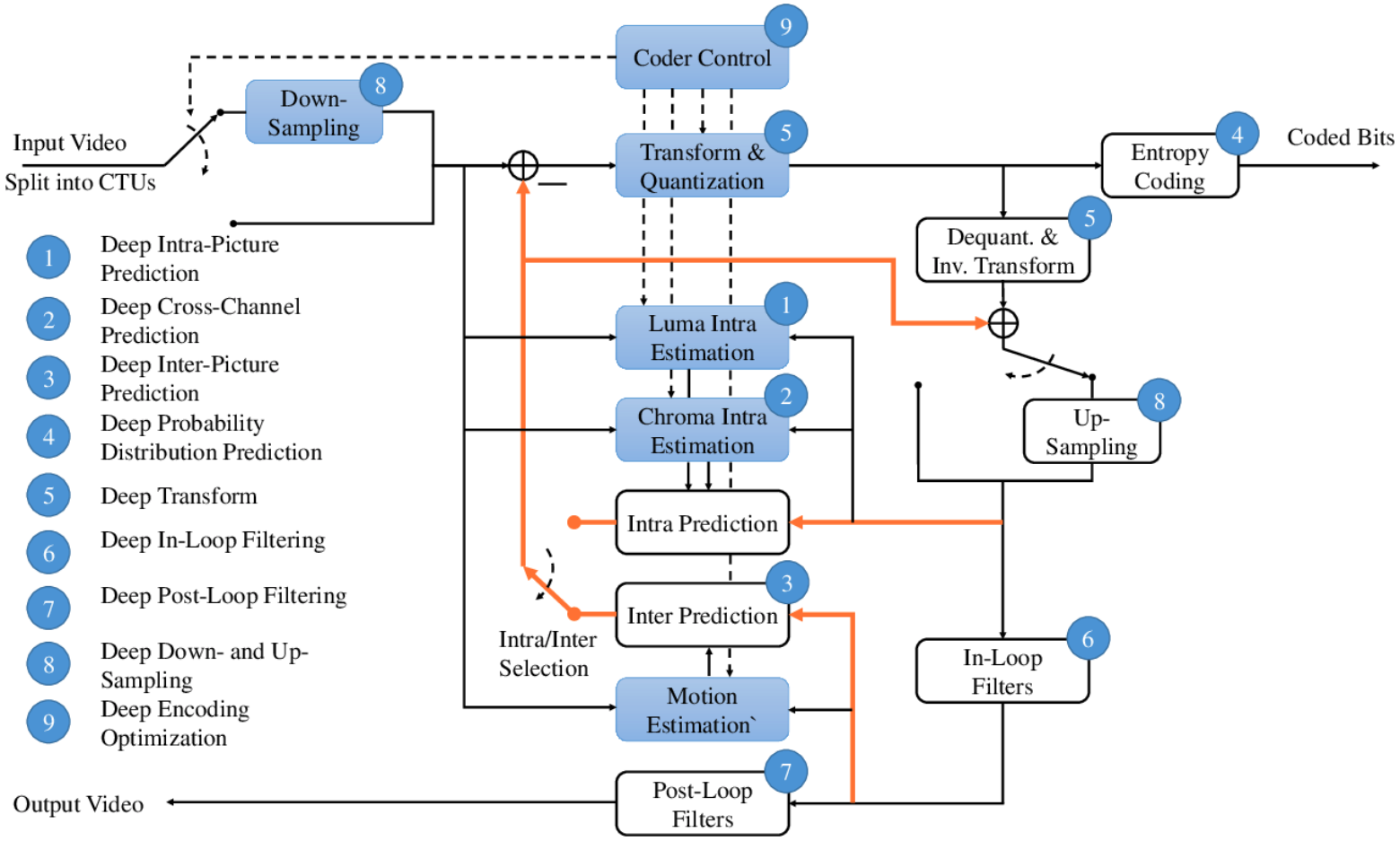
Towards Efficiency and Information Filtering by Pruning for Deep Neural Networks

Journée du GDR-IASIS
Soutenabilité et empreinte carbone de la chaine vidéo
13/03/2025

Enzo Tartaglione
Maître de Conférences, équipe MM, dept.IDS, LTCI
Hi!PARIS chair holder

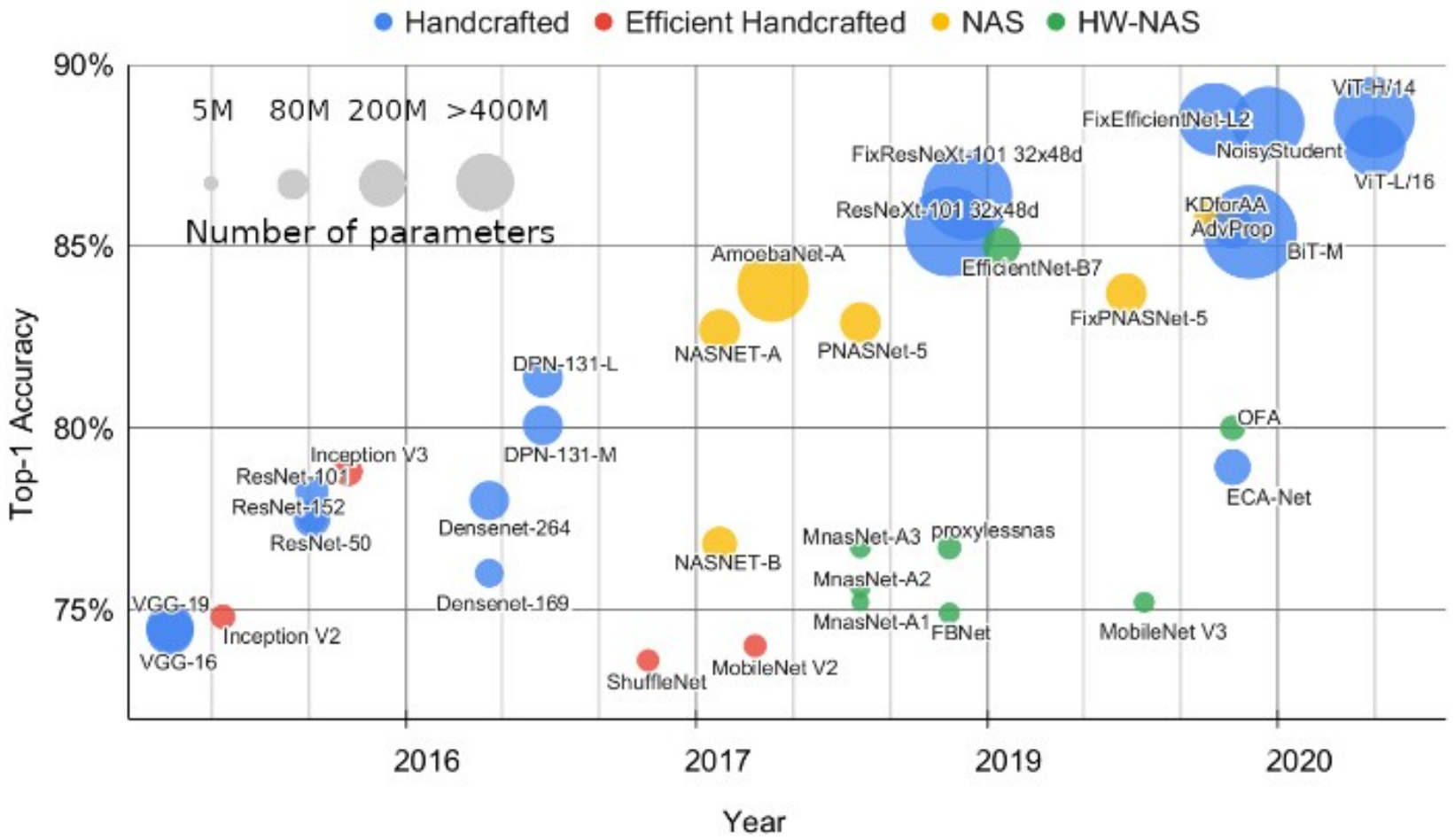
enzo.tartaglione@telecom-paris.fr

Why Deep Learning in Video Coding?



Liu, Dong et al. “Deep Learning-Based Video Coding.” ACM Computing Surveys (CSUR) 53 (2019): 1 - 35.

Less recent trends in ANNs



More recent trends in ANNs

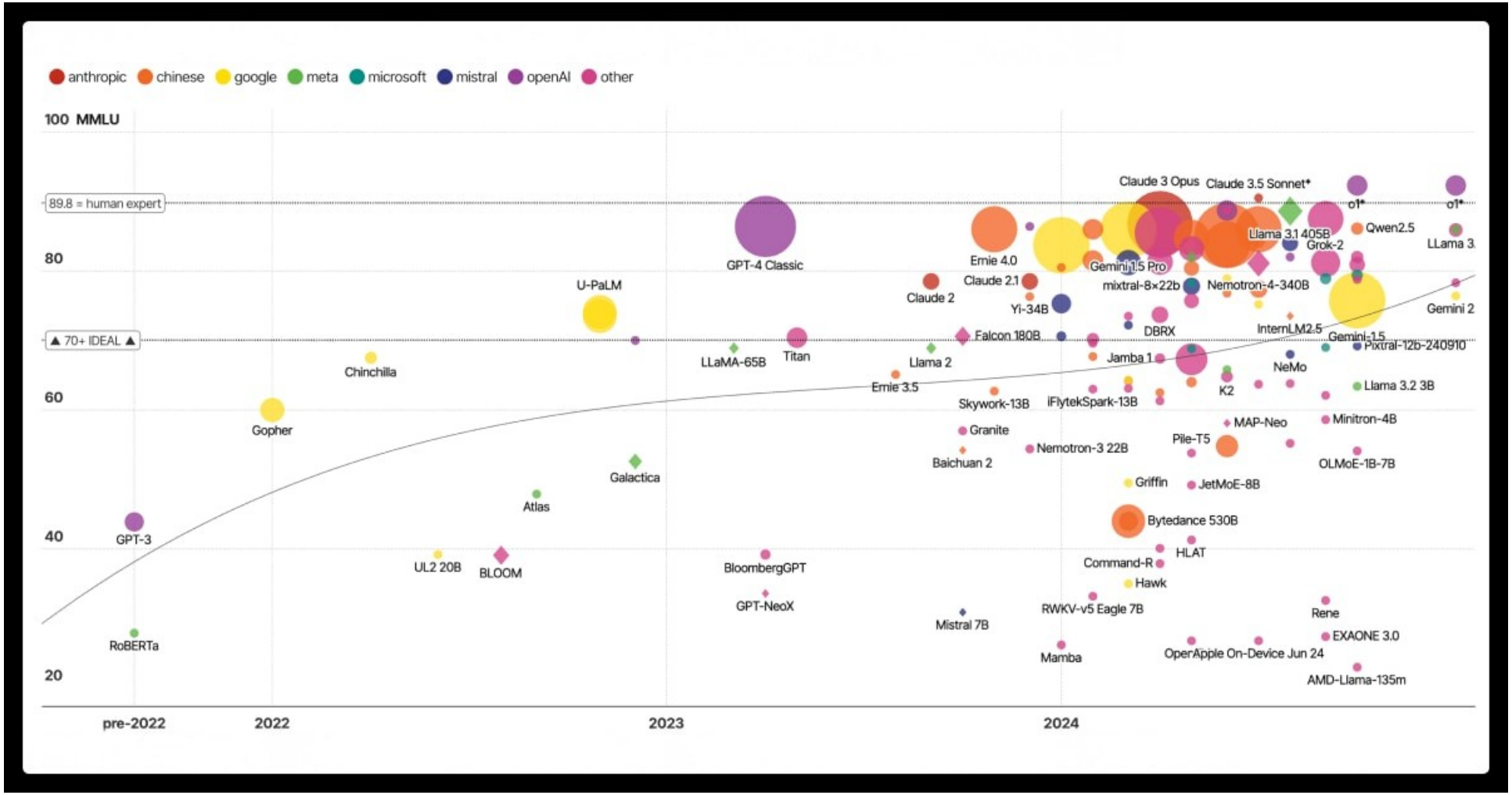


Image from https://labelyourdata.com/articles/llm-model-size

Frugality in AI

- **Perform training with little data**
  - Better optimizers
  - New loss functions
  - Ease the features extraction with priors
  - Transfer learning
  - …

Frugality in AI

- **Perform training with little data**
  - Better optimizers
  - New loss functions
  - Ease the features extraction with priors
  - Transfer learning
  - …
- **Use the trained AI models with little computational resources**
  - Pruning & Quantization
  - Knowledge distillation
  - Neural Architecture Search
  - …

Frugality in AI

- **Perform training with little data**
  - Better optimizers
  - New loss functions
  - Ease the features extraction with priors
  - Transfer learning
  - …
- **Use the trained AI models with little computational resources**
  - Pruning & Quantization
  - Knowledge distillation
  - Neural Architecture Search
  - …
- **Train with little computational resources**
  - Ensure fast convergence for models
  - Reduce the training cost
  - …

Pruning: a definition

- With pruning we refer to the process of removing parameters (synapses) or entire units from the deep learning model.

