University of Antwerp

Artificial Neural Networks

[2500WETANN]

José Oramas

Course Announcements

- **Papers for the Research Assignment will be distributed this week**

  Let me know in case there are issues with the assigned paper

- **Time and Place to be Confirmed**

Transfer Learning

[Reuseing Existing Models]

José Oramas

University of Antwerp

Based on your experience...

What problems did you encounter when training models?

Based on your experience...

What problems did you encounter when training models?

Today: Existing models → New models

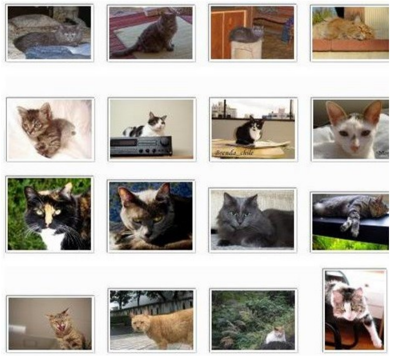
University of Antwerp

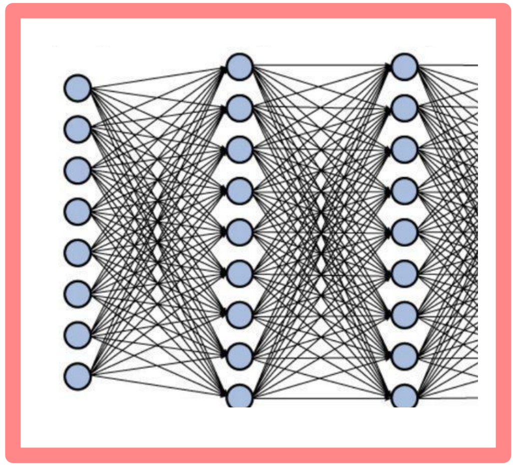
Transfer Learning

Why re-use/re-train a model?

Feature extraction

dataset





Output?
Dowstream Task?

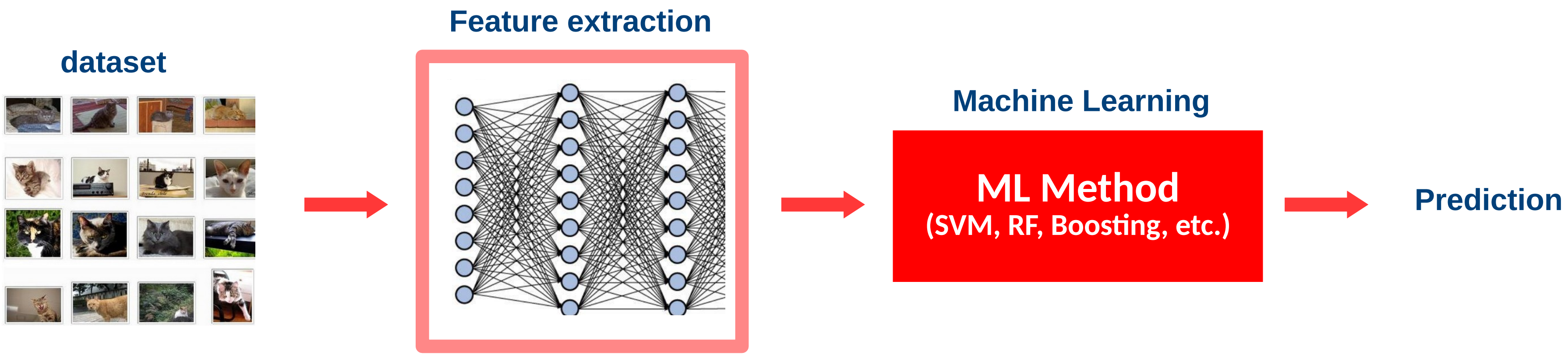
Models as Feature Extractors

[while we wait for GPUs to get better]

Transfer Learning

1- Pre-trained Models as Feature Extractors

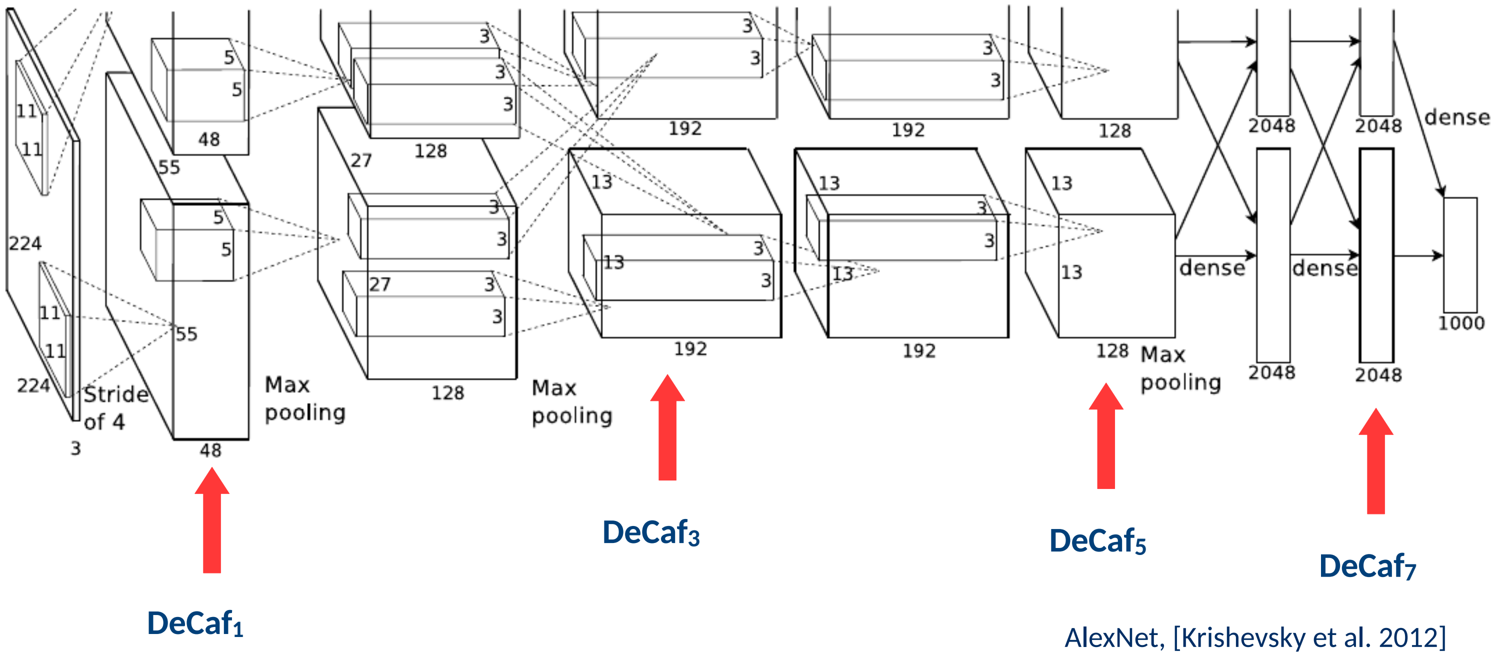
- Push data through the model
- Collect activations in a given layer
- Use activations as input features in a classical ML method



Transfer Learning

1- Pre-trained Models as Feature Extractors

– **Example:** DeCaf [Donahue et al., ICML’14]



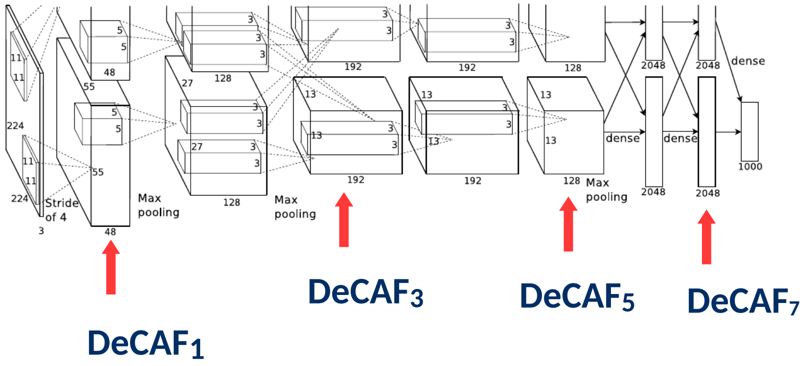
AlexNet, [Krishevsky et al. 2012]

- Take a pre-tranined model [AlexNet trained on ImageNet]
- Extract activations from given parts of the network. [*DeCaf$_i$*]
- Train a standard ML methods based on the extracted activations.

Transfer Learning

1- Pre-trained Models as Feature Extractors

– **DeCaf – Feature Visualization**



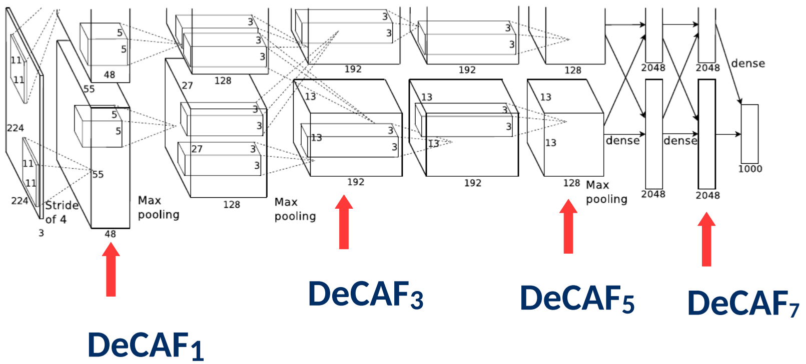
(c) DeCAF_1 (d) DeCAF_6

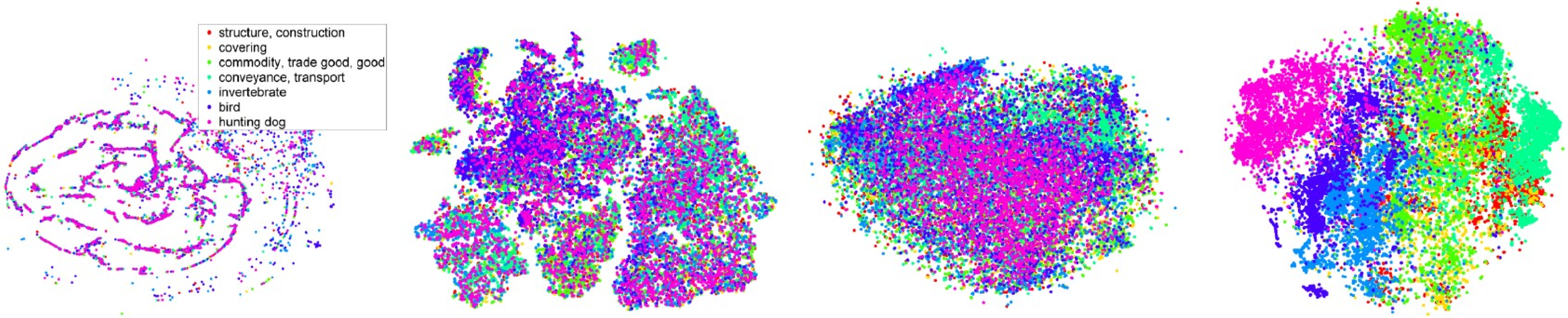
- Visualizing the representation via t-SNE [van der Maaten et al., 2008]

