Learning agents that **communicate**

Co-training

Distillation

Multi-task Learning

MDLp: Minimum Description Length Principle

Antoine Cornuéjols

AgroParisTech – INRAE UMR MIA Paris-Saclay

antoine.cornuejols@agroparistech.fr

We continue our journey about Out-Of-Distribution learning

- What can be gained ... or lost

By resorting to **collaboration** between learning algorithms?

Questions

- Which **learning agents**?
- How to **combine** their findings?
- What kind of **information** should they **exchange**?
- How to ensure the **convergence** of the collaboration?
- If convergence takes place, **toward what**?

Outline

1. Co-learning
2. Distillation
3. Multi-task learning
4. The Minimum Description Length principle (MDLP)

Co-learning

The **co-learning** scenario

- Suppose we want to **classify web pages** as faculty member web pages or not

Blum, A., & Mitchell, T. (1998, July). *Combining labeled and unlabeled data with co-training*. In Proc. of the 11th annual conference on Computational Learning Theory (pp. 92-100).

The **co-learning** scenario

- Suppose we want to **classify web pages** as faculty member web pages or not

Blum, A., & Mitchell, T. (1998, July). *Combining labeled and unlabeled data with co-training*. In Proc. of the 11th annual conference on Computational Learning Theory (pp. 92-100).

The **co-learning** assumptions

- Examples are described using two sets of features: $x = \langle x_1, x_2 \rangle$
  - Each should be **sufficient**
  - They can be made **consistent**, i.e. $\exists\, c_1, c_2$ s.t. $c_1(x_1)=c_2(x_2)=c^*(x)$

Iterative **co-learning**

- Idea **1**: Use small set of almost certain labeled examples to **learn initial hypotheses** h_1 and h_2
  - E.g. h_1 = "My advisor" pointing to a page **xxx** is a good indicator that **xxx** is a *faculty home page*
  - E.g. h_2 = "I am teaching" on a **web page** is a good indicator that this **web page** is a *faculty home page*

- Idea **2**: Use **unlabeled** data to **propagate** learned information
  1. Look for **unlabeled** examples where **one** hypothesis is **confident** AND the **other is not**
  2. Have it **label the examples** so that the other learning algorithm can use it

Iterative **co-learning**

- Repeat
  1. Look through **unlabeled** data to find examples where one of the h_i is **confident** but the other **is not**
  2. Have the confident h_i label it for algorithm A_{3-i}

h_1 and h_2 are initially learnt on a subset of common examples where they find consistent labeling

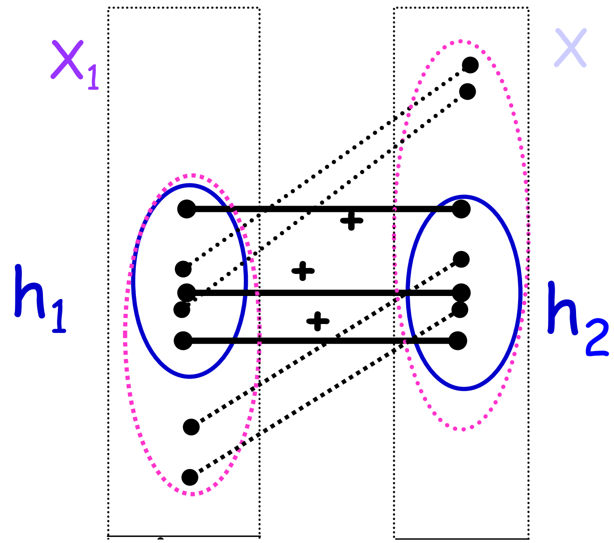


Illustration on Webpage classification

- 12 **labeled** examples
- 1000 unlabeled

Results for 5-folds cross validation

Default prediction: negative (22% test error)

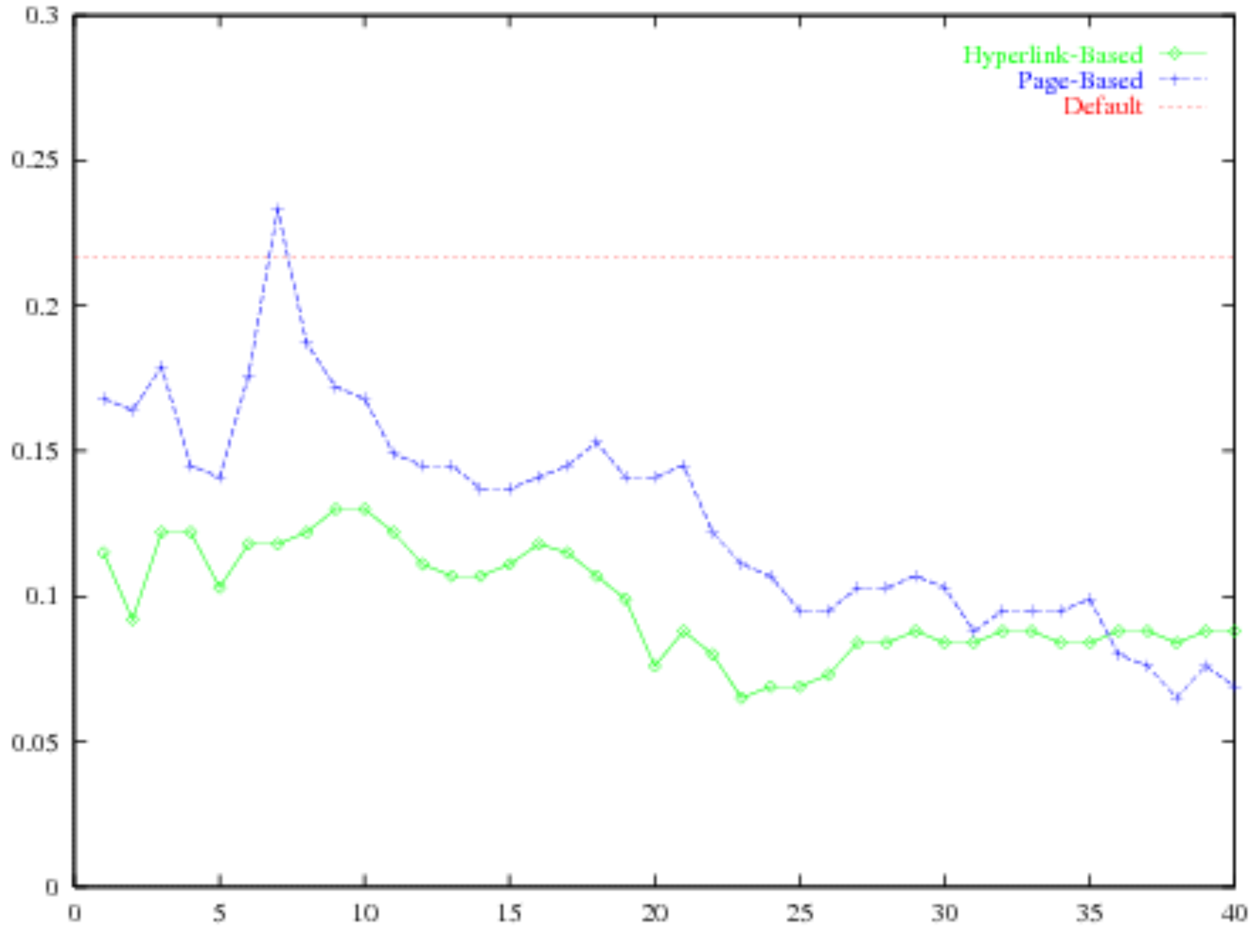
	Page-based classifier	Hyperlink-based classifier	Combined classifier
Supervised training	12.9	12.4	11.1
Co-training	6.2	11.6	5.0

Table 2: Error rate in percent for classifying web pages as course home pages. The top row shows errors when training on only the labeled examples. Bottom row shows errors when co-training, using both labeled and unlabeled examples.

Blum, A., & Mitchell, T. (1998, July). *Combining labeled and unlabeled data with co-training*. In Proc. of the 11th annual conference on Computational Learning Theory (pp. 92-100).

Classification of Webpages

...



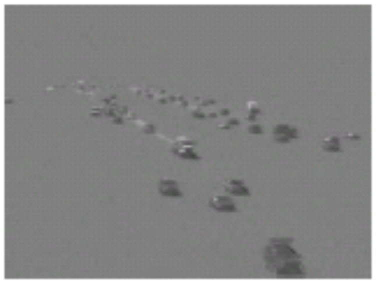
Percent error on test data

Co-learning **iterations**

Applied in many other settings

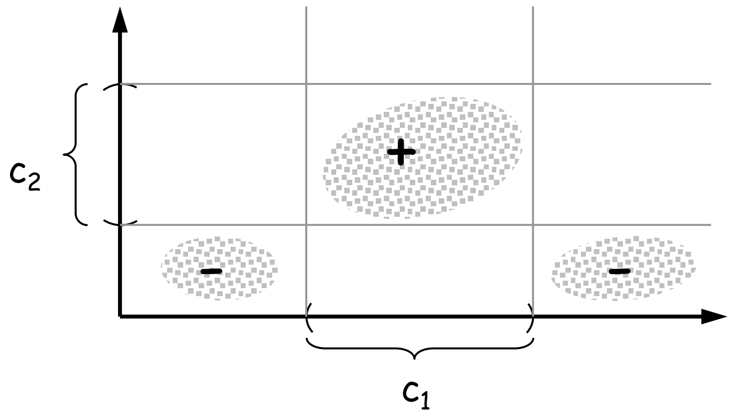
- Named-entity extraction [Collins & Singer, 99]
  - "I arrived in London yesterday"

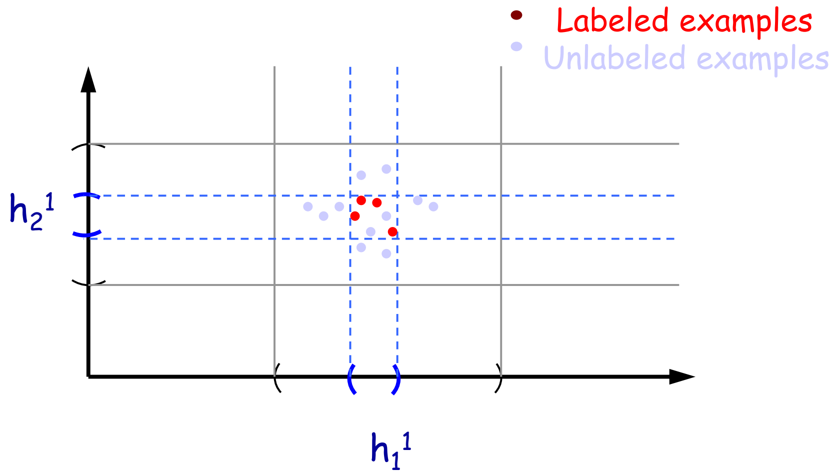
- Identifying objects in images using two different types of preprocessing [Levin, Viola, Freund, 03]




Iterative **co-learning**: simple example

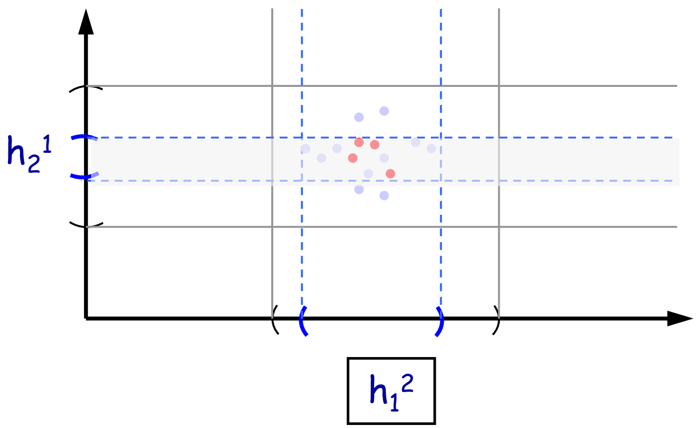
- Learning intervals

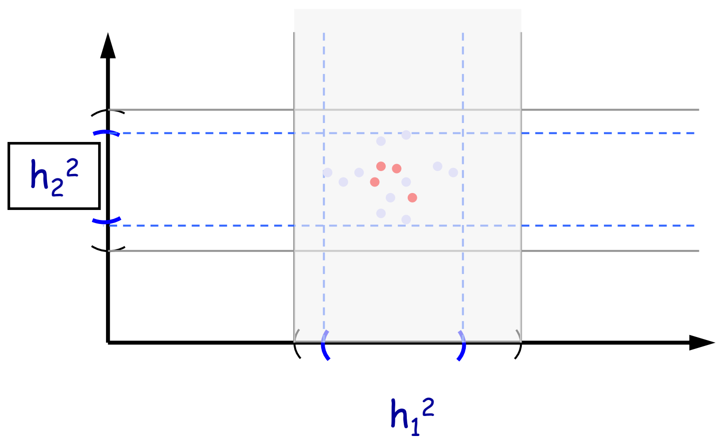




Use labeled data to learn h_1^1 and h_2^1

Use unlabeled data to bootstrap





Co-learning and multi-view Semi Supervised Learning

- Given

$$\mathcal{S}_l = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{m_l}, y_{m_l})\}$$

$$\mathcal{S}_u = \{(\mathbf{x}_{m_l+1}, y_{m_l+1}), \ldots, (\mathbf{x}_{m_u}, y_{m_u})\}$$

Find h_1 and h_2

$$\operatorname*{ArgMin}_{h_1,h_2} \underbrace{\sum_{l=1}^{2} \sum_{i=1}^{m_l} \ell(h_l(\mathbf{x}_i), y_i)}_{\text{Small labeling error}} + \lambda \underbrace{\sum_{j=m_l+1}^{m_u} \text{agreement}(h_1(\mathbf{x}_j), h_2(\mathbf{x}_j))}_{\text{Regularizer to encourage agreement over unlabeled data}}$$

Small **labeling** error

Regularizer to encourage agreement over **unlabeled** data

[Bartlett, Rosenberg, AISTATS-2007], [Sridharan, Kakade, COLT-2008]

Analysis

- Co-training is a method for **using unlabeled data** when examples can be partitioned into **two views** such that:

  1. each **view** in itself is **at least roughly sufficient** to achieve good classification,

  2. and yet the views are **not too highly correlated**.

