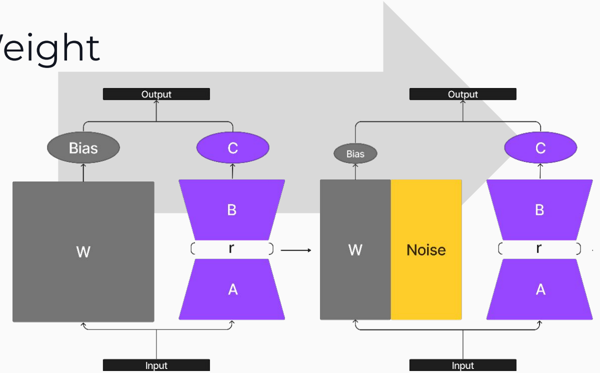


Part.2 **Toy Project_** Noise

Compensate decayed weight matrix **with noise**

Assumption: Noise is **independent** from Original Weight





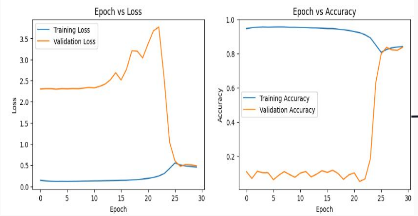
$$W \sim N(m, \sigma^2)$$

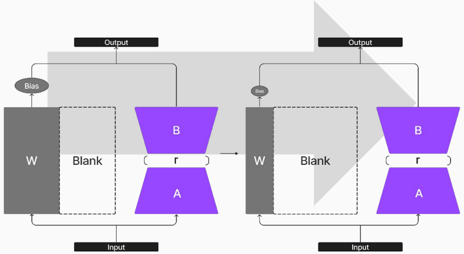
$$W' \sim N(d \cdot m, d^2 \cdot \sigma^2)$$

$$noise \sim N((1-d) \cdot m, (1-d^2) \cdot \sigma^2)$$

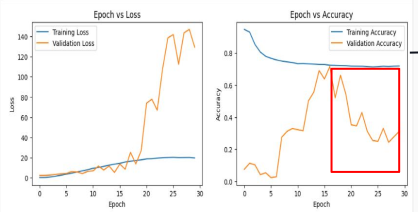
$$W' + noise \sim N(m, \sigma^2)$$

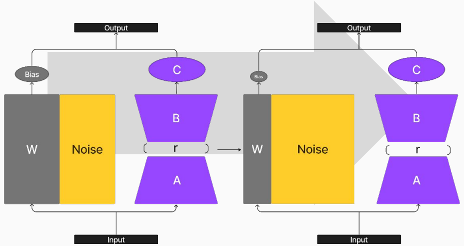
Part.2 **Project Concept**





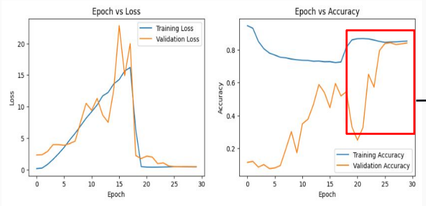
Only WDS*

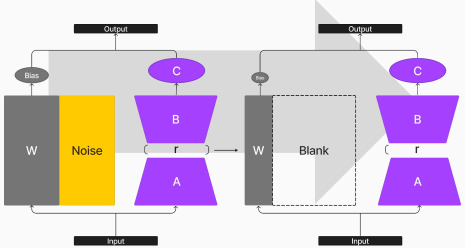




+ Noise + Bias

-Noise exists until End of learning





+ Scheduled Noise + Bias

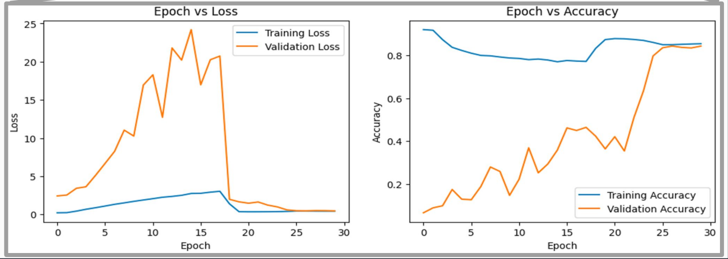
-Noise gone at 0.3 decay factor

WDS* = Weight Decay Scheduling

Part.2 **Project Concept**

Compression ratio (%)	Rank	Accuracy(%)			
		Only WDS	Bias (C_weights)	Scheduled Noise	Scheduled Noise & Bias
100	-	87.79			
9.9	16	82.1	82.73	81.59	**83.74(-4.05%p)**
19.5	32	83.2	82.55	83.44	**83.93**
38.9	64	82.87	83.68	82.38	**84.23**

The results show that C-LoRA efficiently compress the number of parameters (e.g, 9.9%) with marginal performance degradation (- 4%p).