Pruning: a definition

- With pruning we refer to the process of removing parameters (synapses) or entire units from the deep learning model.

- Pruning relates with **sparsification**: the weight matrix (representing a layer) becomes, indeed, sparse!

Pruning: a definition

- With pruning we refer to the process of removing parameters (synapses) or entire units from the deep learning model.
- Pruning relates with **sparsification**: the weight matrix (representing a layer) becomes, indeed, sparse!
- The removed parameters (if they are removed in an “unstructured” way) still need to be encoded, with a “0”.
- Hence, in general, the position with a “missing” connection still needs to be encoded in some way (while using general frameworks), producing some representation overhead.
- Do we have advantages with a pruned architecture?

Pruning: a definition

- With pruning we refer to the process of removing parameters (synapses) or entire units from the deep learning model.
- Pruning relates with **sparsification**: the weight matrix (representing a layer) becomes, indeed, sparse!
- The removed parameters (if they are removed in an “unstructured” way) still need to be encoded, with a “0”.
- Hence, in general, the position with a “missing” connection still needs to be encoded in some way (while using general frameworks), producing some representation overhead.
- Do we have advantages with a pruned architecture?
  - The number of parameters is reduced. For special designs (like ASICs) the gain is real.

Pruning: a definition

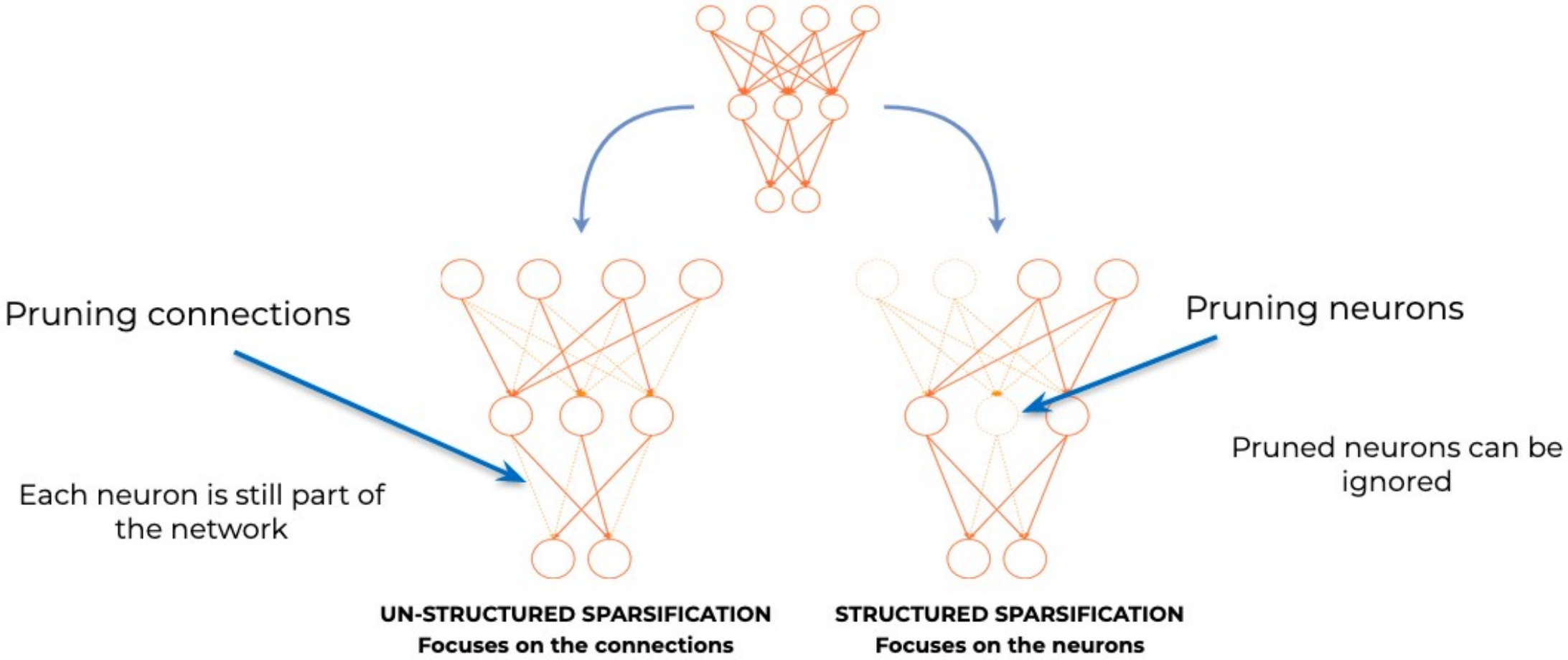
- With pruning we refer to the process of removing parameters (synapses) or entire units from the deep learning model.
- Pruning relates with **sparsification**: the weight matrix (representing a layer) becomes, indeed, sparse!
- The removed parameters (if they are removed in an “unstructured” way) still need to be encoded, with a “0”.
- Hence, in general, the position with a “missing” connection still needs to be encoded in some way (while using general frameworks), producing some representation overhead.
- Do we have advantages with a pruned architecture?
  - The number of parameters is reduced. For special designs (like ASICs) the gain is real.
  - If they are removed in a “structured” way (entire blocks), there is no representation overhead, and the gain is real even in general frameworks!

Pruning in deep learning: why?

- Reducing the storage memory (for a compressed model).
- Reducing the memory footprint.
- Reducing the FLOPs at inference time.
- Reducing energy consumption?
- Enhancing generalization?

Countless approaches to achieve sparse models…

Structured vs Unstructured sparsity



Can we prune entire layers?

Layer collapse

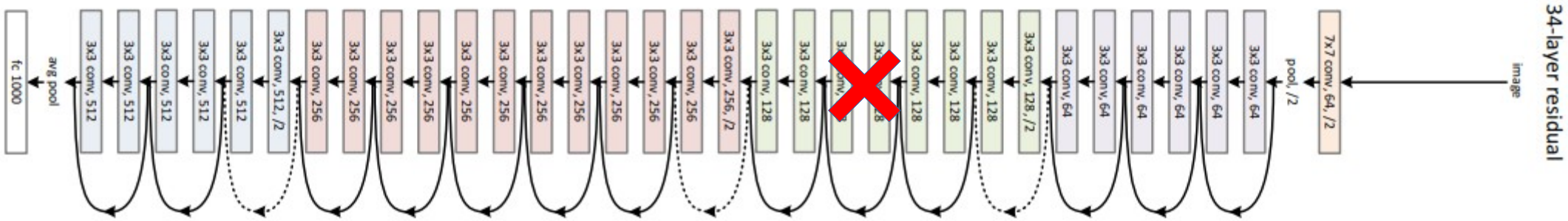
In deep learning, **layer collapse** refers to a phenomenon where a specific layer in a neural network fails to effectively differentiate the input features, causing all outputs from the layer to converge to similar or identical values regardless of input.

Layer collapse

In deep learning, **layer collapse** refers to a phenomenon where a specific layer in a neural network fails to effectively differentiate the input features, causing all outputs from the layer to converge to similar or identical values regardless of input.

Although this effect is traditionally seen as a downside of poor initialization or hyper-parameter fine-tuning (ie. the model does not learn), what if *the model learns, while some layers are collapsed?*

- If that happens, we can completely skip computation of such layer!



Layer collapse- without skip connections?

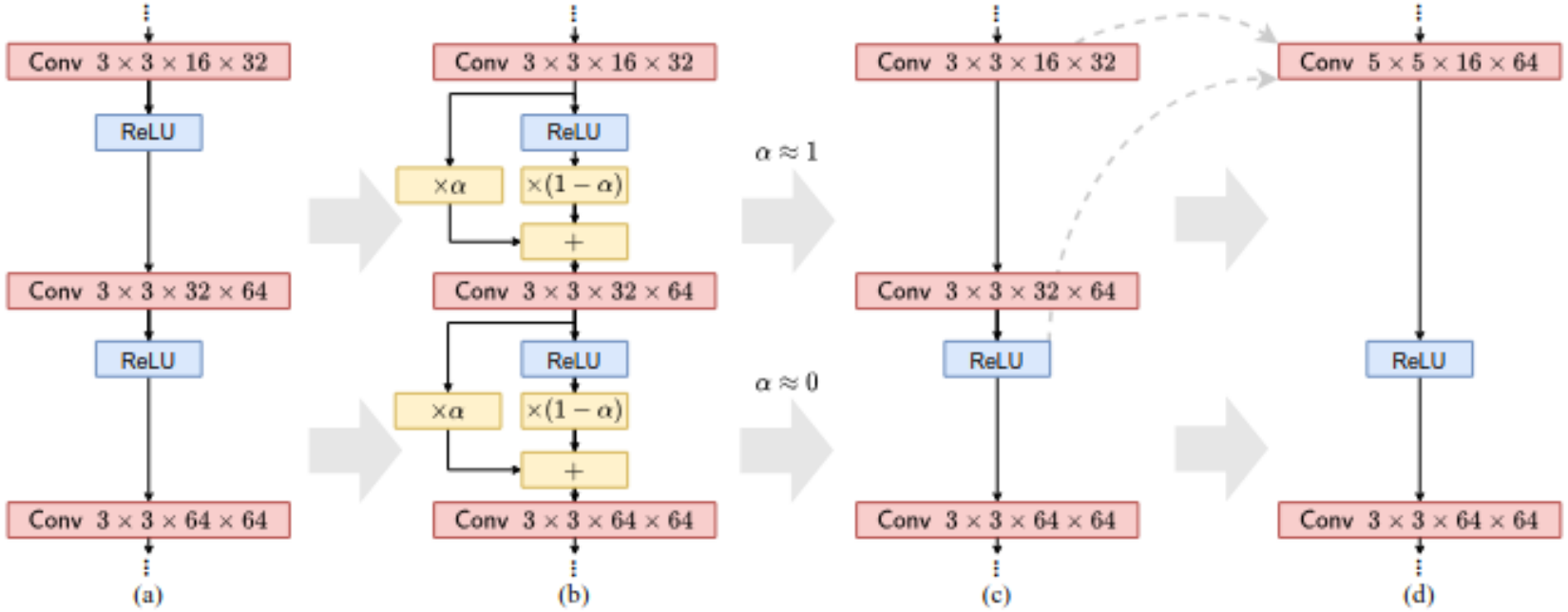
Layer collapse- without skip connections?

Is it really possible?

Layer collapse- without skip connections? [Dror et al. 2021]

Is it really possible?

We can attack the problem differently. One way to reduce the model's depth is to *remove non-linearities!* Layer fold is one approach parametrizing the negative slope of a PReLU.



Is it straightforward? [Pilo et al. 2025]

Case of study: fully-connected layers

$$\boldsymbol{y} = (\boldsymbol{x}W_1^T + b_1)W_2^T + b_2,$$

Pilo, G., Hezbri, N., e Ferreira, A. P., Quétu, V., & Tartaglione, E. (2025). LayerFold: A Python library to reduce the depth of neural networks. SoftwareX, 29, 102030.

