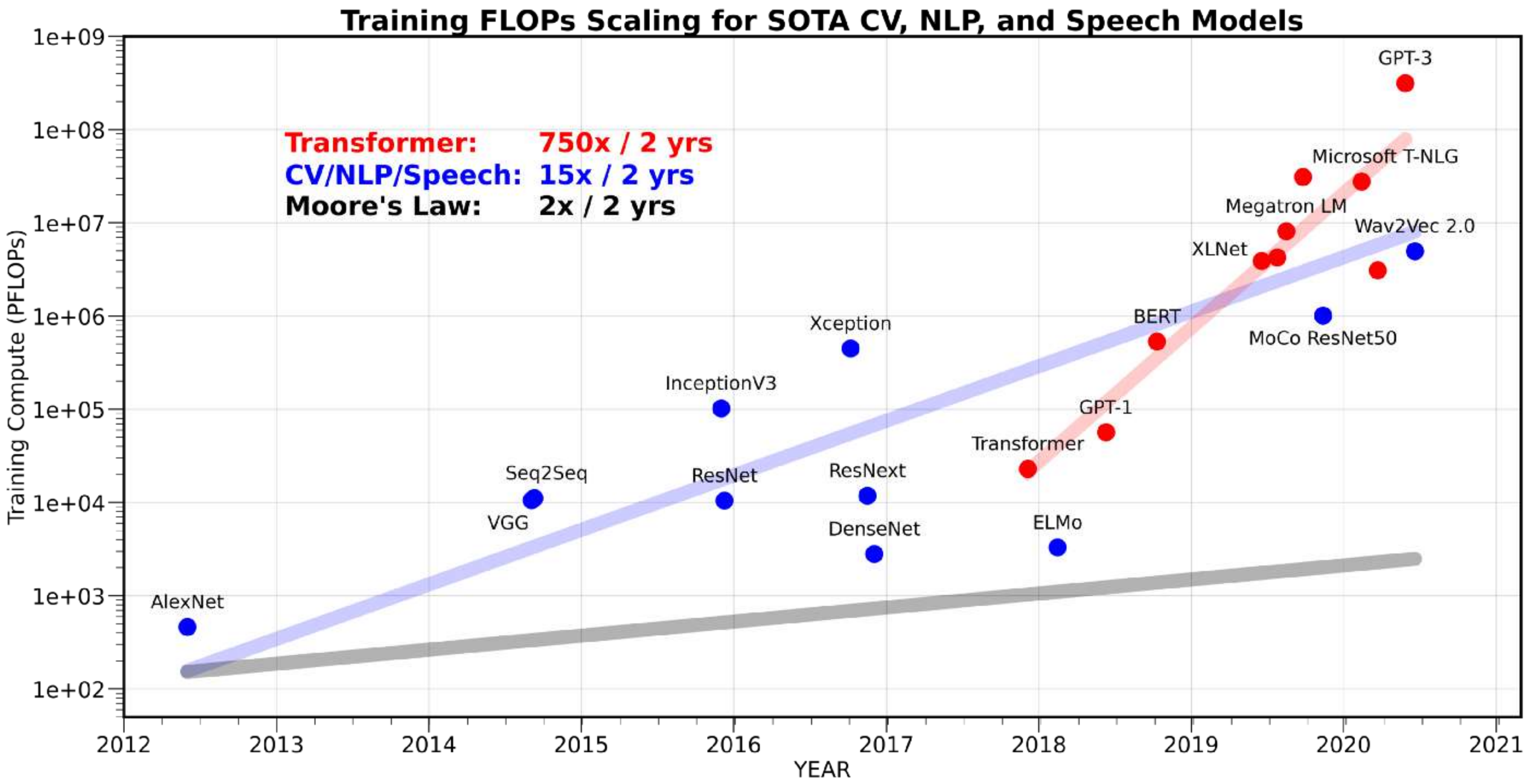
Natural Language Processing

第十一周 网络剪枝与知识蒸馏

庞彦

yanpang@gzhu.edu.cn

Foundation Models

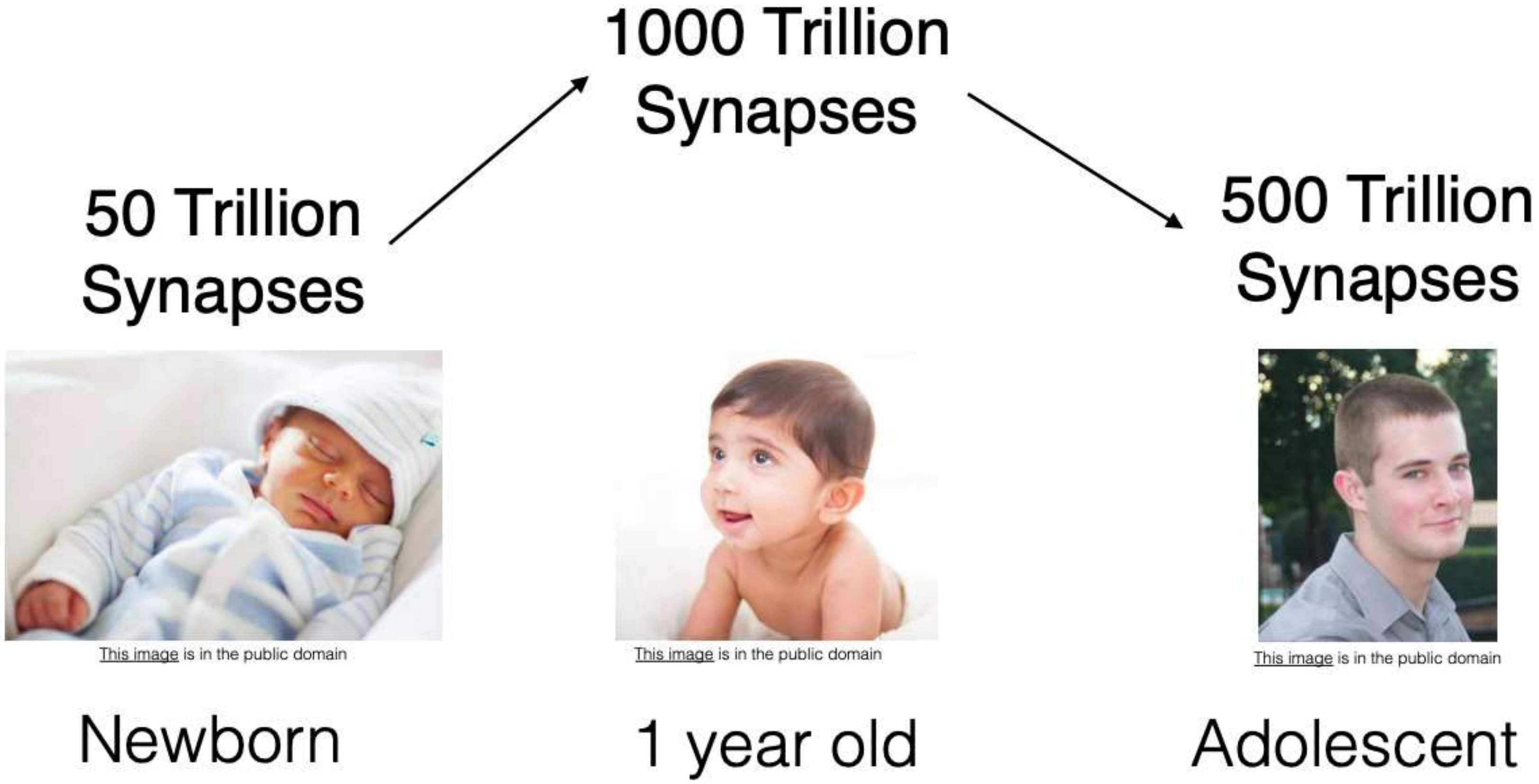


01 Network Pruning

网络剪枝

Network Pruning

Biological Inspiration for Pruning



Network Pruning

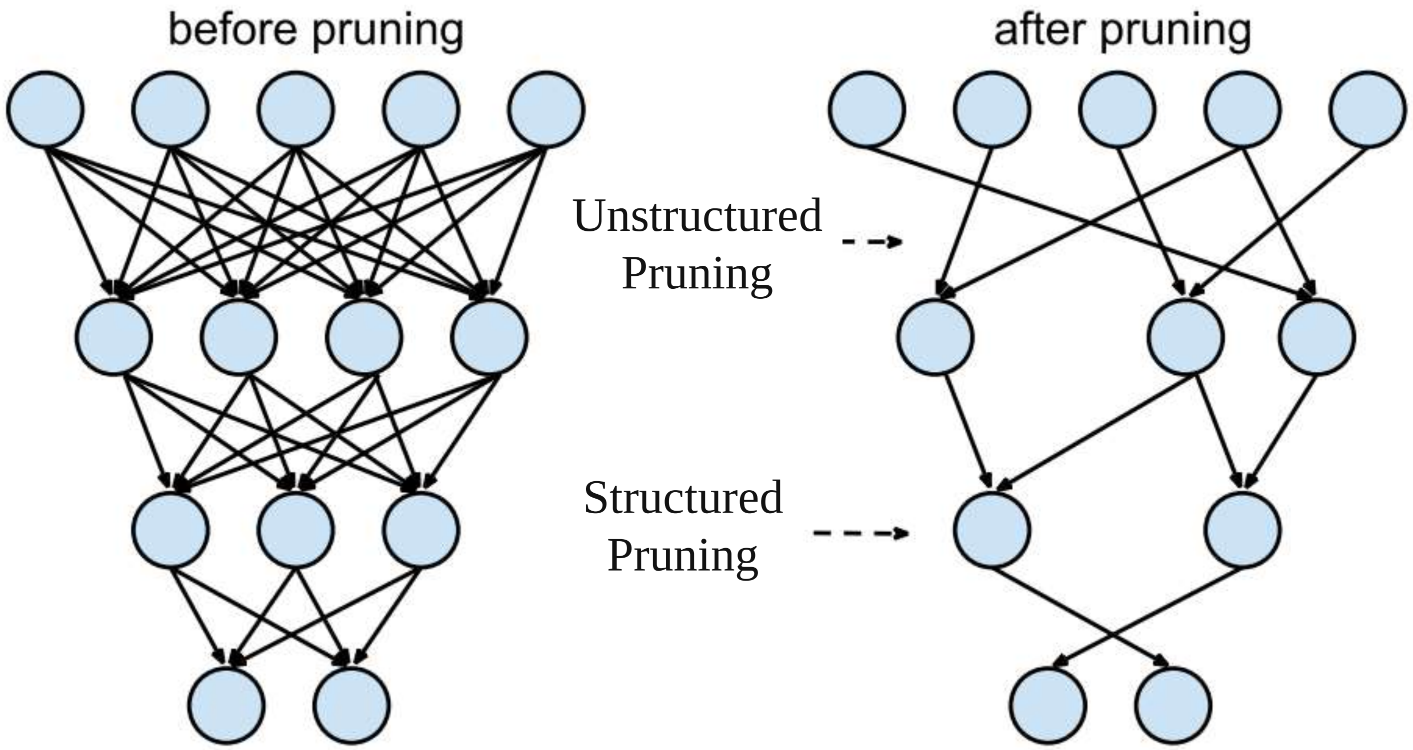
Neural network pruning is a method that revolves around the intuitive idea of removing superfluous parts of a network that performs well but costs a lot of resources.

What kind of part should I prune? — **Pruning Structures**

How to tell which parts can be pruned? — **Pruning Criteria**

How to prune parts without harming the network? — **Pruning Methods**.

Pruning Structures

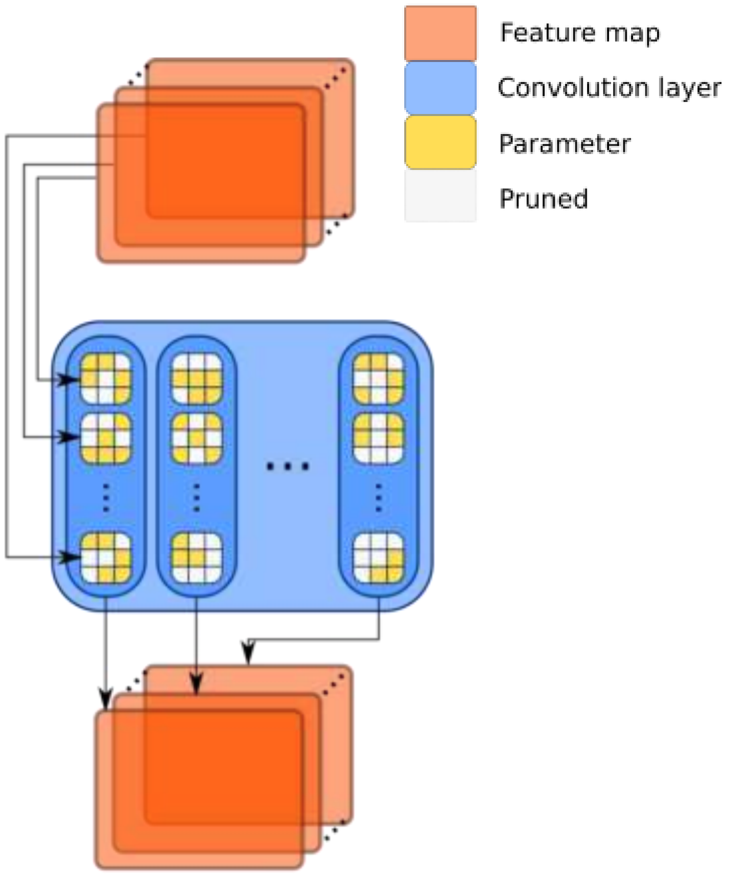


Unstructured Pruning

