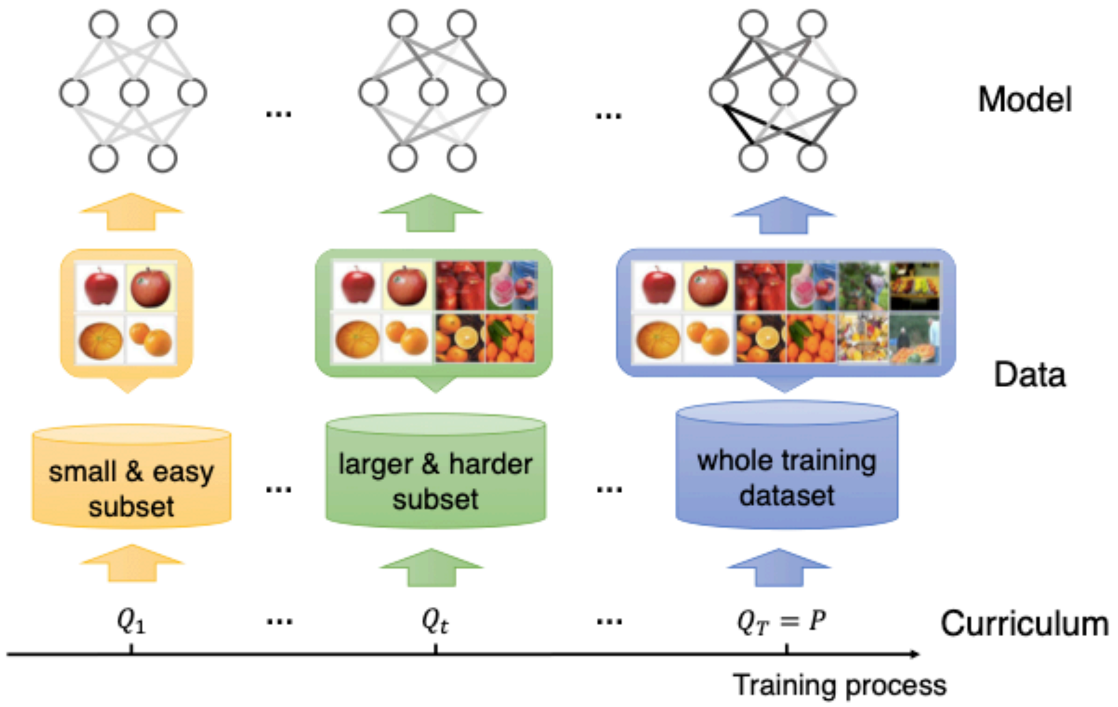
Universität
Bremen



Wang et al, "A Survey on Curriculum Learning", TPAMI 2021

Part 1 - Learning in the Present

Difficulty and Learning Curricula

Lifelong Machine Learning
Summer 2025

Prof. Dr. Martin Mundt

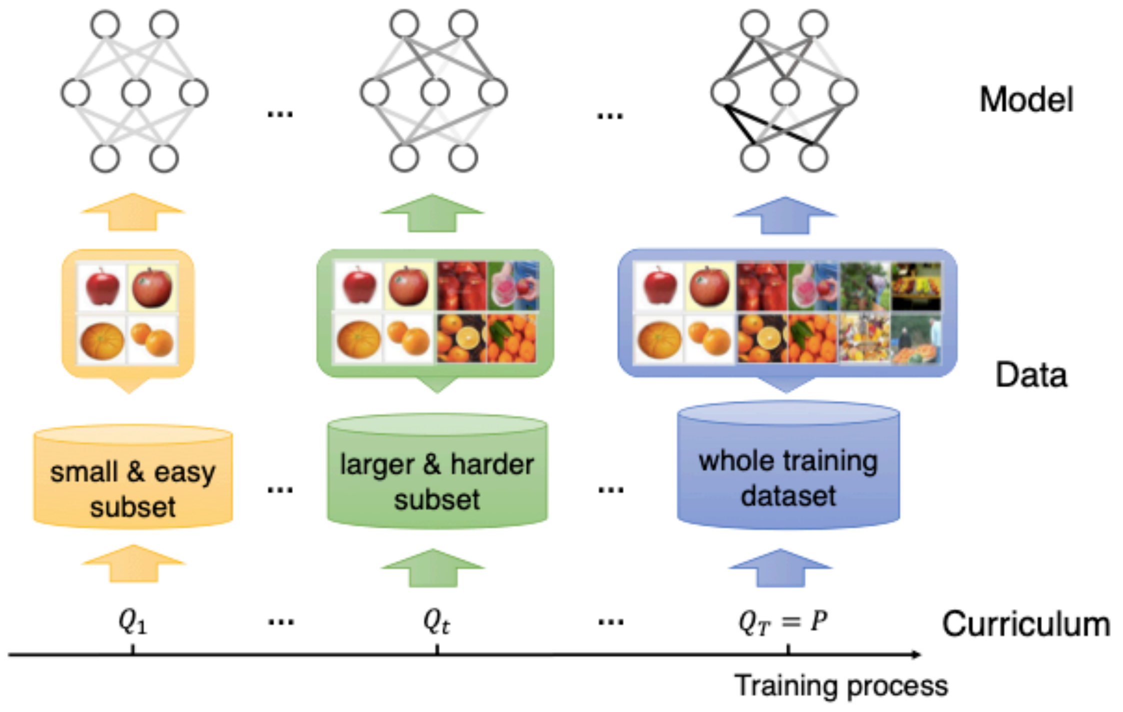
Universität Bremen

Question Time

Assume a static dataset & model: Does the order in which you introduce the data (or shuffle) matter?

It depends

- Some data (subsets) may be "easier" to learn, while some are harder
- Some data may be noisy, perturbed, anomalous
- Some data is more informative with respect to the task we are solving