Is it straightforward? [Pilo et al. 2025]

Case of study: fully-connected layers

$$\boldsymbol{y} = (\boldsymbol{x}W_1^T + b_1)W_2^T + b_2,$$

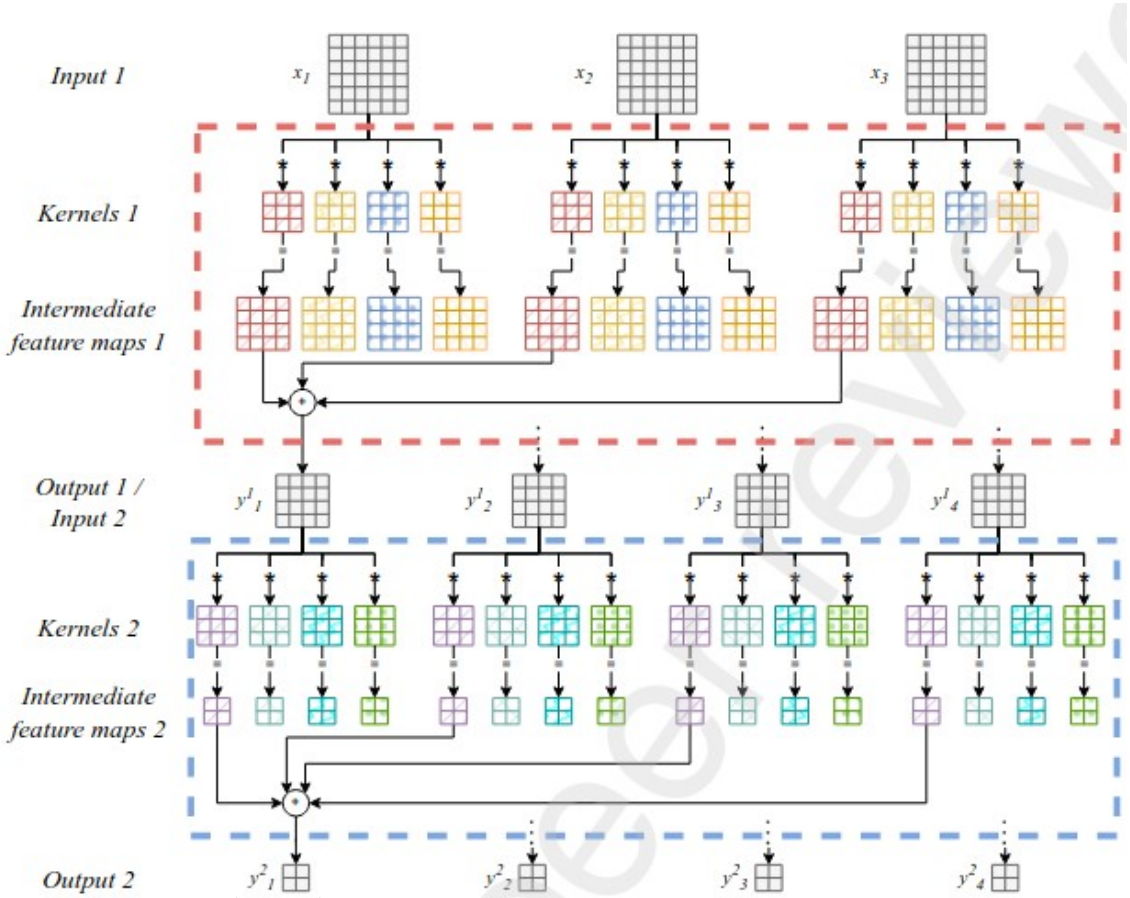
$$W_{eq} = W_2 W_1 \qquad b_{eq} = b_1 W_2^T + b_2.$$

The solution is straightforward.

Pilo, G., Hezbri, N., e Ferreira, A. P., Quétu, V., & Tartaglione, E. (2025). LayerFold: A Python library to reduce the depth of neural networks. SoftwareX, 29, 102030.

Is it straightforward? [Pilo et al. 2025]

Case of study: convolutional layers, stride=1, no padding



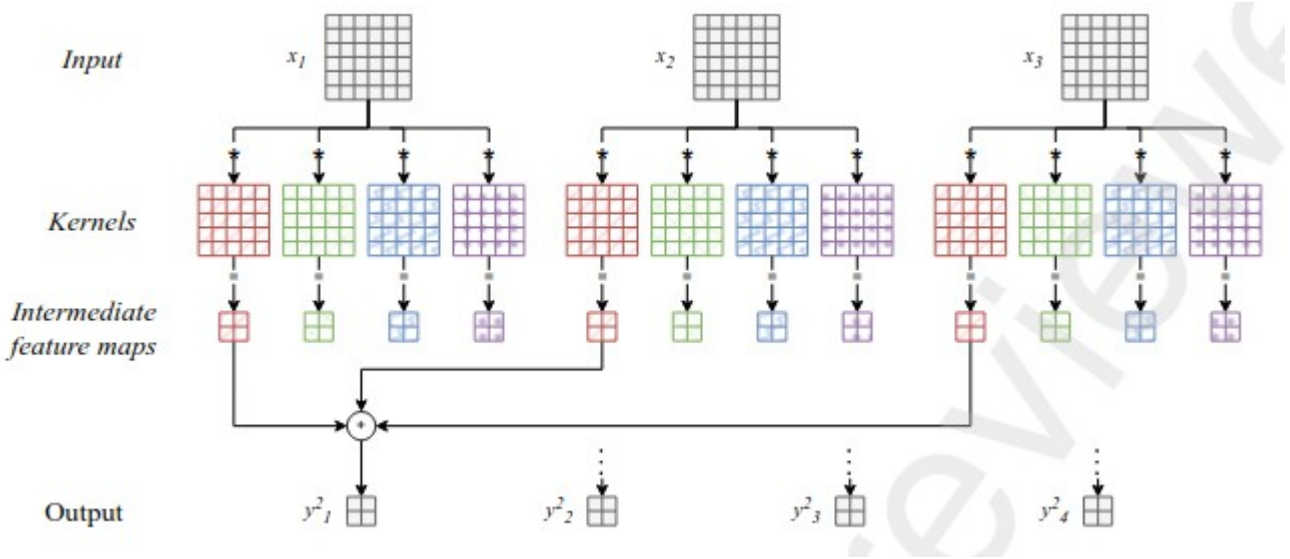
Pilo, G., Hezbri, N., e Ferreira, A. P., Quétu, V., & Tartaglione, E. (2025). LayerFold: A Python library to reduce the depth of neural networks. SoftwareX, 29, 102030.

Is it straightforward? [Pilo et al. 2025]

Case of study: convolutional layers, stride=1, no padding

$$W_{eq} = W_1 * W_2$$

$$b_{eq,j} = b_{2,j} + \sum_{i=1}^{c_1} b_{1,i} \sum_{l=1}^{k_2} \sum_{m=1}^{k_2} w_{2,i,j}.$$



The solution is straightforward.

Pilo, G., Hezbri, N., e Ferreira, A. P., Quétu, V., & Tartaglione, E. (2025). LayerFold: A Python library to reduce the depth of neural networks. SoftwareX, 29, 102030.

Is it straightforward? [Pilo et al. 2025]

Case of study: convolutional layers, stride=1, padding=1

Pilo, G., Hezbri, N., e Ferreira, A. P., Quétu, V., & Tartaglione, E. (2025). LayerFold: A Python library to reduce the depth of neural networks. SoftwareX, 29, 102030.

Is it straightforward? [Pilo et al. 2025]

Case of study: convolutional layers, stride=1, padding=1

To the moment, no closed-form solutions!

Is this case that much frequent?

Pilo, G., Hezbri, N., e Ferreira, A. P., Quétu, V., & Tartaglione, E. (2025). LayerFold: A Python library to reduce the depth of neural networks. SoftwareX, 29, 102030.

Is it straightforward? [Pilo et al. 2025]

Case of study: convolutional layers, stride=1, padding=1

To the moment, no closed-form solutions!

Is this case that much frequent?

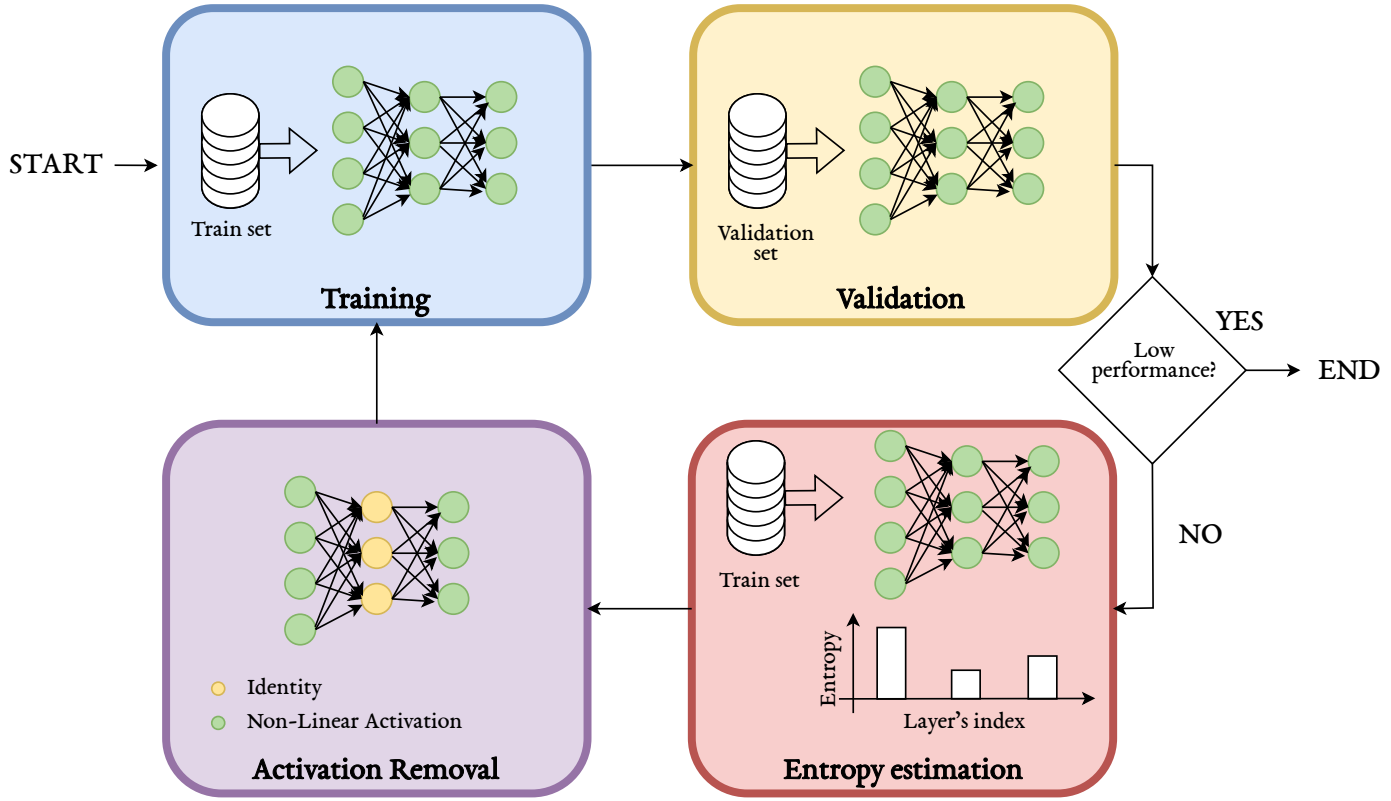
Most of the architectures using skip connections make use of padding to maintain same dimensionality of the output!

Current research is finding ways to bypass this issue….

Pilo, G., Hezbri, N., e Ferreira, A. P., Quétu, V., & Tartaglione, E. (2025). LayerFold: A Python library to reduce the depth of neural networks. SoftwareX, 29, 102030.

OK, how can we remove layers?