Transfer Learning

1- Pre-trained Models as Feature Extractors

– **DeCaf – Feature Visualization**





(a) LLC (b) GIST (c) DeCAF$_1$ (d) DeCAF$_6$

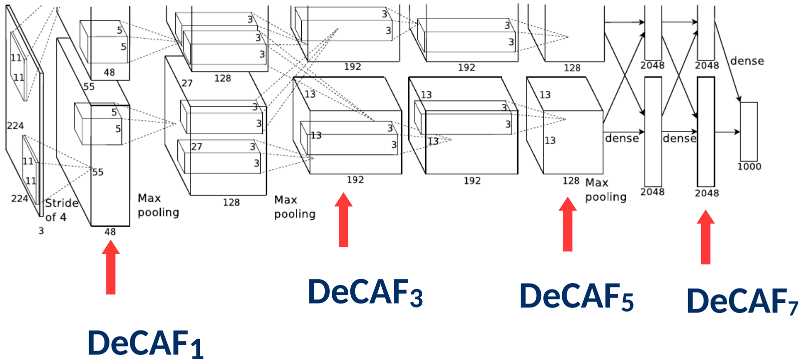
- Visualizing the representation via t-SNE [van der Maaten et al., 2008]

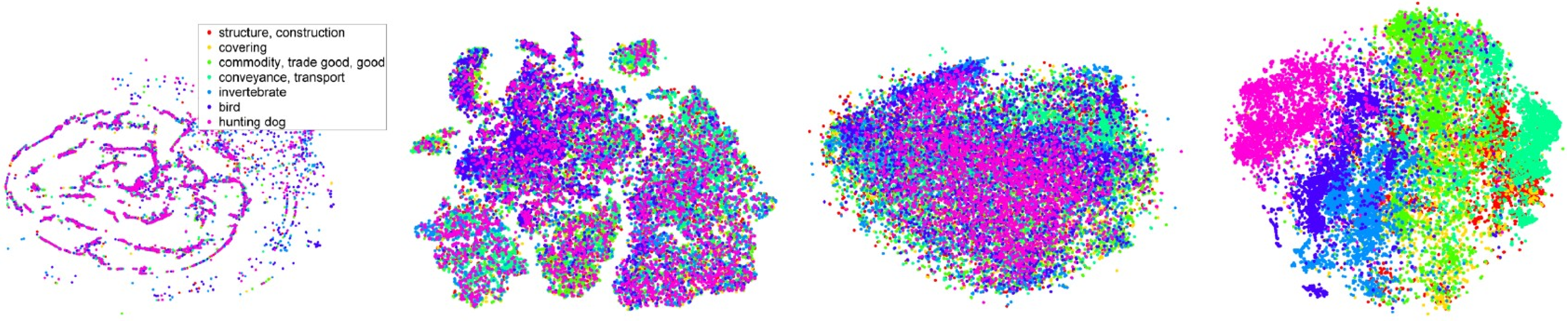
University of Antwerp

Transfer Learning

1- Pre-trained Models as Feature Extractors

– **DeCaf – Feature Visualization**





(a) LLC (b) GIST (c) DeCAF_1 (d) DeCAF_6

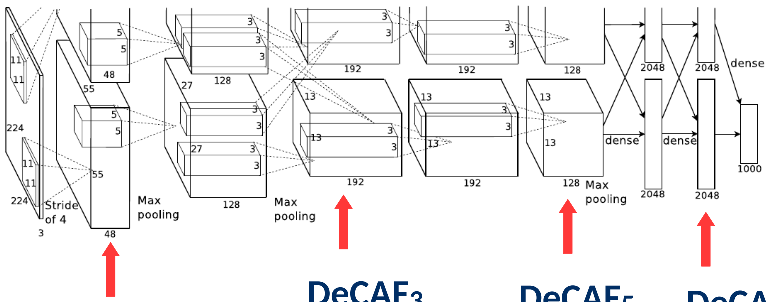
- Visualizing the representation via t-SNE [van der Maaten et al., 2008]

Q: Why DeCAF_1 and DeCAF_6 look so different?

Transfer Learning

1- Pre-trained Models as Feature Extractors

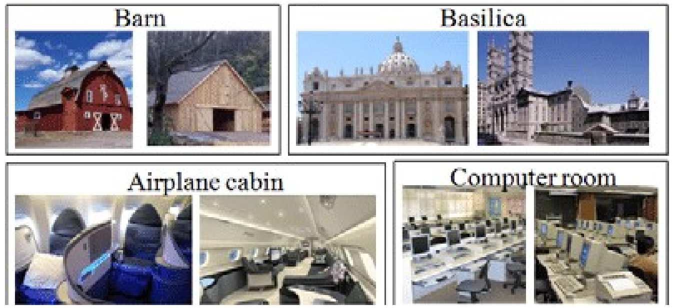
– DeCaf – Feature Generalization



DeCAF$_1$ DeCAF$_3$ DeCAF$_5$ DeCAF$_7$

ILSVRC'12

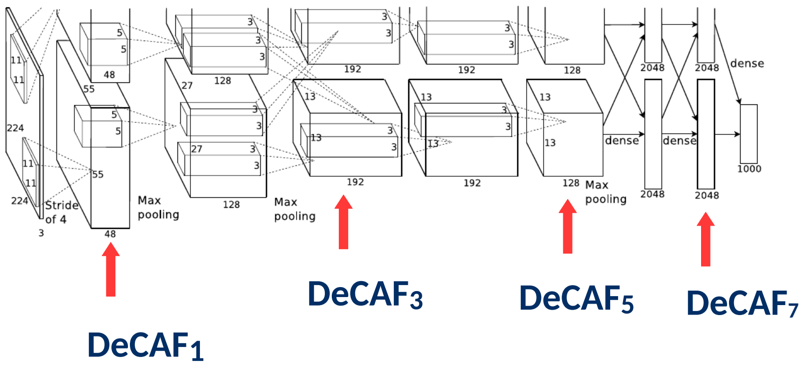
SUN-397



Transfer Learning

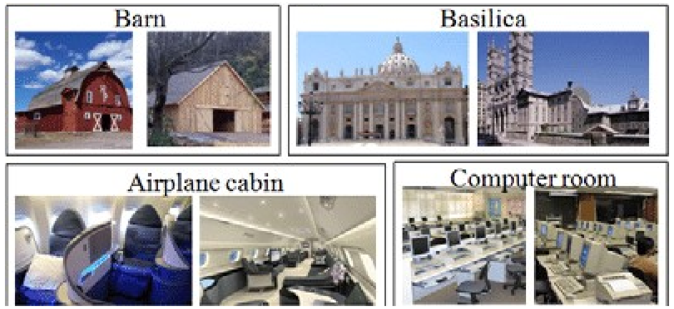
1- Pre-trained Models as Feature Extractors

