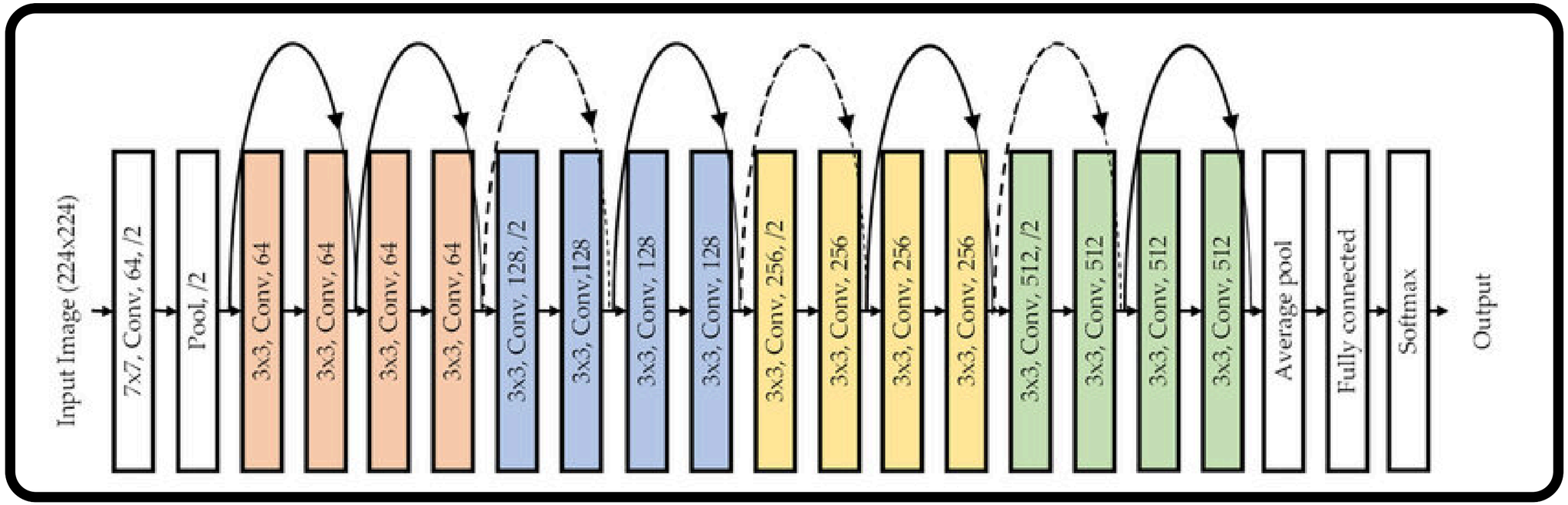
# ResNet

(He, Zhang, Ren, & Sun, 2015)




Xueyun WENG & Léa MIQUEU

# Workflow

**Learning capacity**
*(overfitting)*

- **identifying :**
  - **#epochs**
- **tuning :**
  - **learning rate**
  - **batch size**

**Improving generalization**
*(fight overfitting)*

**Regularization**

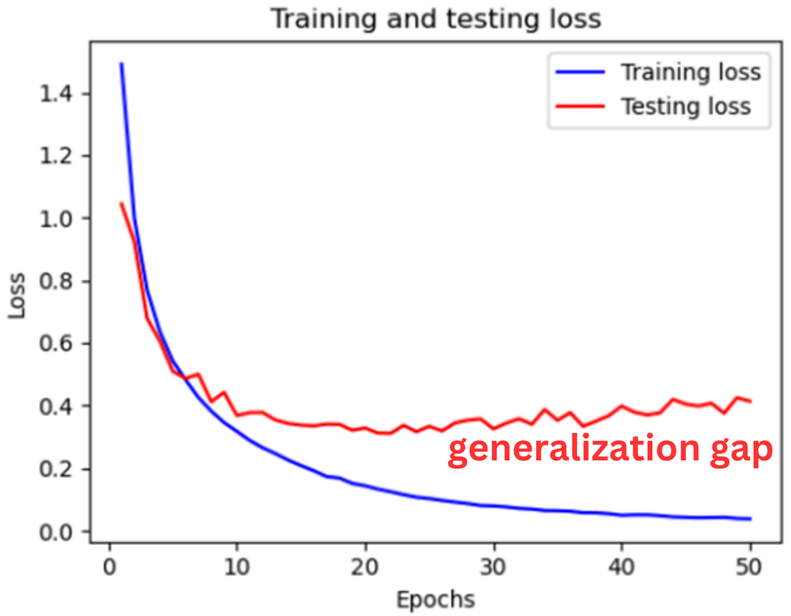
- **weight decay** on the scheduler

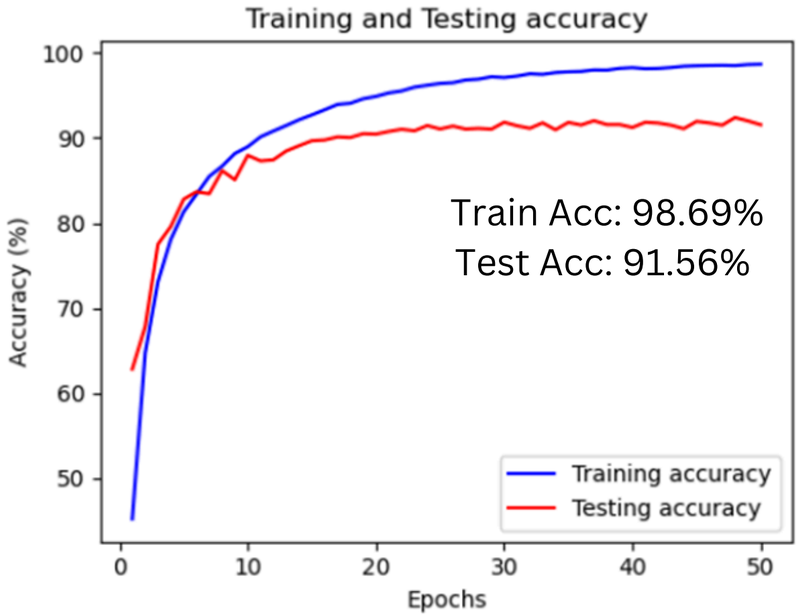
**Data Augmentation**

- **Mixcut**
- **Crop, flip, rotation, scaling**

# Learning capacity

| Parameter | Value |
|---|---|
| # epochs | 50 |
| batch size | 32 |
| optimizer | Adam |
| learning rate | 0.001 |