Directly **pruning parameters** has many advantages:

- **Simple**:
  Since replacing the value of their weight with zero, within the parameter tensors, is enough to prune a connection.

- **Fine granularity**:
  The greatest advantage of pruning connections remains yet that they are the smallest, most fundamental elements of networks and, therefore, they are numerous enough to prune them in large quantities without impacting performance.

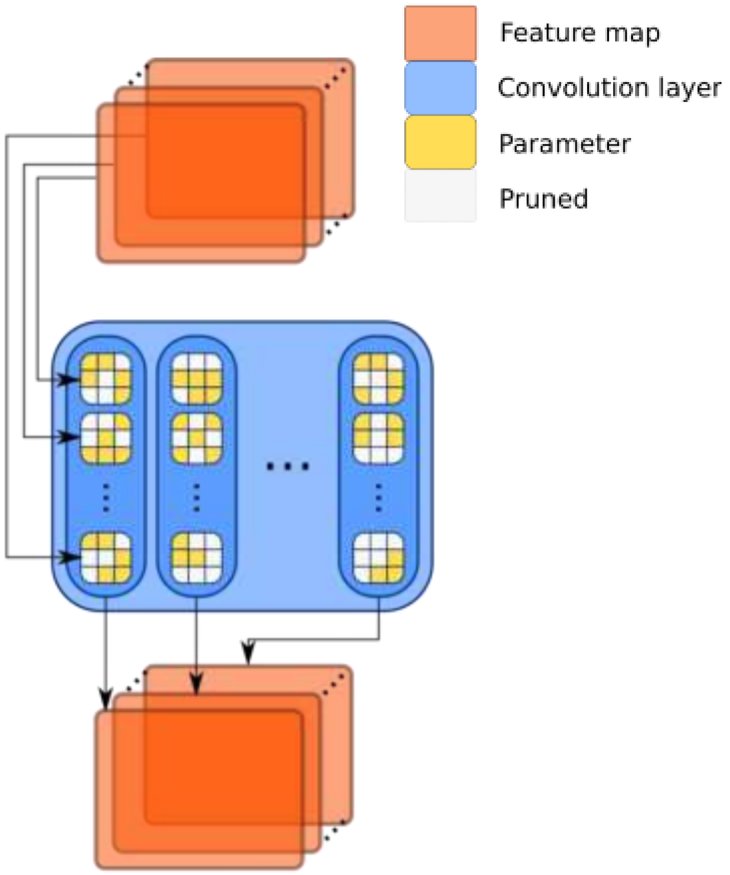


"Unstructured" : weight pruning

Unstructured Pruning

Fatal Drawback:

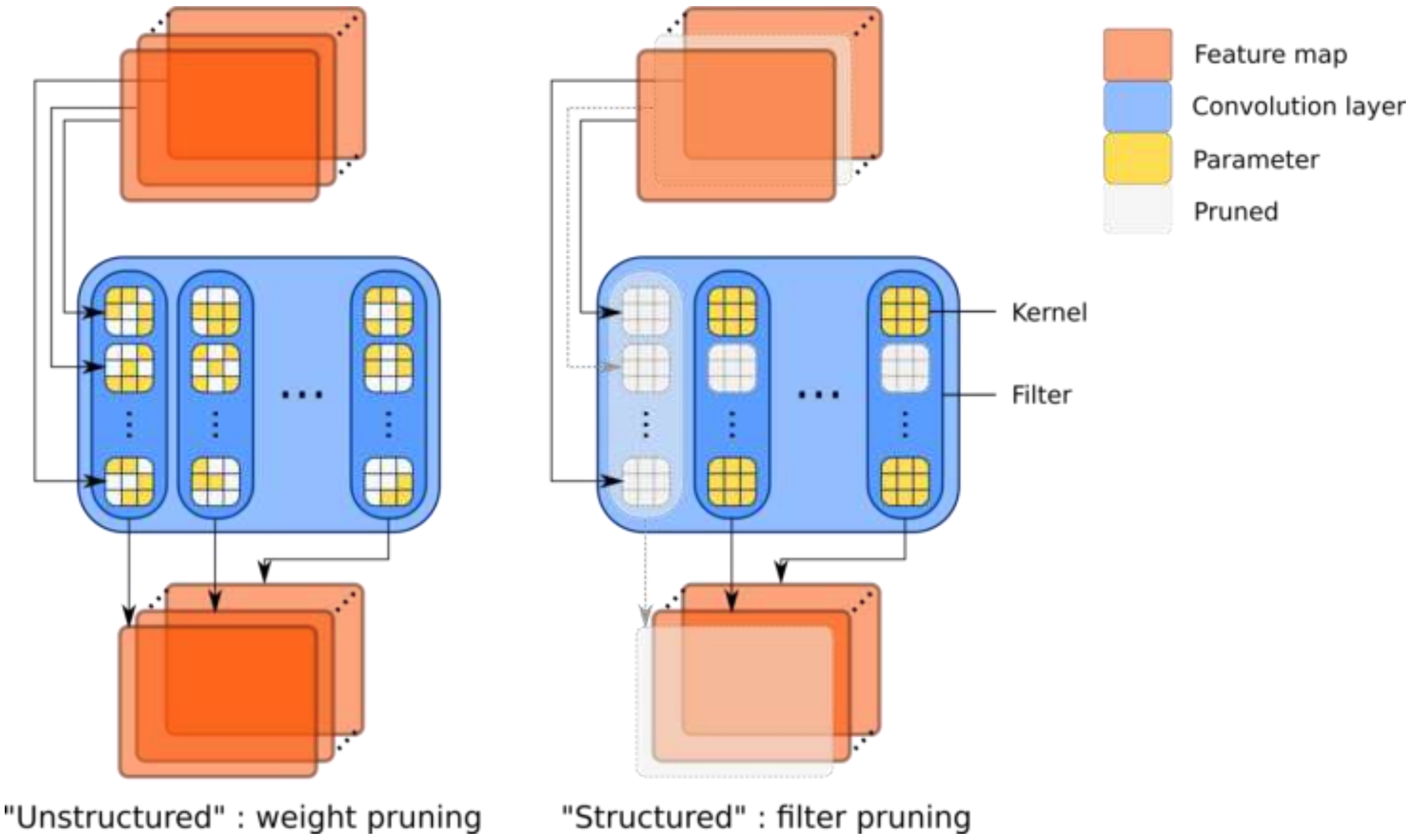
Most frameworks and hardware cannot accelerate sparse matrices' computation, meaning that no matter how many zeros you fill the parameter tensors with, it will not impact the actual cost of the network.