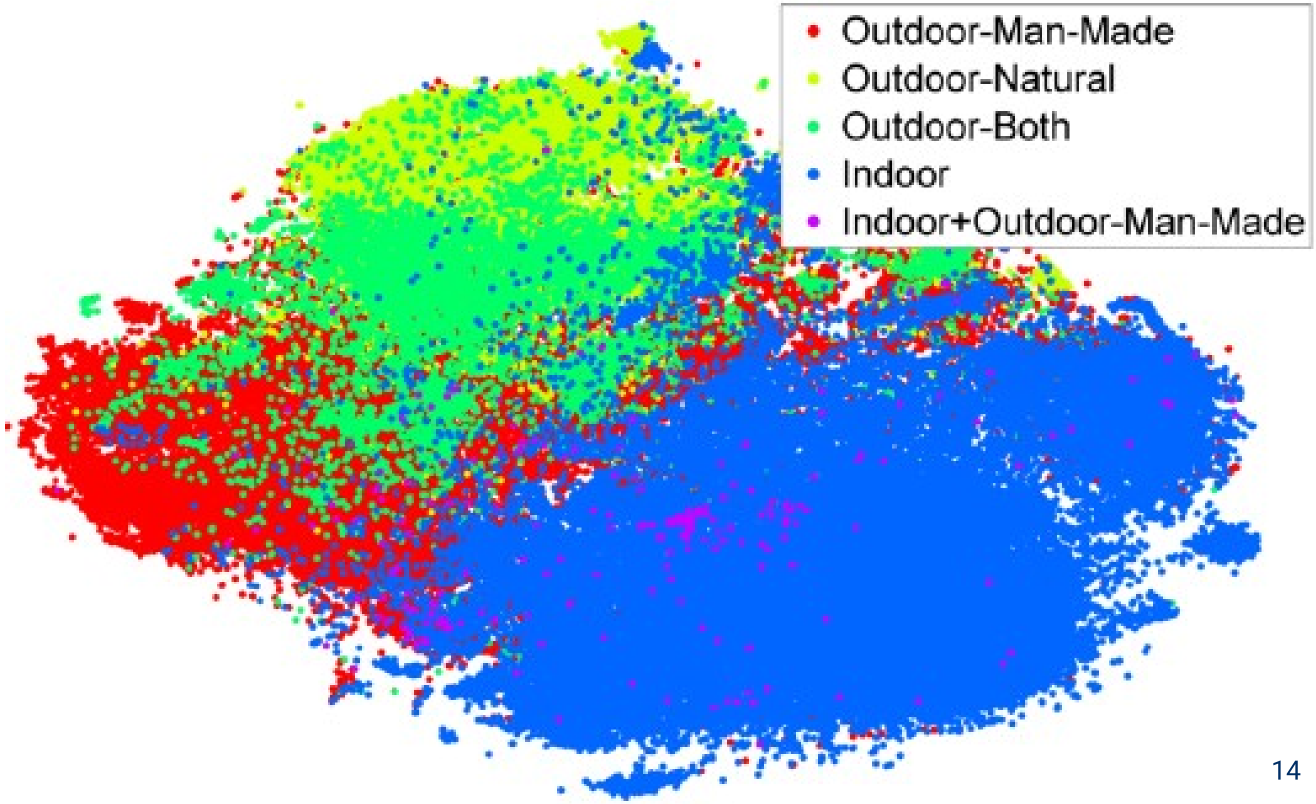
– DeCaf – Feature Generalization



ILSVRC'12

SUN-397



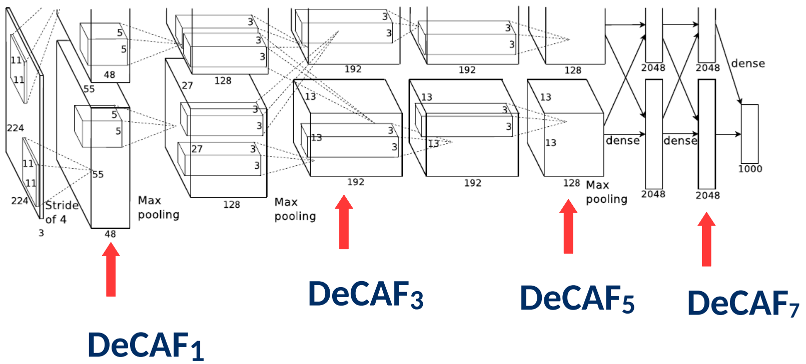


University of Antwerp

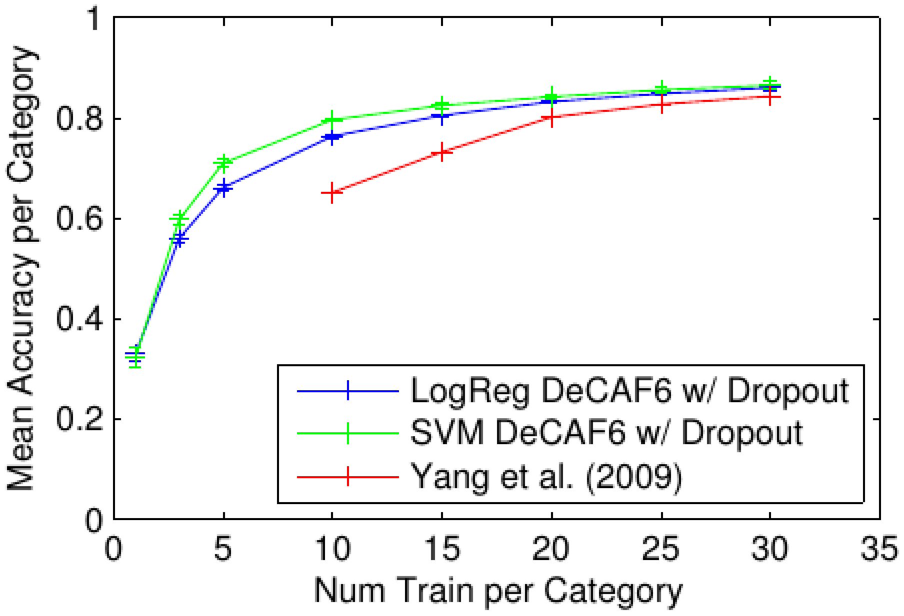
Transfer Learning

1- Pre-trained Models as Feature Extractors

– DeCaf – Quantitative Analysis



	DeCAF_5	DeCAF_6	DeCAF_7
LogReg	63.29 ± 6.6	84.30 ± 1.6	84.87 ± 0.6
LogReg with Dropout	-	86.08 ± 0.8	85.68 ± 0.6
SVM	77.12 ± 1.1	84.77 ± 1.2	83.24 ± 1.2
SVM with Dropout	-	$\mathbf{86.91 \pm 0.7}$	85.51 ± 0.9
Yang et al. (2009)		84.3	
Jarrett et al. (2009)		65.5	

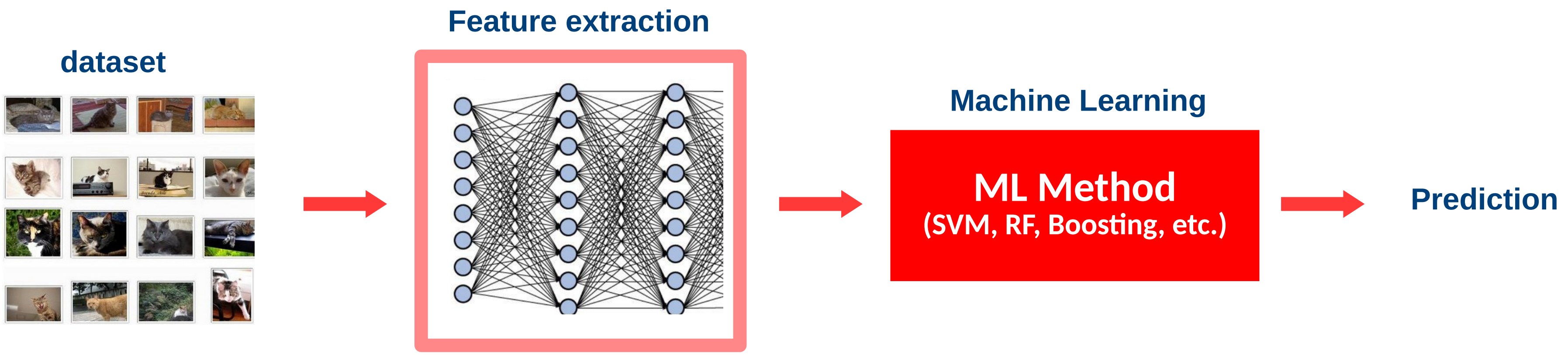


- Classification performance on the Caltech-101 dataset [30 training images/class]

Transfer Learning

1- Pre-trained Models as Feature Extractors

- Push data through the model
- Collect activations in a given layer
- Use activations as input features in a classical ML method



Q: Any potential problem or limitation? (2)

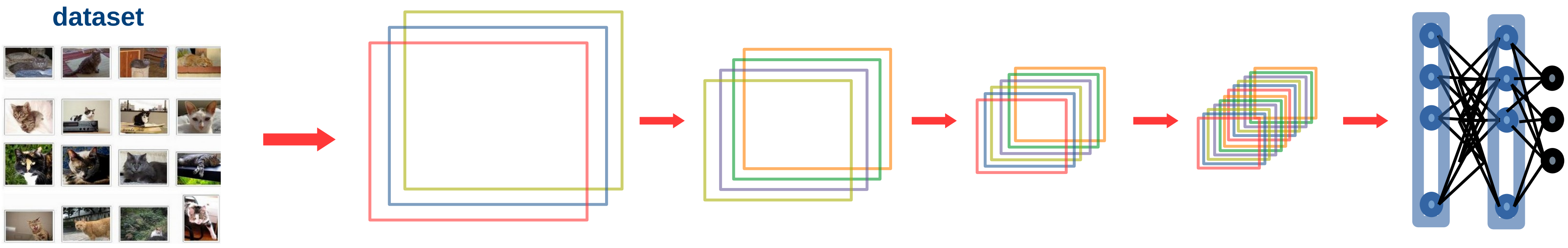
Adapt an Existing Model

[the standard practice]

Transfer Learning

2- Adapting a Pre-trained Model (aka “Fine-Tuning”)

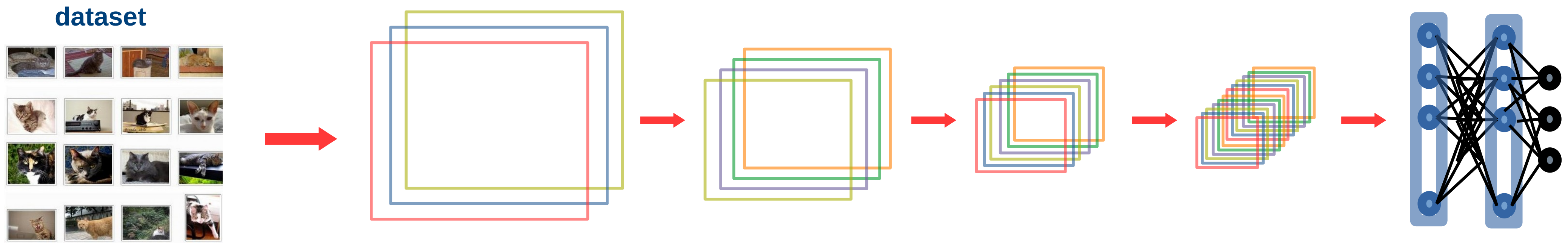
- Adjust Final Layer
- Update the weights of some layers to adapt to new tasks
- Freeze some weights, retrain others



Transfer Learning

2- Adapting a Pre-trained Model (aka "Fine-Tuning")

- Adjust Final Layer
- Update the weights of some layers to adapt to new tasks
- Freeze some weights, retrain others



Q: Any potential problem or limitation?

Break

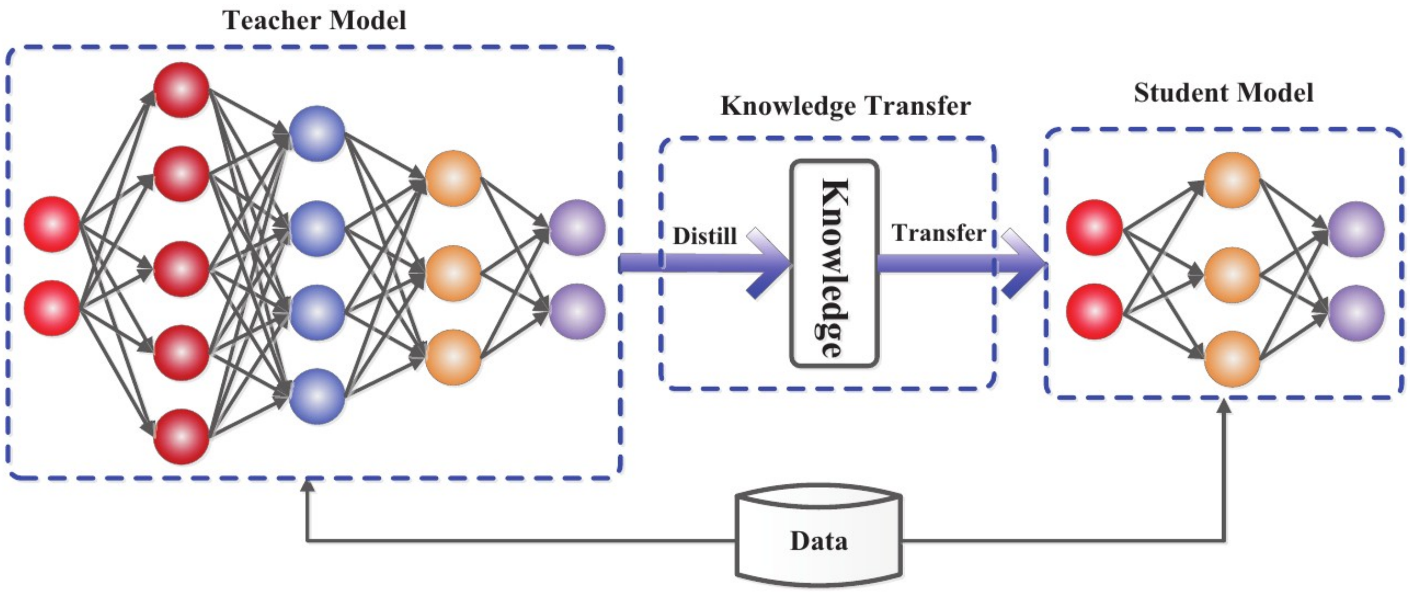
Extracting Relevant Information

[so my model can run without a GPU cluster :D]

Transfer Learning

3- Extracting Relevant Information (aka. "Distillation")

- Move knowledge from a large model to an optimized one
- Teacher-student Architecture.