[Blum & Mitchell, COLT-98]

1. Independence of examples given the labels
2. Algorithm for learning from random classification noise

[Balcan, Blum & Yang, NIPS-2004]

1. Property of distributional expansion on the examples
2. Algorithm for learning from positive data only

Blending

[Mark Turner, Gilles Fauconnier: *The Way We Think. Conceptual Blending and the Mind's Hidden Complexities*. New York: Basic Books 2002]

Blending effect [Fauconnier & Turner]

The **Riddle** of the Buddhist Monk:

A Buddhist monk begins **at dawn** one day **walking up a mountain**, reaches the top **at sunset**, meditates at the top overnight until, **at dawn**, he begins to **walk back** to the foot of the mountain, which he reaches **at sunset**.

Blending effect [Fauconnier & Turner]

A Buddhist monk begins **at dawn** one day **walking up a mountain**, reaches the top **at sunset**, meditates at the top overnight until, **at dawn**, he begins to **walk back** to the foot of the mountain, which he reaches **at sunset**.

- Make **no assumptions** about his **starting** or **stopping** or about his **pace** during the trips.

- Riddle: ***is there a place*** *on the path that the monk occupies* ***at the same hour of the day*** *on the two trips*?

- As we went to press, Rich Wilson and Bill Biewenga, on *Great America II*, their catamaran, were barely **maintaining a 4.5 day lead** over the clipper *Northern Light* whose record run from San Francisco to Boston, in 1853, was 76 days and 8 hours.

Watch out, they are sailing in **1993**, 140 years later,
and they have a **4.5 day lead**!!?

(as if they were in a race!)

Outline

1. Co-learning
2. Distillation
3. Multi-task learning
4. The Minimum Description Length principle (MDLP)

First example:

Learning Neural Networks

using "distillation"

Motivation

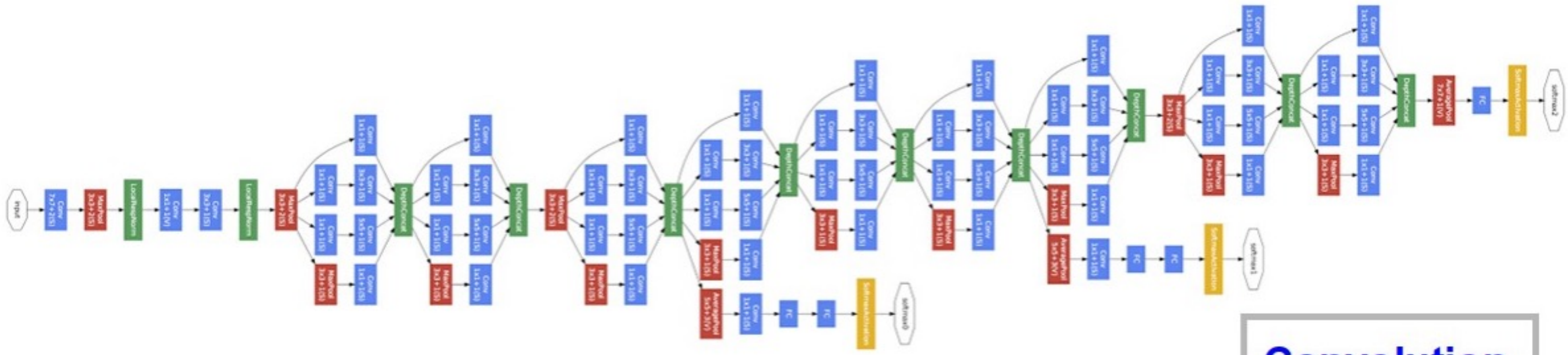
1. We would like to deploy a classifier (NN) on a **computationally limited device** (e.g. *a smartphone*)
   – A deep NN cannot be used

2. The **learning task is difficult** and requires a large data set and a sophisticated learning method (e.g. a deep NN)

Question: can we use the learned deep NN as a **teacher** to help the **student** (i.e. the limited device) learn a simpler classifier?

Motivation

Example: A sophisticated learning technique - GoogLeNet

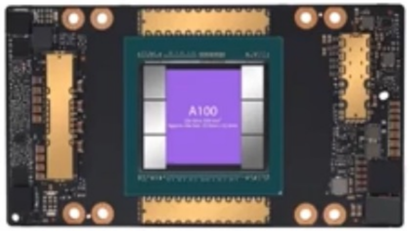


Convolution
Pooling
Softmax
Other

Quite a costly machine to train

AND to use for prediction

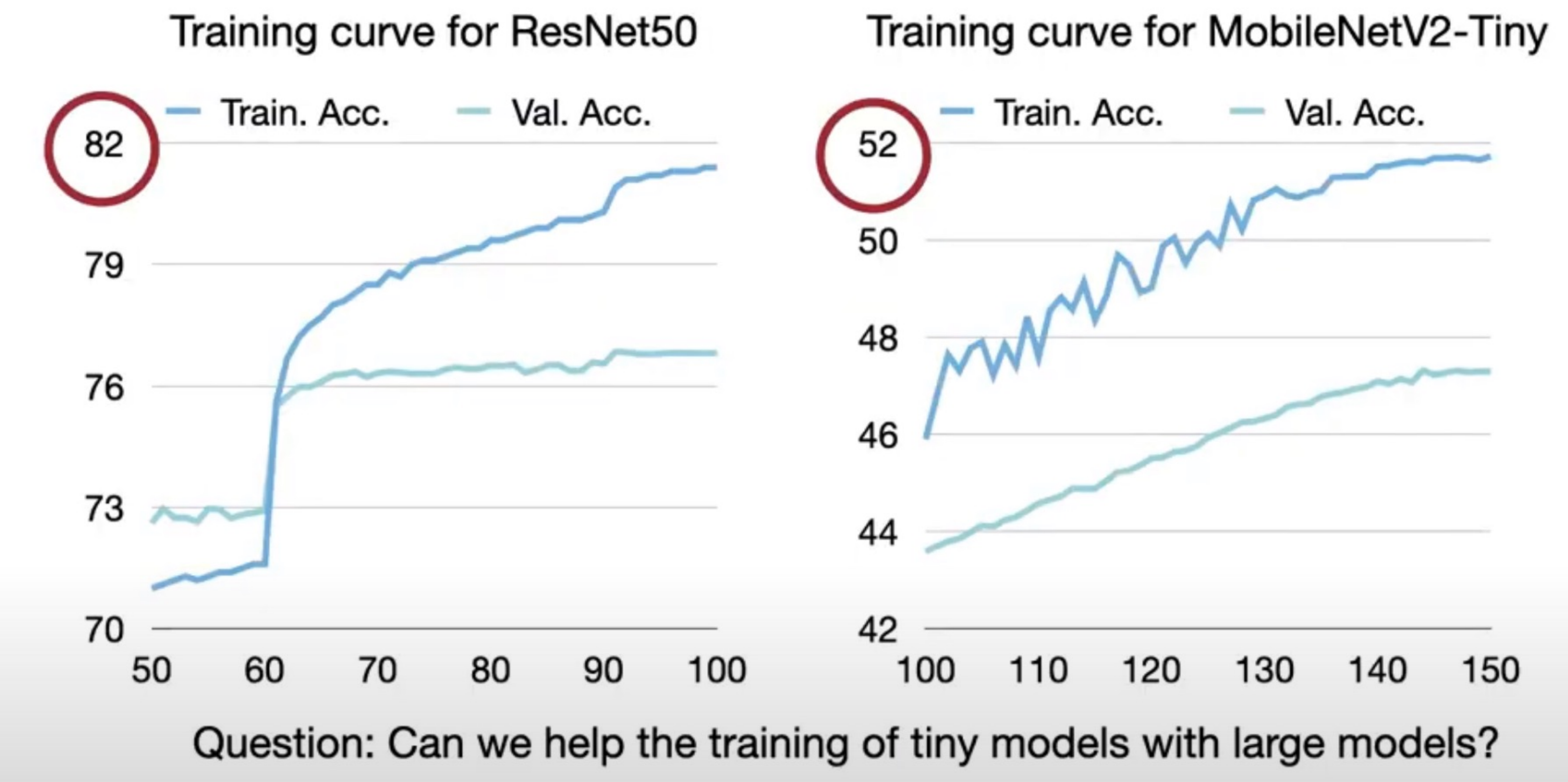
Motivation



	Cloud AI	Tiny AI
Computation (fp32)	19.5 TFLOPS	MFLOPs
Memory	80GB	256kB
Neural Network	ResNet ViT-Large ...	MCUNet MobileNetV2-Tiny ...

- Neural network must be **tiny** to run efficiently on tiny edge devices.

...


Training curve for ResNet50
Train. Acc.
Val. Acc.
82
79
76
73
70
50
60
70
80
90
100
Training curve for MobileNetV2-Tiny
Train. Acc.
Val. Acc.
52
50
48
46
44
42
100
110
120
130
140
150
Question: Can we help the training of tiny models with large models?

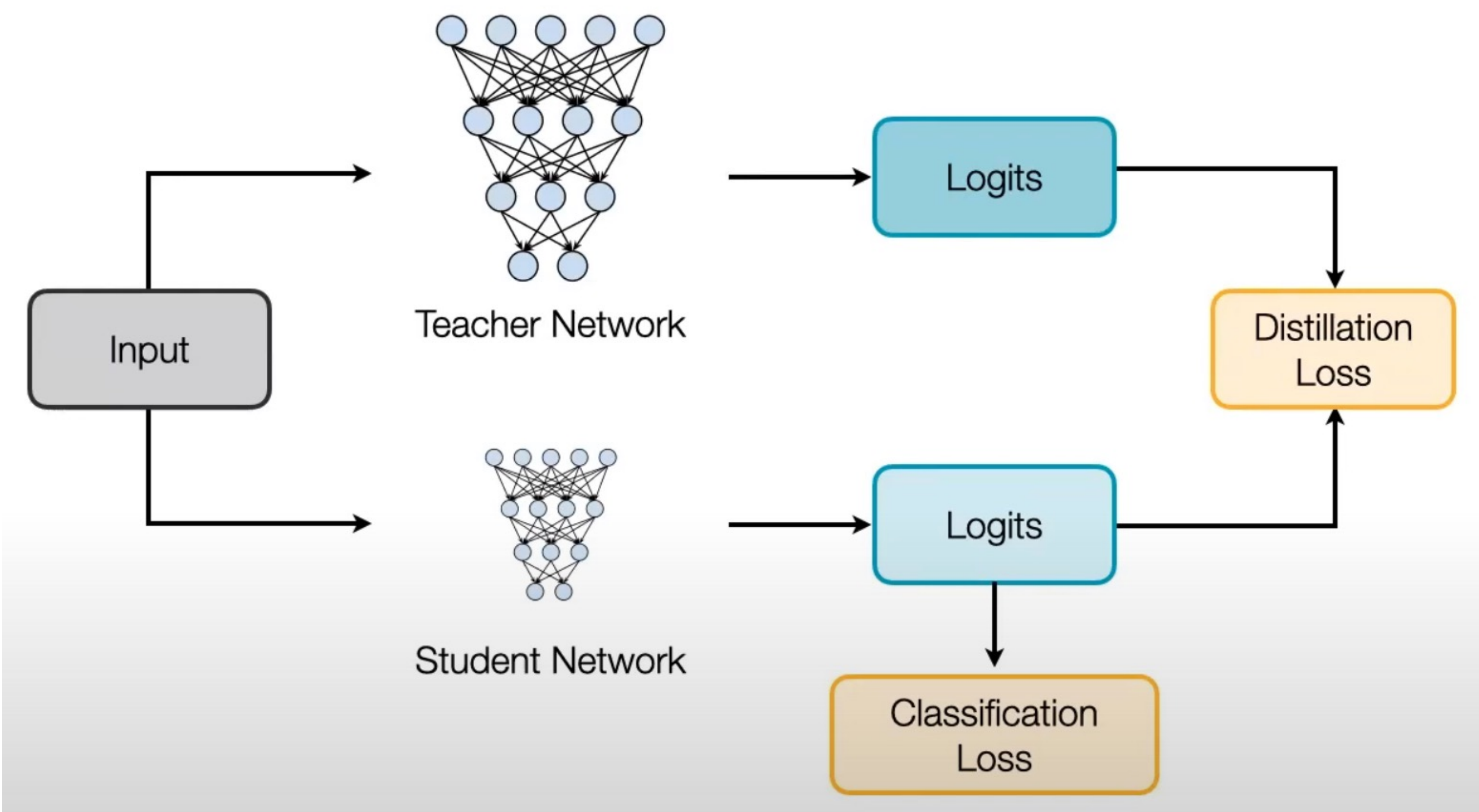
...