"Unstructured" : weight pruning

Structured Pruning

Removes both convolution filters and rows of kernels instead of just pruning connections, which leads to fewer feature maps within intermediate representations.

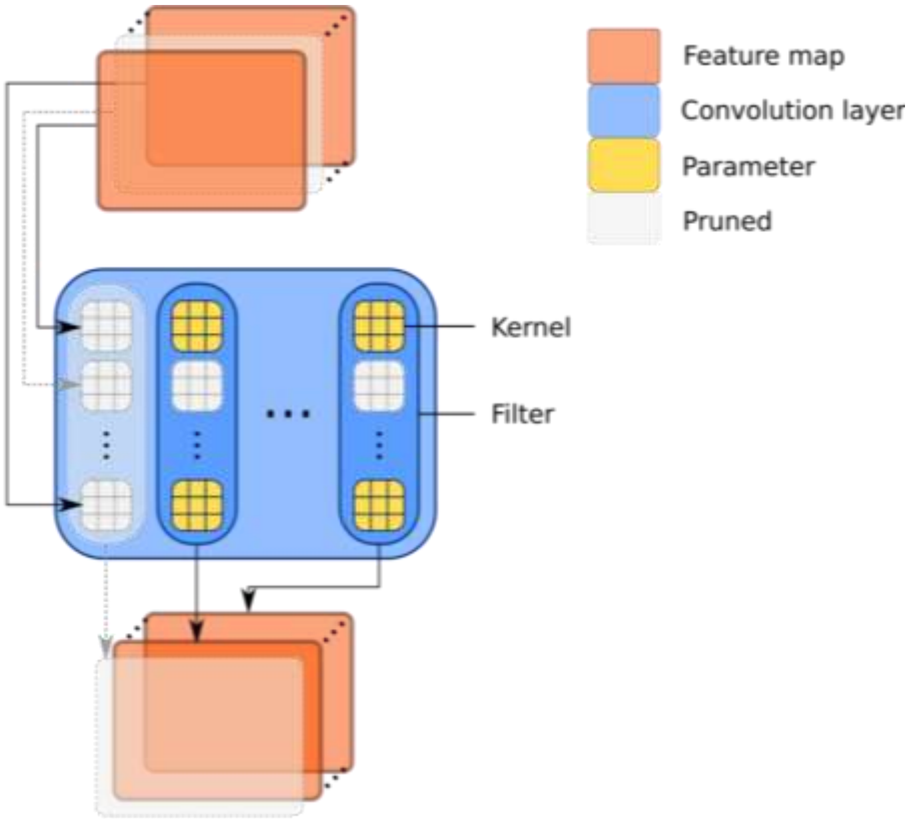


Structured Pruning

Advantages:

Not only are such networks lighter to store, due to fewer parameters, but also they require less computations and generate lighter intermediate representations, hence needing less memory during runtime.

Structured Pruning is sometimes more beneficial to **reduce bandwidth** rather than the parameter count.



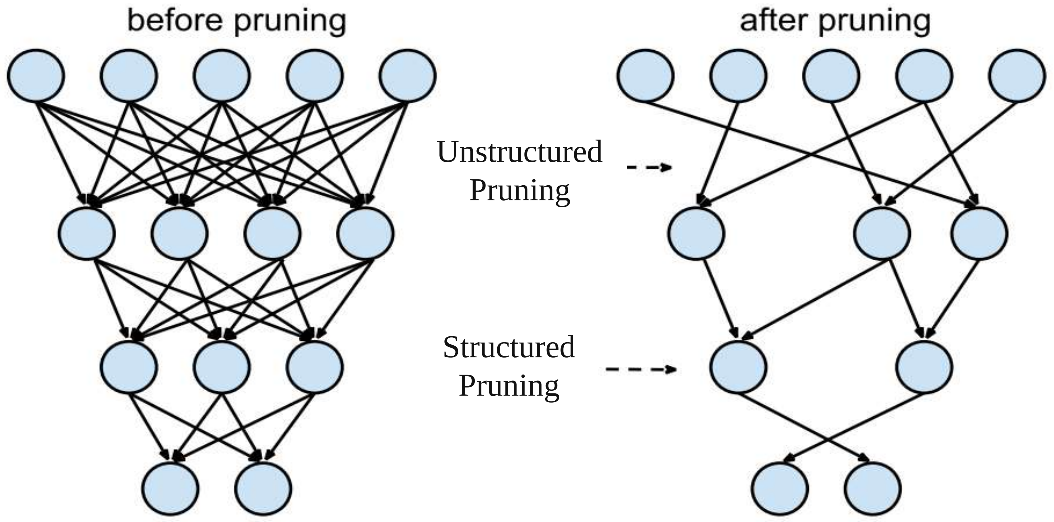
"Structured" : filter pruning

Pruning Criteria

Once one has decided what kind of structure to prune, the next question one may ask could be:

Now, how do I figure out which ones to keep and which ones to prune?

To answer that one needs a proper pruning criteria, that will **rank** the relative importance of the parameters, filters or else.



Weight Magnitude Criterion

One criterion that is quite intuitive and surprisingly efficient is pruning weights whose absolute value or magnitude is the smallest.

0.2	0.1	-0.4	0.8	0.1	-0.8	0.2	0.6
0.8	-0.4	0.6	0.1	0.2	-0.6	0.1	0.7
-0.7	0.1	0.4	0.2	0.6	-0.1	-0.5	0.1
0.1	0.5	-0.3	0.2	-0.4	0.1	0.2	0.7

0.2			0.8			0.2	0.6
0.8		0.6		0.2			0.7
		0.4	0.2	0.6			0.1
	0.5		0.2			0.2	0.7

Gradient Magnitude Pruning

Gradient Magnitude Pruning accumulate gradients over a minibatch of training data and prune on the basis of the product between this gradient and the corresponding weight of each parameter.

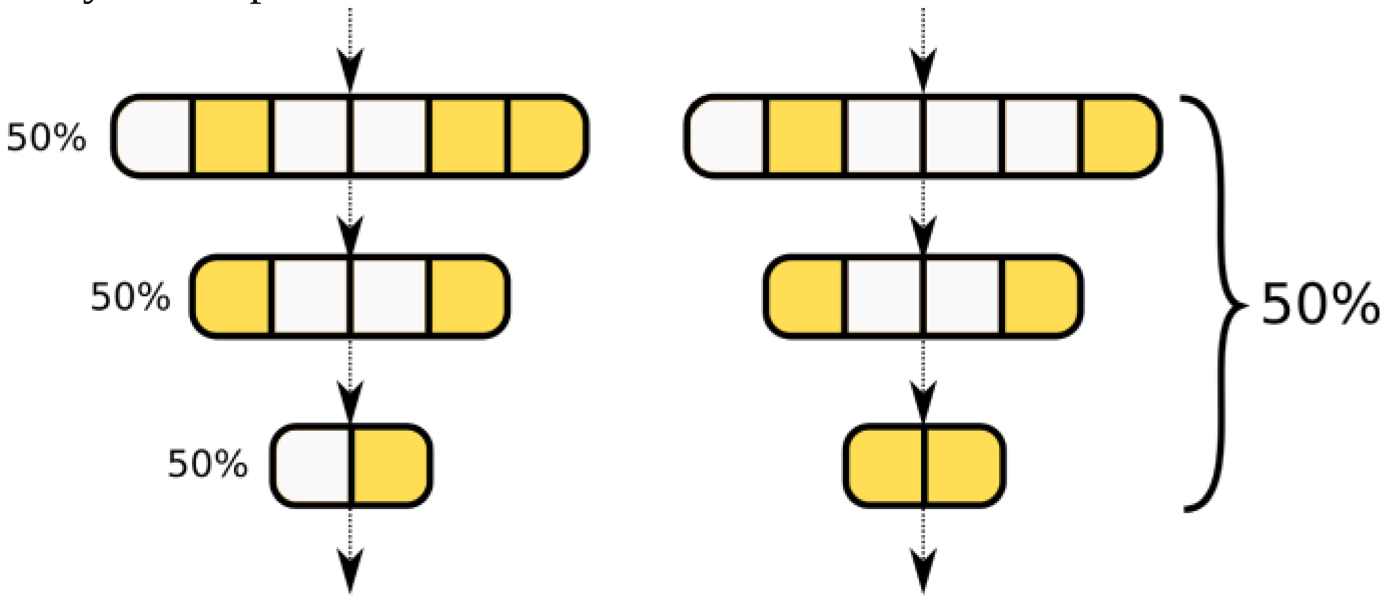
$$\underset{\text{Gradient Matrix}}{\overset{\nabla_{R_1}}{\begin{bmatrix} r_{00} & r_{01} & r_{02} & r_{03} \\ r_{10} & r_{11} & r_{12} & r_{13} \\ r_{20} & r_{21} & r_{22} & r_{23} \\ r_{30} & r_{31} & r_{32} & r_{33} \end{bmatrix}}} \otimes \underset{\text{Weight Matrix}}{\overset{W}{\begin{bmatrix} w_{00} & w_{01} & w_{02} & w_{03} \\ w_{10} & w_{11} & w_{12} & w_{13} \\ w_{20} & w_{21} & w_{22} & w_{23} \\ w_{30} & w_{31} & w_{32} & w_{33} \end{bmatrix}}} \underset{\boxed{\nabla Act + \nabla Top_k}}{=} \underset{\text{Sparse Matrix}}{\overset{\nabla_{A_1}}{\begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{01} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix}}}$$

Global or Local Pruning

One final aspect to take into consideration is whether the chosen criterion is applied globally to all parameters or filters of the network, or if it is computed independently for each layer.