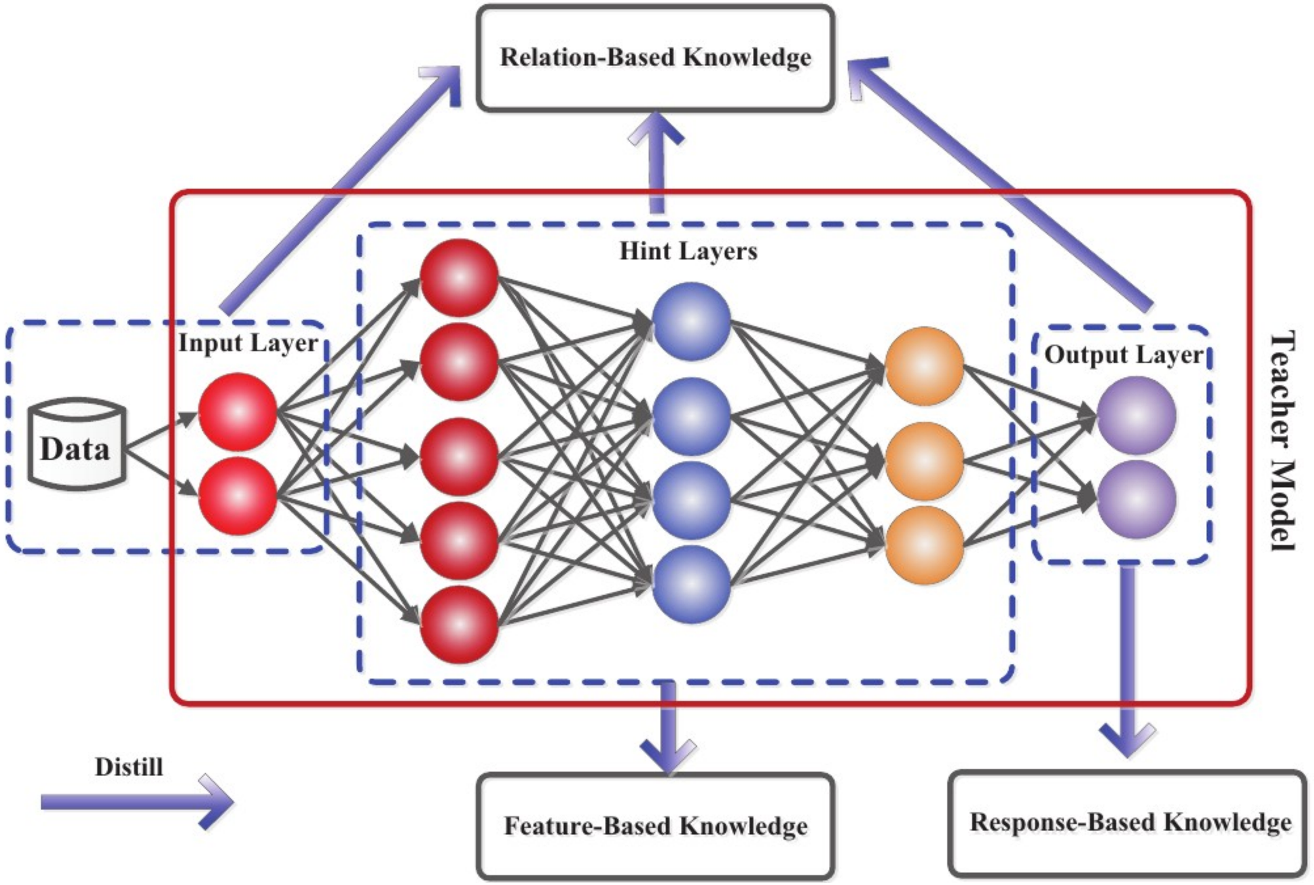
Directions

- Parameter Pruning and Sharing
  [Network Quantization]
- Identify Redundant Parameters
  [low-rank factorization]
- Compression of Conv. filters
- Knowledge Distillation

[Gou et al., 2021]

Transfer Learning

3- Extracting Relevant Information (aka. “Distillation”)



Knowledge Variants

- Responses
  - Mimic the outputs
- Features
  - Mimic outputs and intermediate states
- Relations
  - Model internal relationships

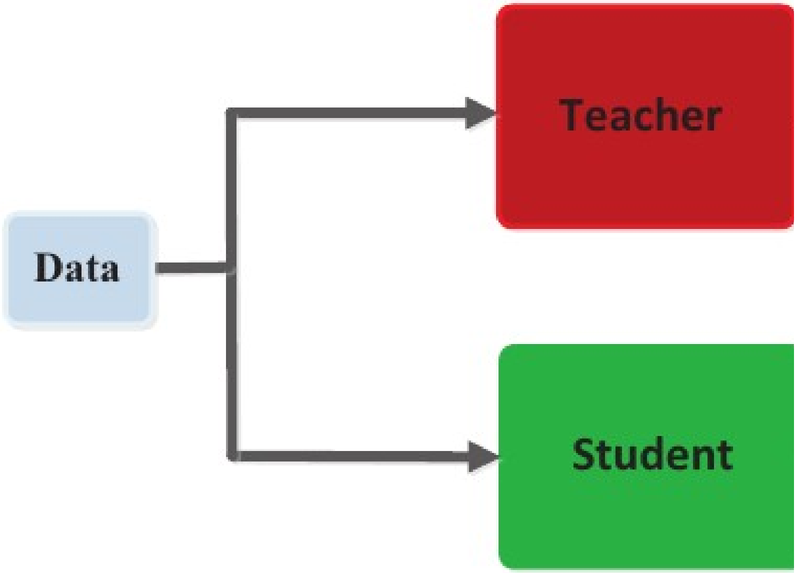
[Gou et al., 2021]

Transfer Learning

3- Extracting Relevant Information (aka. “Distillation”)

– **Response-based Knowledge**

→ Mimic the outputs



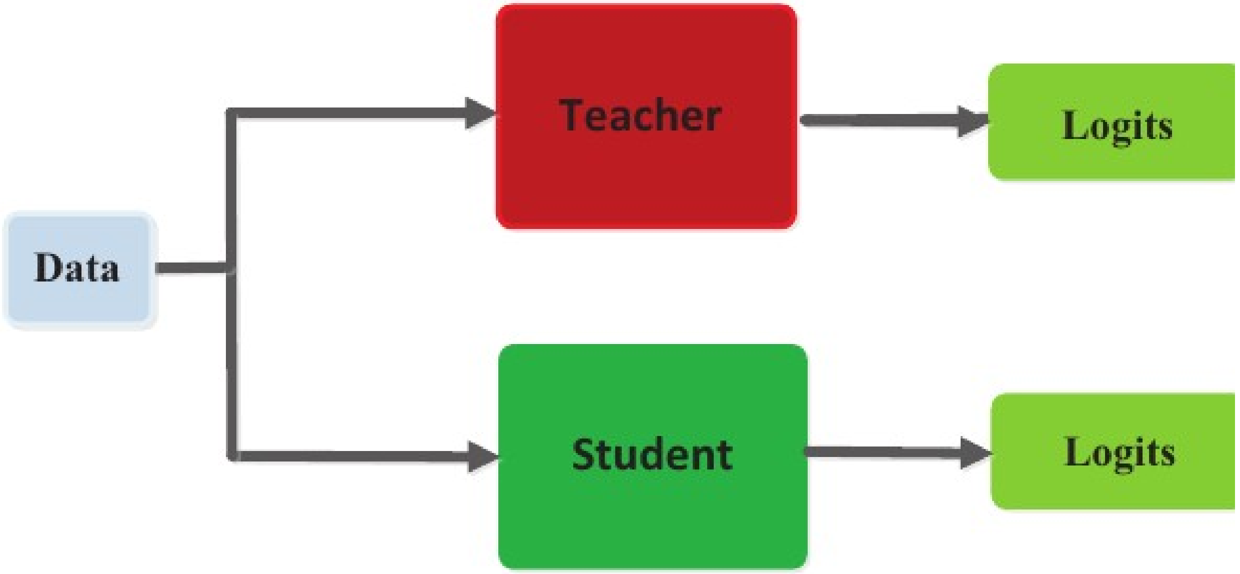
[Gou et al., 2021]

Transfer Learning

3- Extracting Relevant Information (aka. “Distillation”)

– **Response-based Knowledge**

→ Mimic the outputs



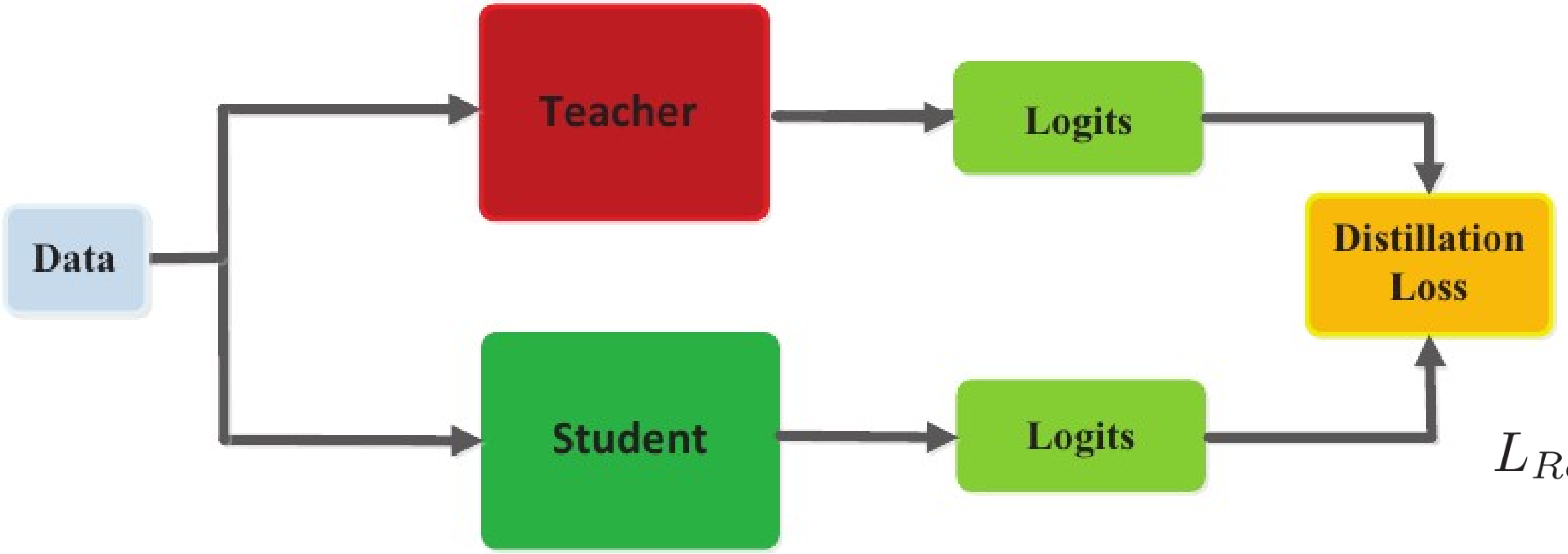
[Gou et al., 2021]

Transfer Learning

3- Extracting Relevant Information (aka. “Distillation”)