Learning techniques for “distillation”

1. Gradually changing the **targets**

2. Gradually changing the **inputs**

3. Gradually changing the **learning task**


Input
Teacher Network
Student Network
Logits
Logits
Distillation Loss
Classification Loss

...

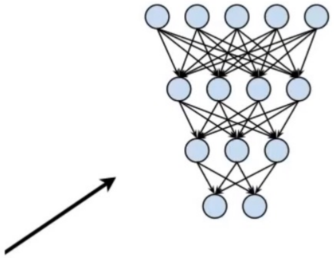
Learning techniques for “distillation”

1. Gradually changing the **targets**

2. Gradually changing the **inputs**

3. Gradually changing the **learning task**

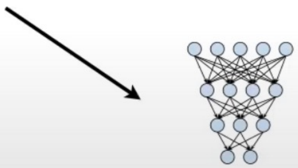
Matching prediction probabilities between teacher and student



Teacher Network

	Logits	Probabilities
Cat	5	0.982
Dog	1	0.017

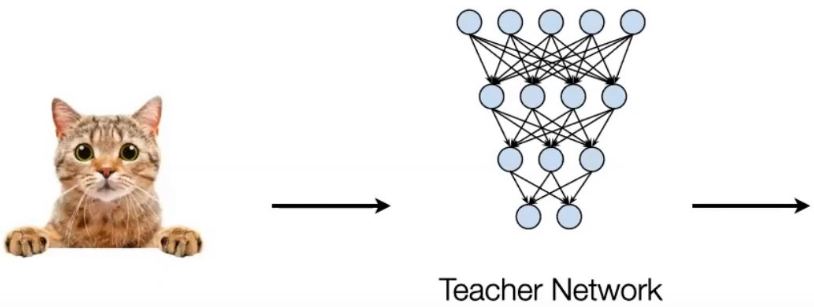
$$\frac{\exp(5)}{\exp(5) + \exp(1)}$$



Student Network

	Logits	Probabilities
Cat	3	0.731
Dog	2	0.269

...



$$\frac{\exp(5/1)}{\exp(5/1)+\exp(1/1)}$$

	Logits	Probabilities (T=1)	Probabilities (T=10)
Cat	5	0.982	0.599
Dog	1	0.017	0.401

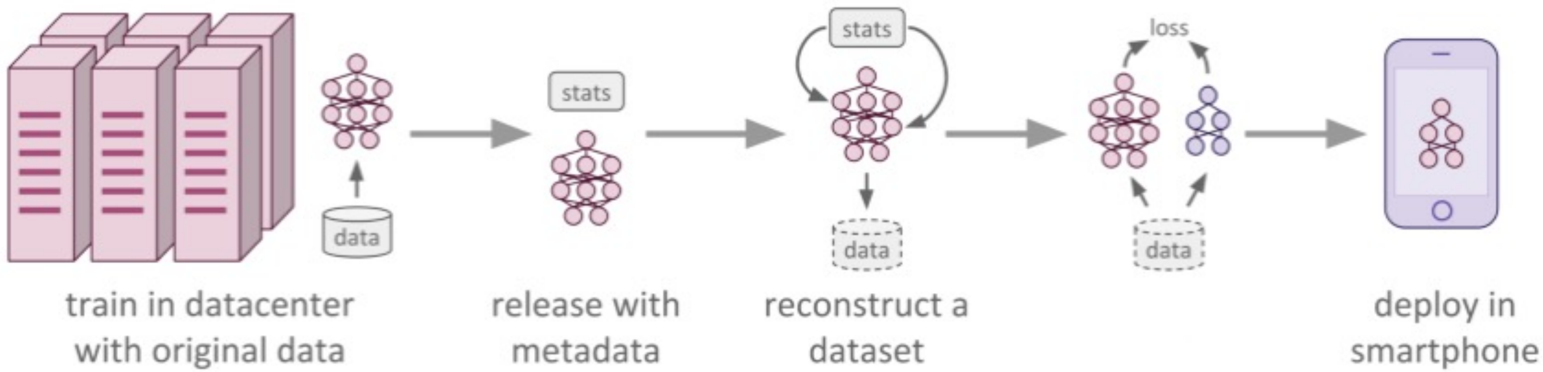
$$\frac{\exp(5/10)}{\exp(5/10)+\exp(1/10)}$$

A larger temperature smooths the output probability distribution.

Changing the **target**

1. Use the sophisticated learning method (teacher) to learn to predict the target classes with a **membership measure**

2. Ask the student to *learn to predict the* ***membership measure*** computed by the teacher instead of the hard classes (on the training set)



Changing the target

1. The teacher uses a softmax function for the values of its output

$$q_i = \frac{e^{(z_i/T)}}{\sum_{j \in \text{classes}} e^{(z_j/T)}}$$

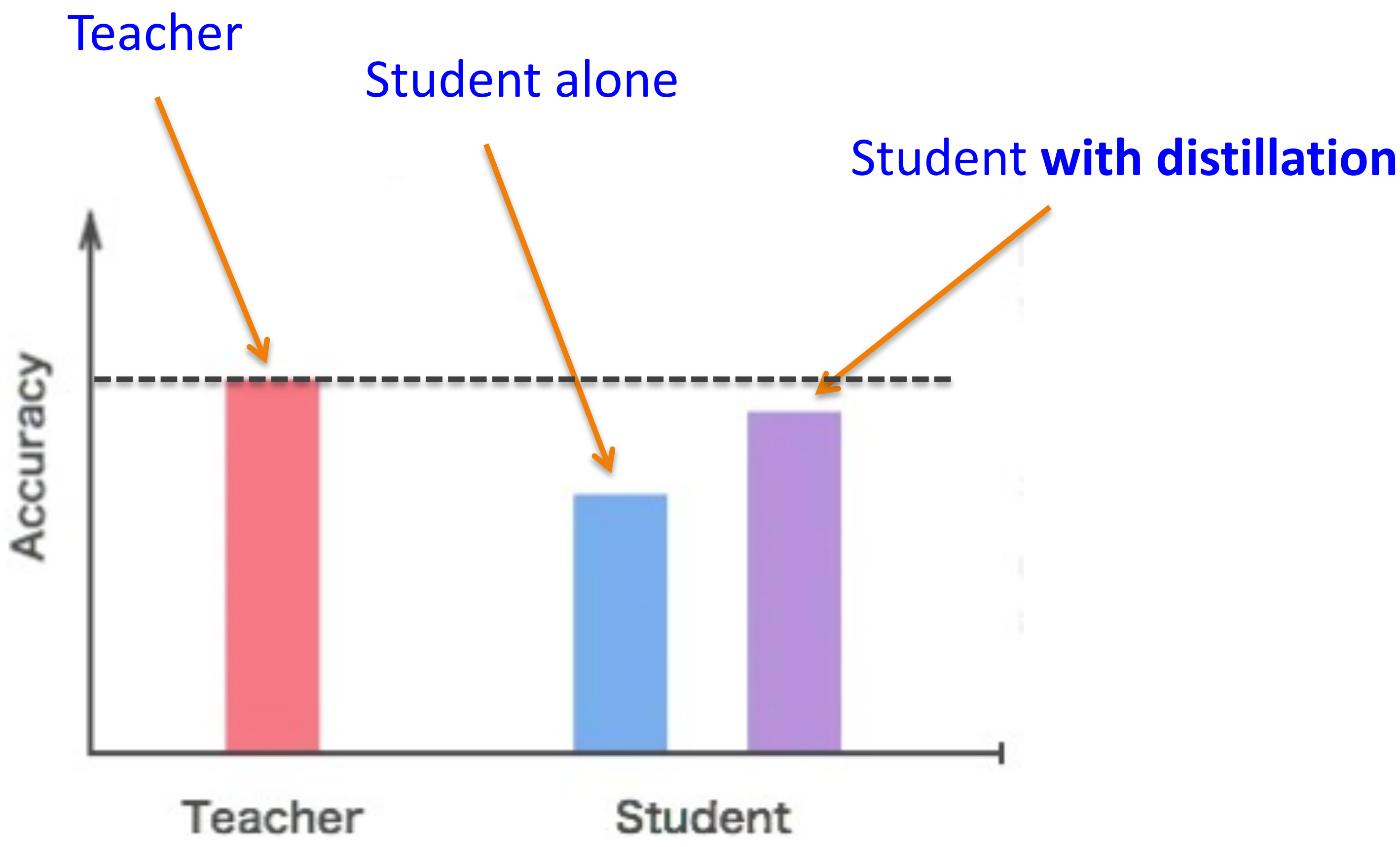
   T is the temperature (the highest *T*, the less different are the outputs)

2. The student *learns to predict the membership measure* first with *T* high, and then, progressively, with *T* decreasing to 1.

When the soft targets have high entropy, they **provide much more information per training case** than hard targets and **much less variance in the gradient** between training cases, so the small model can often be trained on much less data than the original cumbersome model while using a much higher learning rate.

Changing the target

...



Learning techniques for “distillation”

1. Gradually changing the **targets**

2. Gradually changing the **inputs**

3. Gradually changing the **learning task**

Changing the **inputs**

- Idea: friendly training vs. adversary learning
  - Modifies the inputs so as **to facilitate** the training
- **Modifies** the descriptions of the **examples**
  - According to the current training stage $\tilde{x}_i \;=\; x_i + \delta_i$
  - So as to minimize: $$L(\mathcal{B}, w) \;=\; \frac{1}{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} \ell\big(f(\tilde{x}_i, w), y_i\big)$$

Marullo, S., Tiezzi, M., Gori, M., & Melacci, S. (2021). **Being Friends Instead of Adversaries: Deep Networks Learn from Data Simplified by Other Networks**. *arXiv preprint arXiv:2112.09968.*

Neural Friendly Training

- But the modifications are **independently** applied to all training examples
- We would rather like global deformations that help to learn the decision function

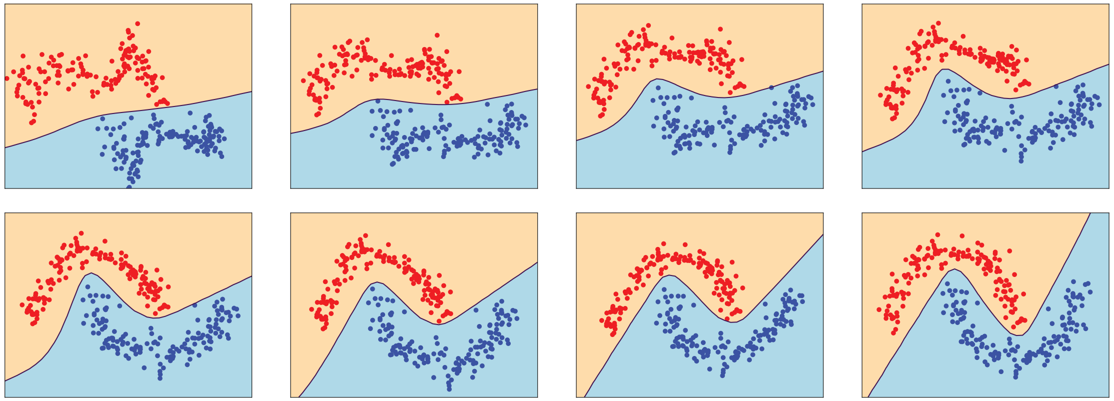
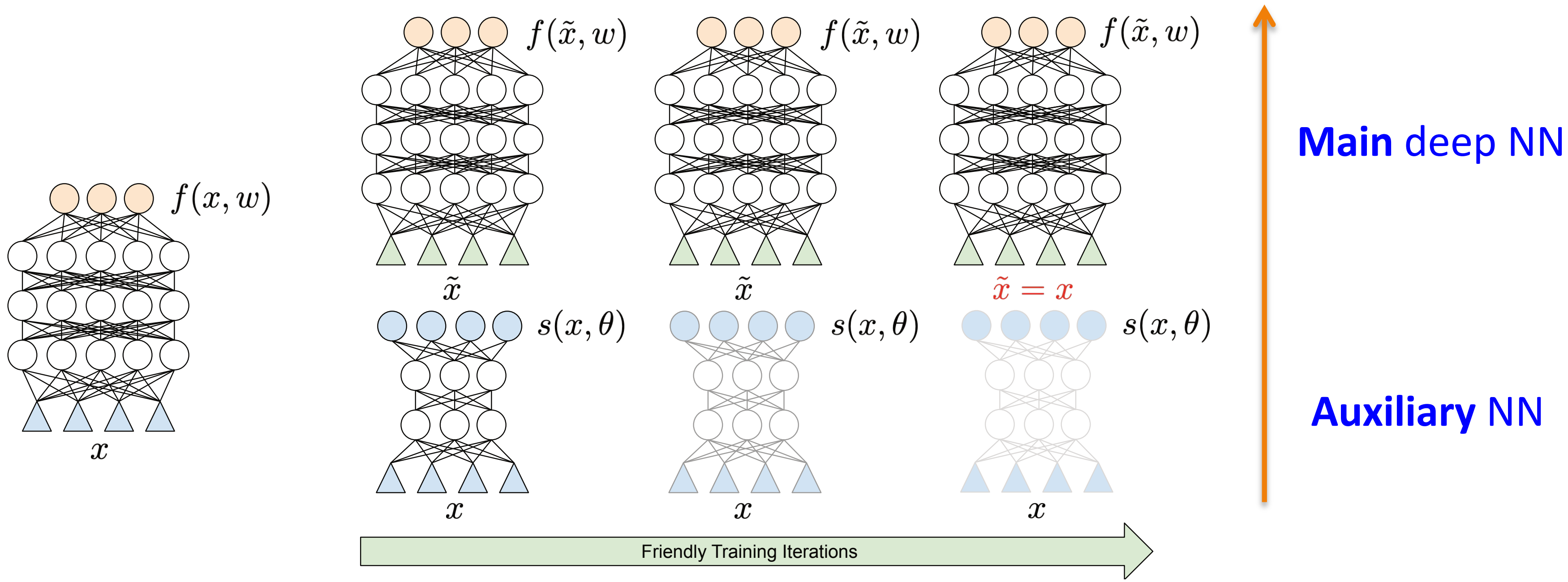