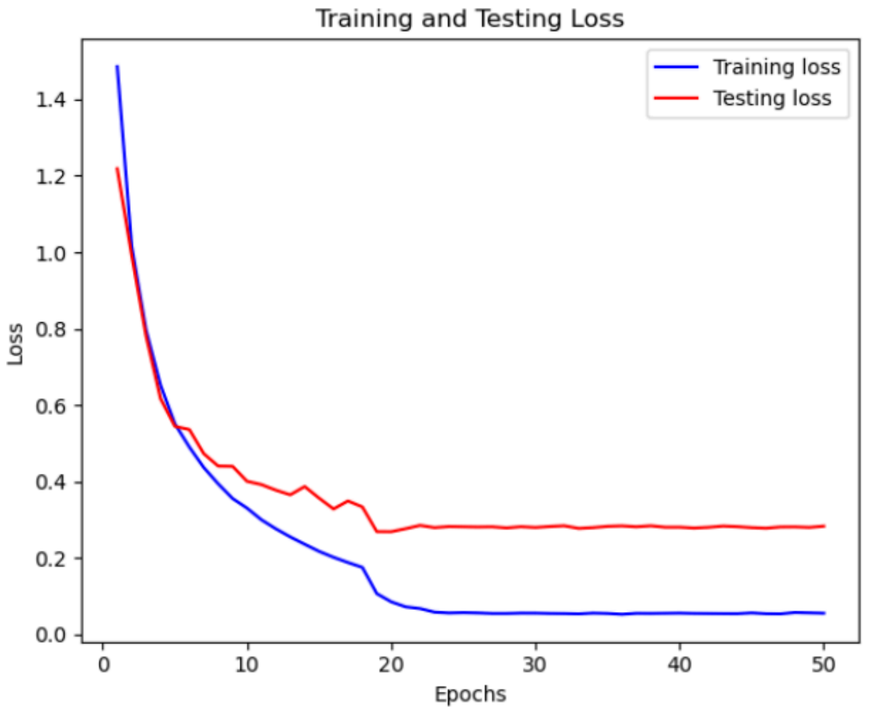

Observation:

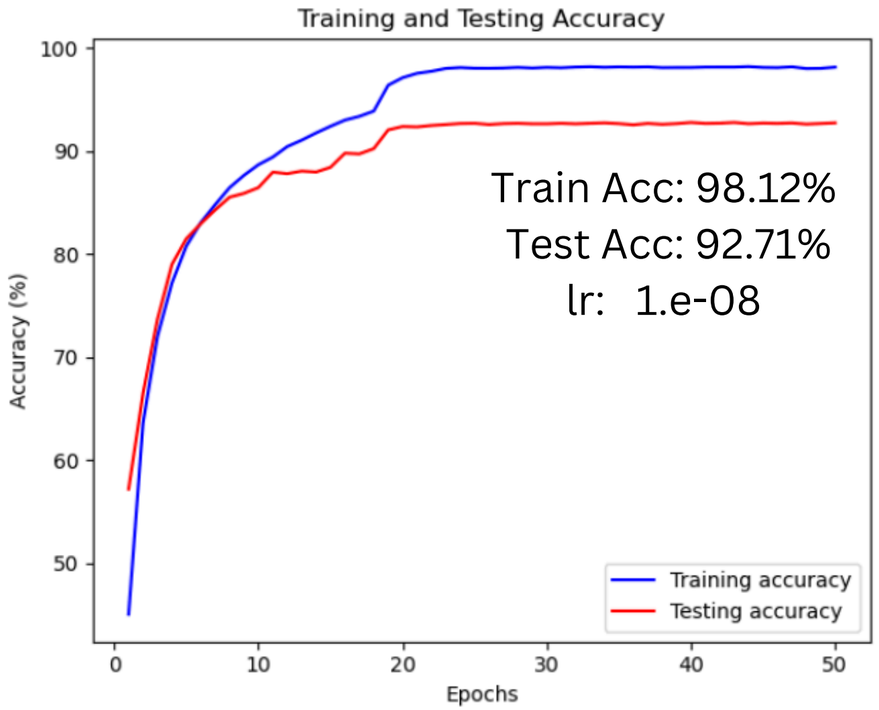
🚨the model overfits rapidly --> good learning capacity on training set but can't generalize

change the **batch size** for *Regularization Effect*

# Learning capacity

| Parameter | Value |
|---|---|
| # epochs | 50 |
| batch size | 64 |
| optimizer | Adam |
| learning rate | 0.001 |
| scheduler | On Plateau (0.10) |



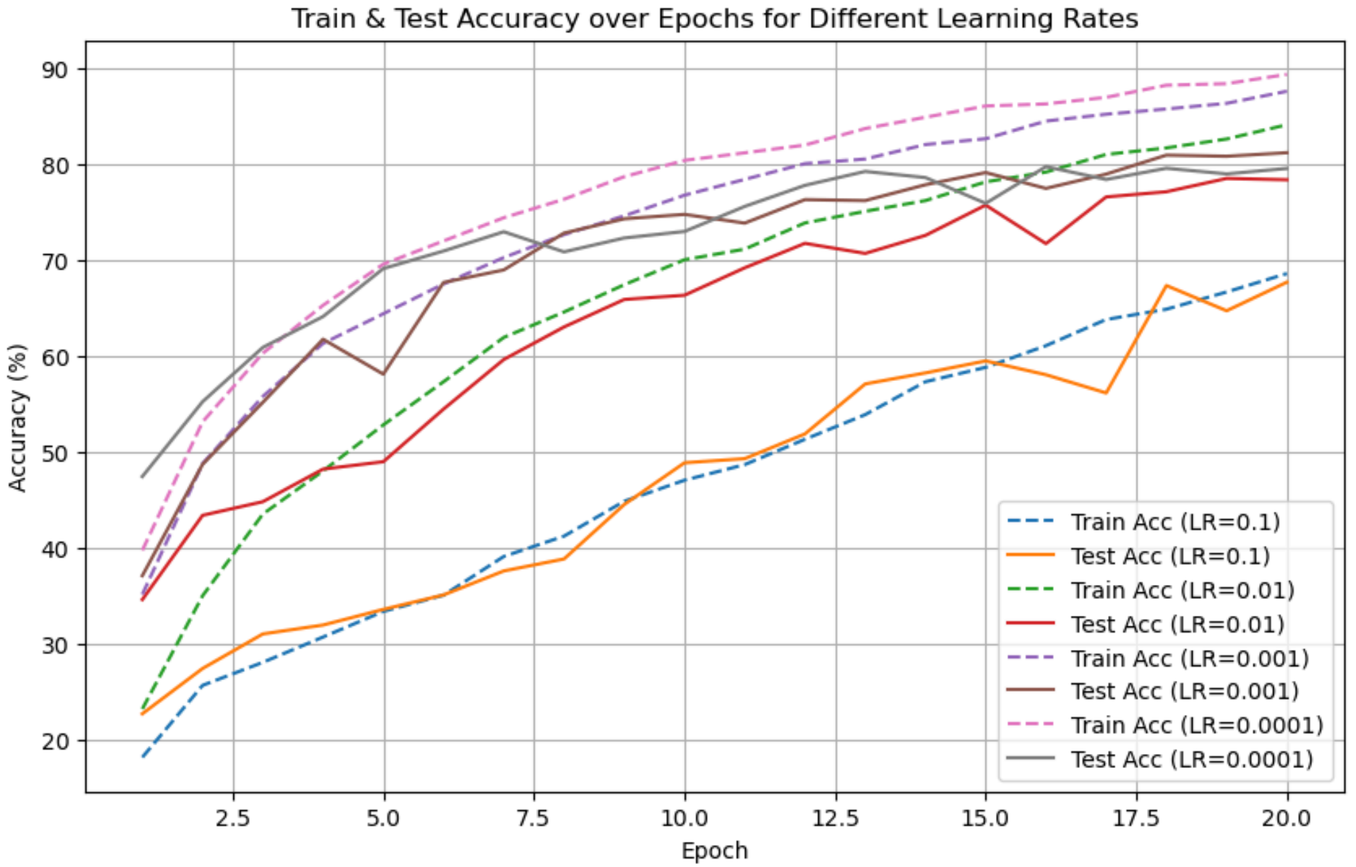




Observation:

✅ generalization (difference train/test : 7,13 --> 5.41)

Next: fight overfitting through model regularization and data augmentation

# Starting Learning Rate

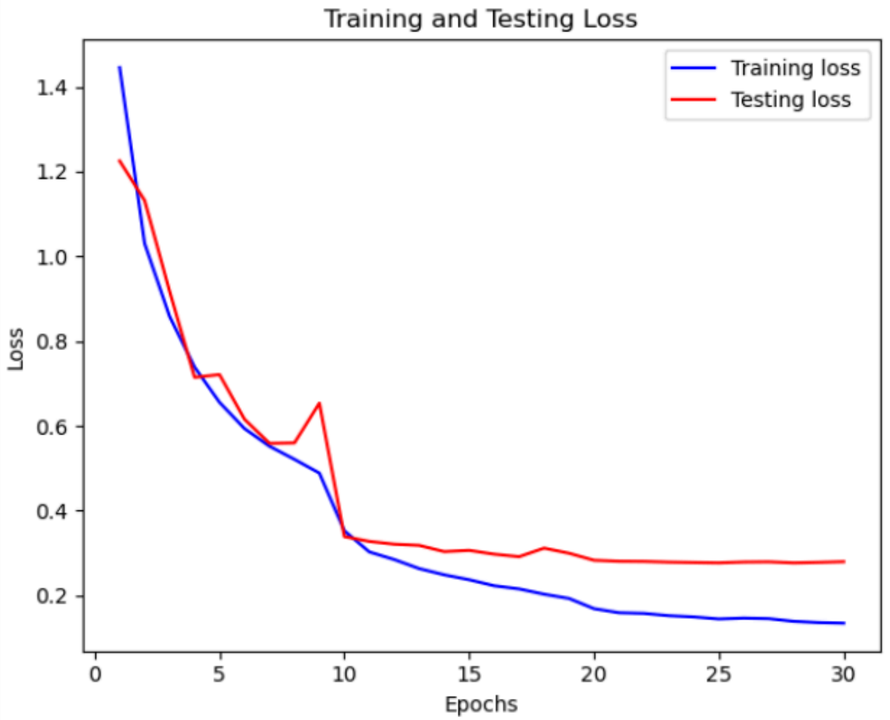


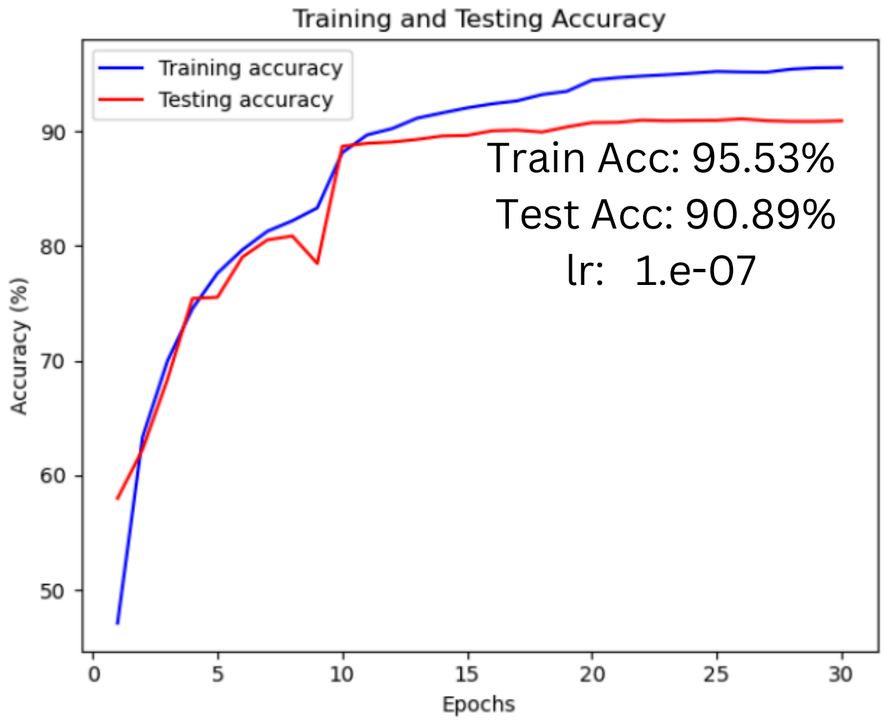

| Parameter | Value |
|---|---|
| # epochs | 50 |
| batch size | 64 |
| optimizer | Adam |
| learning rate | 0.001 |
| scheduler | On Plateau (0.10) |

# Regularization wih weight decay

ℓ2 penatly term is added to the loss, limits the growth of model weights

| Parameter | Value |
|---|---|
| # epochs | 30 |
| batch size | 64 |
| optimizer | Adam |
| learning rate | 0.001 |
| scheduler | On Plateau (0.10) |
| weight decay | 0.0005 |