- **Response-based Knowledge**

  → Mimic the outputs



$$L_{ResD}(z_t, z_s) = \mathcal{L}_R(z_t, z_s)$$

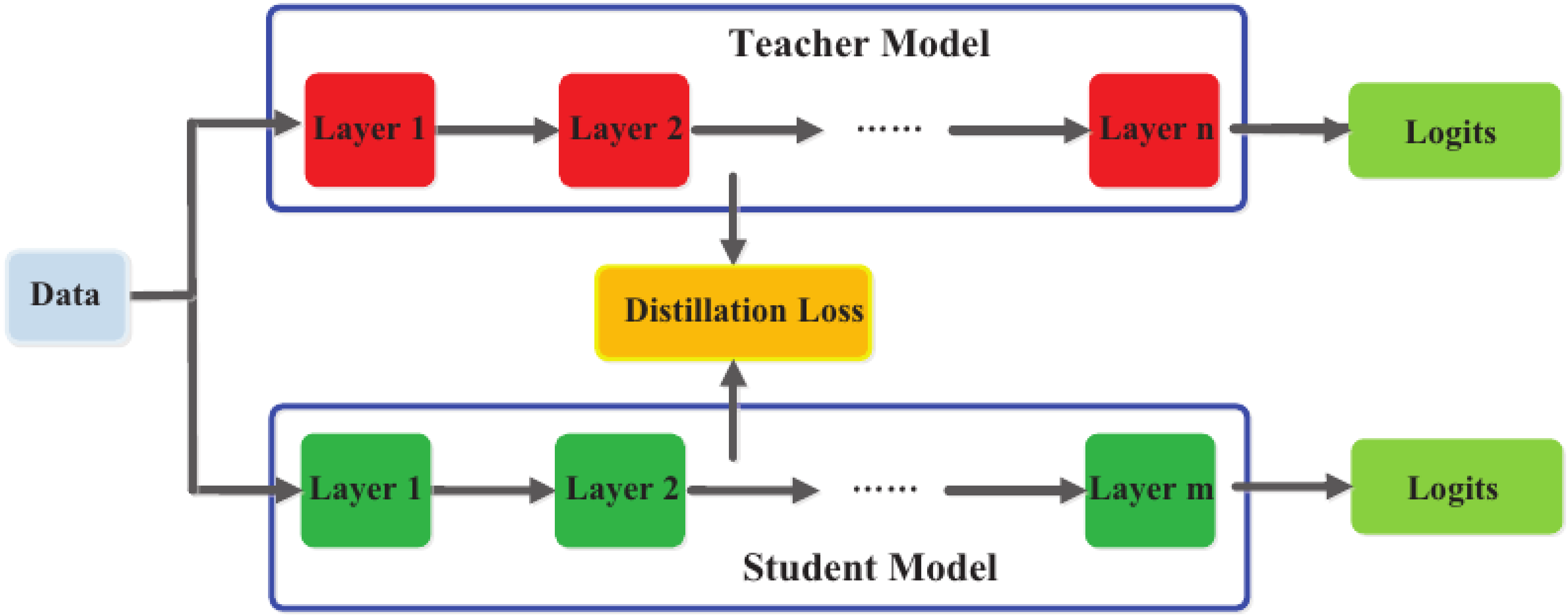
[Gou et al., 2021]

Transfer Learning

3- Extracting Relevant Information (aka. “Distillation”)

– **Feature-based Knowledge**

→ Mimic the outputs and intermediate states



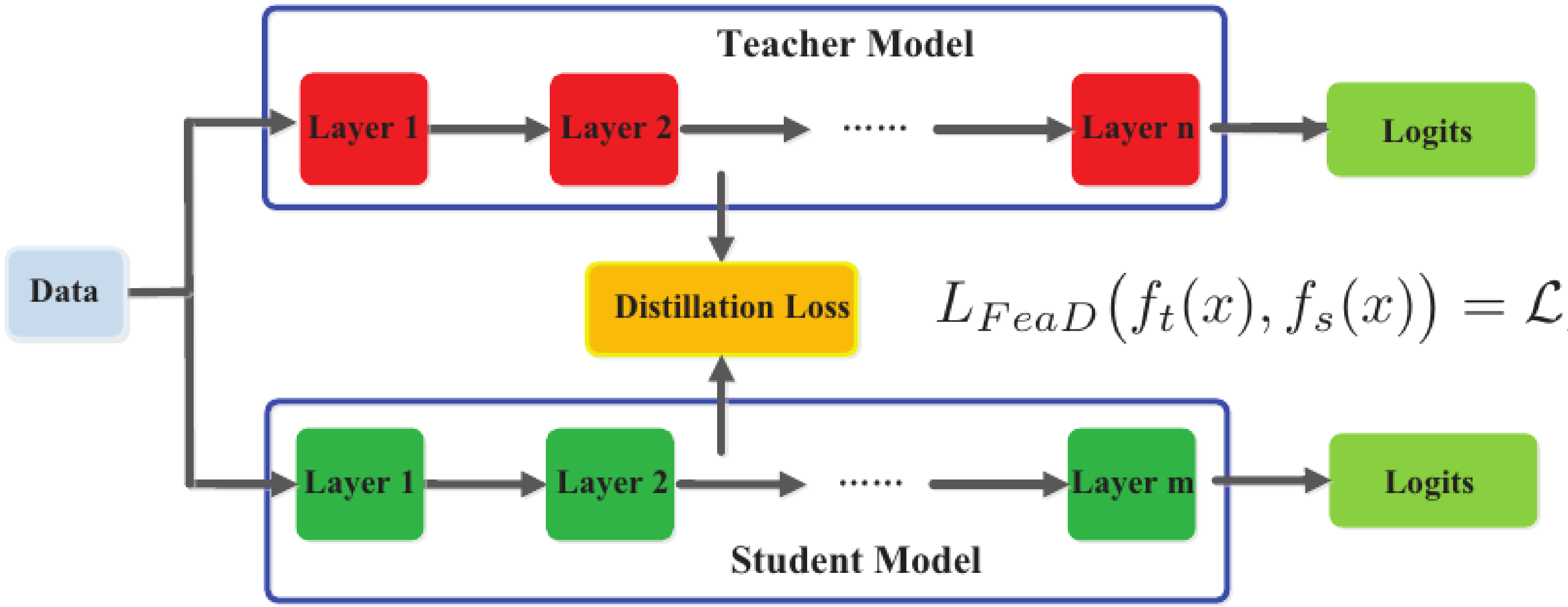
[Gou et al., 2021]

Transfer Learning

3- Extracting Relevant Information (aka. "Distillation")

- **Feature-based Knowledge**

  → Mimic the outputs and intermediate states



$$L_{FeaD}\big(f_t(x), f_s(x)\big) = \mathcal{L}_F\big(\Phi_t(f_t(x)), \Phi_s(f_s(x))\big)$$

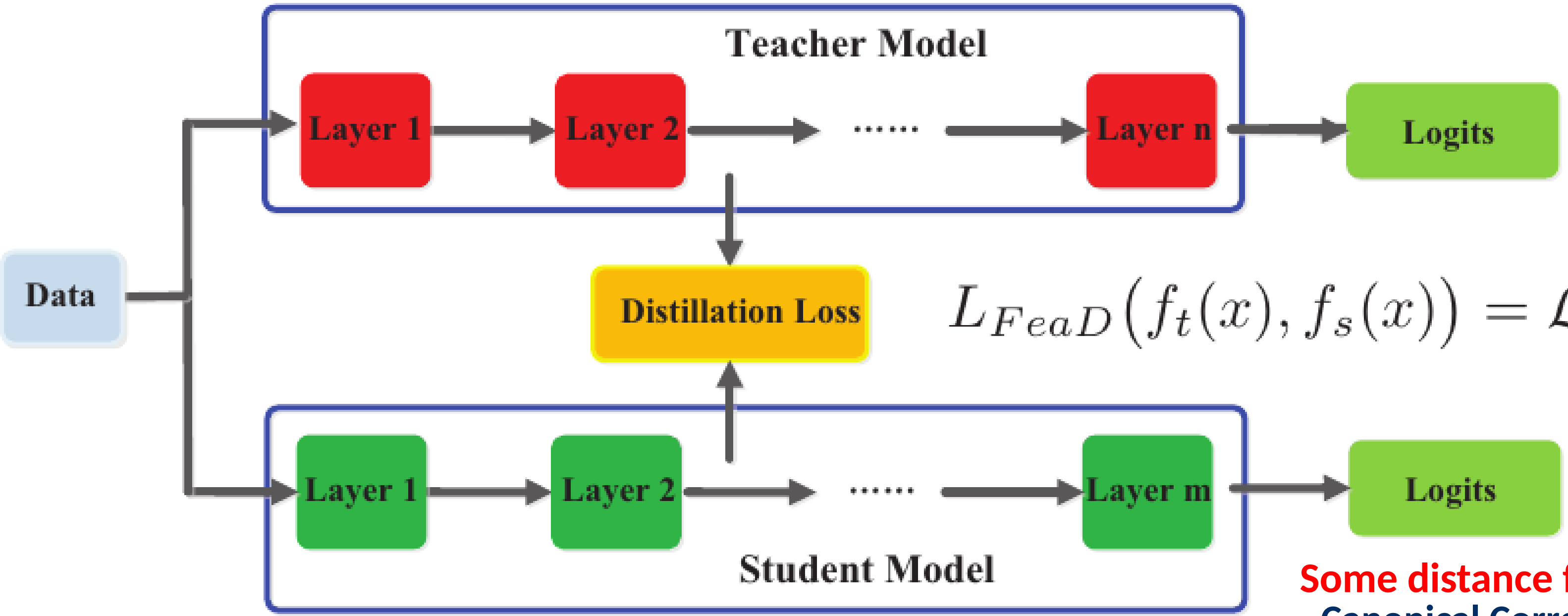
[Gou et al., 2021]

Transfer Learning

3- Extracting Relevant Information (aka. "Distillation")

– Feature-based Knowledge

→ Mimic the outputs and intermediate states



$$L_{FeaD}\left(f_t(x), f_s(x)\right) = \mathcal{L}_F\left(\Phi_t(f_t(x)), \Phi_s(f_s(x))\right)$$

[Gou et al., 2021]

Some distance functions

- **Canonical Correlation Analysis (CCA)**
- **Centered Kernel Alignment (CKA)**, [Kornblith et al. 2019]
- **Orthogonal Procrustes**, [Ding et al. 2021]

University of Antwerp

Transfer Learning

3- Extracting Relevant Information (aka. “Distillation”)

- **Relation-based Knowledge**

  → Exploit relationships between feature maps or data samples

Similarity Function

$$L_{RelD}(f_t, f_s) = \mathcal{L}_{R^1}\left(\Psi_t(\hat{f}_t, \check{f}_t), \Psi_s(\hat{f}_s, \check{f}_s)\right)$$