Figure 1: Left-to-right, top-to-bottom: evolution of the decision boundary developed by a single hidden layer classifier (5 neurons) in the 2-moon dataset, in Neural Friendly Training. Each plot is about a different training iteration (γ); in the last plot data are not transformed anymore.

$$\tilde{x}_i = s(x_i, \theta)$$

Neural Friendly Training



$$L(\mathcal{B}, w, \theta) = \frac{1}{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} \Bigg(\ell\big(f(\underbrace{s(x_i, \theta)}_{\tilde{x}_i}, w), y_i\big) + \eta \Big\| \underbrace{s(x_i, \theta) - x_i}_{\delta_i} \Big\|^2 \Bigg),$$

...

Neural Friendly Training

FC-A: **F**ully **C**onnected MLP **CNN-A**: Convolutional **NN**

Structured perturbations with CNNs only, **emphasizing the digit areas**

Really **poor**

Globally more **satisfying**

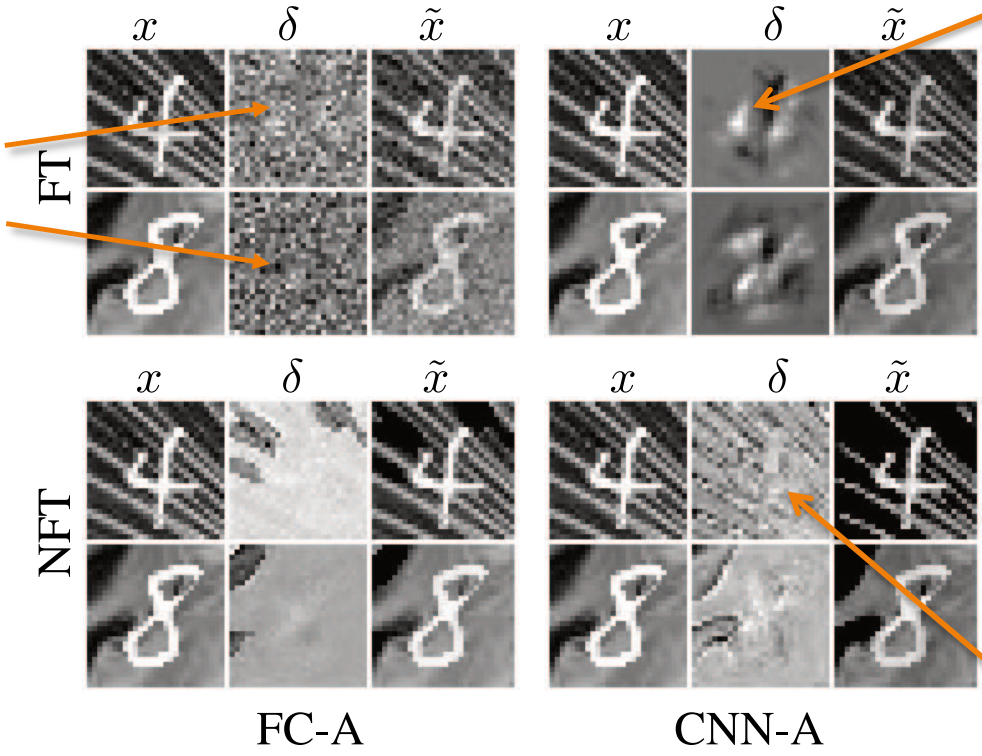


Figure 4: MNIST-BACK-IMAGE. Original data x , perturbation δ (normalized) and resulting "simplified" images $\tilde{x}$ for FC-A and CNN-A at the end of the 1st epoch. Some simplifications are hardly distinguishable. Top: FT. Bottom: NFT.

FT: **F**riendly **T**raining

Independent transformation for each example

NFT: **N**eural **F**riendly **T**raining

Using an **auxiliary** NN

Perturbations **removing distracting cues**

...

Learning techniques for "distillation"

1. Gradually changing the **targets**

2. Gradually changing the **inputs**

3. Gradually changing the **learning task**

Changing the learning task

- The **classical** distillation scenario (adapted)

$$\mathcal{L}_{KD} = (1-\alpha)\,\underbrace{H(y, q_s(\theta))}_{} + \alpha\, T^2\, \underbrace{H(p_t, q_s(\theta))}_{}$$

Classical cross-entropy between **output** and **target** values

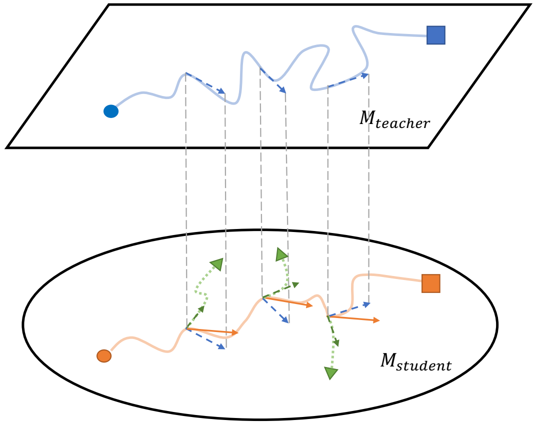
Cross-entropy between **teacher** and **student's** outputs

Changing the learning task

- Idea: train the student network through a sequence of **intermediate** learning **tasks**.
- Question: **how to choose** the intermediate learning tasks?
  1. They should be **easily achievable** by the student
  2. Consequence: the **teacher should be aware** of the student's progress
- Co-evolution between student and teacher
  1. The teacher converges toward the **goal**, but stay close to the **learner**

$$\theta_t^{m+1} = \min_{\theta_t} H(y, p_{\theta_t}) \quad \text{s.t. } D_{\mathrm{KL}}(q^m_{\theta_s}, p_{\theta_t}) \leq \epsilon$$

$$\hat{\mathcal{L}}_{\theta_t} = (1-\lambda) H(y, p_{\theta_t}) + \lambda H(q_{\theta_s}, p_{\theta_t})$$



  2. The student follows the **teacher** at each step

$$\theta_s^{m+1} = \theta_s^m - \eta_s \nabla \mathcal{L}_s(\theta_s, p_{\theta_t^{m+1}}), \quad \mathcal{L}_s(\theta_s) = H(p_{\theta_t}, q_{\theta_s})$$

Changing the learning task

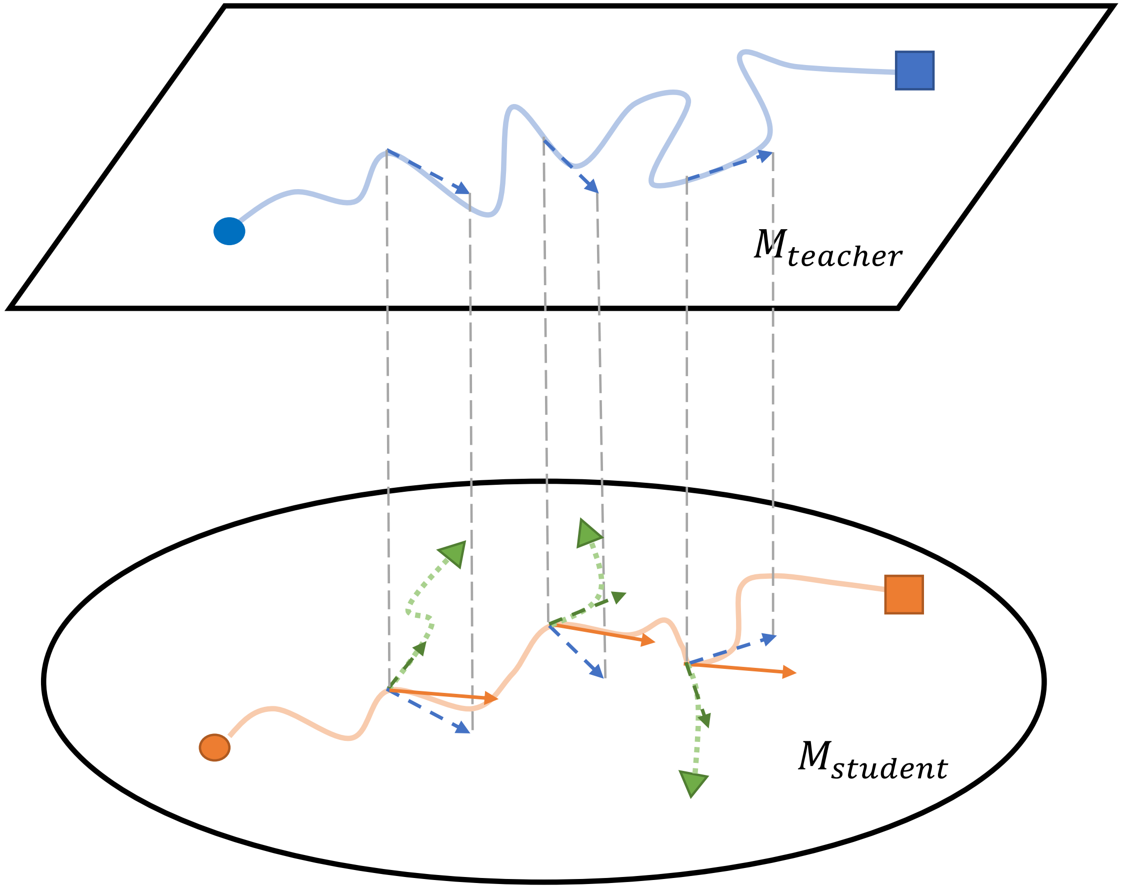


Fig. 1: $\mathcal{M}_{teacher}$ and $\mathcal{M}_{student}$ refer to the output manifolds of student model and teacher model. The lines between circles (●,●) to squares (■,■) imply the learning trajectories in the distribution level. The intuition of ProKT is to avoid bad local optimas (triangles (▲)) by conducting supervision signal projection.

...

Changing the learning task

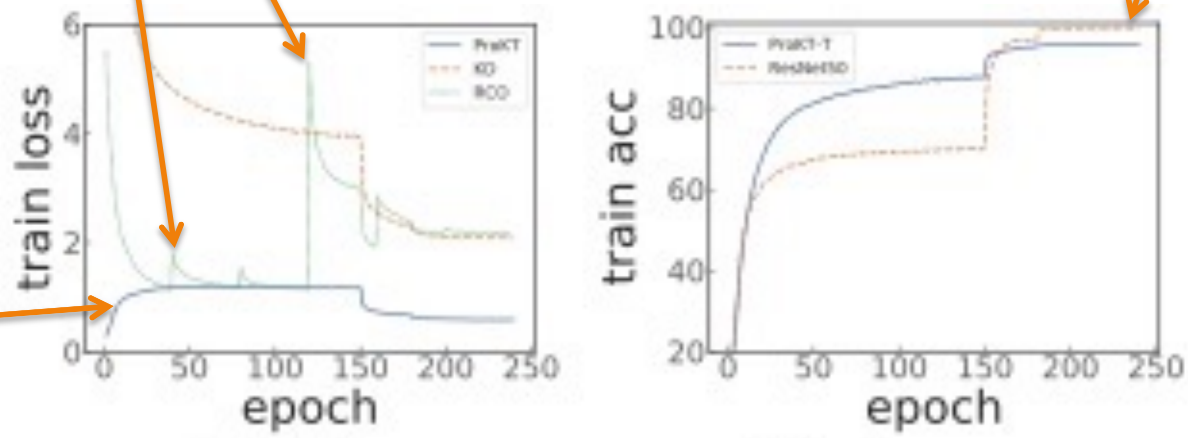
KD : classical **K**nowledge **D**istillation

RCO : use intermediate models obtained during the teacher's training process

ProKT : their method

Lower performance of the teacher, but better student in the end

The divergence between teacher and student in ProKT is **smooth** and **well bounded**



(a) Train loss of **student** (b) Train accuracy of **teacher**

Shi, W., Song, Y., Zhou, H., Li, B., & Li, L. (2021, September). **Follow your path: a progressive method for knowledge distillation**. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 596-611). Springer.