While global pruning has proven many times to yield better results, it can lead to layer collapse.

A simple way to avoid this problem is to resort to layer-wise local pruning, namely pruning the same rate at each layer, when the used method cannot prevent layer collapse.



Difference between local pruning (left) and global pruning (right): local pruning applies the same rate to each layer while global applies it on the whole network at once.

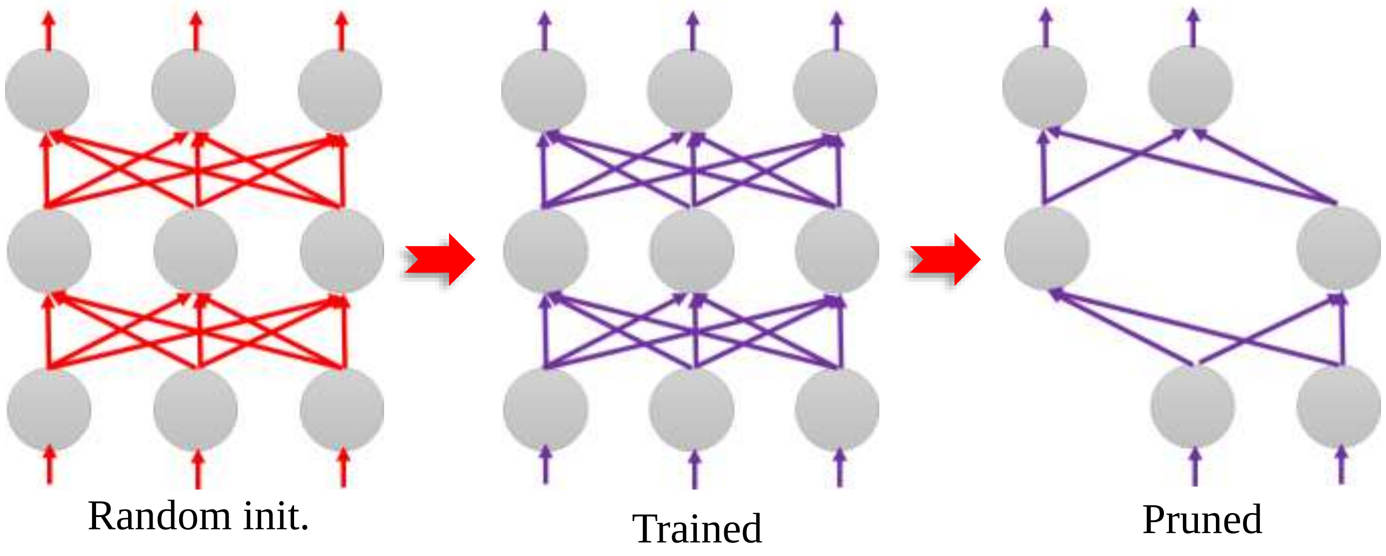
Pruning Method

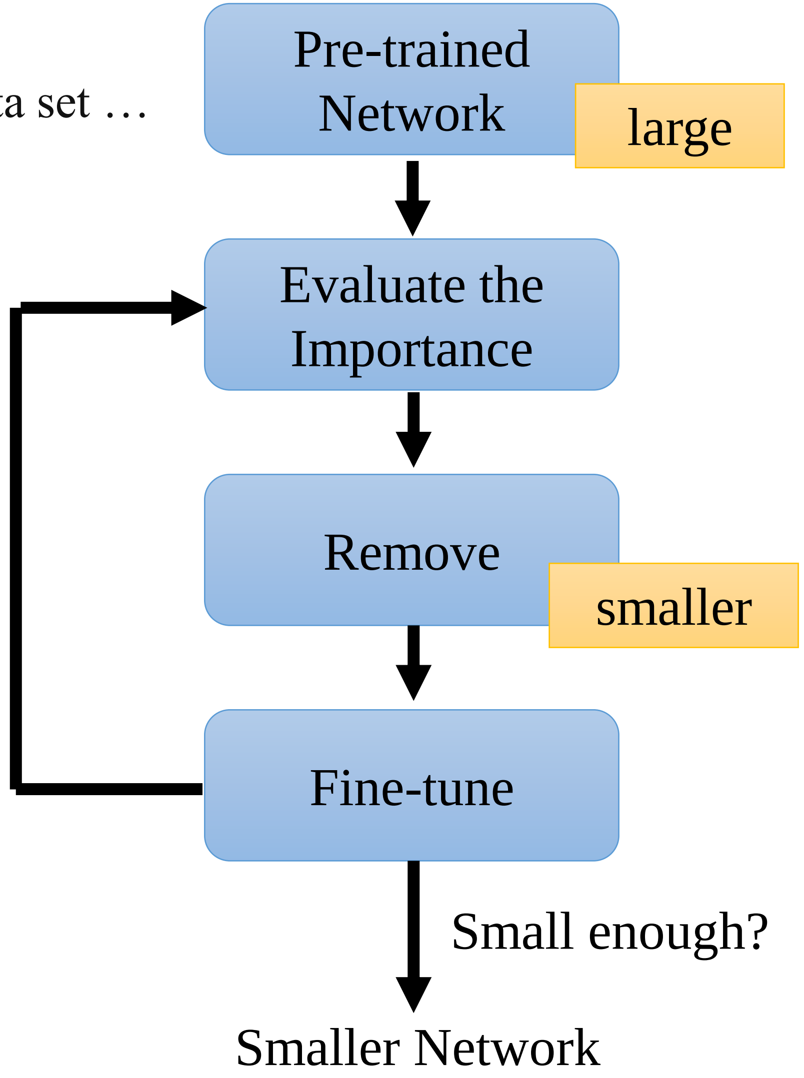
Importance of a weight: absolute values, life long …
Importance of a neuron: the number of times it wasn't zero on a given data set …

After pruning, the accuracy will drop (hopefully not too much);

Fine-tuning on training data for recover;

Don't prune too much at once, or the network won't recover.

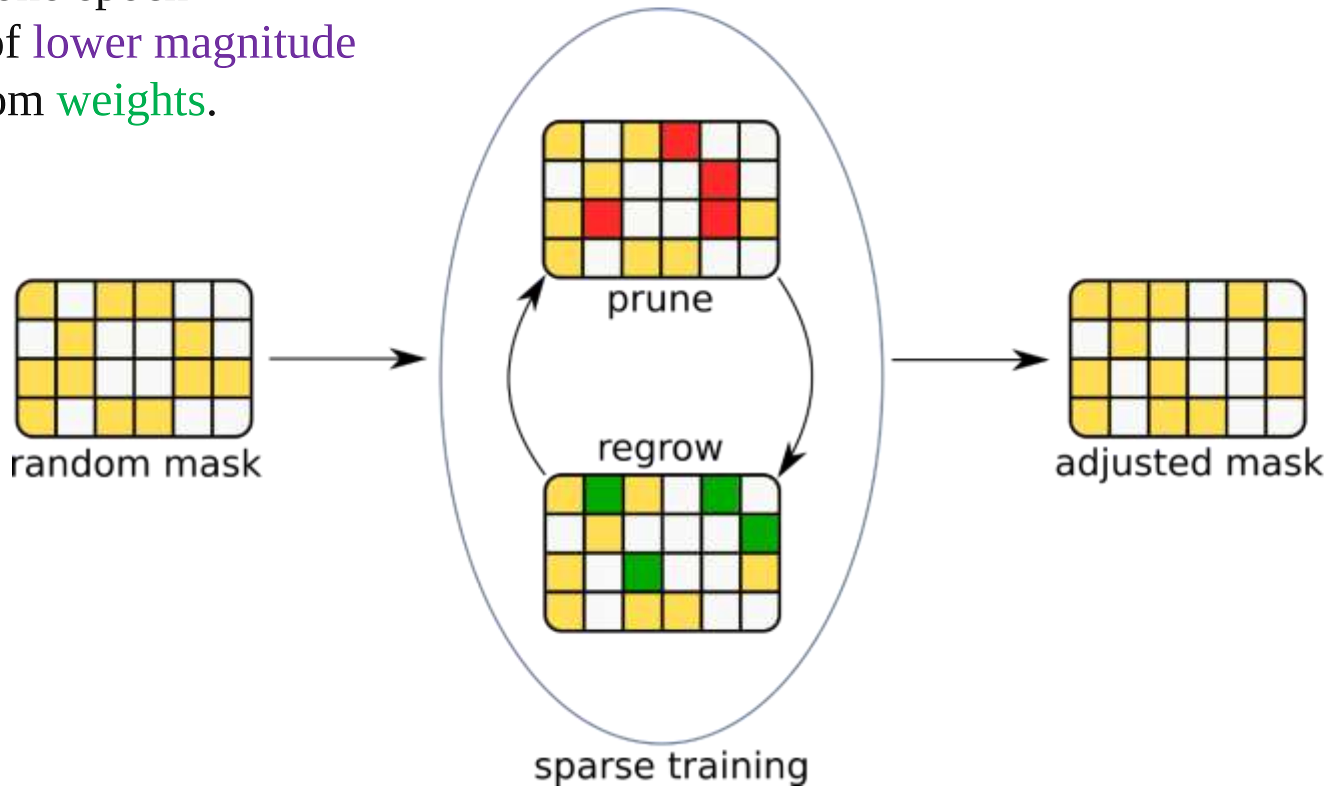




Pruning Method

Sparse Training consists in enforcing a constant rate of sparsity during training while its distribution varies and is progressively adjusted.