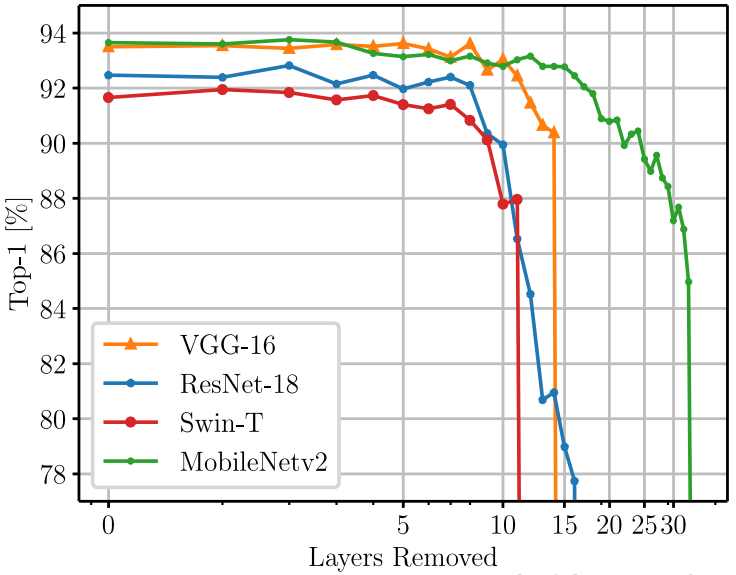
The naive way: EASIER [Quetu et al. 2024]



Quétu, V., Liao, Z., & Tartaglione, E. (2024, August). The simpler the better: An entropy-based importance metric to reduce neural networks' depth. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 92-108). Cham: Springer Nature Switzerland.

The naive way: EASIER [Quetu et al. 2024]

EASIER applied on ResNet-18, VGG-16, Swin-T and MobileNetv2 on CIFAR-10. For each model, we gradually remove non-linear layers.



Quétu, V., Liao, Z., & Tartaglione, E. (2024, August). The simpler the better: An entropy-based importance metric to reduce neural networks' depth. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 92-108). Cham: Springer Nature Switzerland.

The naive way: EASIER [Quetu et al. 2024]

EASIER applied on ResNet-18, VGG-16, Swin-T and MobileNetv2 on CIFAR-10. For each model, we gradually remove non-linear layers.

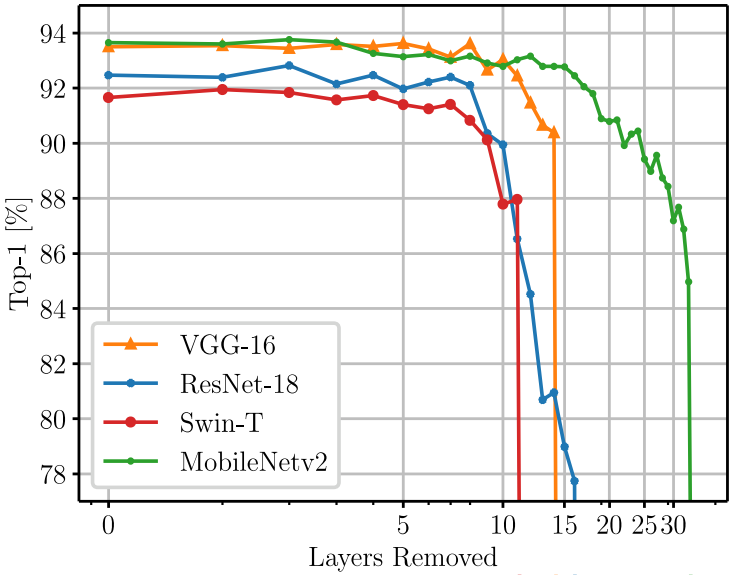
Limitations:

Performance degradation

→ Difficult to compress parameter-efficient architectures that are not overfitting without compromising performance.
For example: Swin-T on Tiny-ImageNet-200.

Training efficiency

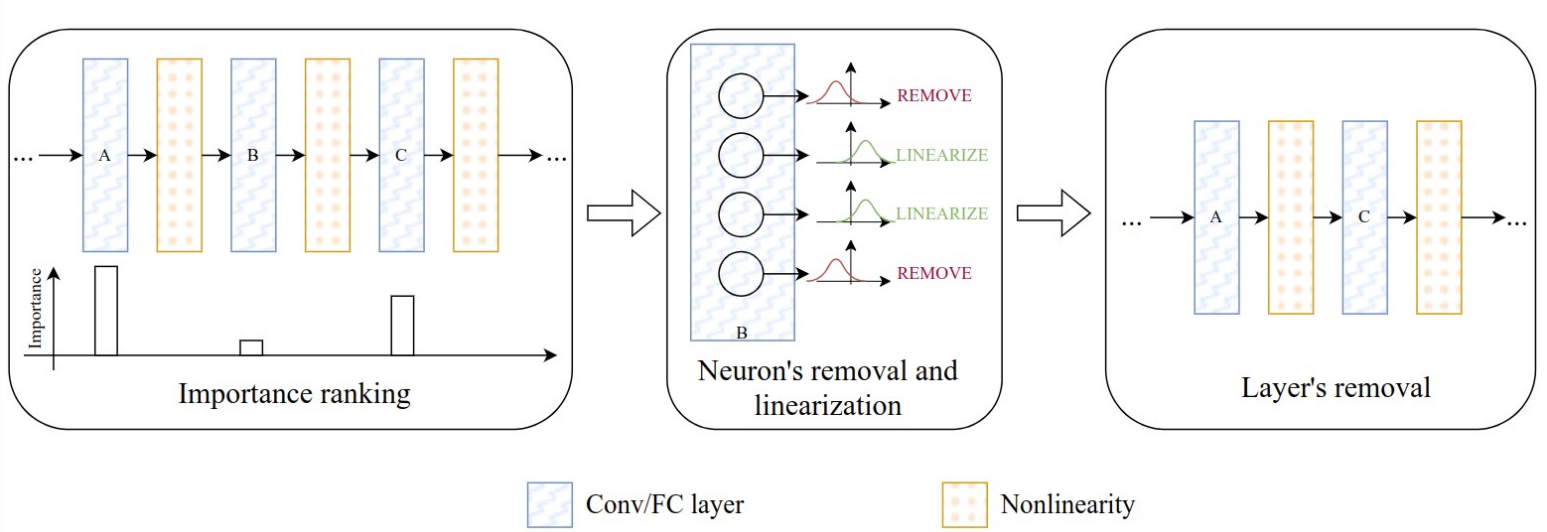
→ Longer training time due to the iterative nature of EASIER.



Models trained on CIFAR-10.

Quétu, V., Liao, Z., & Tartaglione, E. (2024, August). The simpler the better: An entropy-based importance metric to reduce neural networks' depth. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 92-108). Cham: Springer Nature Switzerland.

A better way: TLC [Liao et al. 2025]



Works by simply observing distributions in batch norms and in layer norms.

$$\hat{x}_{l,i} = \frac{x_{l,i} - \mu^B_{l,i}}{\sqrt{(\sigma^B_{l,i})^2 + \epsilon}}; \qquad z_{l,i} = \gamma_{l,i}\hat{x}_{l,i} + \beta_{l,i};$$

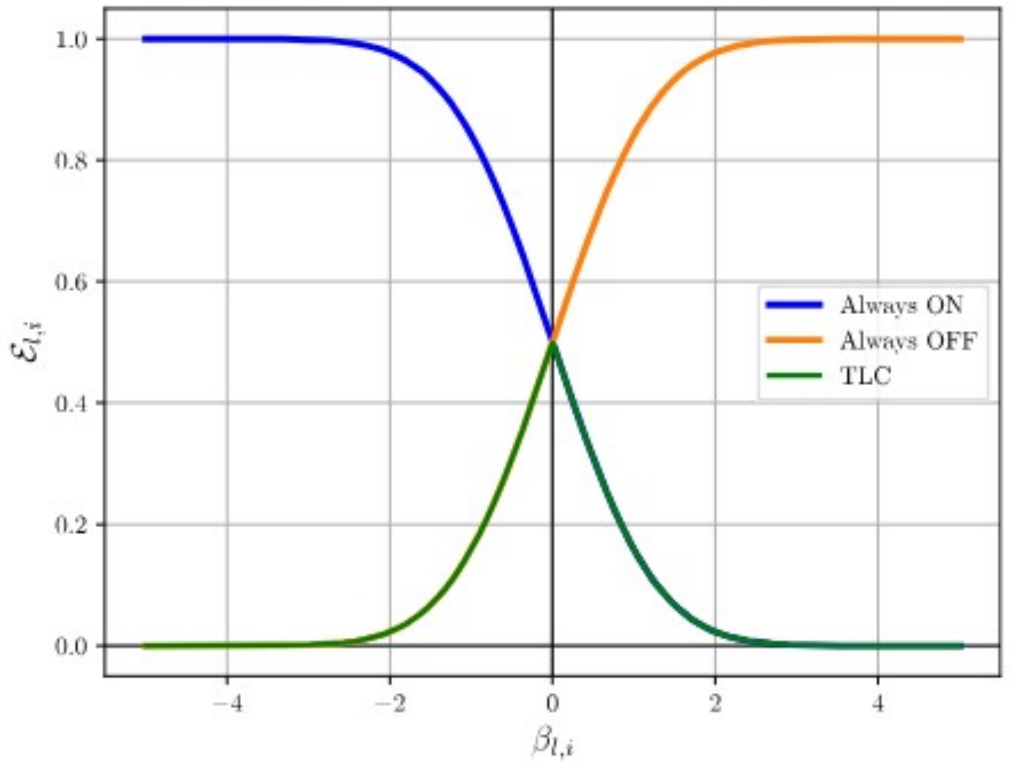
Still, sharing iterative fine-tuning nature, so computationally expensive….

Liao, Z., Hezbri, N., Quétu, V., Nguyen, V. T., & Tartaglione, E. (2025). Till the Layers Collapse: Compressing a Deep Neural Network through the Lenses of Batch Normalization Layers. AAAI 2025.

A better way: TLC [Liao et al. 2025]

$$\hat{x}_{l,i} = \frac{x_{l,i} - \mu^B_{l,i}}{\sqrt{(\sigma^B_{l,i})^2 + \epsilon}}; \qquad z_{l,i} = \gamma_{l,i}\hat{x}_{l,i} + \beta_{l,i};$$

$$\mathcal{E}_{l,i} = \Phi\left(-\frac{|\beta_{l,i}|}{\gamma_{l,i}}\right)$$



Liao, Z., Hezbri, N., Quétu, V., Nguyen, V. T., & Tartaglione, E. (2025). Till the Layers Collapse: Compressing a Deep Neural Network through the Lenses of Batch Normalization Layers. AAAI 2025.

Browsing the literature…

"Deeper layers are typically quite similar to each other and can be more easily dropped."

The Unreasonable Ineffectiveness of the Deeper Layers, ICLR 2024

Browsing the literature…

"Deeper layers are typically quite similar to each other and can be more easily dropped."

The Unreasonable Ineffectiveness of the Deeper Layers, ICLR 2024

"The changes in parameters and output representations between adjacent layers within the LLM are not particularly significant."

LaCo: Large Language Model Pruning via Layer Collapse, ICLR 2024

Browsing the literature…

"Deeper layers are typically quite similar to each other and can be more easily dropped."

The Unreasonable Ineffectiveness of the Deeper Layers, ICLR 2024

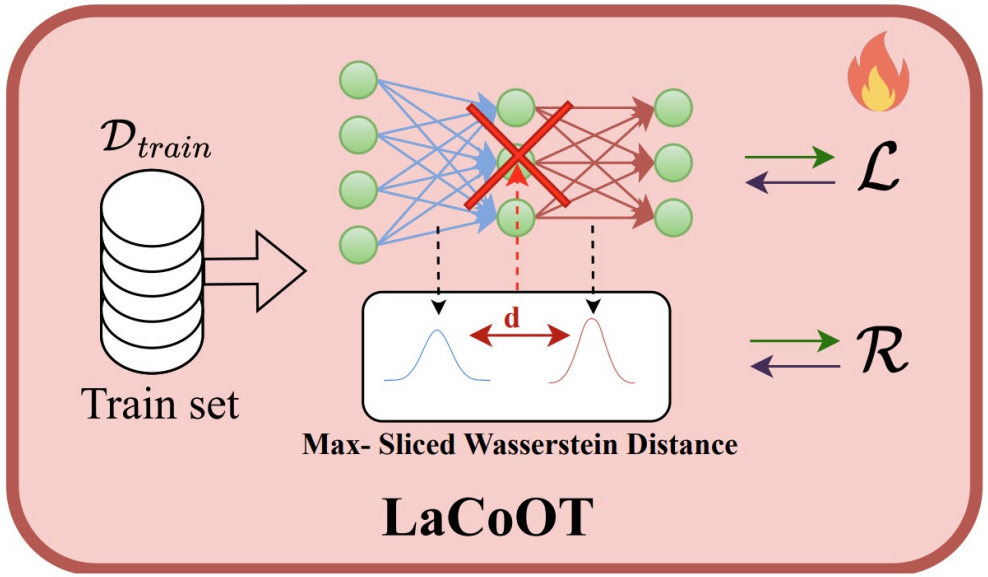
"The changes in parameters and output representations between adjacent layers within the LLM are not particularly significant."