Changing the learning task

KD : classical **K**nowledge **D**istillation

RCO : use intermediate models obtained during the teacher's training process

ProKT : their method where the teacher stays close to the student

Using Kullback-Leibler (KD)loss

	Teacher	vgg13	ResNet50	ResNet50	resnet32x4	resnet32x4	WRN-40-2
	Student	MobileNetV2	MobileNetV2	vgg8	ShuffleNetV1	ShuffleNetV2	ShuffleNetV1
	Teacher	74.64	79.34	79.34	79.42	79.42	75.61
Without distillation	Student	64.6	64.6	70.36	70.5	71.82	70.5
With distillation	KD*	67.37	67.35	73.81	74.07	74.45	74.83
	RCO	68.42	68.95	73.85	75.62	**76.26**	75.53
	ProKT	**68.79**	**69.32**	**73.88**	**75.79**	75.59	**76.02**
	CRD	69.73	69.11	74.30	75.11	75.65	76.05
	CRD+KD	**69.94**	69.54	74.58	75.12	76.05	76.27
	CRD+ProKT	69.59	**69.93**	**75.14**	**76.0**	**76.86**	**76.76**

Using Constrastive Representation Distillation (CRD) loss

Shi, W., Song, Y., Zhou, H., Li, B., & Li, L. (2021, September). **Follow your path: a progressive method for knowledge distillation**. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 596-611). Springer.

Lessons

- **Careful** distillation is useful
- Points to the idea of **curriculum** learning

Distillation: **other** approaches

- Match intermediate **weights**

- Match intermediate **features**

- Match gradients (**attention maps**)

Outline

1. Co-learning

2. Distillation

3. Multi-task learning

4. The Minimum Description Length principle (MDLP)

What is Multi-Task learning (MTL)?

- As soon you try to optimize more than one loss function

  - E.g. From someone's picture, trying to guess both
    - The **gender**
    - The **age**
    - The **emotion**

Why Multi-Task learning (MTL)?

- (IF) The tasks at hand are **not unrelated**
  - E.g. From someone's picture, trying to guess both
    - The **gender**
    - The **age**
    - The **emotion**

- It may help to consider them all together:

  better performance with **less** computing resources

  - E.g. guessing the *gender* may help recognize the *emotion* and vice-versa

Rk: There are links with the LUPI framework

Assumption behind MTL

- The **combined learning** of multiple related tasks **can outperform learning each task in isolation**

- MTL allows for **common information** shared between the tasks to be used in the learning process, which leads to better generalization if the tasks are related

- E.g. Learning to **predict the ratings** for several different critics (in different countries) can lead to better performances for **each separate task** (predict the restaurant ratings for a specific critic)

- Learning to **recognize a face** and the **expression** (fear, disgust, anger, …)

- **Multi modality learning**: e.g. vision and proprioception

Possible **relations** between tasks

- All functions to be learn are **close** to each other **in some norm**
  - E.g. functions capturing preferences in users' modeling problems

- Tasks that share a **common underlying representation**
  - E.g. in *human vision*, all tasks use the **same set of features** learnt in the first stages of the visual system (e.g. local filters similar to wavelets)
  - Users may also *prefer* different types of things (e.g. books, movies, music) based on the **same set of features** or **score** functions

Question

How do we choose to

model the shared information between the tasks?

- Idea: Some shared underlying constraints
  - E.g. a **low dimensional representation** shared across multiple related tasks
    - By way of a **shared hidden layer** in a neural network
    - By explicitly constraining the **dimensionality of a shared representation**

- T binary classification tasks defined over $\mathcal{X} \times \mathcal{Y}$

$$\mathcal{S} = \{\{(\mathbf{x}_{11}, y_{11}), (\mathbf{x}_{21}, y_{21}), \ldots, (\mathbf{x}_{m1}, y_{m1})\}, \ldots, \{(\mathbf{x}_{1T}, y_{1T}), (\mathbf{x}_{2T}, y_{2T}), \ldots, (\mathbf{x}_{mT}, y_{mT})\}\}$$

$$h_j(\mathbf{x}) = \mathbf{w}_j \cdot \mathbf{x}$$ **Linear** hypotheses

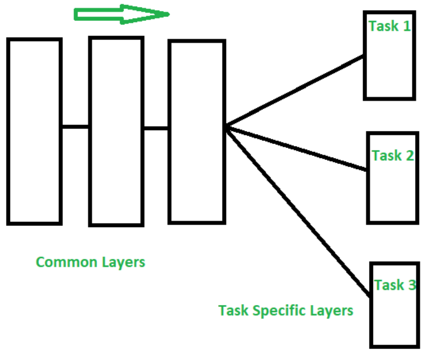
That share a weight vector $$\mathbf{w}_j = \mathbf{w}_0 + \mathbf{v}_j$$

$$h_1^\star, \ldots, h_T^\star = \underset{\mathbf{w}_0, \mathbf{v}_j, \xi_{ij}}{\mathrm{Argmin}} \left\{ \sum_{j=1}^{T} \sum_{i=1}^{m} \xi_{ij} + \frac{\lambda_1}{T} \sum_{j=1}^{T} ||\mathbf{v}_j||^2 + \lambda_2 \, ||\mathbf{w}_0||^2 \right\}$$

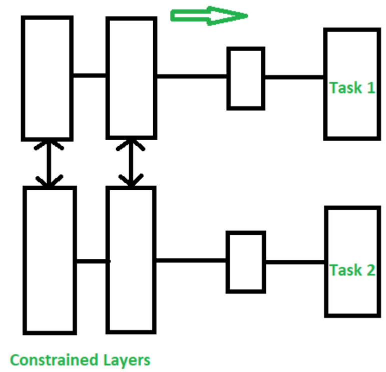
MTL with deep neural networks

- Approaches

1. Sharing **features** (first layers) and have multiple task-specific heads



1. **Soft**-features or parameters sharing



- Multi-Task Learning induces a bias that prefers hypotheses that can “explain” all tasks

- Beware:
  - Can lead to **worse** performance if the tasks are **unrelated** or **adversarially** related

- Question: *how to measure the* ***relatedness*** *of learning tasks*?

- Do you think of a **recent** multi-task learning system?

- Do you think of a **recent** multi-task learning system?

Exploit **universal representations** across modalities

...

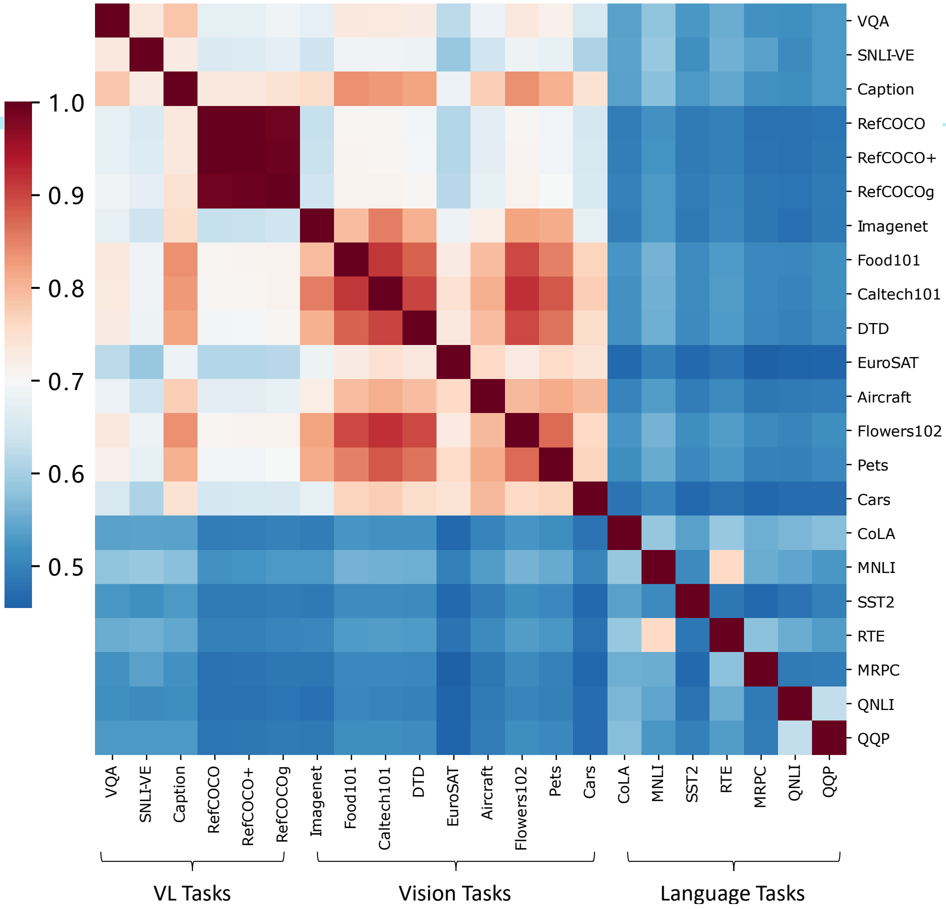


Figure 1. Heatmap of the predicted task similarities, composed of both unimodal and multimodal tasks. Vision-language tasks are more similar to vision tasks compared to language tasks. Best viewed in color.

WU, Chengyue, WANG, Teng, GE, Yixiao, et al. **$\pi $-Tuning: Transferring Multimodal Foundation Models with Optimal Multi-task Interpolation**. In *International Conf. on Machine Learning* (ICML). PMLR, 2023. p. 37713-37727.

- Idea of minimizing a **distance** between the “local” models

What kind of **distance** ?

Outline

1. Co-learning
2. Distillation
3. Multi-task learning
4. The Minimum Description Length principle (MDLP)

Kolmogorov's complexity

Andreï Kolmogorov
(1903 – 1987)

Complexity of a sequence =

Size in bits of the **smallest program**

that can generate that sequence

$$K_M(x) = \min_{p \in P_M} \{l(p), s(p) = x\}$$

x	: the sequence
P_M	: program coded on machine M
l(p)	: size of p

Kolmogorov's complexity

- True randomness
  - No structure
  - Smallest program = the sequence itself
- Pi
  - Lots of structure, very simple!

$$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k}\left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6}\right)$$

  - Infinite sequence of integers → but a small program

Solomonoff's induction

Ray Solomonoff
(1926 - …)

- *Look* for the smallest program
  that can generate a given sequence
  - Almost all induction problems can be cast as the prediction of a binary sequence
- Unfortunately, this is **NOT computable**…
  - Even if it exists, it is not possible to find it in the general case *(Gödel's theorem, stopping problem, …)*
- It is possible to approximate it

Minimum Description Length Principle (MDLP)

- The **best hypothesis** (given training data) is the one that **minimize** the sum of

  1. The **length** in bits of the description of the **hypothesis**
  2. The **length** in bits of the description of the **data given the hypothesis**

$$h^\star \;=\; \underset{h \in \mathcal{H}}{\mathrm{ArgMin}} \Big\{ \underbrace{L(h)} + \underbrace{L(\mathcal{S}|h)} \Big\}$$