1) Initialize the network with a random mask that prunes a certain proportion of the network
2) Train this pruned network during one epoch
3) Prune certain amount of weights of lower magnitude
4) Regrow the same amount of random weights.



02 Knowledge Distillation

知识蒸馏

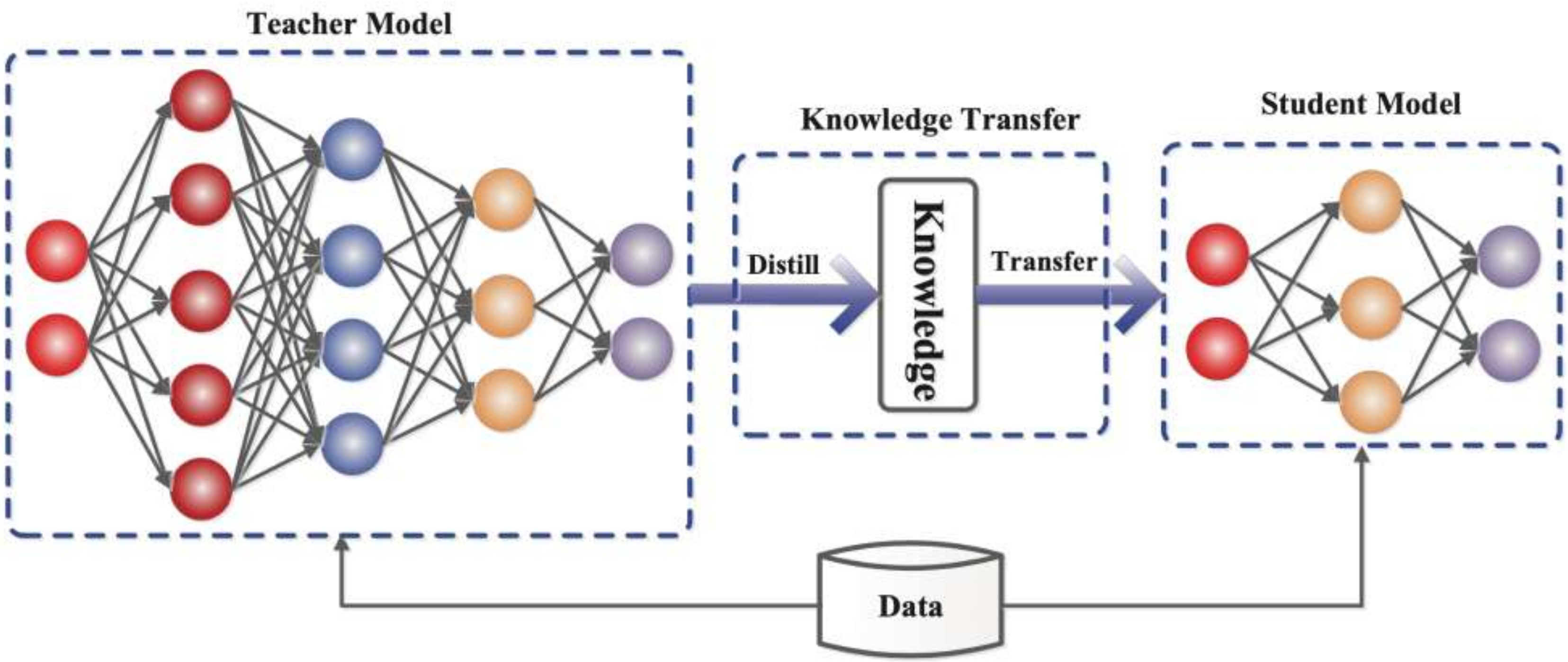
Knowledge Distillation

Knowledge distillation refers to the process of transferring the knowledge from a large unwieldy model or set of models to a single smaller model that can be practically deployed under real-world constraints.

With the advent of deep learning in the last decade, and its success in diverse fields including speech recognition, image recognition, and natural language processing, knowledge distillation techniques have gained prominence for practical real-world applications

Knowledge Distillation

In knowledge distillation, a small "student" model learns to mimic a large "teacher" model and leverage the knowledge of the teacher to obtain similar or higher accuracy.

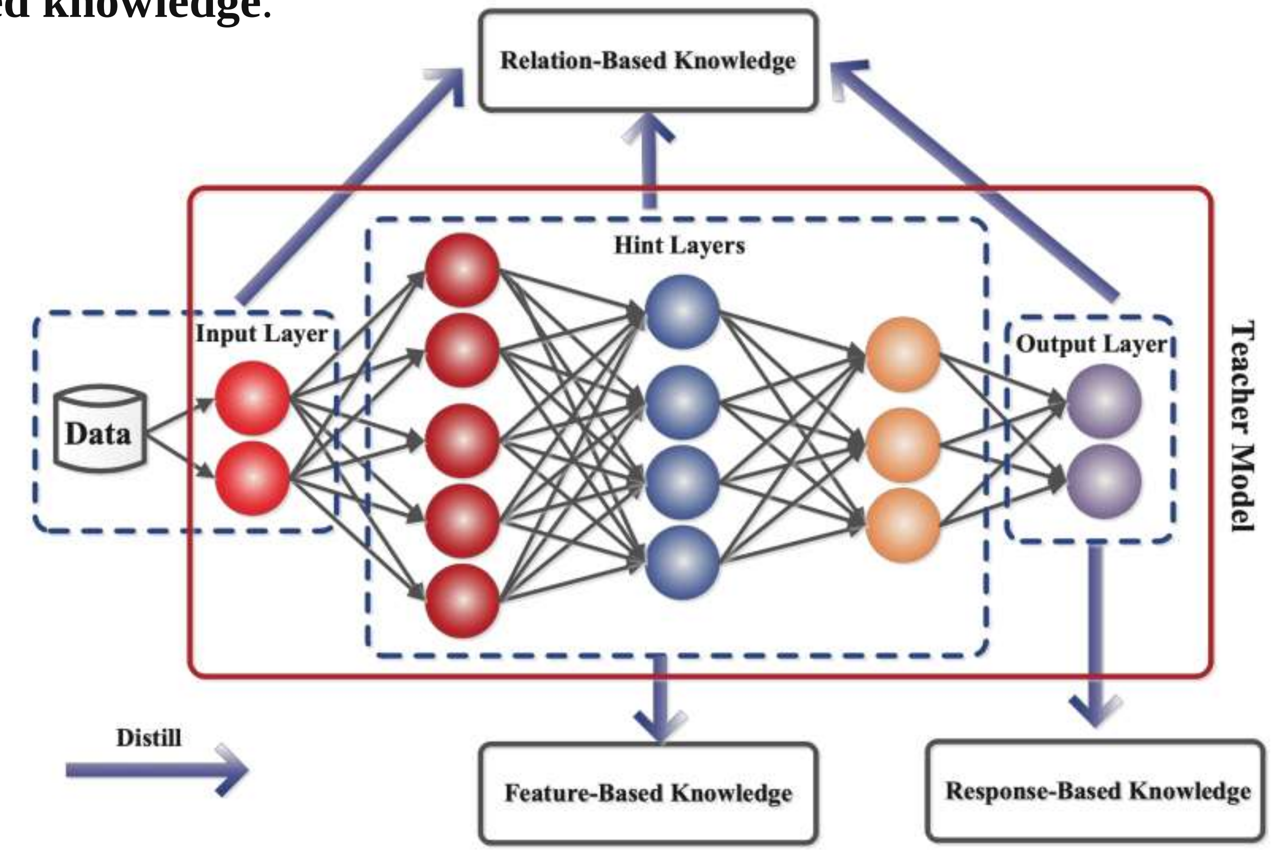


A knowledge distillation system consists of three principal components: the **knowledge**, the **distillation algorithm**, and the **teacher-student architecture**.

Knowledge

In a neural network, **knowledge** typically refers to the learned weights and biases.