Features

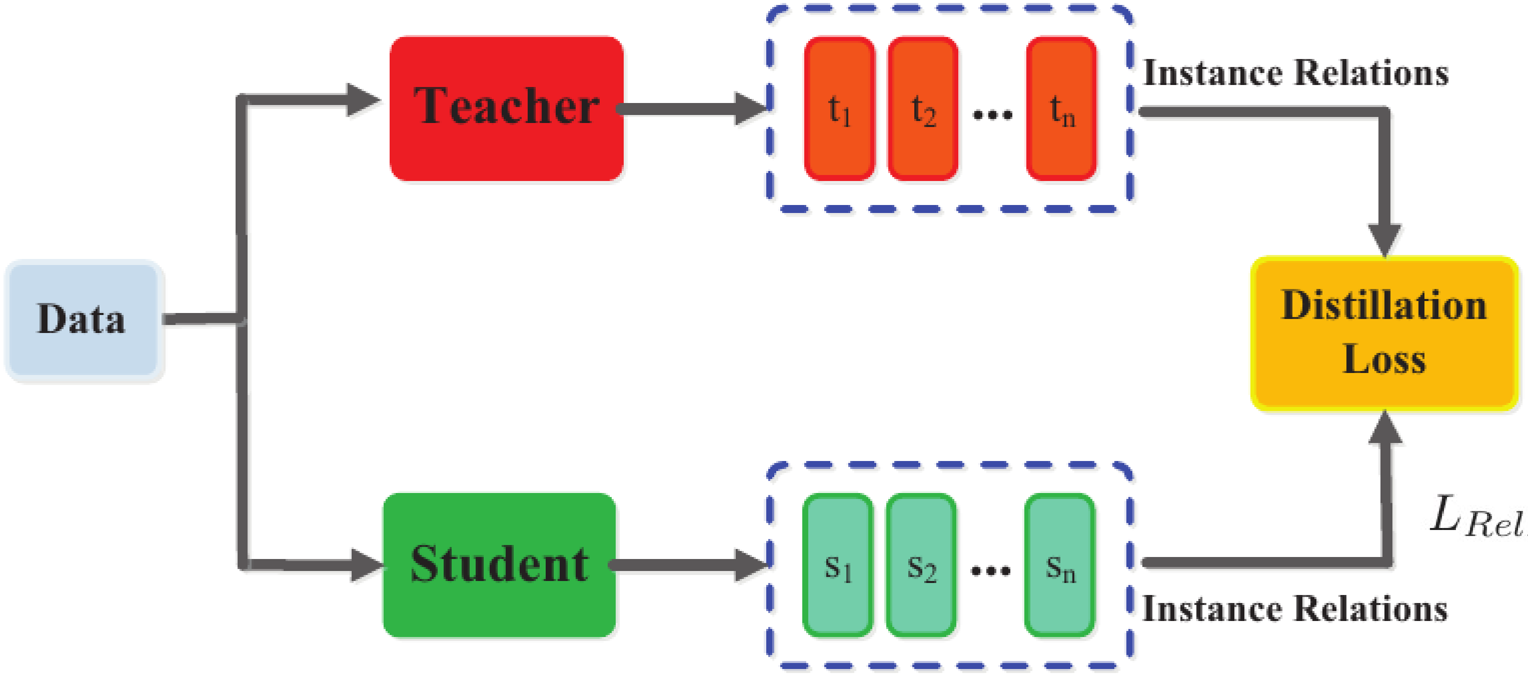
[Gou et al., 2021]

Transfer Learning

3- Extracting Relevant Information (aka. "Distillation")

- **Relation-based Knowledge**

  → Exploit relationships between feature maps or data samples



- Imposing data-level relationships

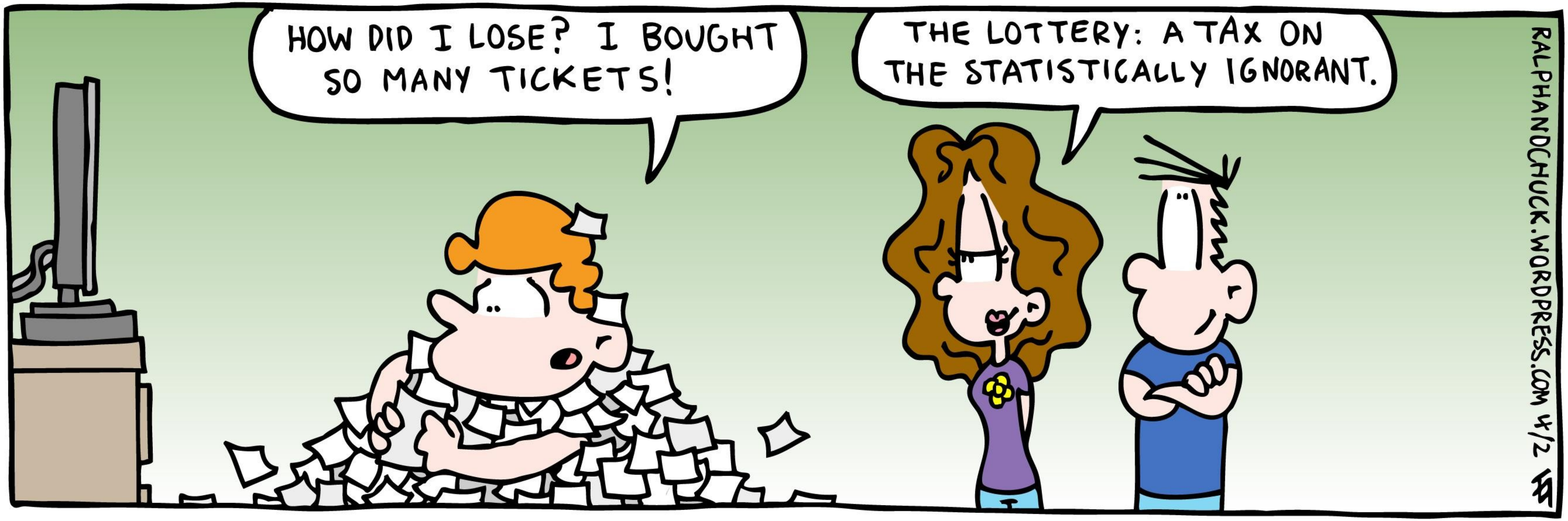
$$L_{RelD}(F_t, F_s) = \mathcal{L}_{R^2}(\psi_t(t_i, t_j), \psi_s(s_i, s_j))$$

[Gou et al., 2021]

Yes nice, but...

Is it always possible to reach a smaller model?

Lottery Ticket Hypothesis

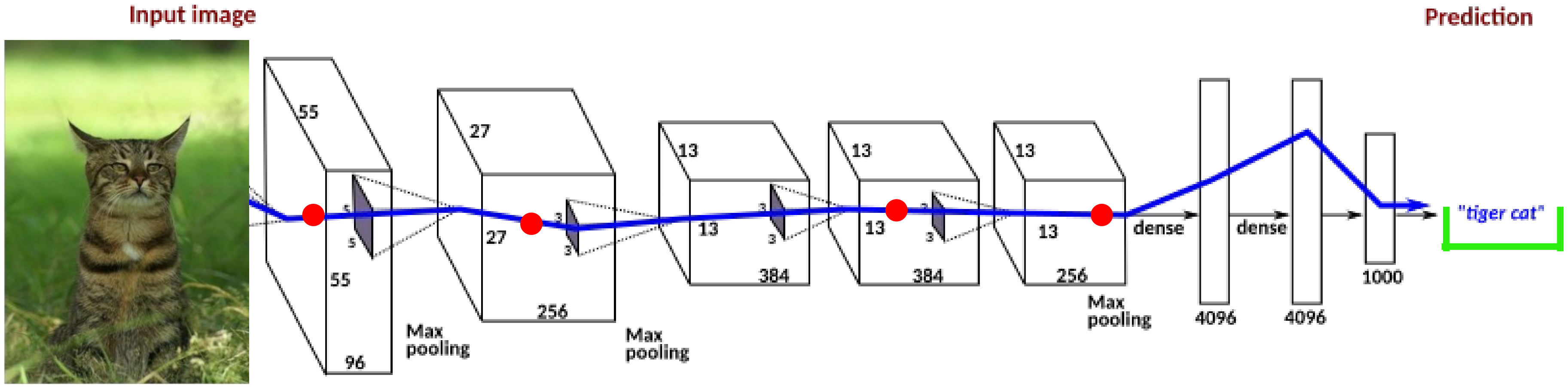


The Lottery Ticket Hypothesis. *Training succeeds for a given network if one of its subnetworks (a "winning ticket") has been randomly initialized such that it can be trained in isolation to high accuracy in at most the number of iterations necessary to train the original network.*

[Frankle et al., 2019]

Lottery Ticket Hypothesis

Fingerprinting Internal Network Activations [Oramas et al., 2019]

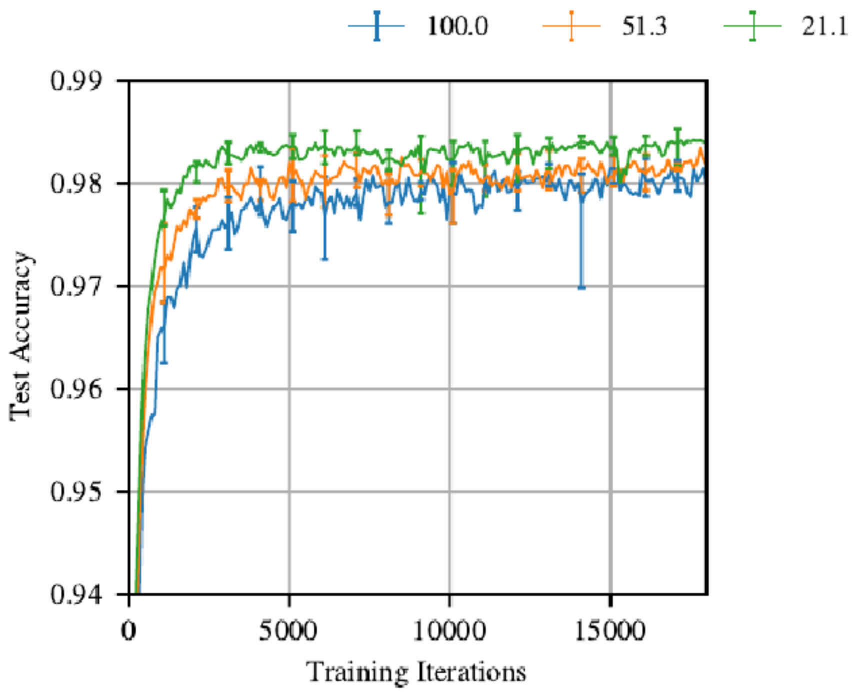


- Given a large amount of possible network initializations
- The network sucessfully trains because a sub-network was initialized properly
- This sub-network can reach similar performance to that of the complete model

[Frankle et al., 2019]

Lottery Ticket Hypothesis

- Given a large amount of possible network initializations
- The network sucessfully trains because a sub-network was initialized properly
- This sub-network can reach similar performance to that of the complete model

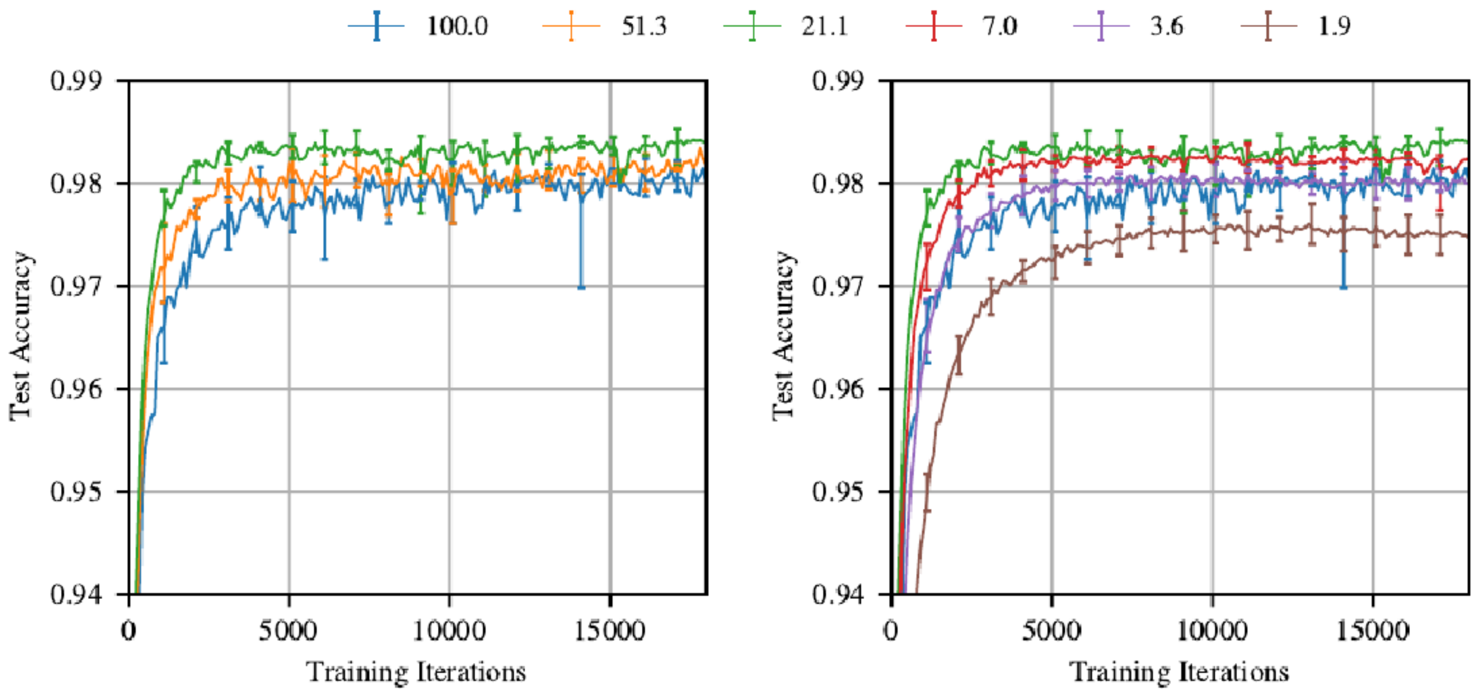


MNIST Digit Recognition with LeNet

[Frankle et al., 2019]