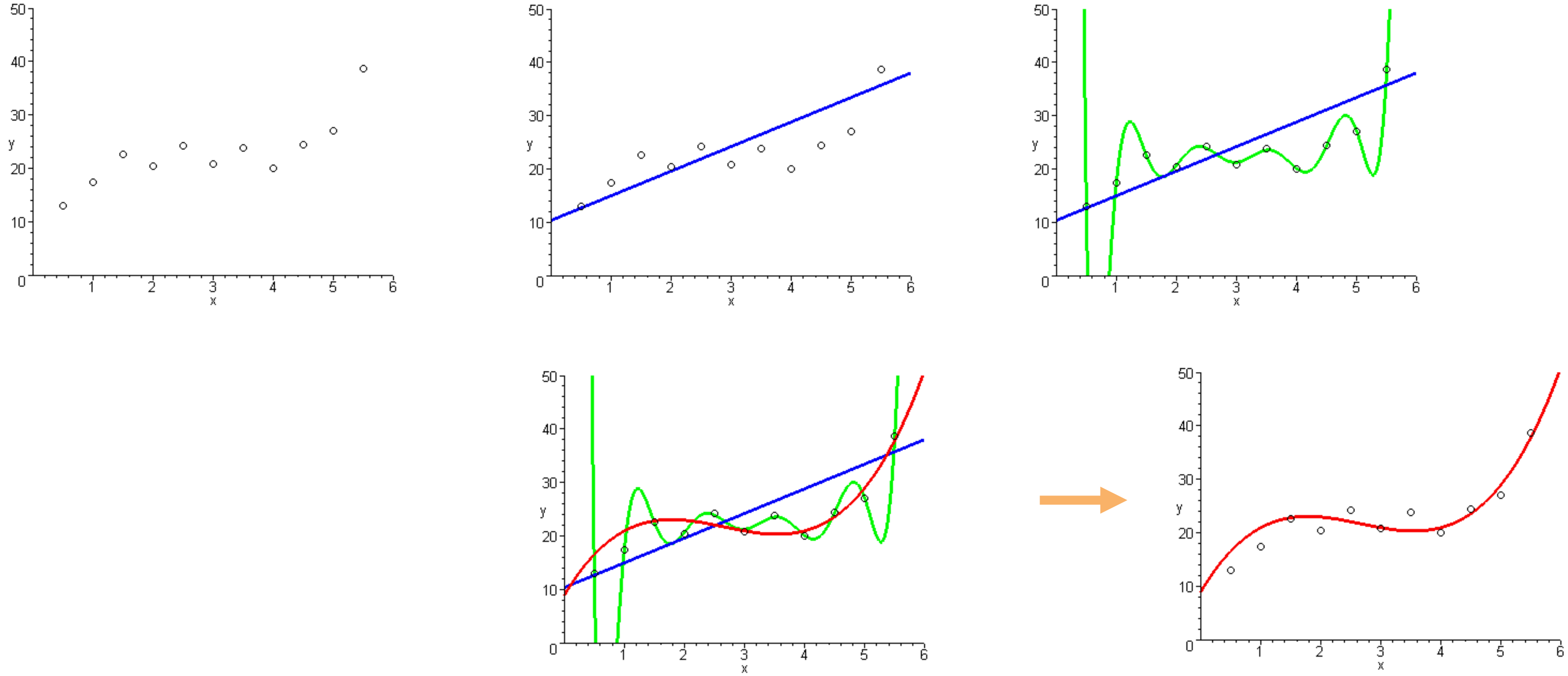
Strong relationship with

$$P(h|\mathcal{S}) \;=\; \frac{P(h) \times P(\mathcal{S}|h)}{P(\mathcal{S})}$$

$$h^\star = \underset{h \in \mathcal{H}}{\mathrm{ArgMax}} \; P(\mathcal{S} \mid h) \; P(h)$$

Example: regression

- **Complexity** of model:
  - the **degree** of a polynomial (to be described up to a given precision)
- **Error**
  - The size of the **corrections** wrt to the predictions



Minimum Description Length Principle (MDLP)

You have to define a code with which to describe the hypothesis and the data

⟷ a **bias** (prior knowkedge)

$$h^\star \;=\; \underset{h \in \mathcal{H}}{\mathrm{ArgMin}} \Big\{ \underbrace{L(h)} + \underbrace{L(\mathcal{S}_m | h)} \Big\}$$

- **Multi-task** learning
  - **Simultaneous** learning phases

Maximizing the **agreement** between learners

- **Transfer** learning
  - **Successive** learning phases

Maximizing the **agreement** (**??**) between learners

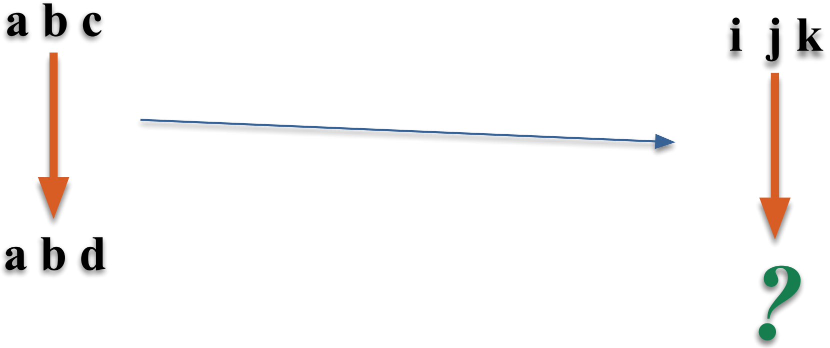
Analogy making

Copycat

Douglas Hofstadter
(1945 – …)

- Mitchell & Hofstadter – 1993

a b c	→	a b d
k j i	→	?



a b c

a b d

i i j j k k

?

- a b d
- i i j j k d
- i i j j k l
- i i j j k k
- ?

Copycat

a b c	→	a b d
i j k	→	?
k j i	→	?
c	→	?
a b c d e	→	?
m	→	?
x y z	→	?
f p c	→	?
i i j j k k	→	?
a a b b c c	→	?
i j j k k k	→	?
a b b c c c	→	?

a b c

a b d

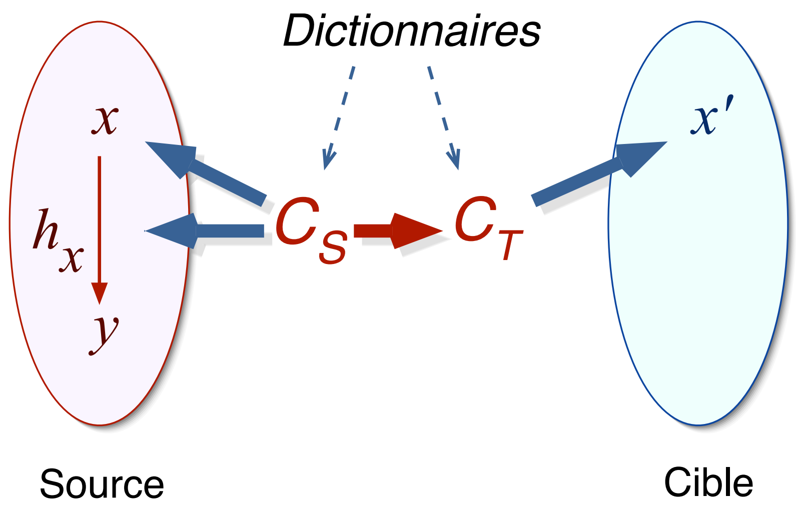
a a b a b c

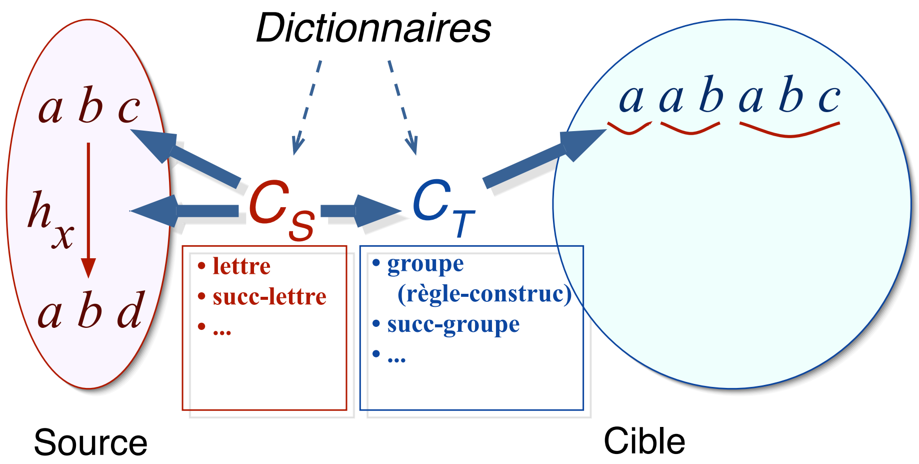
?

Domain adaptation & analogie

- Learn both :
  - A good **representation**
    - Of the source domain
    - Of the target domain
  - A good **transformation** rule

    ??





Copycat

- Successor and predecessor
  - a → b, b → a, 1 → 2, …
- Sequence
  - abcd…
- Sequence of sequences
  - aaabbbccc…
- First, last, …
- Opposite(first, last),
  Opposite(successor, predecessor), …

Various solutions

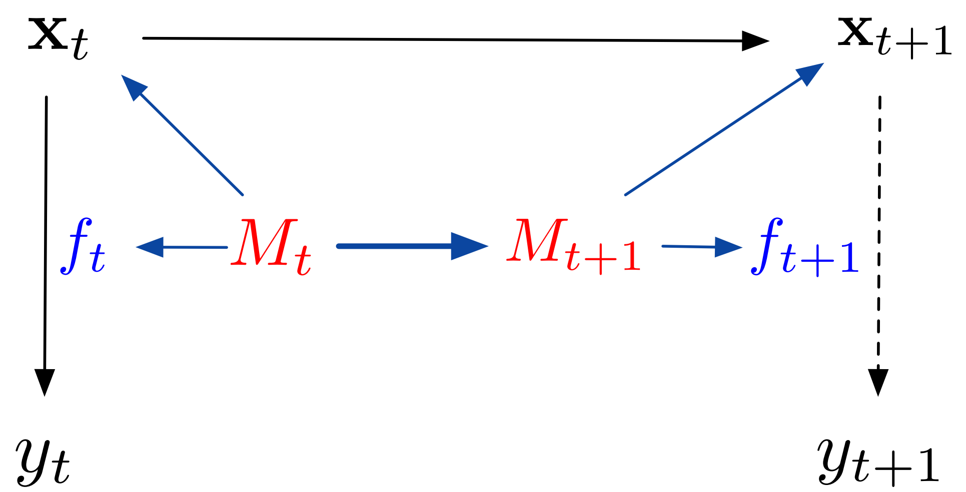
a b c	→	a b d	Comment
i j k	→	i j l	Replace *last letter* by its *successor*
	→	i j k	Replace **c** by **d**
	→	i j d	Replace *last letter* by **d**
	→	i j	Remove last letter and if this a '**c**' replace by **d**
	→	a b d	Replace by **a b d**
	→	i j k l	**c**=3, **d**=4, length(**ijk**)=3, length(**ijkl**)=4
	→	i j f	Replace last letter by **d** if this a '**c**' otherwise by **f**

Cornuéjols [1994 – 2020 - …]

- *Minimum Description Length Principle* + Copycat
  - MDLp = approximation of Kolmogorov’s complexity for learning

- Analogy making:
  1. Minimize the description of the known terms A:B :: C:? (production) or A:B :: C:D (evaluation)
  2. Choose the smallest description

An approach to analogy: using Kolmogorov complexity

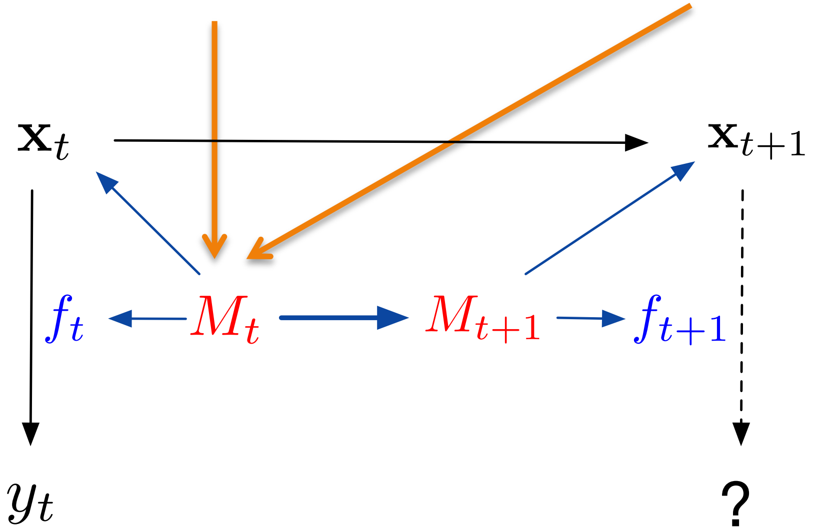


$$K(M_t) + K(\mathbf{x}_t|M_t) + K(f_t|M_t) + \underbrace{K(M_{t+1}|M_t)}_{\text{Change of system of reference}} + K(\mathbf{x}_{t+1}|M_{t+1}) + K(f_{t+1}|M_{t+1})$$

[Cornuéjols, 1996, 1997, 1998, 2016]

Une formalisation

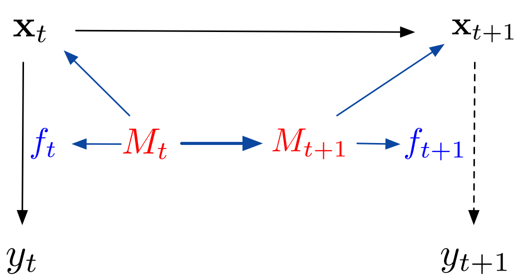
- Kolmogorov's Complexity
  - Uses a **dictionary** (with associated description lengths)
  - Which **depends** on the a **priori knowledge** and the **past experiences**



$$K(M_t) + K(\mathbf{x}_t|M_t) + K(y_t|M_t) + K(M_{t+1}|M_t) + K(\mathbf{x}_{t+1}|M_{t+1}) + K(f_{t+1}|M_{t+1})$$

[A. Cornuéjols (1996) « Analogie, principe d'économie et complexité algorithmique »]

An approach to analogy: using Kolmogorov complexity



- Descripteurs utilisés dans la définition des structures :
  - orientation (-> / <-) *1 bit*
  - cardinalité ou nombre d'éléments : n ***log$_2$(n) + 1** bits*
  - type d'éléments (voir en-dessous)
  - longueur : l ***log$_2$(l) + 1** bits*
  - commençant ou se terminant par l'élément = x ***L(x)** bits*
- **Lettre** *(1/2) -> **1** bit*

  Une lettre particulière (e.g. 'd') *(1/2.26) -> **6** bits*
- **Chaîne** (orientation,éléments) *(1/8) -> **3** bits*

  L = 3 + L(orientation) + ∑ L(éléments)

  e.g. L('a3bd' avec orientation = ->) = 3 + 1 + $\log_2((1/2.26)^3)$ + L(3)

  = 3 + 1 + 18 + 3 = 25 bits
- **Ensemble** (type d'éléments, cardinalité, éléments) *(1/8) -> **3** bits*

  L = 3 + L(type) + L(cardinalité) + ∑ L(éléments)
- **Groupe** (type d'éléments, nombre d'éléments, éléments) *(1/8) -> **3** bits*

  L = 3 + L(type) + L(nb él.) + ∑ L(éléments)
- **Séquence** (orientation, type d'éléments, loi de succession ou nombre d'éléments, longueur, commençant ou se terminant par) *(1/8)*

  L = 3+ L(orient.) + L(type) + L(loi) or L(nb él.) + L(long) + L(début/fin)
- Description et longueur d'une loi de **succession**

  succ(type-of-el.,n,x) ≡ le nième successeur de l'élément x du type type-of-el.