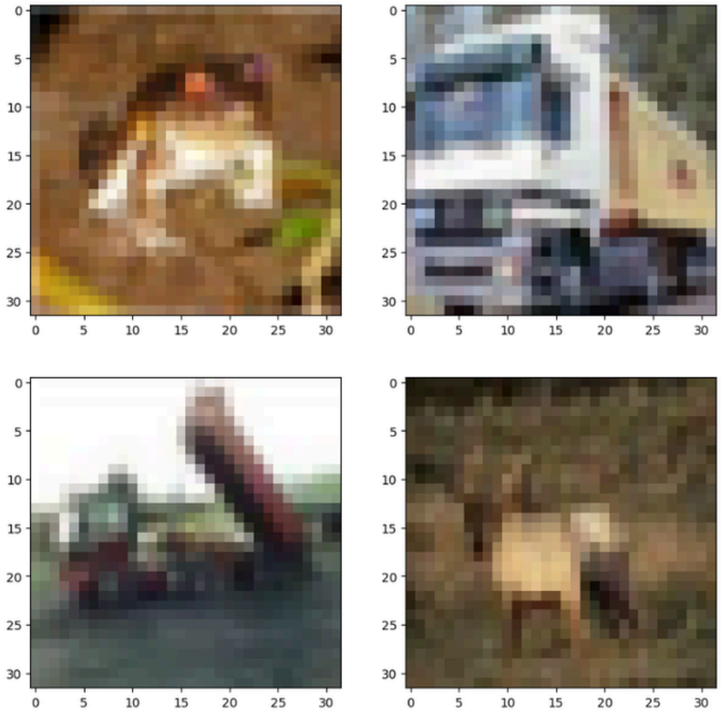

**Obeservation:**

- **slightly getting rid of overfitting**

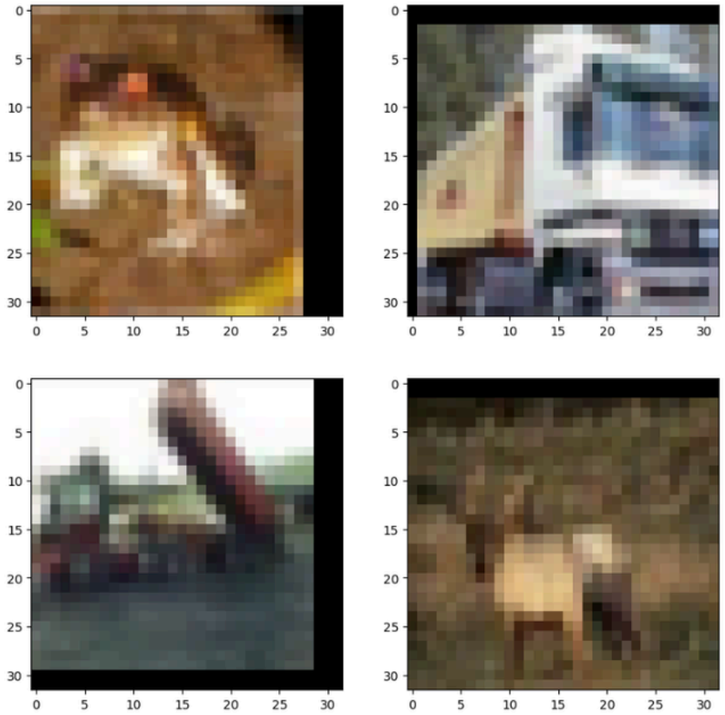
# Data Augmentation

Help generalization by sampling training examples from a larger distribution using randomized transforms