03

Part.3

- VGG16
- BERT

Part.3 **VGG16**

Model structure: Pretrained vgg16('imagenet')
- 13 convolution layer, 3 dense layer

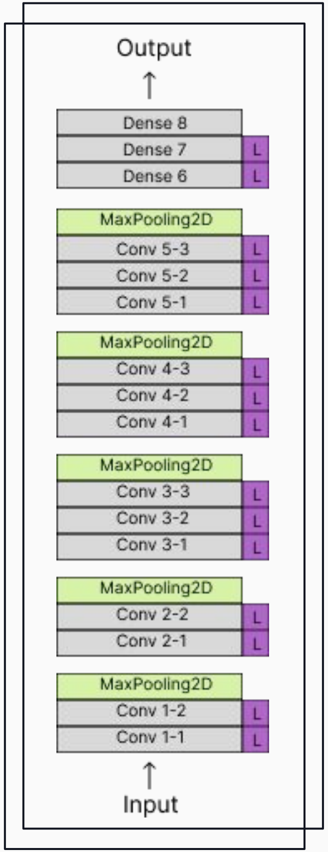
Dataset:CIFAR 10
(32*32,5000 train dataset, 1000 test dataset, 10 categories)

Compression target layers:
- All Conv, dense_6, dense_7 layer

Hyperparameters : Batch Size: 30 Learning Rate:0.0001

Total params: 1.9M

Original val_accuracy: 85.36%



Part.3 Experiment on **VGG16**

Compression Ratio (%)	**Rank**		**Accuracy**(%)			
	Conv LoRA	**Dense LoRA**	**Only WDS**	**Bias (C_weights)**	**Scheduled noise**	**Scheduled noise & Bias**
100	-	-	85.36			
6.90	16	16	74.98 ±1.34	**75.79** ±0.30	74.70 ±0.78	75.57 ±0.40
7.68	16	32	75.39 ±0.85	75.42 ±0.28	74.44 ±0.31	**75.52** ±0.17
9.21	16	64	75.72 ±0.56	74.77 ±0.61	75.46 ±0.63	**75.83** ±0.61
12.94	32	16	**76.96** ±0.91	75.99 ±0.16	76.08 ±0.36	76.49 ±0.53
13.72	32	32	**76.81** ±0.85	76.04 ±0.91	76.69 ±0.93	76.38 ±0.54
15.25	32	64	76.32 ±1.07	77.17 ±0.60	76.92 ±0.19	**77.51** ±1.87

The results show that C-LoRA efficiently compress the number of parameters (e.g, 6.9%~15.25%), while the performance degradation (-10.92 %p~-7.85%p) is marginal.

Part.3 Experiment on **Bert-Tiny & Bert-Small**

Model structure:
- Bert-Tiny (2 Encoder Blocks) , Bert-Small (4 Encoder Blocks)
- Pretrained on English Wikipedia + BooksCorpus

Dataset: IMDB Dataset (Text Classification Dataset)
(25000 training dataset, 25000 test dataset , 2 categories)

Compression target layers: Self attention block's layers

- Query, Key, Value, Output Dense layer (Einsum Dense Layer)

Hyperparameters : Batch Size: 32 Learning Rate: 0.00005
Total params: 4.3M(Bert Tiny) 28M (Bert Small)

Part.3 **Bert-Tiny**

Compression ratio of Transformer Encoder Blocks (%)	Rank	Accuracy(%)			
		Only WDS	Bias (C_weights)	Scheduled Noise	Scheduled Noise & Bias
100	-	83.03 (With Early Stopping Method, Achieved its best val-loss on Epoch 4)			
8.8	16	80.00 ±0.28	79.97 ±0.36	80.68 ±0.05	**80.79** ±0.33
17.0	32	81.26 ±0.15	81.20 ±0.19	81.36 ±0.23	**81.43** ±0.38
33.6	64	**81.91** ±0.26	81.84 ±0.16	81.57 ±0.26	81.70 ±0.08

The results show that C-LoRA efficiently compress the number of parameters (e.g, 8.8%~33.6%), while the performance degradation (- 3.06%p ~ -1.12%p) is marginal.