Lottery Ticket Hypothesis

- Given a large amount of possible network initializations
- The network sucessfully trains because a sub-network was initialized properly
- This sub-network can reach similar performance to that of the complete model

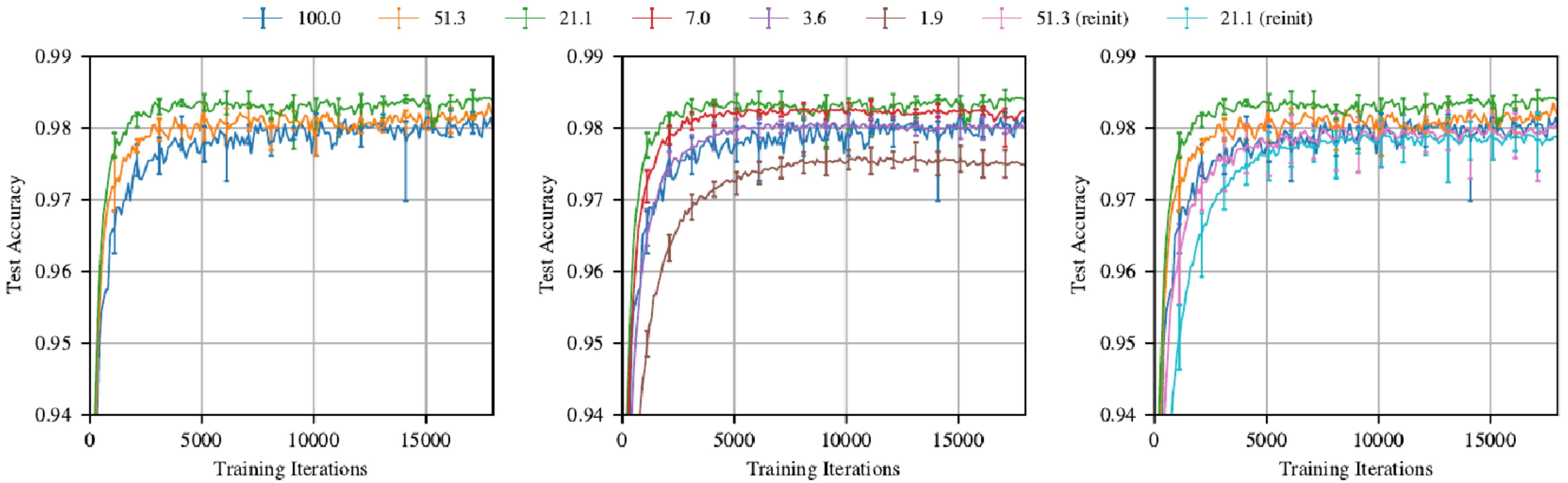


MNIST Digit Recognition with LeNet

[Frankle et al., 2019]

Lottery Ticket Hypothesis

- Given a large amount of possible network initializations
- The network sucessfully trains because a sub-network was initialized properly
- This sub-network can reach similar performance to that of the complete model



MNIST Digit Recognition with LeNet

[Frankle et al., 2019]

Lottery Ticket Hypothesis

- Given a large amount of possible network initializations
- The network sucessfully trains because a sub-network was initialized properly
- This sub-network can reach similar performance to that of the complete model

Finding the Winning Ticket

→ Iterative Magnitude Prunning (IMP) Algorithm

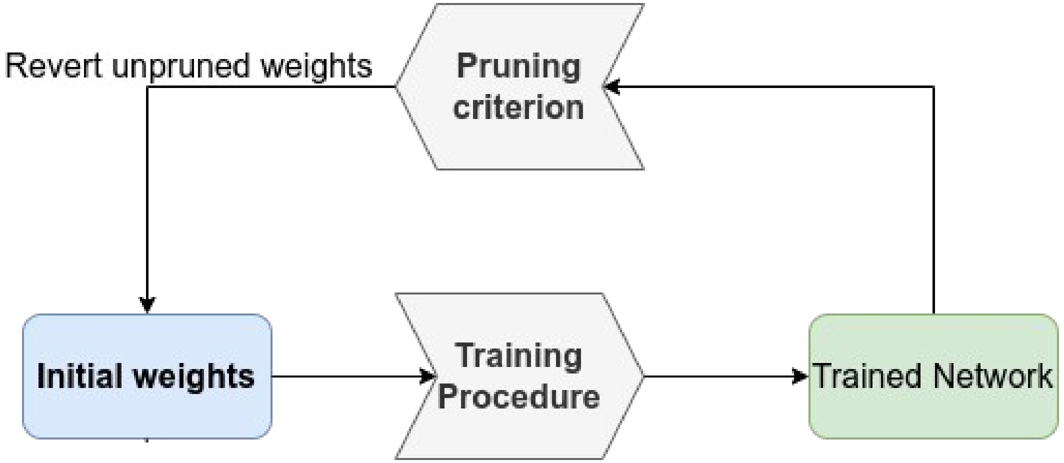
1. Randomly initialize a neural network.
2. Train the network until it converges.
3. Prune a fraction of the network.
4. Reset the weights of the remaining portion of the network to their values from (1)

   (i.e., the initializations they received before training began).

[Frankle et al., 2019]

Lottery Ticket Hypothesis

- Given a large amount of possible network initializations
- The network sucessfully trains because a sub-network was initialized properly
- This sub-network can reach similar performance to that of the complete model



Finding the Winning Ticket

→ Iterative Magnitude Prunning (IMP) Algorithm

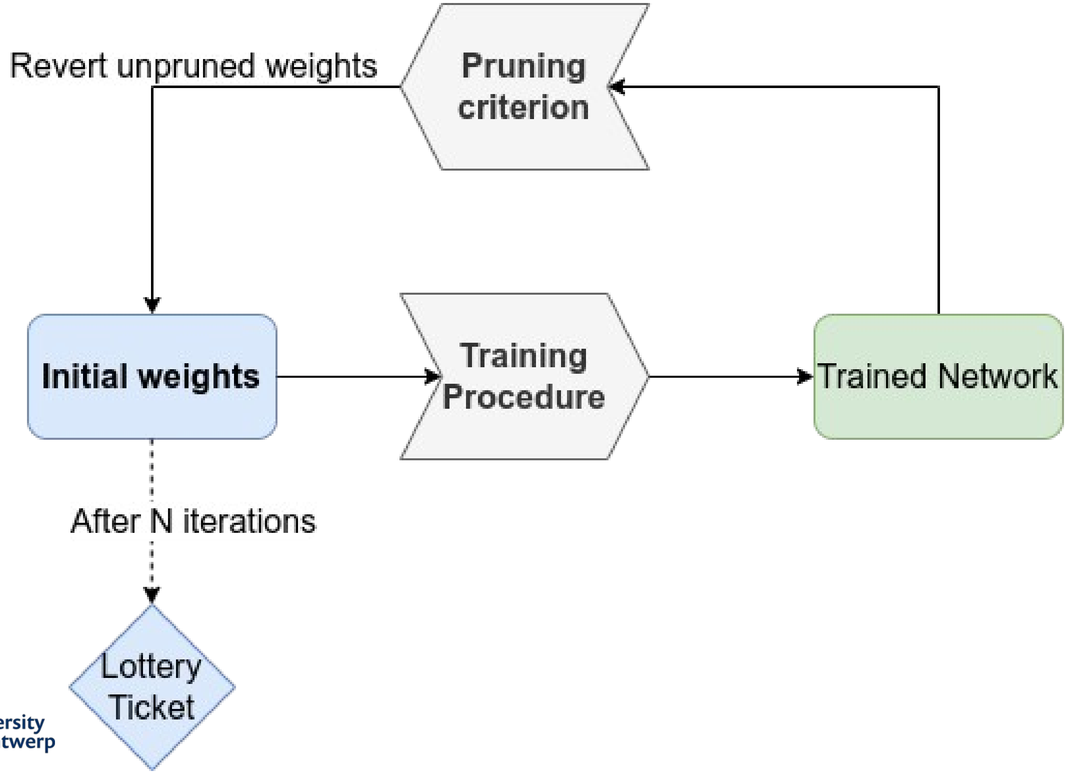
1. Randomly initialize a neural network.
2. Train the network until it converges.
3. Prune a fraction of the network.
4. Reset the weights of the remaining portion of the network to their values from (1)

   (i.e., the initializations they received before training began).