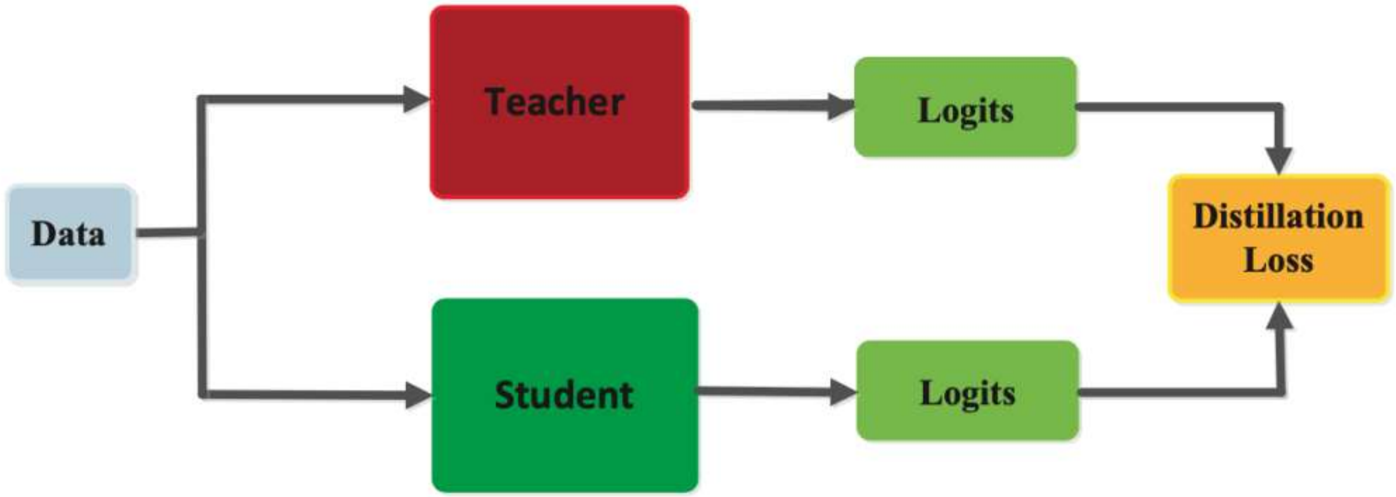
The different forms of knowledge are categorized into three different types: **Response-based knowledge**, **Feature-based knowledge**, and **Relation-based knowledge**.



Response-based Knowledge

The **response-based knowledge** focuses on the final output layer of the teacher model.

The hypothesis is that the student model will learn to mimic the predictions of the teacher model.



Response-based Knowledge

Cross-entropy minimization

Learning target

“1”: 0.7, “7”: 0.2, “9”: 0.1

?

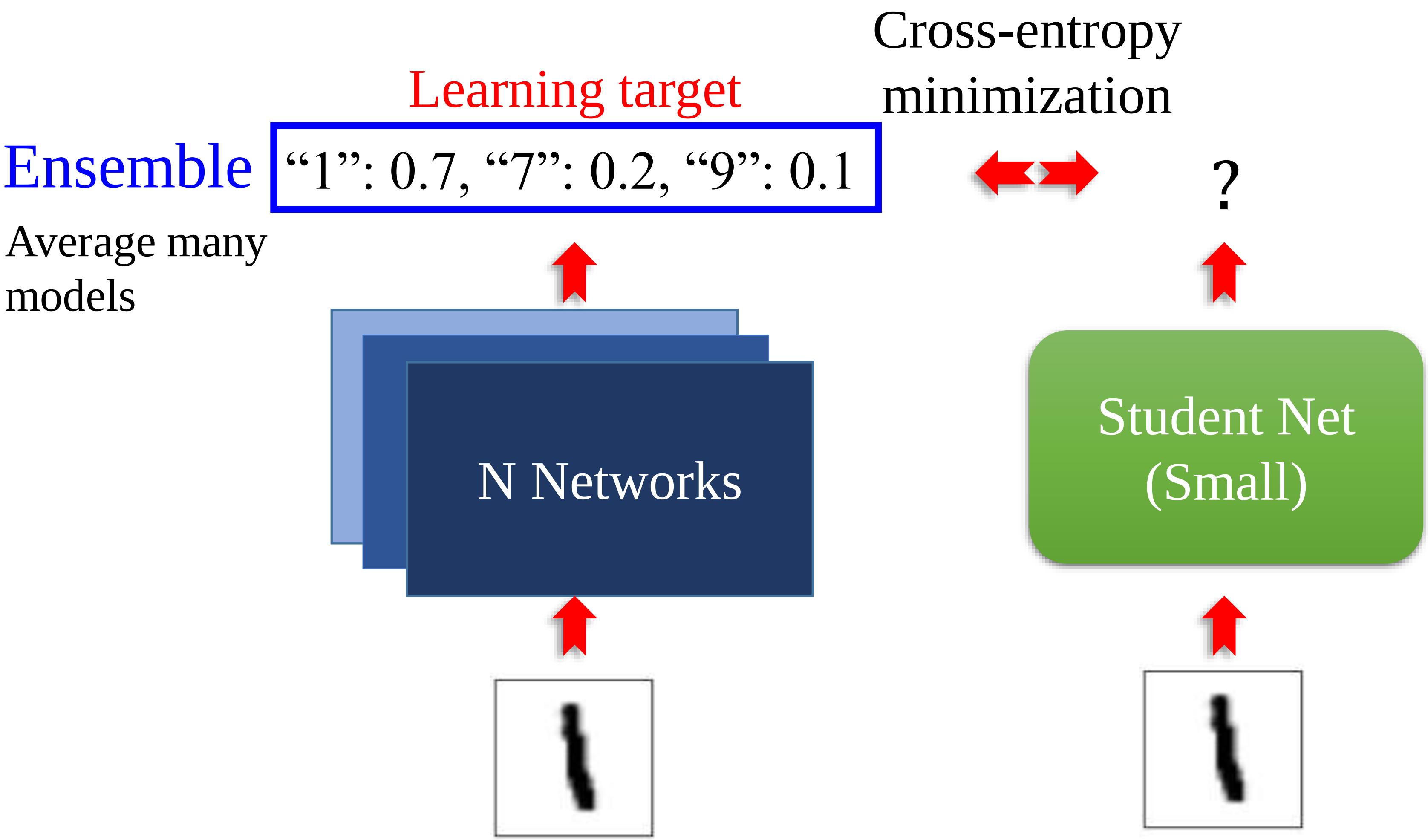
Teacher Net (Large)

Student Net (Small)

Providing the information that “1” is similar to “7”

Response-based Knowledge

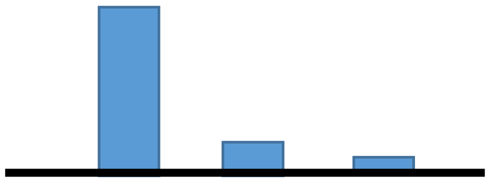
Approaches:

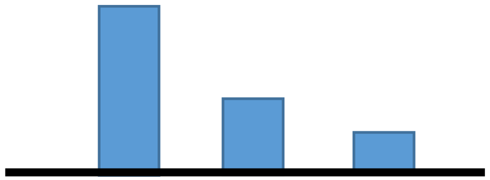


Response-based Knowledge

Temperature for SoftMax

$$y_i' = \frac{exp(y_i)}{\sum_j exp(y_j)} \quad \xrightarrow{T\,=\,100} \quad y_i' = \frac{exp(y_i/T)}{\sum_j exp(y_j/T)}$$





$y_1 = 100 \qquad y_1' = 1$

$y_2 = 10 \qquad y_2' \approx 0$

$y_3 = 1 \qquad y_3' \approx 0$

$y_1/T = 1 \qquad y_1' = 0.56$

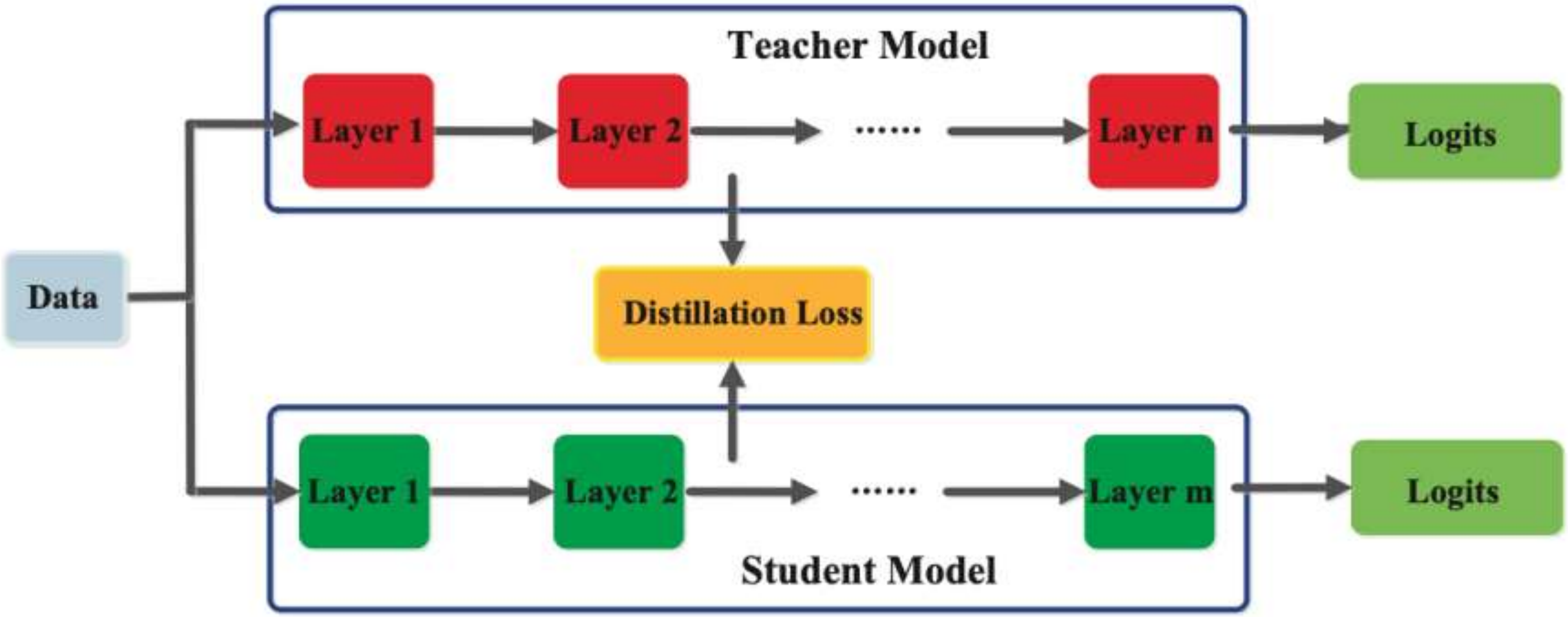
$y_2/T = 0.1 \qquad y_2' = 0.23$

$y_3/T = 0.01 \qquad y_3' = 0.21$

Feature-based Knowledge

The goal of **feature-based knowledge** is to train the student model to learn the same feature activations as the teacher model in its intermediate layers.