- And it heavily depends on the learning strategy, model, and optimizer



Wang et al, "A Survey on Curriculum Learning", TPAMI 2021

Universität Bremen

A motivating example to provide intuition

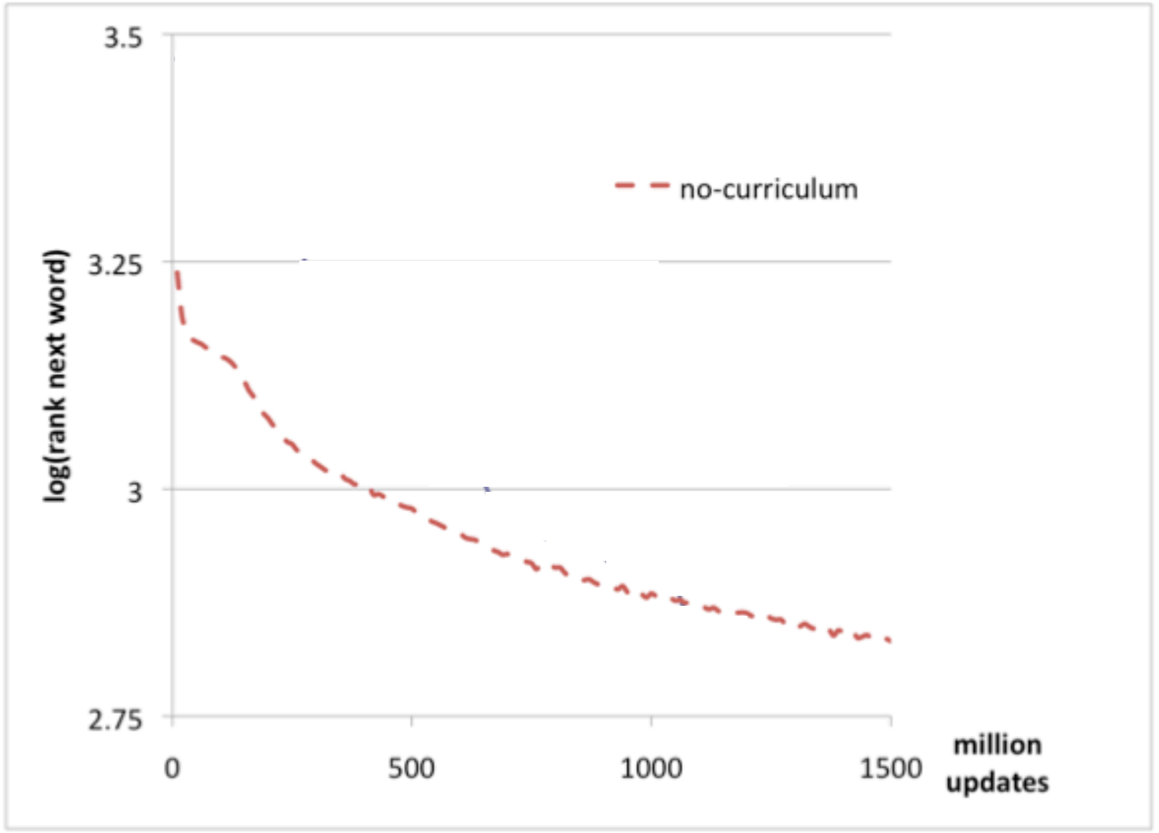
To train Japanese listeners on the English [r]-[l] distinction, McCandliss et al (Success and failure in teaching the [r]-[l] contrast to Japanese adults, Cognitive, Affective & Behavioral Neuroscience, 2002) found that participants could better distinguish the two phonemes if they were given over-articulated (easy) examples first, based on a crafted 1D curriculum that linearly interpolates a vocal tract model.