  L = L(type) + L(n (voir ci-dessous)) + L(x)

  L(n) = L(1/6) si n=1 ou -1 (1er successeur ou prédécesseur)

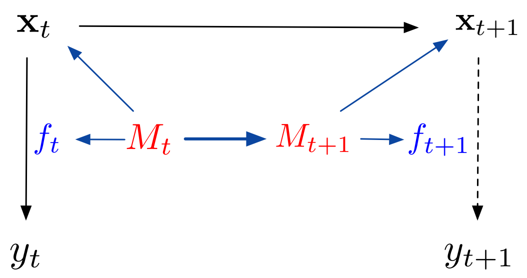
  L(1/3) si n=0 (même élément)

  L((1/3).(1/2)^p) sinon (avec p=n si n≥0, p=-n sinon)
- Premier / Dernier (par rapport à l'orientation définie) ***1** bit*
- nième ***n** bits*

$$K(M_t) + K(\mathbf{x}_t|M_t) + K(f_t|M_t) + \underbrace{K(M_{t+1}|M_t)}_{\text{Change of system of reference}} + K(\mathbf{x}_{t+1}|M_{t+1}) + K(f_{t+1}|M_{t+1})$$

An approach to analogy: using Kolmogorov complexity

[Cornuéjols, 1996, 1997, 1998, 2016]



'abc' ≡ **Chaîne** (1/8)

orientation : -> (1/2)

1er='A', 2ème='B', 3ème='C' $(1/4.26)^3$

TOTAL (longueur) : **21** bits

'abc' ≡ **Ensemble** (1/8)

{'A', 'B', 'C'} $(1/4.26)^3$

TOTAL : **20** bits

'abc' ≡ **Séquence** (1/8)

orientation : -> (1/2)

type d'éléments = lettres (1/2)

loi de succession :

successeur(élt(lettre=x)) = élt(succ(lettre,1,x))

L(lettre) + L(1er succ) + L(x) = L(1/2 . 1/6 . 1)

= l(1/12) = 4 bits

longueur = 3 3 bits

commençant avec l'élément(lettre='A') (1/26)

TOTAL : **17** bits

An approach to analogy: using Kolmogorov complexity

[Cornuéjols, 1996, 1997, 1998, 2016]

Problème 1 : **abc => abd ; iijjkk => ?**

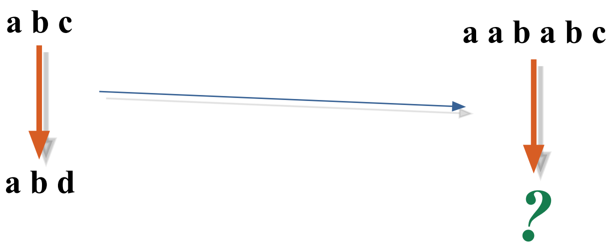
- Solution 1 : "*Remplacer groupe de droite par son successeur*" **iijjkk => iijjll**
- Solution 2 : "*Remplacer lettre de droite par son successeur*" **iijjkk => iijjkl**
- Solution 3 : "*Remplacer lettre de droite par D*" **iijjkk => iijjkd**
- Solution 4 : "*Remplacer 3ème lettre par son successeur*" **iijjkk => iikjkk**
- Solution 5 : "*Remplacer les C par D*" **iijjkk => iijjkk**
- Solution 6 : "*Remplacer groupe de droite par la lettre D*" **iijjkk => iijjd**

	P1;S1	P1;S2	P1;S3	P1;S4	P1;S5	P1;S6
$L(M_S)$	10	9	11	11	12	11
$L(S_S \mid M_S)$	8	18	18	18	22	15
$L(\beta_S \mid M_S)$	4	4	3	7	8	3
$L(M_C \mid M_S)$	5	0	0	0	0	17
$L(S_C \mid M_C)$	8	36	36	36	42	15
$L(\beta_C \mid M_C)$	6	4	3	7	8	3
Total-1 (bits)	**41**	**71**	**71**	**79**	**93**	**65**
Total-2 (bits)	**35**	**67**	**68**	**72**	**85**	**62**
Rang	1	3	4	4	6	2

Copycat + MDLp

a b c	→	a b d	Length in bits
i i j j k k	→	i i j j l l	35
i i j j k k	→	i i j j k l	67
i i j j k k	→	i i j j k d	68
i i j j k k	→	i i k j k k	72
i i j j k k	→	i i j j k k	85
i i j j k k	→	i i j j d	62

Results




```
'abc' ≡ Chaîne                                      (1/8)
        orientation : ->                            (1/2)
        1er='A', 2ème='B', 3ème='C'                 (1/4.26)^3
                                 TOTAL (longueur) :  21 bits
```

```
'abc' ≡ Ensemble                                    (1/8)
        {'A', 'B', 'C'}                             (1/4.26)^3
                                           TOTAL :  20 bits
```

```
'abc' ≡ Séquence                                    (1/8)
        orientation : ->                            (1/2)
        type d'éléments = lettres                   (1/2)
        loi de succession :
          successeur(élt(lettre=x)) = élt(succ(lettre,1,x))
          L(lettre) + L(1er succ) + L(x)   = L(1/2 . 1/6 . 1)
                                           = l(1/12) = 4 bits
        longueur = 3                                3 bits
        commençant avec l'élément(lettre='A')       (1/26)
                                           TOTAL :  17 bits
```

[A. Cornuéjols (1996) « Analogie, principe d'économie et complexité algorithmique »]

Analogy making and MDLP

- Application to **language analogies**: how to end words (conjugations, plurals, …)

apte : inapte :: élu : x	x = inélu	*(Prefixation)*
let,?0,:,'i','n',?0,let,mem,0,'apte',::,mem,0,'élu'		
átír : átírunk :: kitart : x	x = kitartunk	*(Suffixation)*
let,?0,:,?0,'u','n','k',let,mem,0,'átír',::,mem,0,'kitart'		
pati : patti :: olo : x	x = olto	*(Insertion)*
let,?0,?1,:,?0,'t',?1,let,mem,0,'pa','ti',::,mem,0,'ol','o'		
pria : pria-pria :: keju : x	x = keju-keju	*(Repetition)*
let,?0,:,?0,'-',?0,let,mem,0,'pria',::,mem,0,'keju'		
vantut : vanttu :: autopilotit : x	x = autopilotti	*(Reduplication)*
let,?0,?1,'t',:,?0,'t',?1,let,mem,0,'van','tu',::,mem,0,'autopilot','i'		

Murena, P. A., Al-Ghossein, M., Dessalles, J. L., & Cornuéjols, A. (2020). **Solving Analogies on Words based on Minimal Complexity Transformation**. In *IJCAI* (pp. 1848-1854).

Analogy making and MDLP

- Application to **language analogies**: how to end words (conjugations, plurals, …)

Language	**#analogies**	NLG_COMP	NLG_PROP	NLG_ALEA
Arabic	165,113	87.18%	**93.33%**	81.91%
Finnish	313,011	**93.69%**	92.76%	78.75%
Georgian	3,066,273	**99.35%**	97.54%	88.42%
German	730,427	**98.84%**	96.21%	95.42%
Hungarian	2,912,310	**95.71%**	92.61%	86.02%
Maltese	28,365	**96.38%**	84.72%	91.84%
Navajo	321,473	81.21%	**86.87%**	78.95%
Russian	552,423	96.41%	**97.26%**	95.46%
Spanish	845,996	**96.73%**	96.13%	94.42%
Turkish	245,721	**89.45%**	69.97%	70.06%
Total	9,181,112	**96.41%**	94.34%	87.93%

Table 2: Proportion of correct answers when solving analogies from the dataset SIGMORPHON'16 using our method NLG_COMP and two state-of-the-art methods NLG_PROP [Fam and Lepage, 2018] and NLG_ALEA [Langlais *et al.*, 2009].

The **results** using **deep NNs** and learnt embeddings were in the range 0.1% to 17%!!

Murena, P. A., Al-Ghossein, M., Dessalles, J. L., & Cornuéjols, A. (2020). **Solving Analogies on Words based on Minimal Complexity Transformation**. In *IJCAI-2020* (pp. 1848-1854).

Overview: which **communications**?

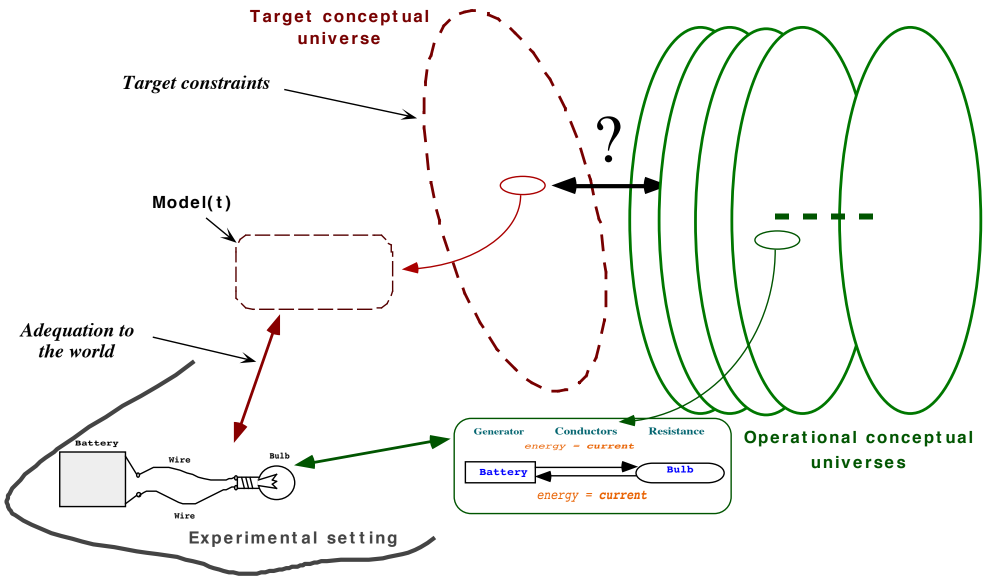
- **Ensemble** learning (e.g. boosting)
- **Co**-learning
- **Distillation**
- **Multi-task** learning
- **Transfer** learning (analogy making)

Overview: which **communications**?