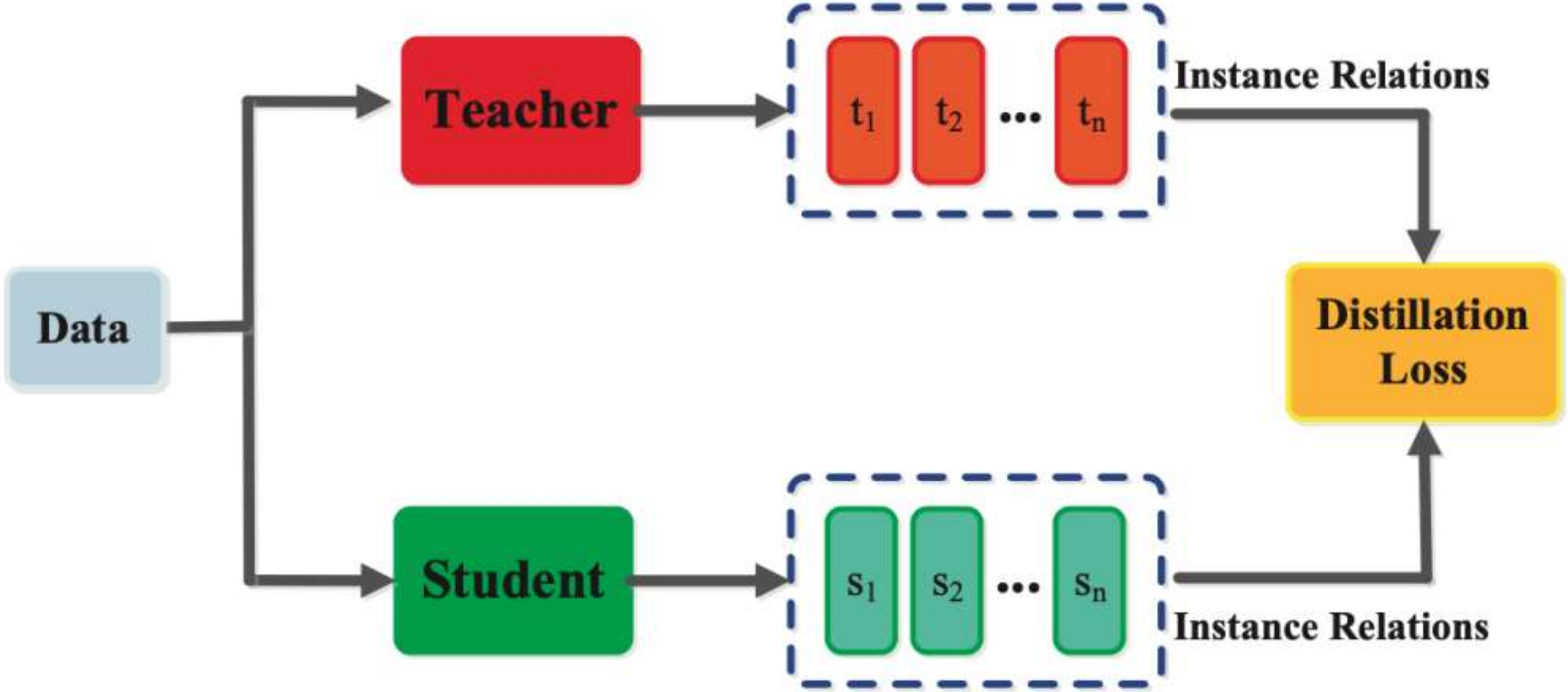
The distillation loss function achieves this by minimizing the difference between the feature activations of the teacher and the student models.



Relation-based Knowledge

The **relation-based knowledge** that captures the relationship between feature maps can also be used to train a student model.

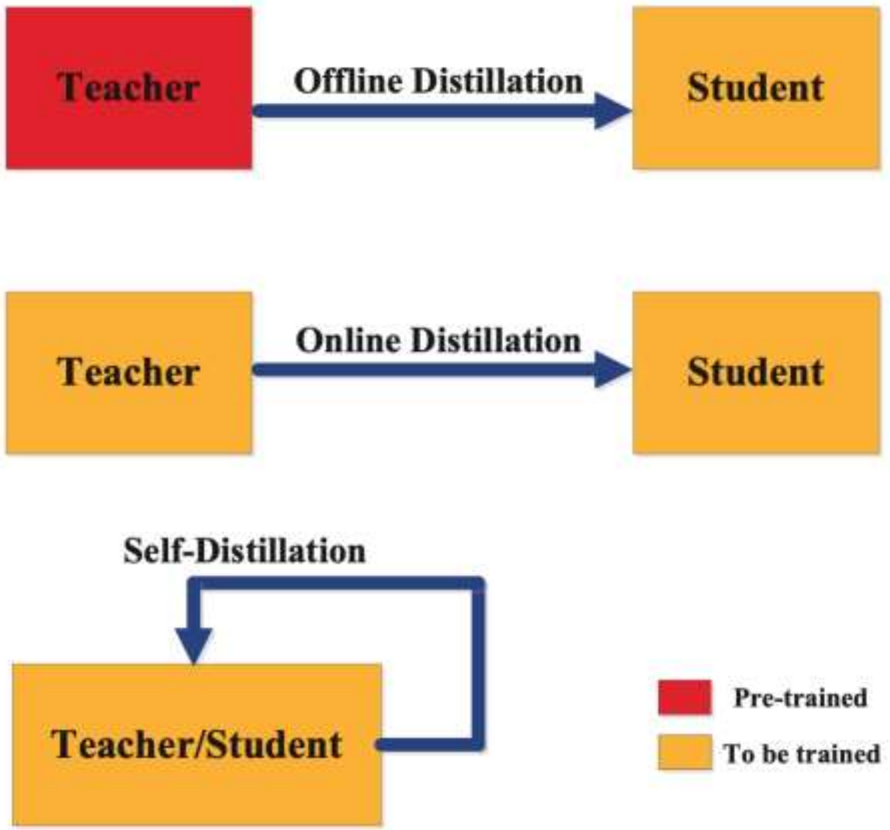
This relationship can be modeled as correlation between feature maps, graphs, similarity matrix, feature embeddings, or probabilistic distributions based on feature representations.



Training

There are three principal types of methods for training student and teacher models, namely **offline**, **online** and **self distillation**.

The categorization of the distillation training methods depends on whether the teacher model is modified at the same time as the student model or not.



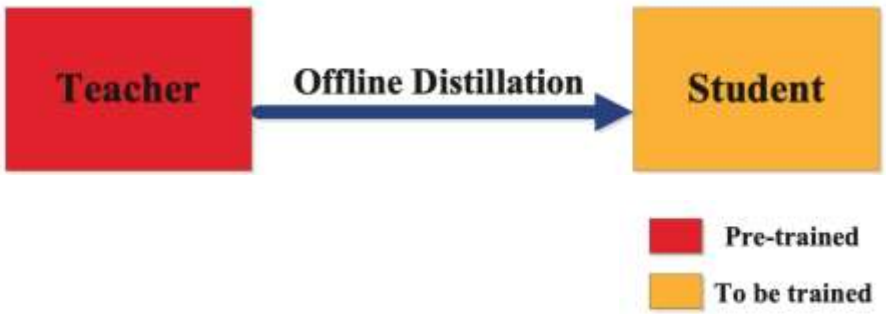
Training: Offline distillation

Offline distillation is the most common method, where a pre-trained teacher model is used to guide the student model.

In this scheme, the teacher model is first pre-trained on a training dataset, and then knowledge from the teacher model is distilled to train the student model.

Given the recent advances in deep learning, a wide variety of pre-trained neural network models are openly available that can serve as the teacher depending on the use case.

Offline distillation is an established technique in deep learning and easier to implement.



Training: Online distillation

The **online distillation** can be used where both the teacher and student models are updated simultaneously in a single end-to-end training process.

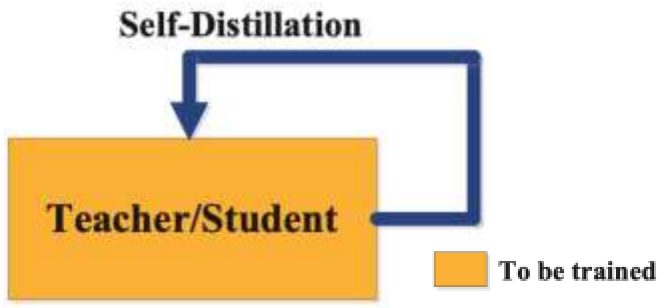
Online distillation can be operationalized using parallel computing thus making it a highly efficient method.

Training: Self-distillation

In **self-distillation**, the same model is used for the teacher and the student models.

For instance, knowledge from deeper layers of a deep neural network can be used to train the shallow layers. It can be considered a special case of online distillation, and instantiated in several ways.

Knowledge from earlier epochs of the teacher model can be transferred to its later epochs to train the student model.



Architecture

The design of the student-teacher network architecture is critical for efficient knowledge acquisition and distillation.

Typically, there is a model capacity gap between the more complex teacher model and the simpler student model.