Universität
Bremen

A motivating example

Ranking language model trained with vs. without curriculum on Wikipedia

“Error” is log of the rank of the next word (within 20k-word vocabulary)



Y. Bengio et al, “Curriculum Learning”, ICML 2009

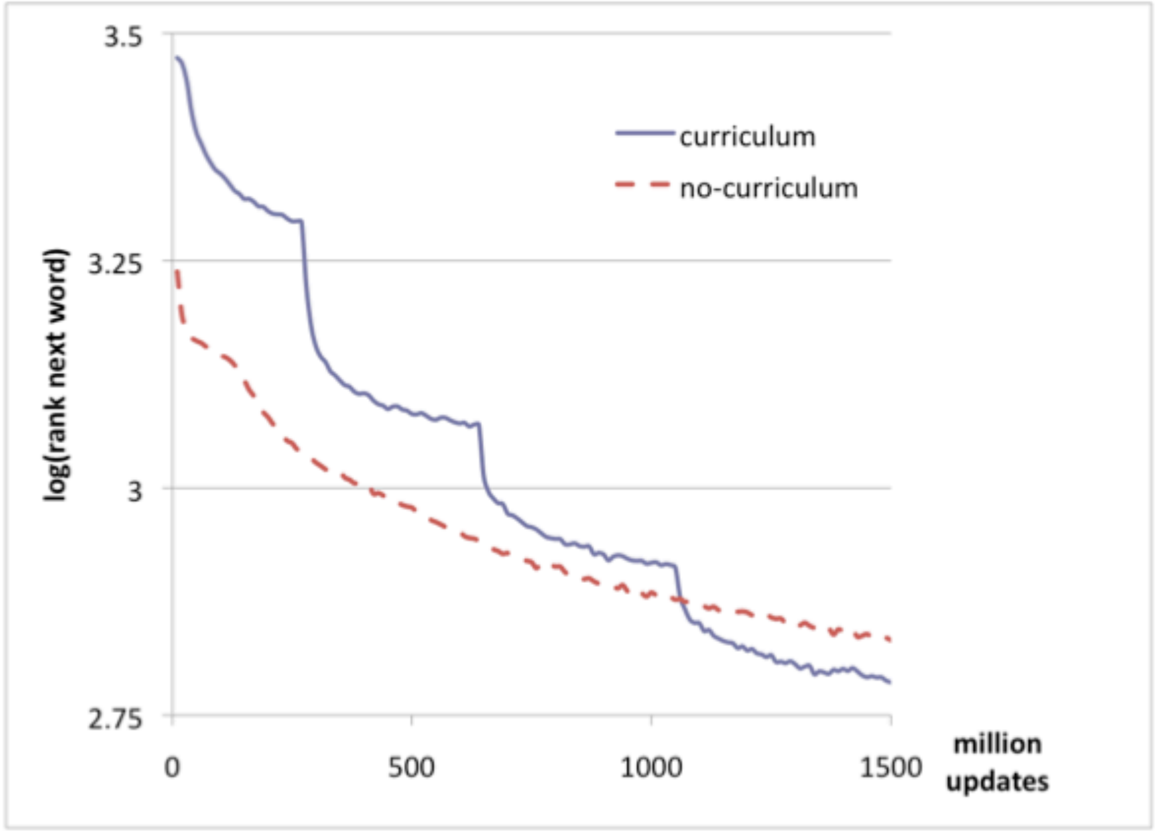
Universität Bremen

A motivating example

Ranking language model trained with vs. without curriculum on Wikipedia