Part.3 **Bert-Small**

Compression ratio of Transformer Encoder Blocks (%)	Rank	Accuracy(%)			
		Only WDS	**Bias (C_weights)**	**Scheduled Noise**	**Scheduled Noise & Bias**
100	-	89.05			
2.1	16	84.83 ± 0.24	84.93 ± 0.25	**85.24** ± 0.24	85.11 ± 0.19
4.2	32	85.66 ± 0.43	**85.80** ± 0.44	85.66 ± 0.23	85.68 ± 0.25
8.4	64	86.01 ± 0.28	85.95 ± 0.28	**86.02** ± 0.21	85.88 ± 0.15

The results show that C-LoRA efficiently compress the number of parameters (e.g, 2.1%~8.4%), while the performance degradation (- 4.2%p~-3%p) is marginal.

04

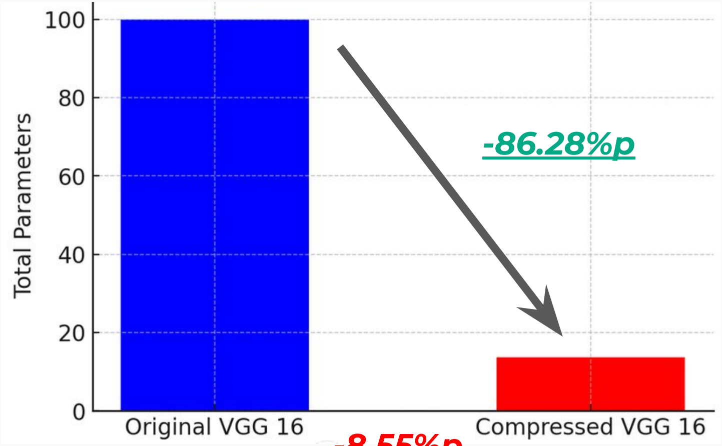
Part.4

- Conclusion
- Future steps

Part.4 **Conclusion**

C- LoRA	VGG 16(%)	Bert-Tiny(%)	Bert-Small(%)
Only WDS	75.63 ±0.58	81.06±0.56	85.50±0.38
Bias	75.84 ±0.41	81.00±0.55	85.56±0.35
Scheduled Noise	75.54±0.50	81.21±0.28	85.56±0.25
Scheduled Noise & Bias	**76.00±0.48**	**81.31±0.30**	**85.64±0.25**

Part.4 **Conclusion**

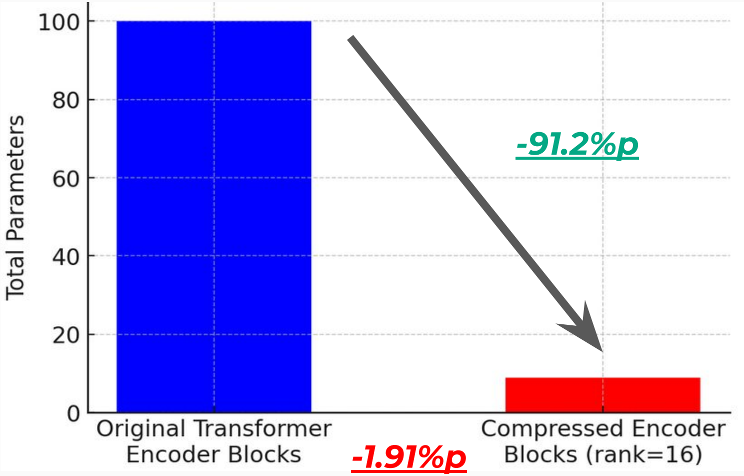


-8.55%p

acc : 85.36% → 76.81%

VGG 16 Compression (13.72%)

_Only WDS



-1.91%p

83.03% → 81.12%

Bert Tiny Compression (8.8%)

_Bias & Scheduled noise

Part.4

Future steps

B & A Matrix Initialization

- Research and Application of Initialization Techniques for Improved Weight Transfer

Noise Implementation Enhancing

- Noise Scheduling Adjustment
- Noise Level Adjustment

Thanks!

Do you have any questions?

Ask Me NOW :)

OR:

3l.aiffelton@gmail.com