LaCo: Large Language Model Pruning via Layer Collapse, ICLR 2024

"LLMs exhibit significant redundancy at the layer level, enabling the simple removal of entire layers without substantially impacting downstream task performance."

ShortGPT: Layers in Large Language Models are More Redundant Than You Expect, ICLR 2024

Layer Collapse by Regularization [Quetu et al. 2024]

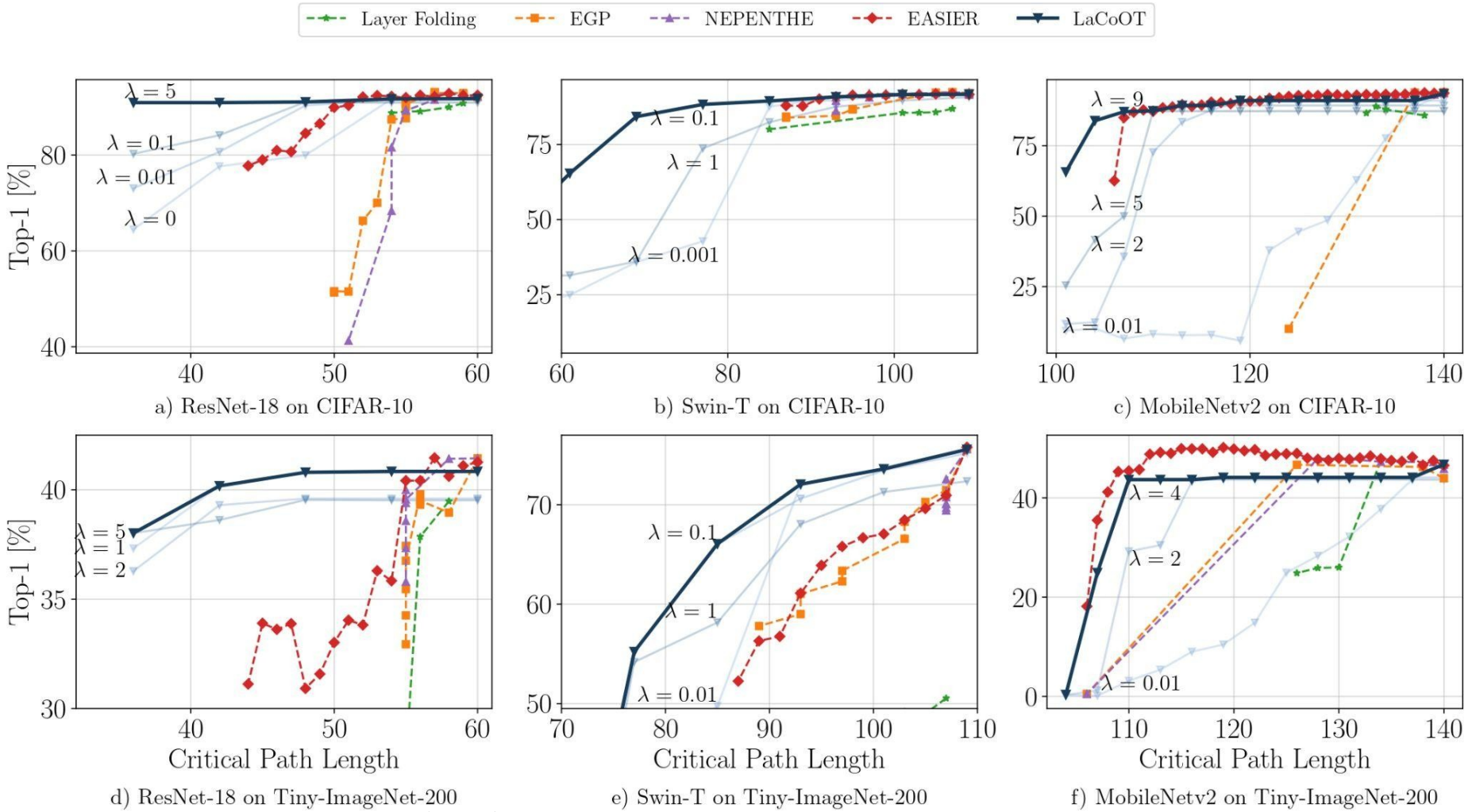


Training performed by minimizing distribution distance between adjacent layers (or group of) through a regularization term.

Layer pruning is performed in one-shot without retraining.

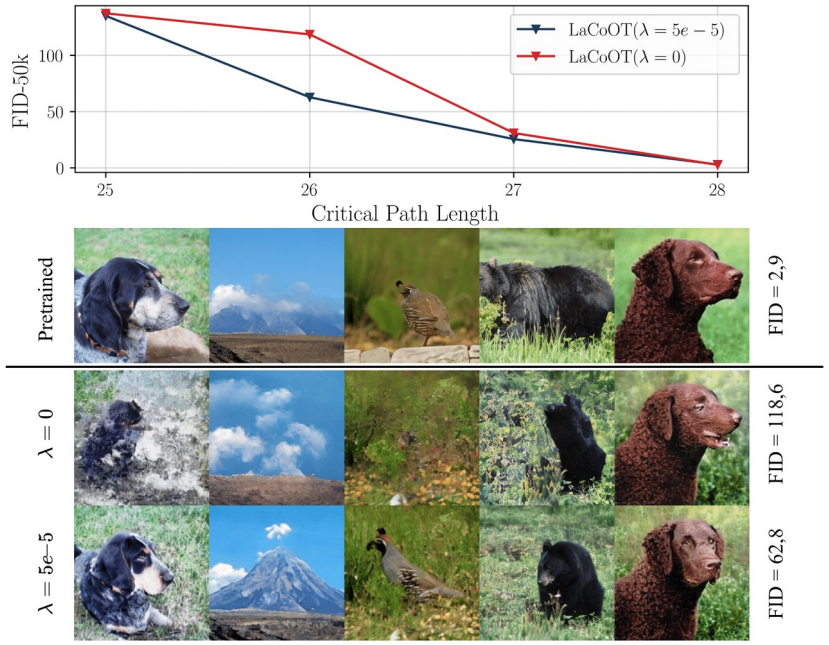
Quétu, V., Hezbri, N., & Tartaglione, E. (2024). LaCoOT: Layer Collapse through Optimal Transport. arXiv preprint arXiv:2406.08933.

Layer Collapse by Regularization [Quetu et al. 2024]



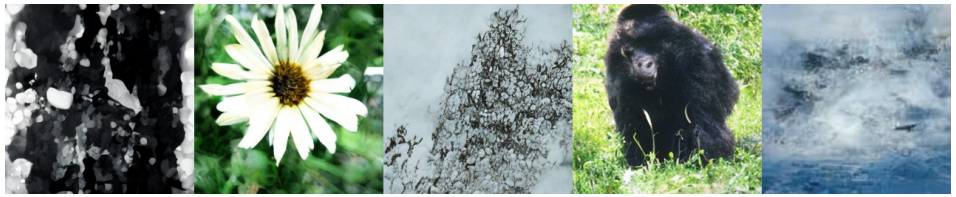
Quétu, V., Hezbri, N., & Tartaglione, E. (2024). LaCoOT: Layer Collapse through Optimal Transport. arXiv preprint arXiv:2406.08933.

Layer Collapse by Regularization [Quetu et al. 2024]



FID-50k as a function of the critical path length achieved by a DiT-XL/2 finetuned on ImageNet.

Pretrained DiT-XL/2



DiT-XL/2 finetuned with λ=0 – 2 blocks removed

DiT-XL/2 finetuned with λ=5e-5 – 2 blocks removed

Quétu, V., Hezbri, N., & Tartaglione, E. (2024). LaCoOT: Layer Collapse through Optimal Transport. arXiv preprint arXiv:2406.08933.

Adapters

Foundation models

A **foundation model** is an AI model that is trained on broad data such that it can be applied across a wide range of use cases.

Foundation models are general-purpose technologies that can support a diverse range of use cases. Building foundation models is often highly resource-intensive (GPT-n, BERT, DALL-E,...).