This structural gap can be reduced through optimizing knowledge transfer via efficient student-teacher architectures.

Architecture

The most common architectures for knowledge transfer include a student model that is:

- a shallower version of the teacher model with fewer layers and fewer neurons per layer,
- a quantized version of the teacher model,
- a smaller network with efficient basic operations,
- a smaller networks with optimized global network architecture,
- the same model as the teacher.

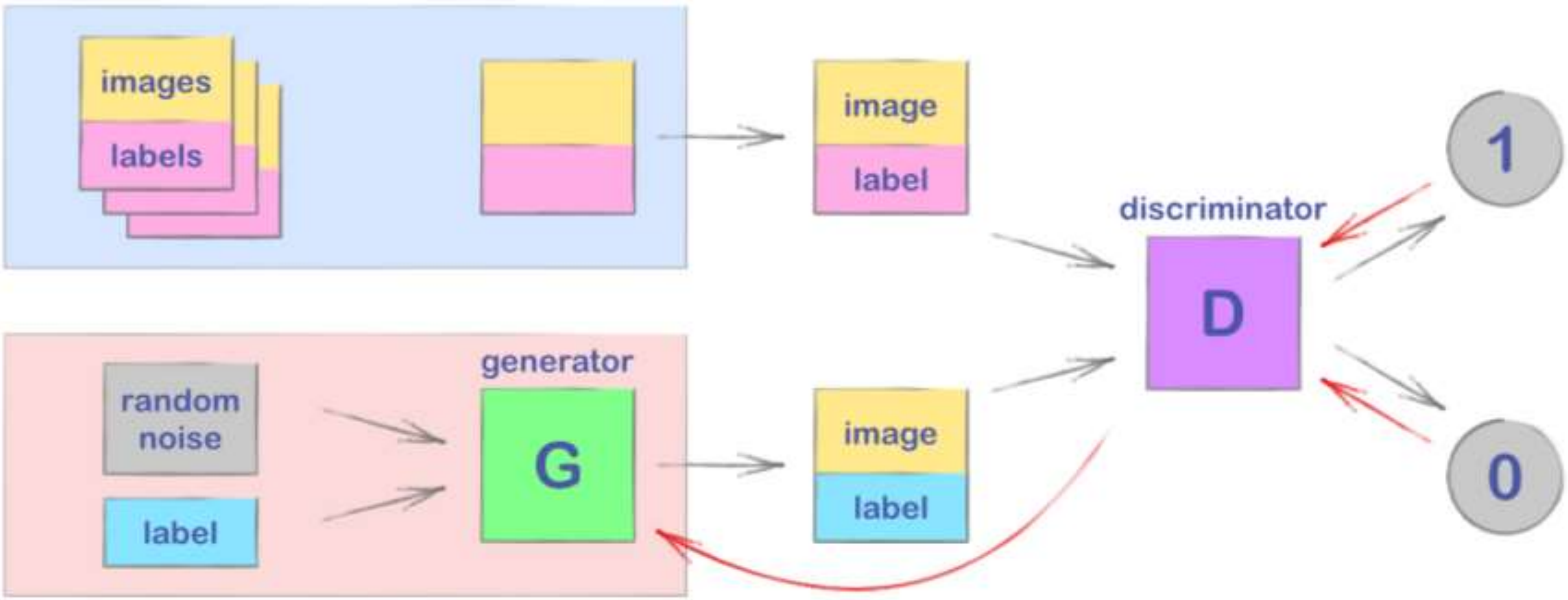
Algorithms

Algorithms for training student models to acquire knowledge from teacher models:

- Adversarial Distillation
- Multi-Teacher Distillation
- Cross-modal Distillation

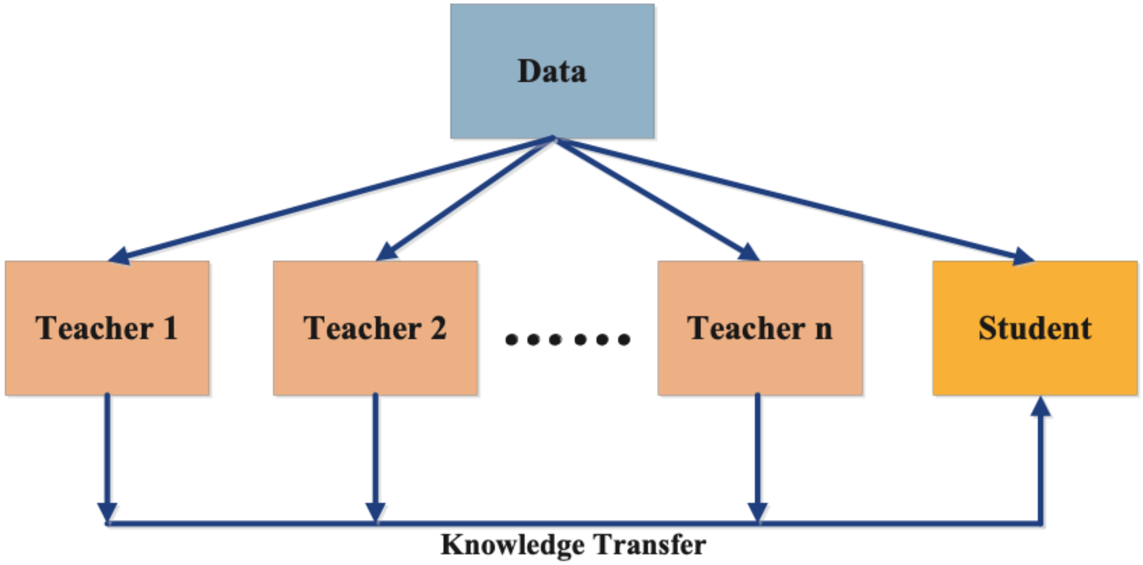
Algorithms: Adversarial distillation

Adversarial distillation enables the student and teacher models to learn a better representation of the true data distribution.



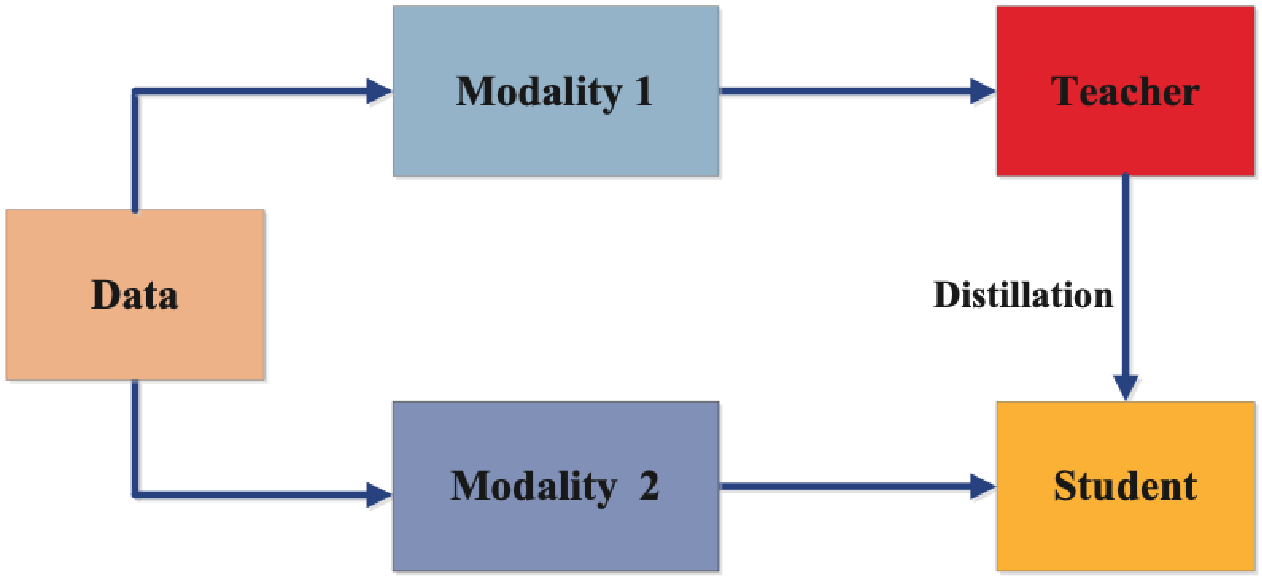
Algorithms: Multi-Teacher Distillation

In multi-teacher distillation, a student model acquires knowledge from several different teacher models.



Algorithms: Cross-modal Distillation

In cross-modal distillation, the teacher is trained in one modality and its knowledge is distilled into the student that requires knowledge from a different modality.



Algorithms

Graph-based distillation captures intra-data relationships using graphs instead of individual instance knowledge from the teacher to the student. Graphs are used in two ways – as a means of knowledge transfer, and to control transfer of the teacher's knowledge. In graph-based distillation, each vertex of the graph represents a self-supervised teacher which may be based on response-based or feature-based knowledge like logits and feature maps respectively.

Data-free distillation is based on synthetic data in the absence of a training dataset due to privacy, security or confidentiality reasons. The synthetic data is usually generated from feature representations of the pre-trained teacher model. In other applications, GANs are also used to generate synthetic training data.

Algorithms

Attention-based distillation is based on transferring knowledge from feature embeddings using attention maps.

Quantized distillation is used to transfer knowledge from a high-precision teacher model (e.g. 32-bit floating point) to a low-precision student network (e.g. 8-bit).

Lifelong distillation is based on the learning mechanisms of continual learning, lifelong learning and meta-learning where previously learnt knowledge is accumulated and transferred into future learning.

Neural architecture search-based distillation is used to identify suitable student model architectures that optimize learning from the teacher models.

Q&A