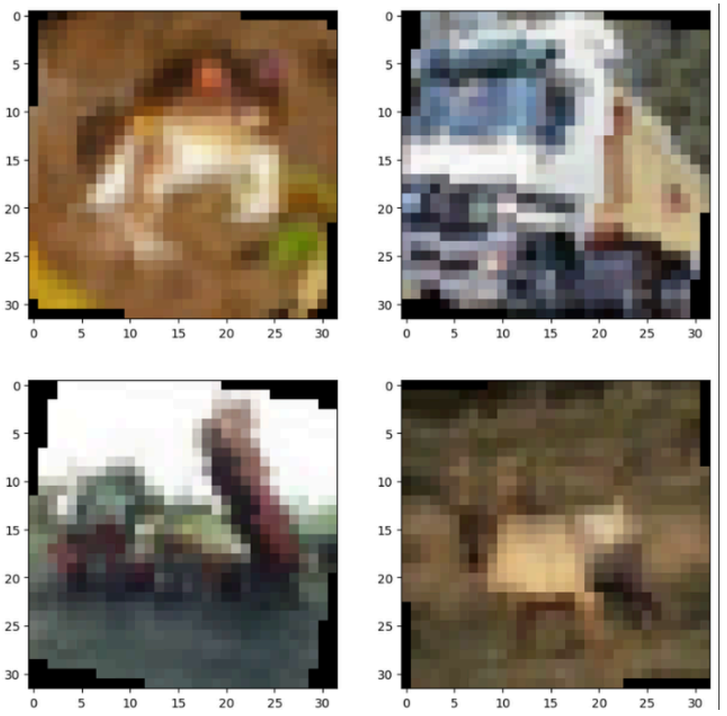
No Data Augmentation



Crop and Flip

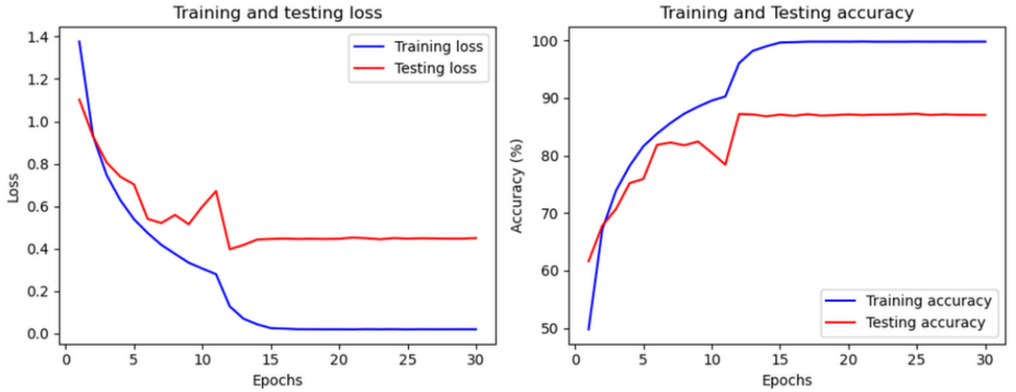


Scaling and Rotation

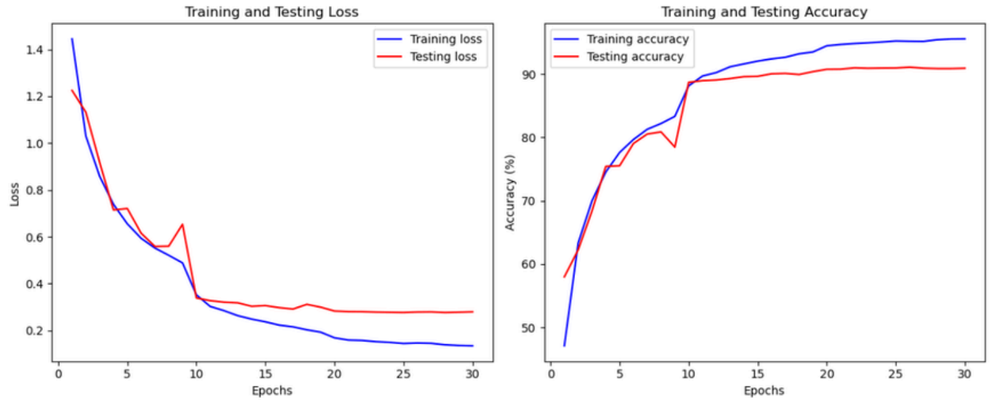


# Data Augmentation

No Data Augmentation

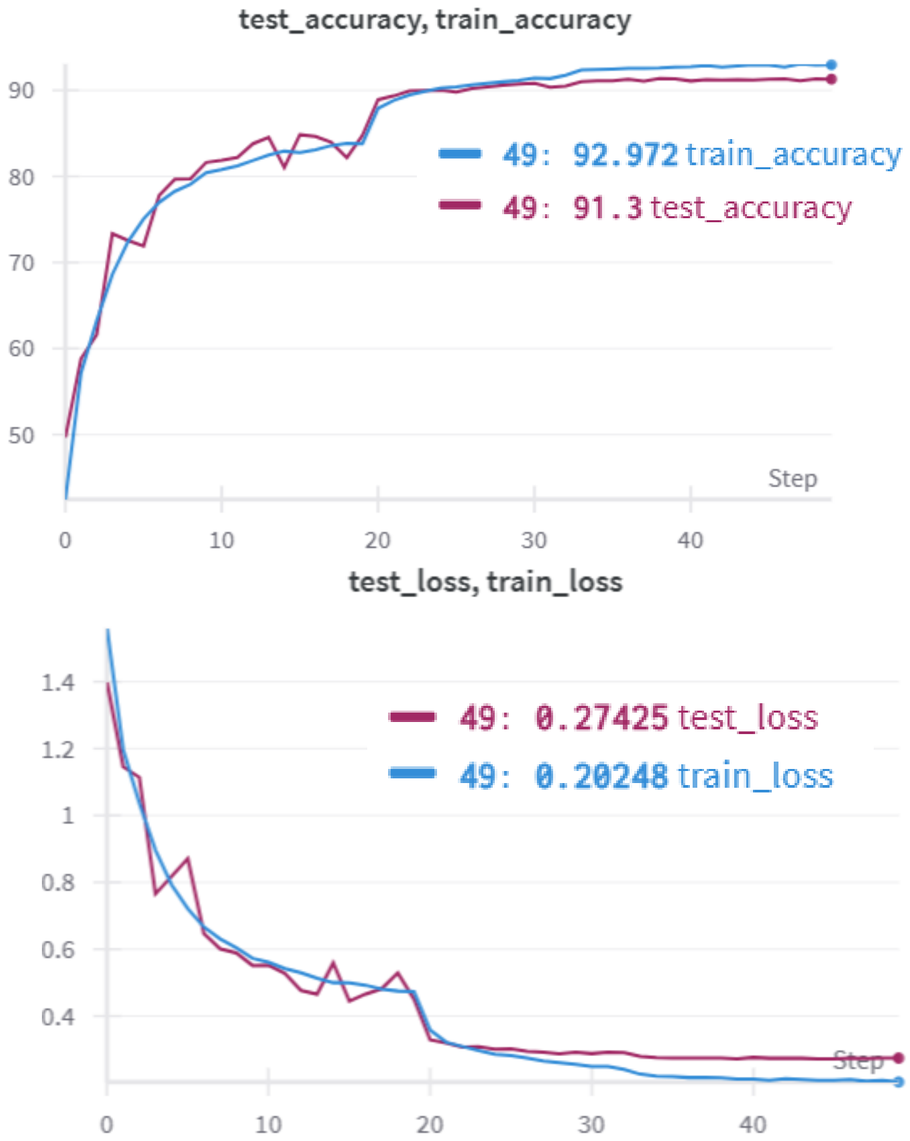



Crop and Flip




Scaling and Rotation

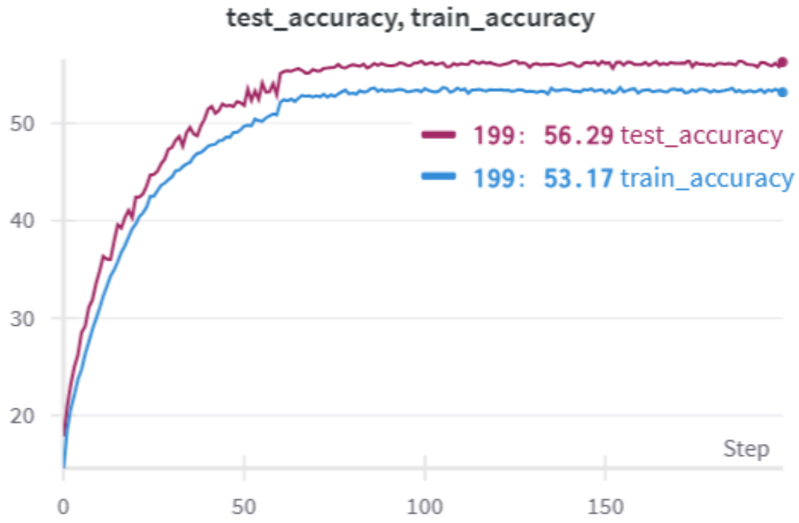
# Data Augmentation

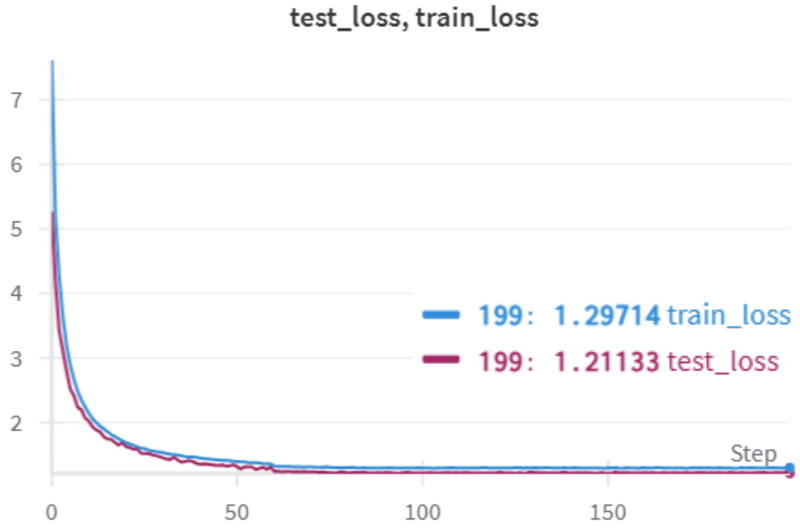



Crop + Flip + Scaling + Rotation

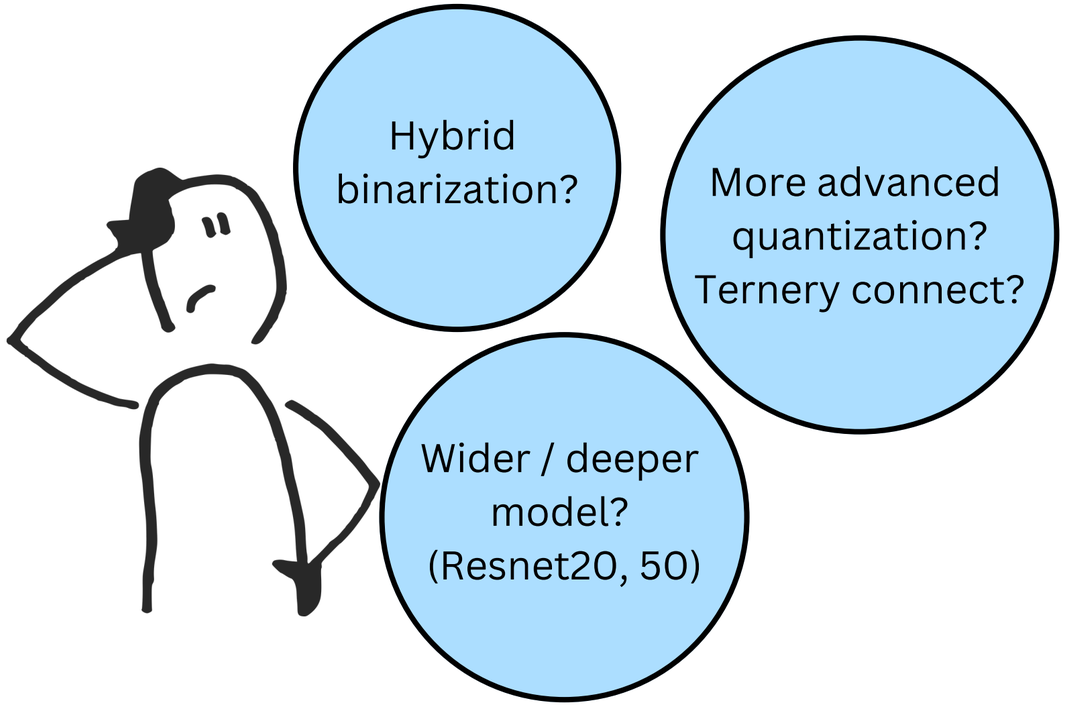
fancy