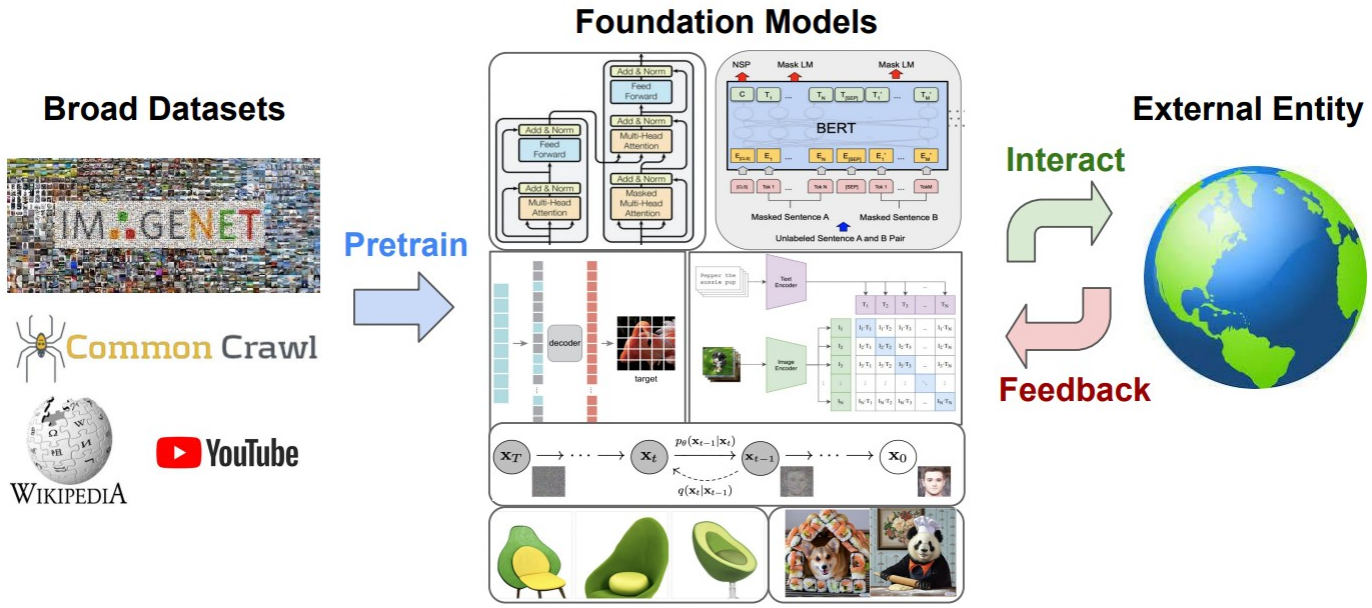


Image from https://www.marktechpost.com/

Foundation models

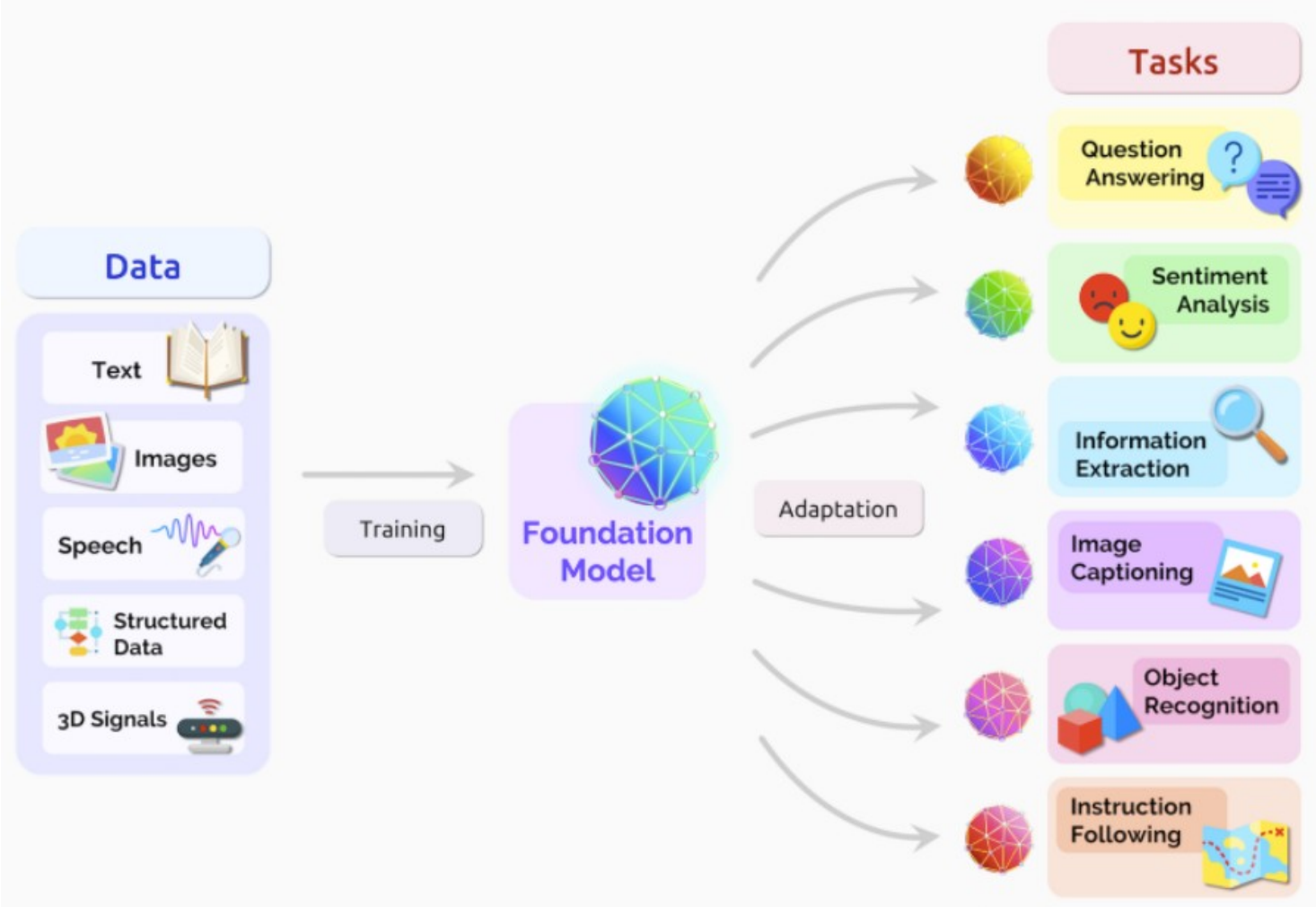


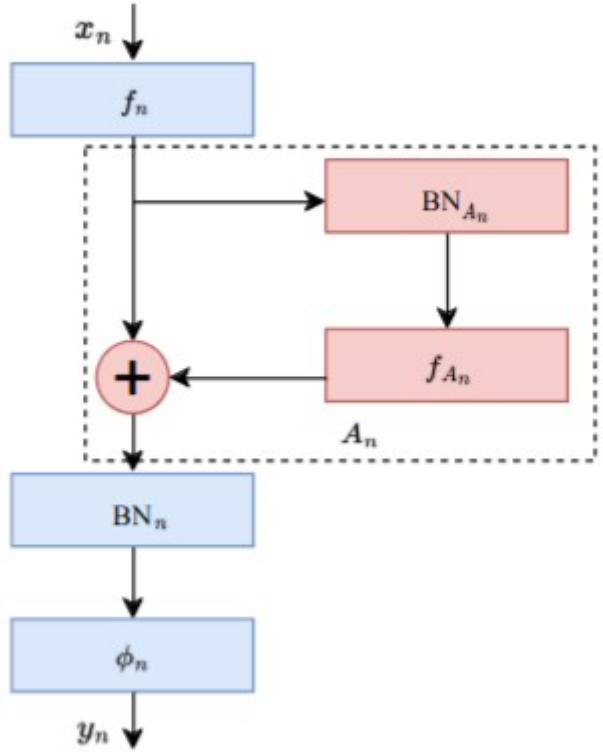
Image from https://billtcheng2013.medium.com/

What is an adapter?

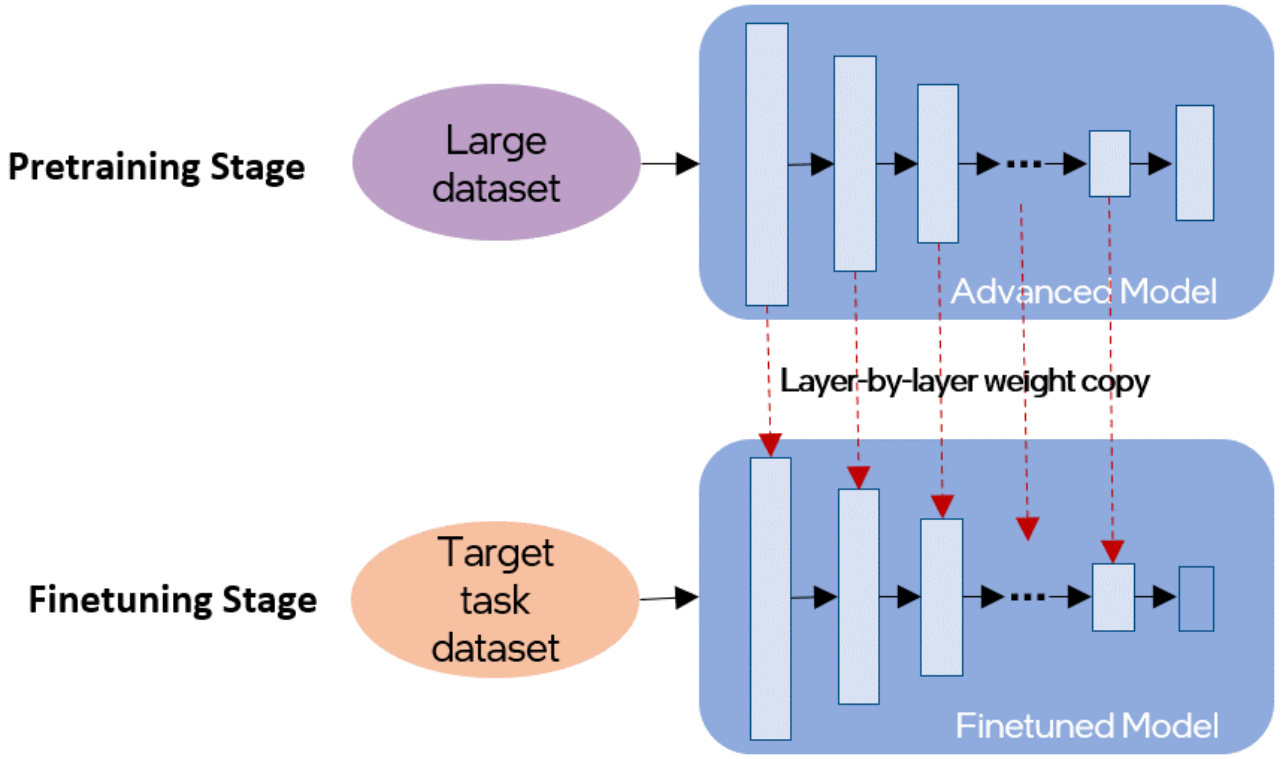
Adapters are **lightweight** alternatives to fully fine-tuned pre-trained models.

Currently, adapters are implemented as small feedforward neural networks that are inserted between layers of a pre-trained model.

They provide a **parameter-efficient**, **computationally efficient,** and modular approach to transfer learning.



Difference with traditional transfer learning



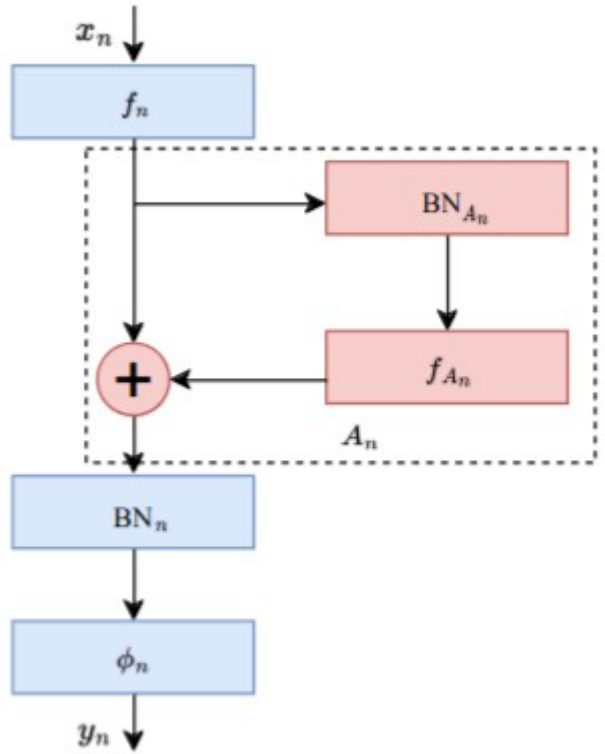
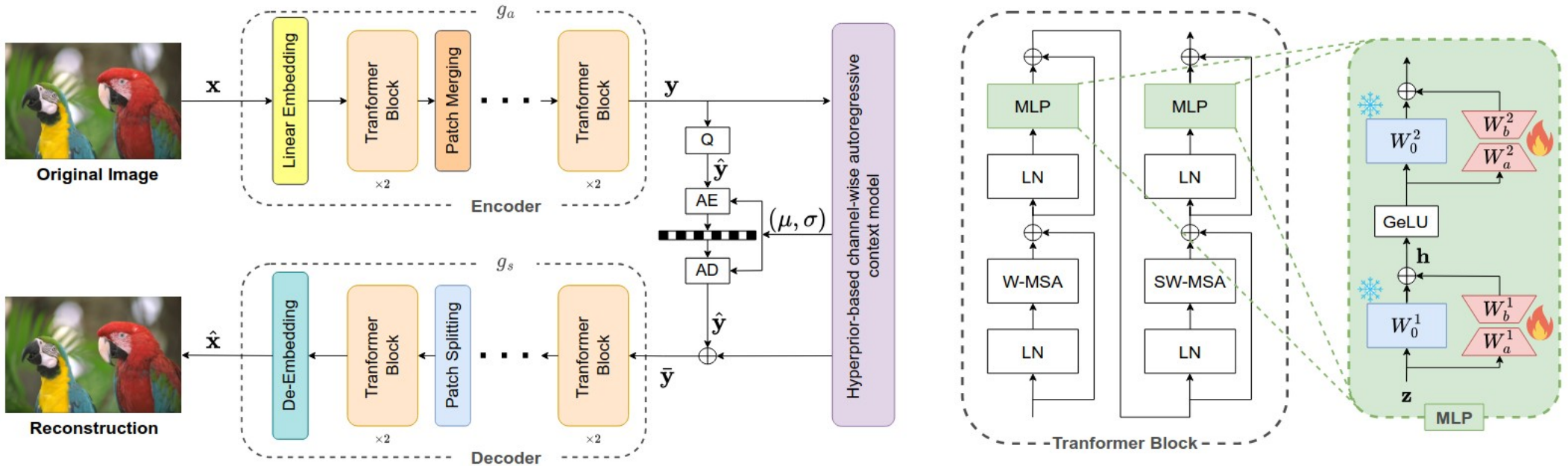


Image from https://community.intel.com

Adapt your Learnable Image Compression modEl for variable bitrates

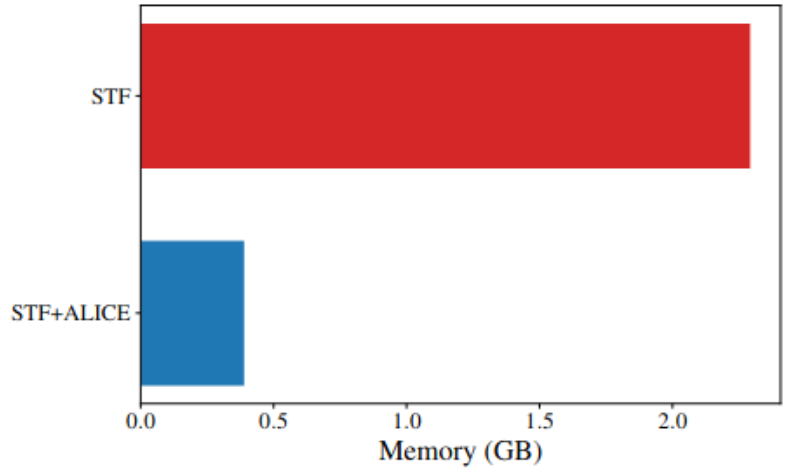
- Idea: instead of training multiple architectures for every rate-distorsion point, why not training a set of adapters that learn how to filter information (ie. starting from a low distorsion model)?
- These adapters can be designed to be linear modules, ie. they can be merged within the original model, causing no inference overhead.