[Frankle et al., 2019]

Lottery Ticket Hypothesis

- Given a large amount of possible network initializations
- The network sucessfully trains because a sub-network was initialized properly
- This sub-network can reach similar performance to that of the complete model



Finding the Winning Ticket

→ Iterative Magnitude Prunning (IMP) Algorithm

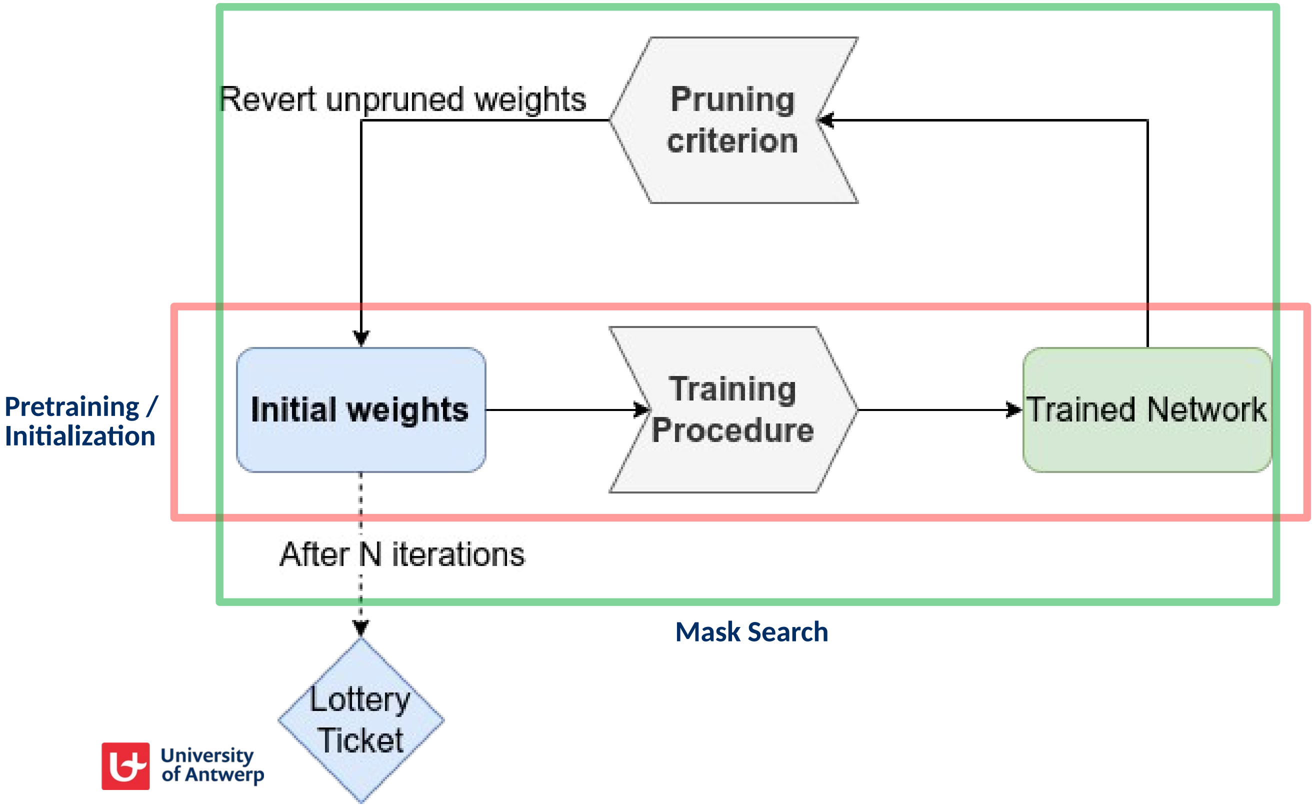
1. Randomly initialize a neural network.
2. Train the network until it converges.
3. Prune a fraction of the network.
4. Reset the weights of the remaining portion of the network to their values from (1)

   (i.e., the initializations they received before training began).

[Frankle et al., 2019]

Lottery Ticket Hypothesis

- Given a large amount of possible network initializations
- The network sucessfully trains because a sub-network was initialized properly
- This sub-network can reach similar performance to that of the complete model



Finding the Winning Ticket

→ Iterative Magnitude Prunning (IMP) Algorithm

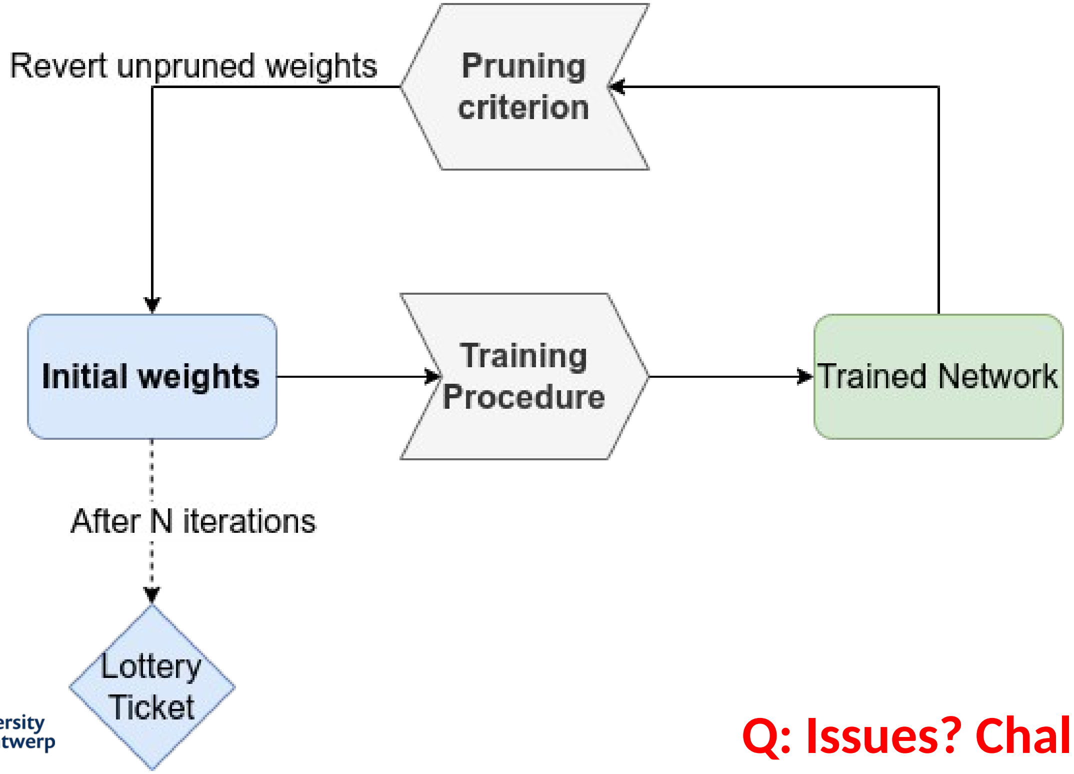
1. Randomly initialize a neural network.
2. Train the network until it converges.
3. Prune a fraction of the network.
4. Reset the weights of the remaining portion of the network to their values from (1)

   (i.e., the initializations they received before training began).

[Frankle et al., 2019]

Lottery Ticket Hypothesis

- Given a large amount of possible network initializations
- The network sucessfully trains because a sub-network was initialized properly
- This sub-network can reach similar performance to that of the complete model



Finding the Winning Ticket

→ Iterative Magnitude Prunning (IMP) Algorithm

1. Randomly initialize a neural network.
2. Train the network until it converges.
3. Prune a fraction of the network.
4. Reset the weights of the remaining portion of the network to their values from (1)

   (i.e., the initializations they received before training began).

Q: Issues? Challenges?

[Frankle et al., 2019]

Summarizing

[Finally :D]

Summarizing

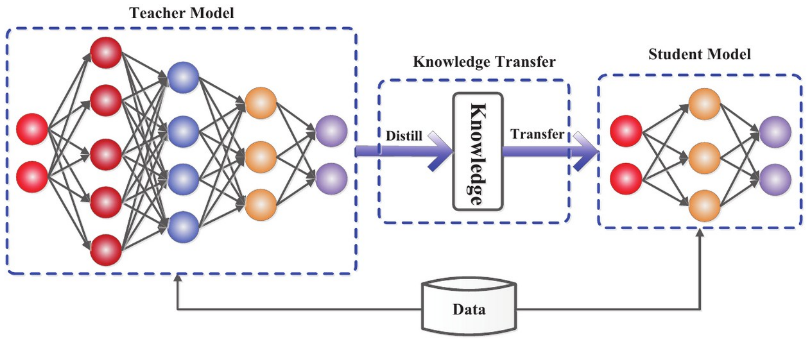
- Multiple ways to reuse pre-trained models

  With different pros and cons

  Reuse features

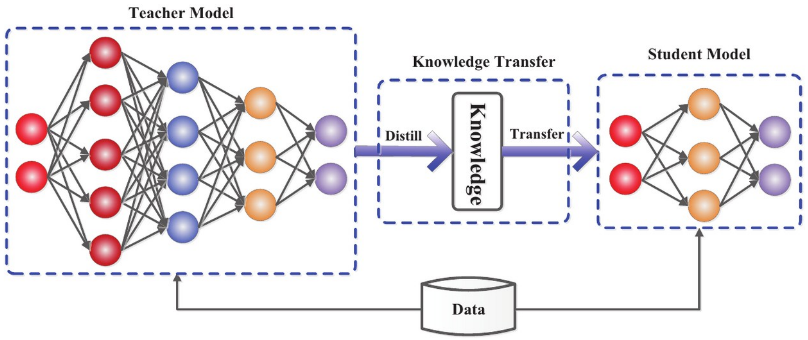
  Reuse architecture

  Optimize the architecture on a given principle

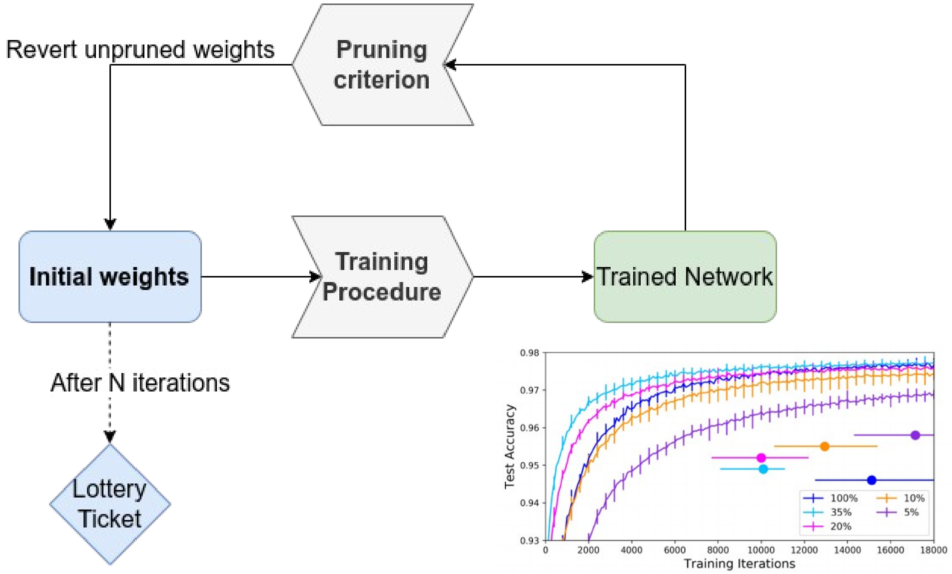


Summarizing

- Multiple ways to reuse pre-trained models
  - With different pros and cons
  - Reuse features
  - Reuse architecture
  - Optimize the architecture on a given principle



- Winning Ticket Representation
  - Sparse → leads to lighter models
  - With generalization capabilities



Questions?

References

- A. Csiszárik, P. Kőrösi-Szabó, Á. K. Matszangosz, G Papp†, D. Varga. **Similarity and Matching of Neural Network Representations**. NeurIPS 2021.
- F. Ding, J. Denain, J. Steinhardt, **Grounding Representation Similarity Through Statistical Testing**, NeurIPS 2021.
- J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, **DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition,** ICML 2014
- J Frankle, M. Carbin, **The Lottery Ticket Hypothesis: Training Pruned Neural Networks,** ICLR 2019
- J. Gou, B. Yu, S. Maybank, D. Tao, **Knowledge Distillation: A Survey,** IJCV 2021
- G. Hinton, O. Vinyals, & J. Dean. **Distilling the knowledge in a neural network**. ArXiv:1503.02531, 2015.
- S. Kornblith, M. Norouzi, H. Lee, G. Hinton. **Similarity of Neural Network Representations Revisited,** ICML 2019.
- L.J.P. van der Maaten and G.E. Hinton. **Visualizing High-Dimensional Data Using t-SNE.** Journal of Machine Learning Research 9, 2008

University of Antwerp

Artificial Neural Networks

[2500WETANN]

José Oramas