# Quantization - Binary Connect

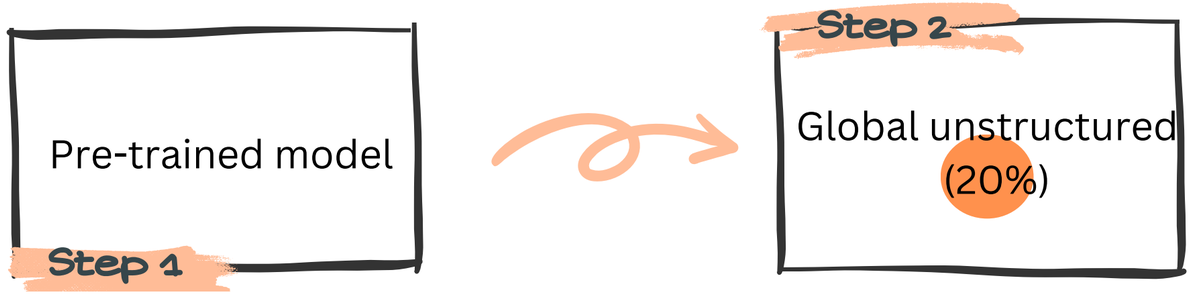


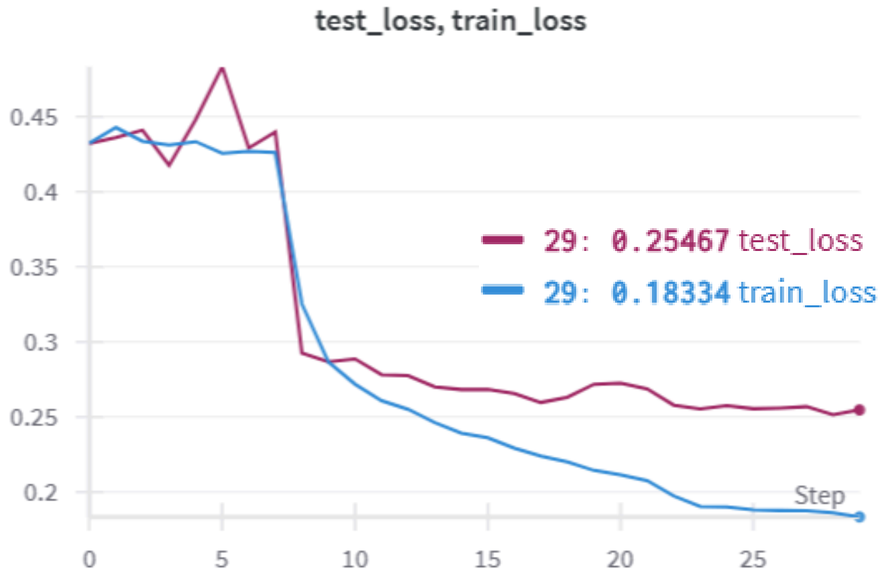



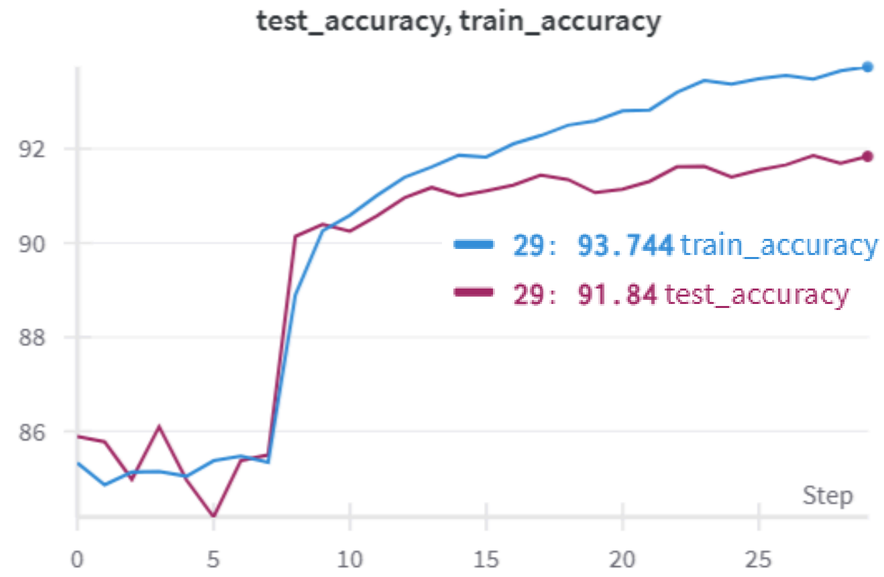

- Bad and saturating accuracies
- Losses stagnate



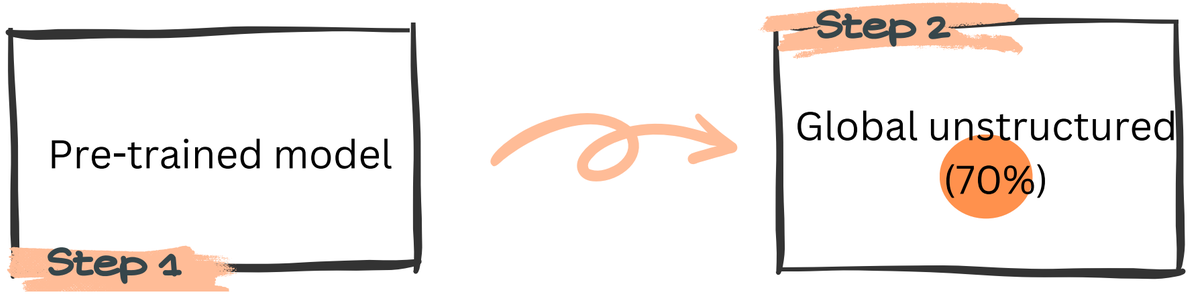
# Simple pruning

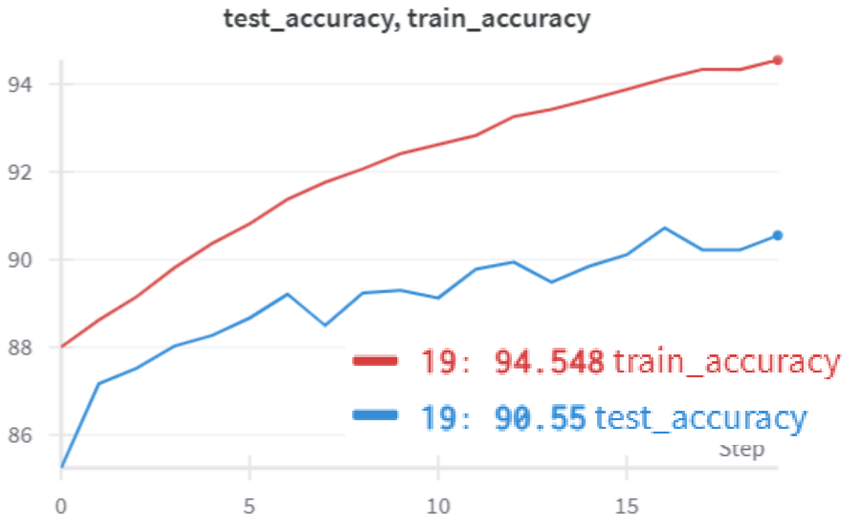


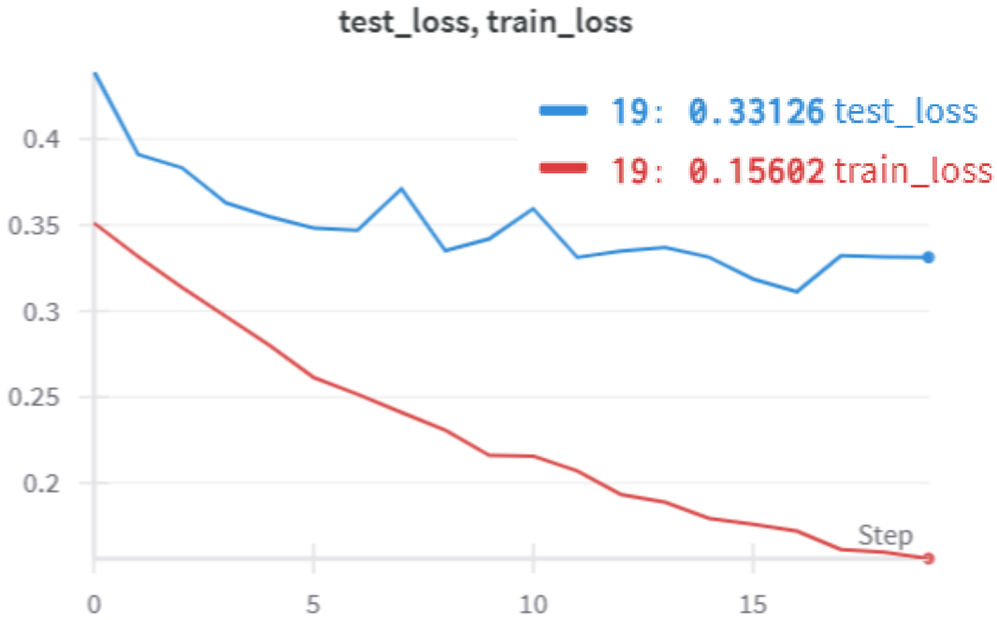



# Simple pruning




- higher sparsity --> lower accuracy
- needs more epoch to stabilize


# Pruning strategy

Pre-trained model

Step 1