Spadaro, G., Ali, M. S., Presta, A., Pilo, G., Bae, S. H., Giraldo, J. H., ... & Tartaglione, E. (2024, December). ALICE: Adapt your Learnable Image Compression modEl for variable bitrates. In 2024 IEEE International Conference on Visual Communications and Image Processing (VCIP) (pp. 1-5). IEEE.

Adapt your Learnable Image Compression modEl for variable bitrates

- Advantage: a lot of storage memory saved (only one model saved, plus set of adapters for various bitrates) at no computation overhead.

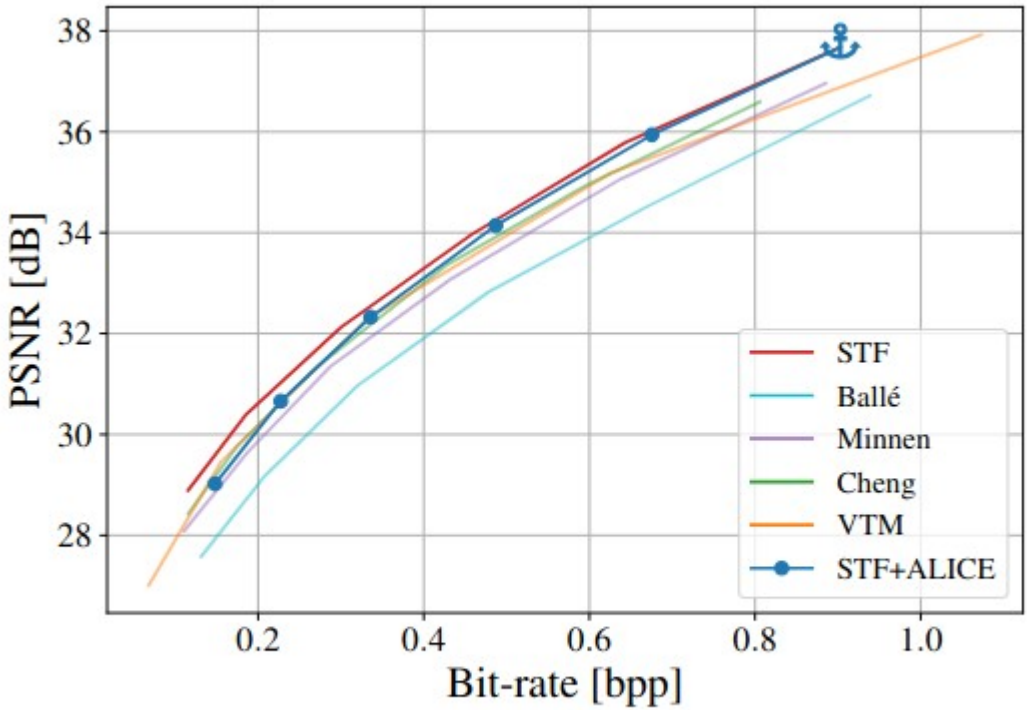


Model		Inference time (s) NVIDIA RTX 2080	NVIDIA Quadro P400	Jetson Orin Nano	Raspberry Pi 5	FLOPs (G)	Memory (GB)
STF	Encoder	$0.155_{\pm 0.002}$	$1.076_{\pm 0.002}$	$0.620_{\pm 0.002}$	$14.573_{\pm 0.066}$	286.377	3.572
	Decoder	$0.205_{\pm 0.002}$	$1.221_{\pm 0.001}$	$0.758_{\pm 0.001}$	$16.469_{\pm 0.123}$	323.711	3.748
STF + ALICE	Encoder	$0.166_{\pm 0.002}$	$1.207_{\pm 0.002}$	$0.691_{\pm 0.003}$	$16.084_{\pm 0.064}$	290.188	4.918
	Decoder	$0.211_{\pm 0.001}$	$1.351_{\pm 0.001}$	$0.816_{\pm 0.003}$	$18.312_{\pm 0.196}$	327.521	5.102
STF + MALICE	Encoder	$0.157_{\pm 0.001}$	$1.075_{\pm 0.001}$	$0.621_{\pm 0.002}$	$14.723_{\pm 0.079}$	286.377	3.529
	Decoder	$0.206_{\pm 0.001}$	$1.221_{\pm 0.002}$	$0.757_{\pm 0.003}$	$16.701_{\pm 0.121}$	323.711	3.708

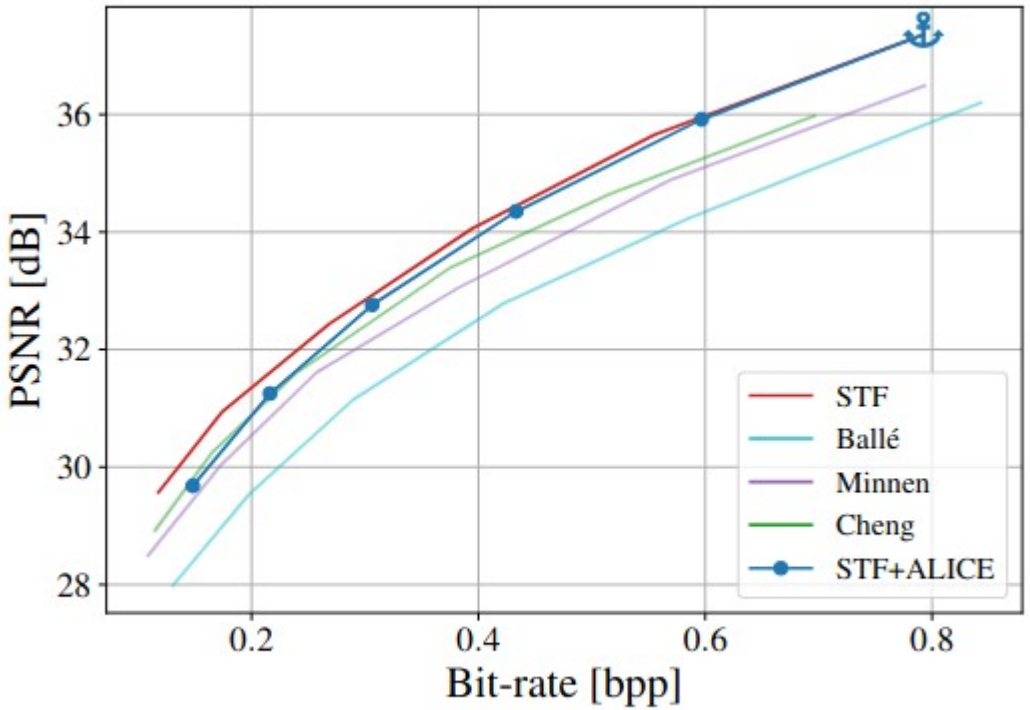
Spadaro, G., Ali, M. S., Presta, A., Pilo, G., Bae, S. H., Giraldo, J. H., ... & Tartaglione, E. (2024, December). ALICE: Adapt your Learnable Image Compression modEl for variable bitrates. In 2024 IEEE International Conference on Visual Communications and Image Processing (VCIP) (pp. 1-5). IEEE.

Adapt your Learnable Image Compression modEl for variable bitrates

- Limit: we are upper-bound by the performance of the original backbone.



(a) Experiments on the Kodak dataset.

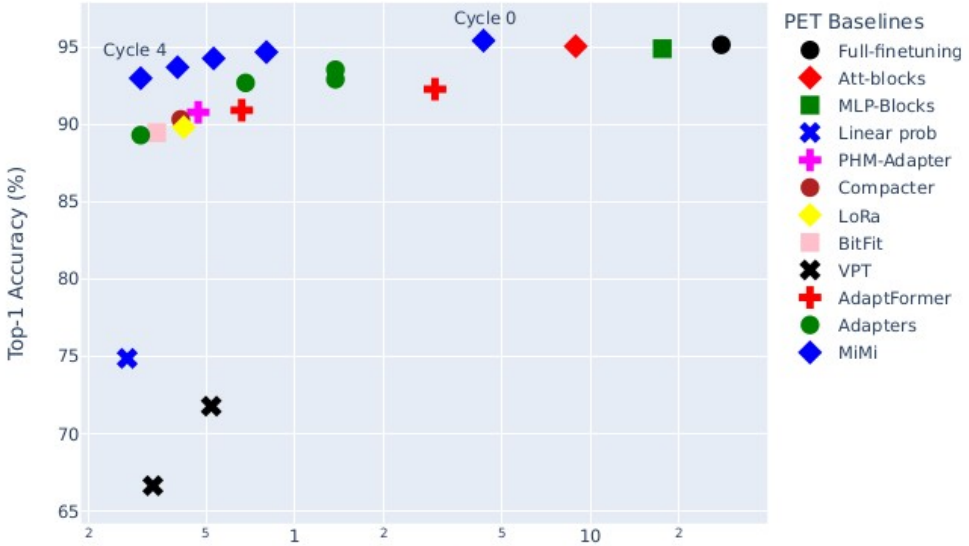


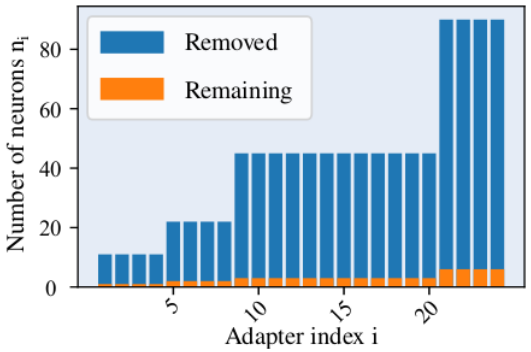
(b) Experiments on the CLIC dataset.

Spadaro, G., Ali, M. S., Presta, A., Pilo, G., Bae, S. H., Giraldo, J. H., ... & Tartaglione, E. (2024, December). ALICE: Adapt your Learnable Image Compression modEl for variable bitrates. In 2024 IEEE International Conference on Visual Communications and Image Processing (VCIP) (pp. 1-5). IEEE.

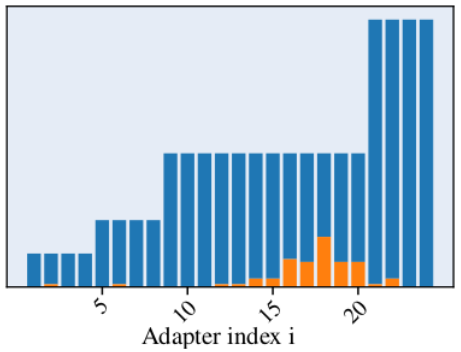
Can we customize the size of the adapters?

- With a proper sizing strategy, it is possible to **properly size by pruning** the size of the adapters as a function of the target task.
- Different task require a different allocation of adapter's complexity.
- The general model does not change. This means that it is possible to store multiple sets of adapters per different tasks with **minimal memory occupancy**.