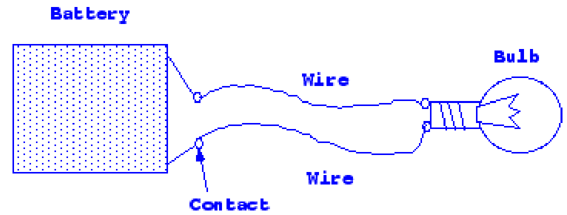
- **Ensemble** learning (e.g. boosting)
  - Communicating a new **input distribution** such as to learn ≠ hypotheses + vote
- **Co**-learning
  - Benefit from **different perspectives** and exchange of **pseudo-labeled** examples
- **Distillation**
  - Easing a student's learning by **modifying** the **outputs**, the **inputs** or the **task**
- **Multi-task** learning
  - **Minimizing** the **disagreement** between the learned hypotheses
- **Transfer** learning (analogy making)
  - **Minimizing** a **distance** (not necessarily symmetrical) between successive models

Cognitive tunnel effect

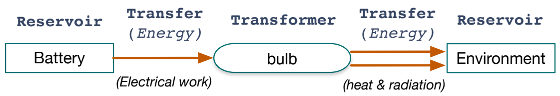
[A. Cornuéjols, A. Tiberghien, G. Collet. *Tunnel Effects in Cognition: A new Mechanism for Scientific Discovery and Education*. Arxiv-1707.04903- Tue, 18 Jul 2017 00:00:00 GMT]



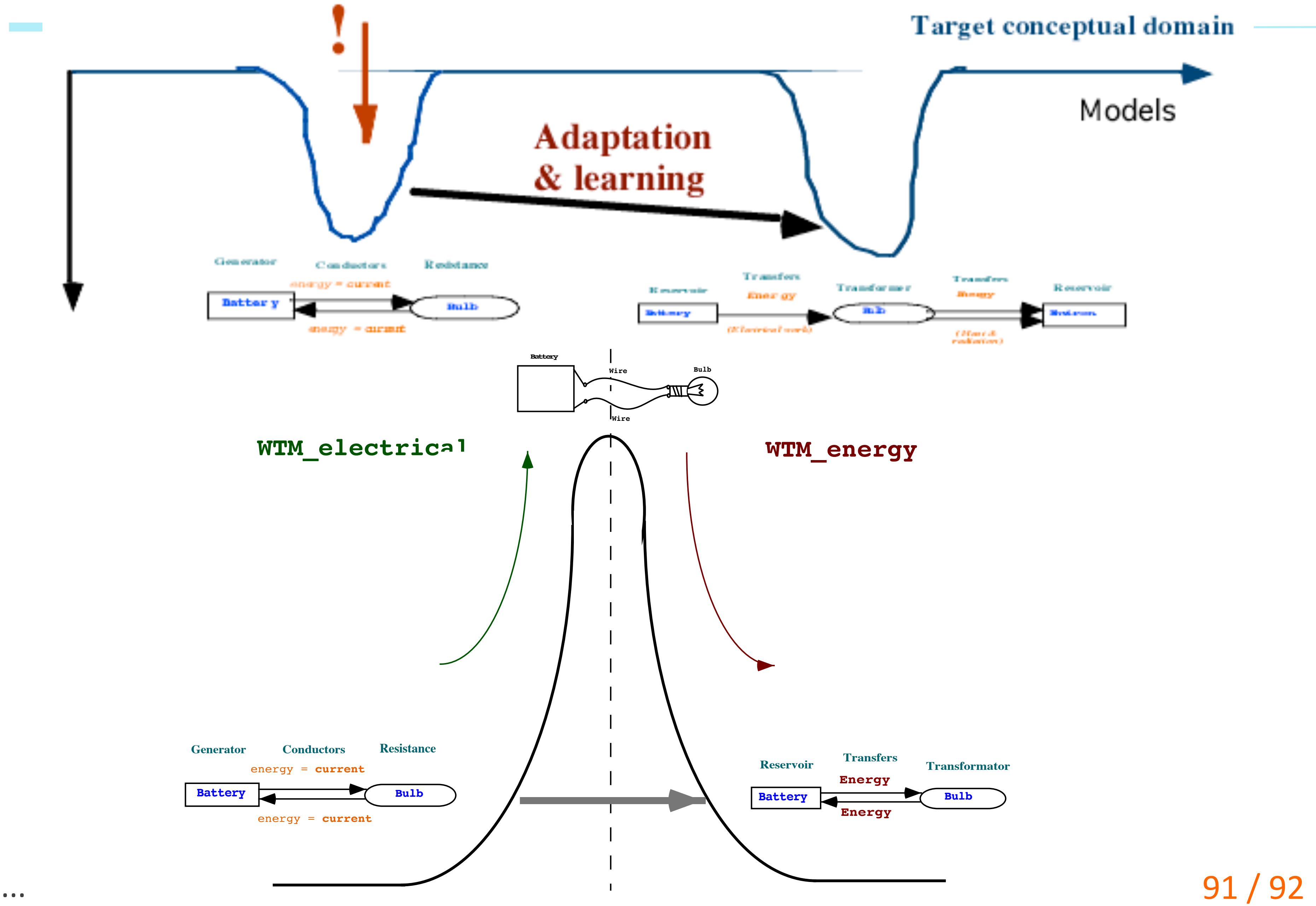
Experimental setting



Conceptual interpretation in terms of **energy chain**



Cognitive tunnel effect



…

Newton’s luggage

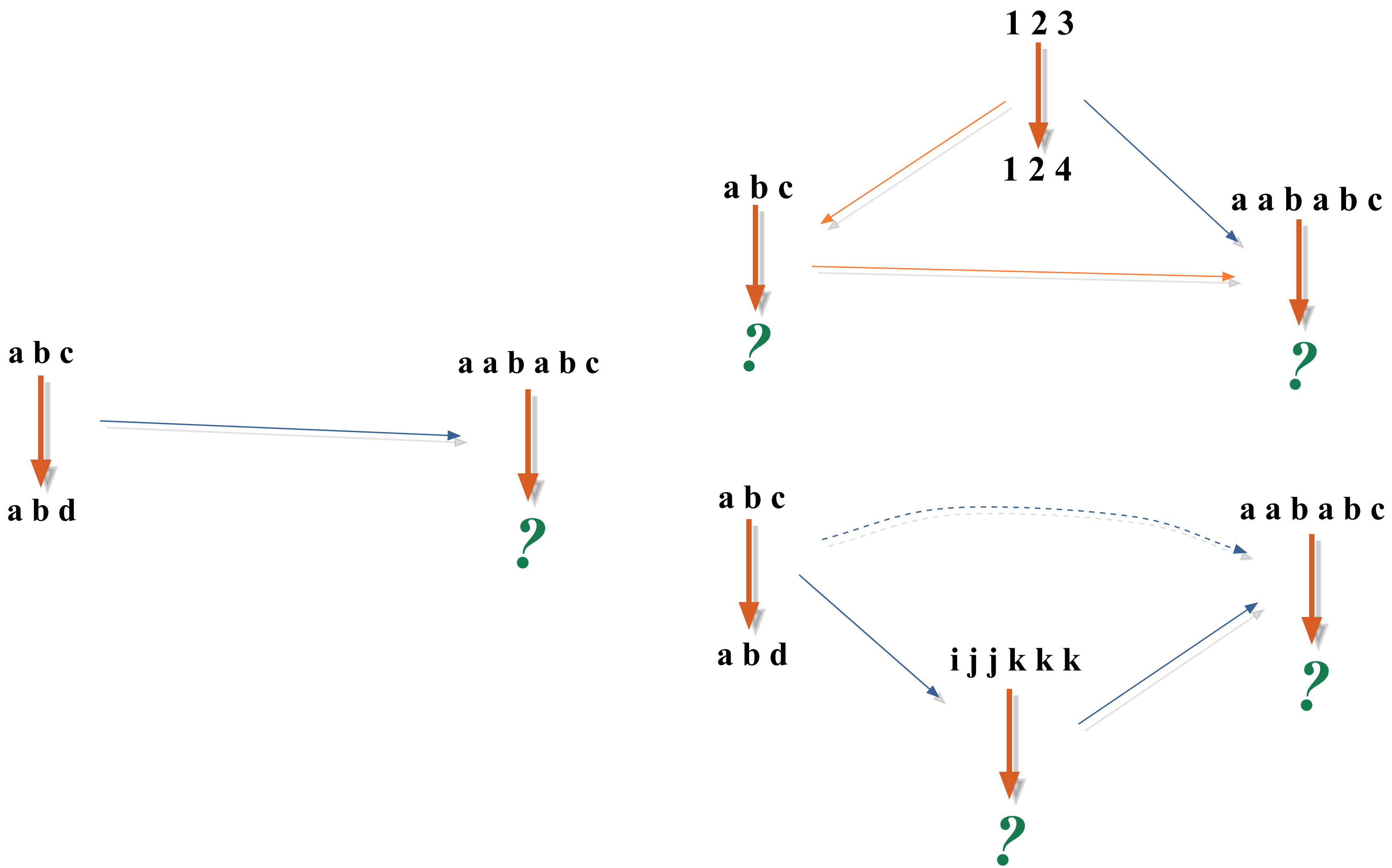
[Loup Verlet. La malle de Newton. Gallimard, NRF, 1993]

- How did Newton arrive to **the theory of gravitation**?
- What were the **sources** of his thoughts?
  - **Alchemy** (among other things)
- What were the **questions of the time**?
  - How transmutation of bread into the corpse of Jesus Christ can arise simultaneously in all churches?

→ **Action at distance**

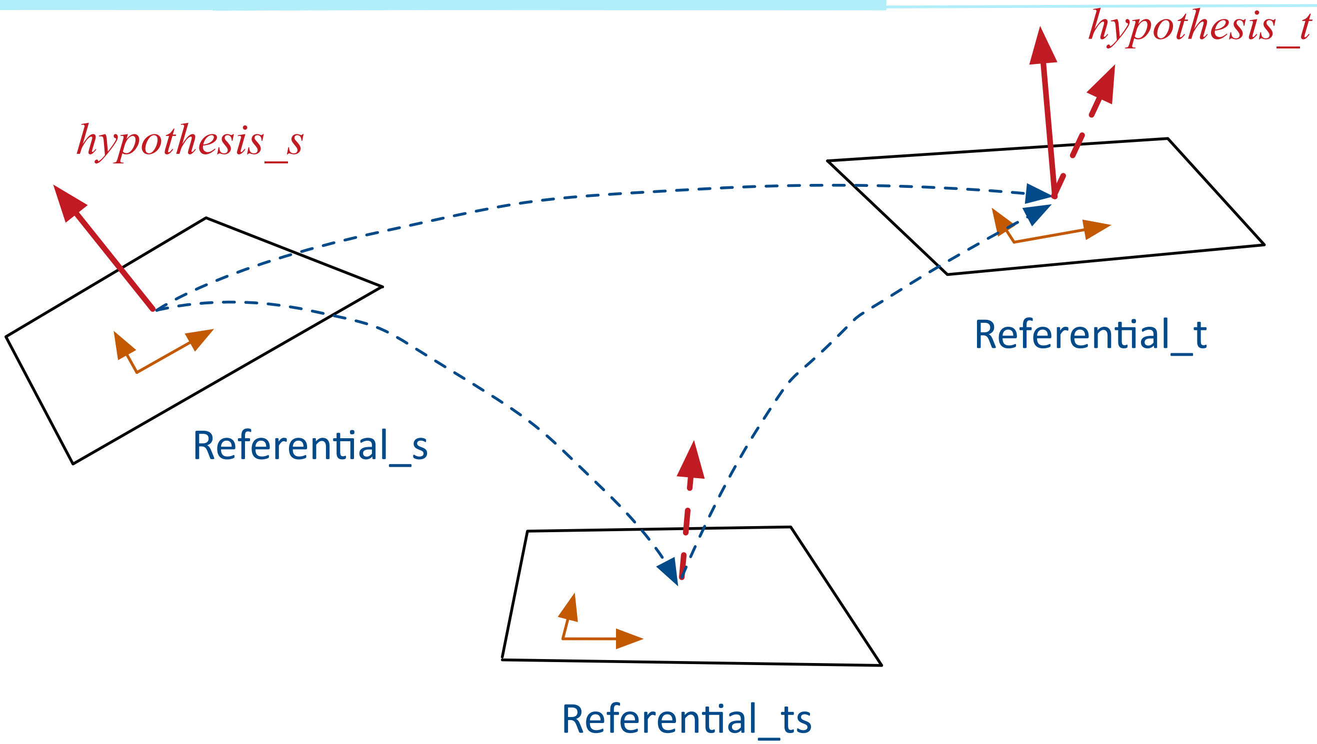
Some speculations

Transfer and sequence effects



...

Transfer and sequence effects



1. **Which equations** for the change of referential and for hypothesis transfer?
2. How to **prove** that these equations are **optimal**?

Conclusions (1)

Transfer learning → mostly heuristical approaches so far

1. **Parallel transport** is a natural way for looking at **transfer** learning
   - The **covariant derivative** is then a measure of difference
     - **How** to compute it?
       - Pioneering works in **computer vision**
     - What about when the **source** and **target** domains are **different**?
       - TransBoost: a **proposal**

2. Transfer learning is **path dependent** in general
   - The study of these path dependencies is **important** ...
     - Curriculum learning
     - Longlife learning
   - ... and a wide **open research question**

Conclusions (2)

- The **theoretical guarantees** for transfer learning:
  - Do not necessarily depend on the **performance of the source hypothesis** h_S

    But depend on the **bias** that h_S determines

  - Involve the **capacity** of the space of **transformations**

    (and **the path** followed between source and target)

Still to be explored

Bibliography

- Ben-David, S., Lu, T., Luu, T., & Pál, D. (2010). Impossibility theorems for domain adaptation. In *International Conference on Artificial Intelligence and Statistics* (pp. 129-136).
- Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., & Vaughan, J. W. (2010). A theory of learning from different domains. *Machine learning*, *79*(1-2), 151-175.
- Cornuéjols A., Murena P-A. & Olivier R. "*Transfer Learning by Learning Projections from Target to Source*". Symposium on Intelligent Data Analysis (IDA-2020), April 27-29 2020, Bodenseeforum, Lake Constance, Germany.
- Kuzborskij, I., & Orabona, F. (2013, February). Stability and hypothesis transfer learning. In *International Conference on Machine Learning* (pp. 942-950).
- Mansour, Y., Mohri, M., & Rostamizadeh, A. (2009). Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*.
- Redko, I., Morvant, E., Habrard, A., Sebban, M., & Bennani, Y. (2019). *Advances in Domain Adaptation Theory*. Elsevier.
- H. Venkateswara, S. Chakraborty, and S. Panchanathan, "Deep-learning systems for domain adaptation in computer vision: Learning transferable feature representations," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 117–129, 2017.
- Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks?. In *Advances in neural information processing systems* (pp. 3320-3328).
- Zhang, C., Zhang, L., & Ye, J. (2012). Generalization bounds for domain adaptation. In *Advances in neural information processing systems* (pp. 3320-3328).