Step 2

- L1 filter local structured (**ps**)
- Fine tuning

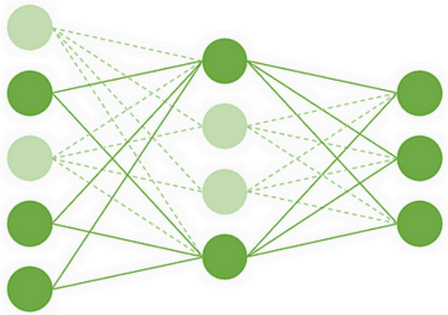
- L1 weight global unstructured (**pu**)
- Fine tuning

Step 3

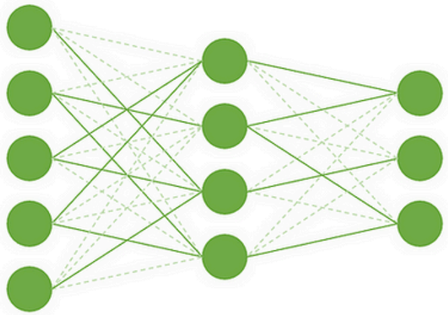
Step 4

Quantization
.half()

Structured Pruning

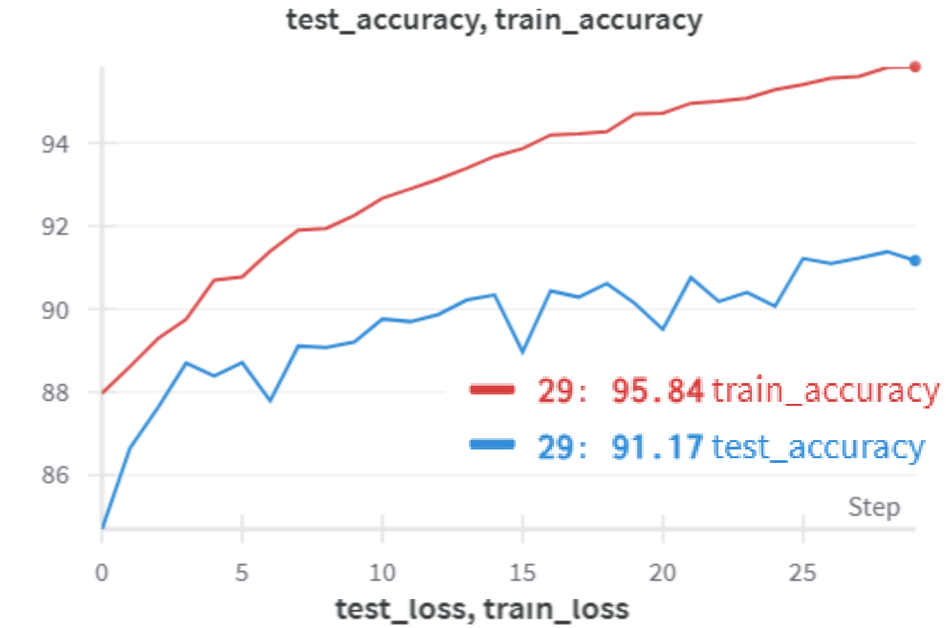


Unstructured Pruning



Start with structured pruning (e.g., remove 30-50% filters).
Fine-tune with unstructured pruning to reach higher sparsity.
Retrain after each pruning step to recover accuracy.