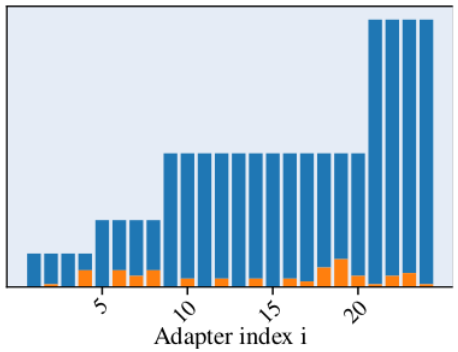




(a) Local pruning: all datasets.



(b) Global pruning: VGG-Flowers.



(c) Global pruning: CIFAR-10.

Marouf, I. E., Tartaglione, E., & Lathuilière, S. (2024). Mini but mighty: Finetuning vits with mini adapters. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (pp. 1732-1741).

Another dimension: token pruning

Why token pruning?

- Tokens process information in parallel in transformers.
- During forward propagation, it can happen that some encode exactly the same information.
- Removing redundant token saves memory and computation.

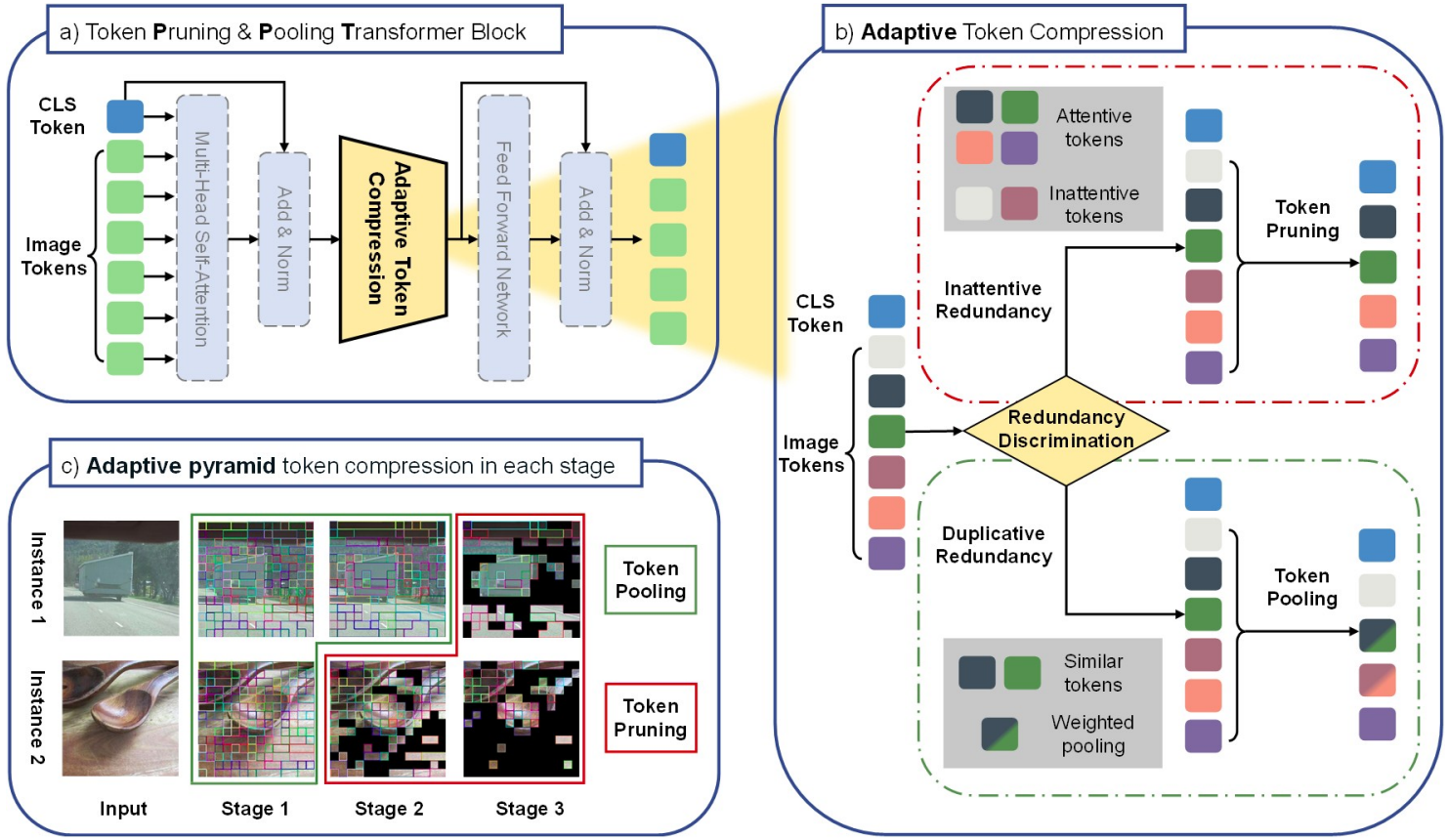
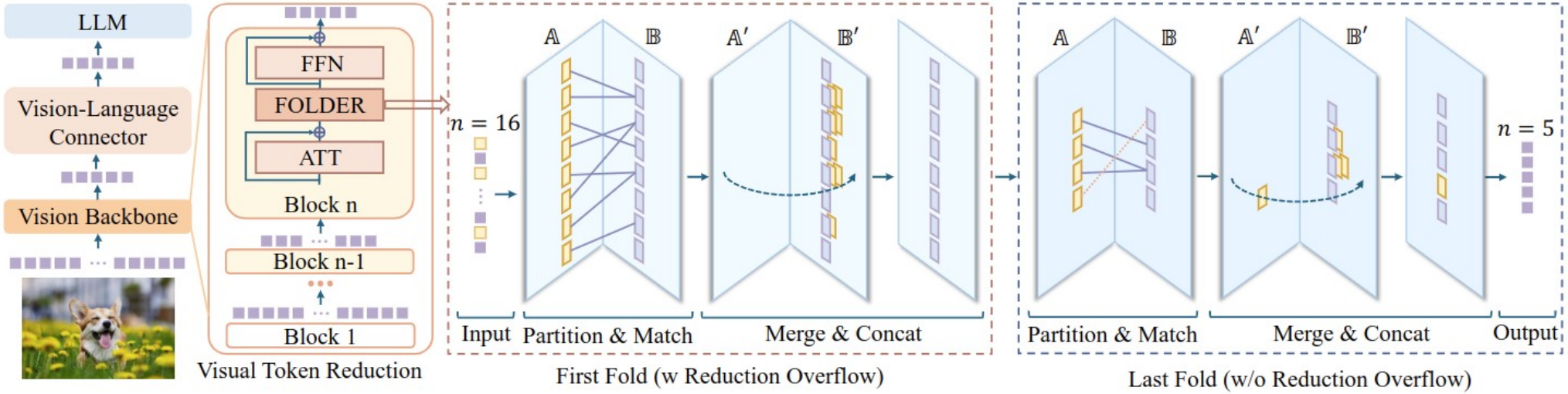


Image from https://paperswithcode.com/paper/ppt-token-pruning-and-pooling-for-efficient

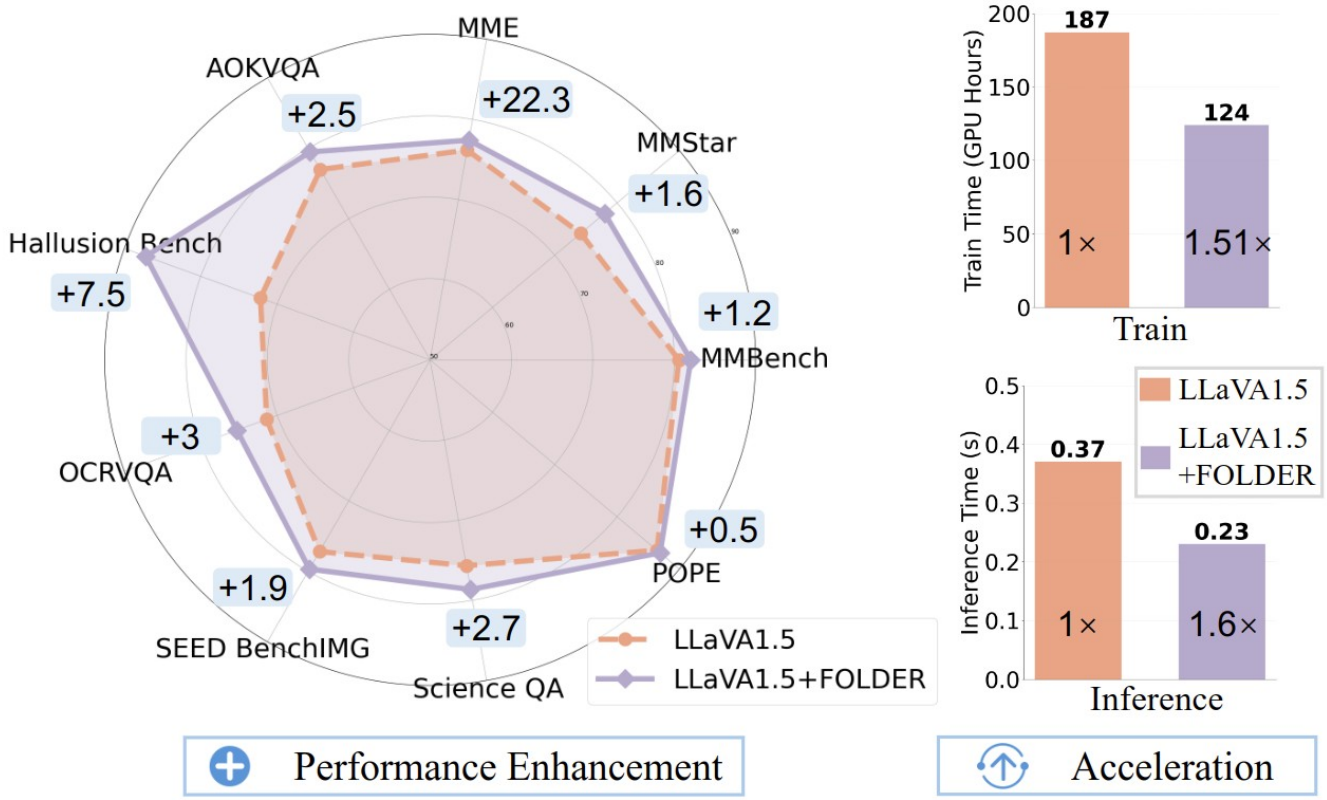
FOLDER [Wang et al. 2025]



As a plug-and-play module for speeding-up training and inference, FOLDER is integrated into the final blocks of the vision backbone (last two here). To deal with reduction overflow, FOLDER automatically executes another FOLD operation when the expected reduction is more than half. The last FOLD, which escapes from reduction overflow, merges tokens according to the remaining reduction numbers.

Wang, H., Yu, Z., Spadaro, G., Ju, C., Quétu, V., & Tartaglione, E. (2025). FOLDER: Accelerating Multi-modal Large Language Models with Enhanced Performance. arXiv preprint arXiv:2501.02430.

FOLDER [Wang et al. 2025]



Wang, H., Yu, Z., Spadaro, G., Ju, C., Quétu, V., & Tartaglione, E. (2025). FOLDER: Accelerating Multi-modal Large Language Models with Enhanced Performance. arXiv preprint arXiv:2501.02430.

How can all of this applied to video coding?

- All the pruning strategies presented can be directly applied to Deep Learning-based approaches present in the coding pipeline.

How can all of this applied to video coding?

- All the pruning strategies presented can be directly applied to Deep Learning-based approaches present in the coding pipeline.

- Variable rate applications are among the most encouraging ones for dynamically reduce the size of the model when latency, bandwidth, or energy are heavily constrained.

How can all of this applied to video coding?

- All the pruning strategies presented can be directly applied to Deep Learning-based approaches present in the coding pipeline.
- Variable rate applications are among the most encouraging ones for dynamically reduce the size of the model when latency, bandwidth, or energy are heavily constrained.
- Training models individually to optimize each rate-distorsion point appears a less-necessary effort: adapters are already working well, and integrating pruning to further reduce the complexity seems promising.


Q&A