# Pruning strategy



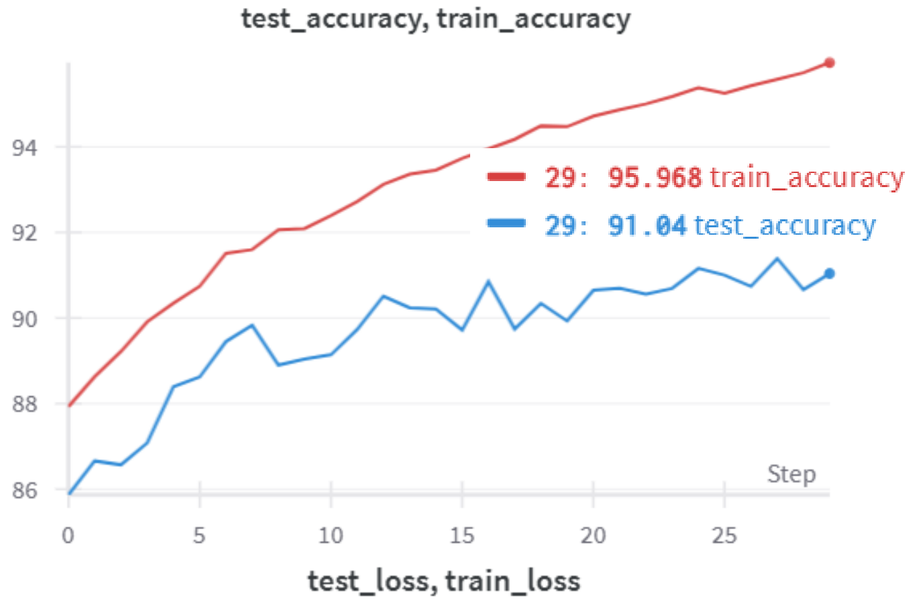


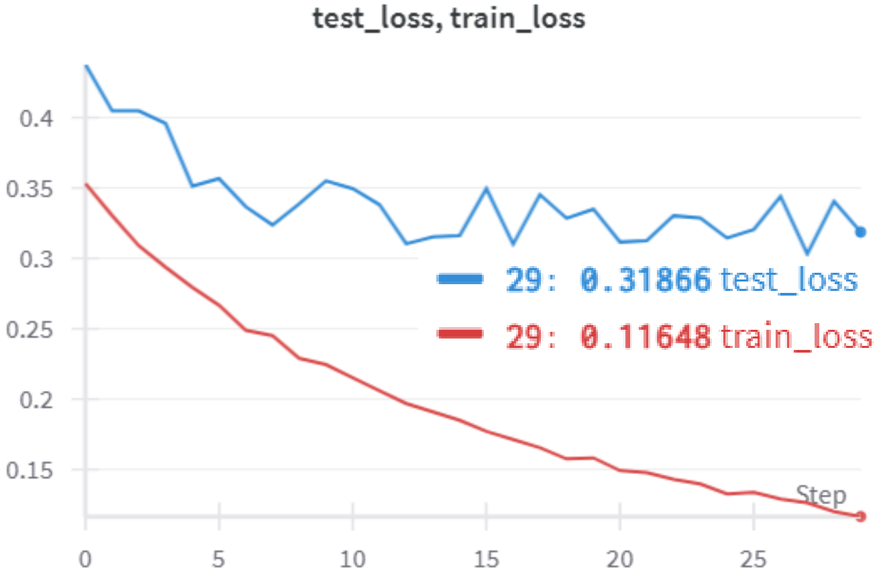
| Parameter | Value |
|---|---|
| ps | 0.2 |
| pu | 0.2 |

- Fine-tuning recovers most of the lost accuracy
- hard-ware efficiency with ps
- good accuracy with pu

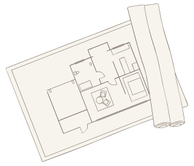
# Pruning strategy

| Parameter | Value |
|---|---|
| ps | 0.3 |
| pu | 0.2 |




- higher ps
  - lower accuracy than ps=0.2
  - larger impact than unstructured alone



# Factorization - Architecture changed

depthwise_from_scratch:

- modified version of the ResNet-18 model by replacing certain convolutional layers with depthwise separable convolutions, depending on the use_depthwise flag.

group: 2 puissance x, where x in an integer

- modified version of the ResNet-18 model by replacing certain convolutional layers with depthwise separable convolutions, depending on the use_depthwise flag.

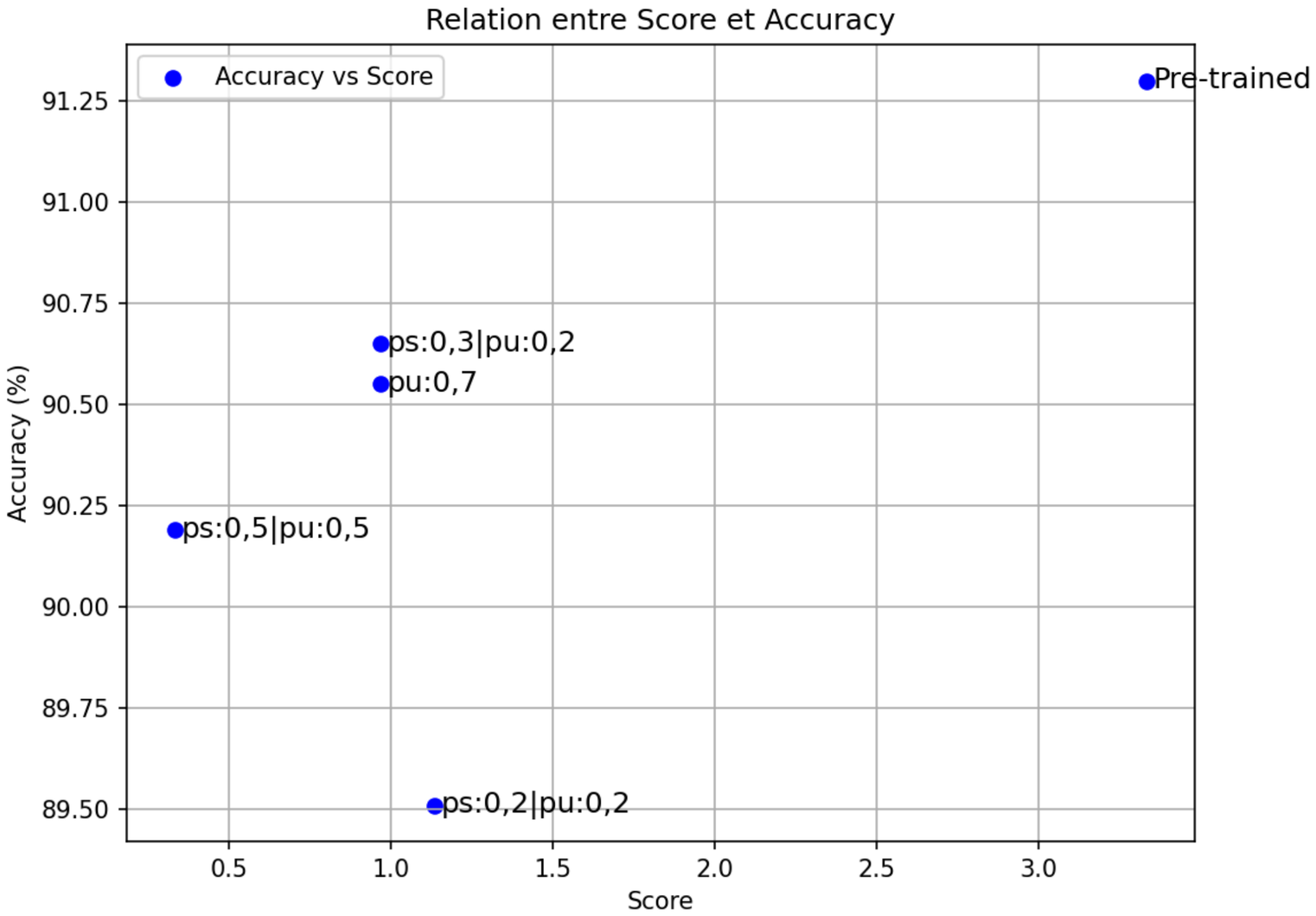
# Score

$$\text{score} = \underbrace{\frac{[1-(p_s+p_u)]\frac{q_w}{32}w}{5.6\cdot 10^6}}_{param} + \underbrace{\frac{(1-p_s)\frac{\max(q_w,q_a)}{32}f}{2.8\cdot 10^8}}_{ops}$$

| Score | ps | pu | qa | qw |
|---|---|---|---|---|
| 3.3337 | - | - | - | - |
| 0.9673 | 0.3 | 0.2 | 16 | 16 |
| 1.1340 | 0.2 | 0.2 | 16 | 16 |
| 0.9685 | - | 0.7 | 16 | 16 |
| | | | | |

pre-trained model

# Conclusion


Relation entre Score et Accuracy
Accuracy vs Score
Pre-trained
ps:0,3|pu:0,2
pu:0,7
ps:0,5|pu:0,5
ps:0,2|pu:0,2
Accuracy (%)
91.25
91.00
90.75
90.50
90.25
90.00
89.75
89.50
0.5
1.0
1.5
2.0
2.5
3.0
Score

To do:

- Present different REsnet architecture
- Trade-off between score and accuracies
- Train depthwise on 100 epochs to see the evolution (can’t track with wandb but can do an inference)