“Error” is log of the rank of the next word (within 20k-word vocabulary)

1. The curriculum trained model skips examples with words outside of the 5k most frequent words
2. Then skips examples outside 10k most frequent words… (up to 20k)



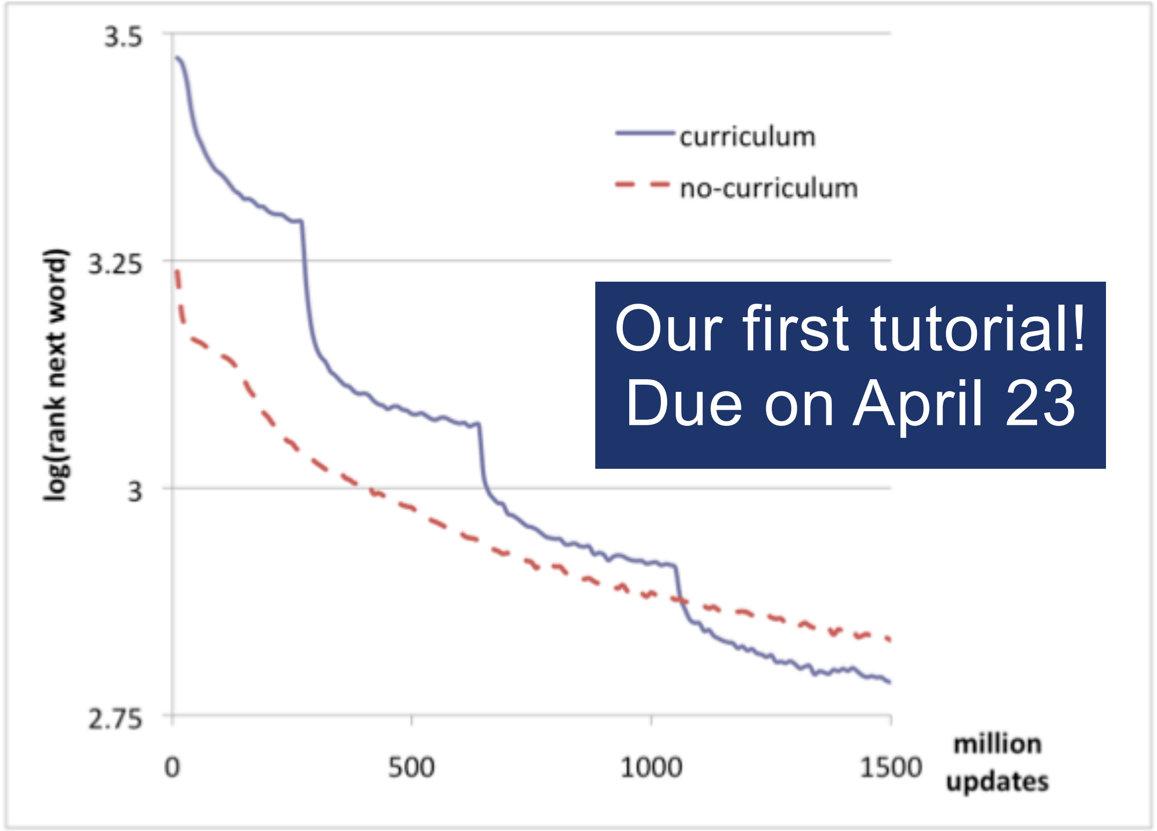
Y. Bengio et al, “Curriculum Learning”, ICML 2009

A motivating example

Ranking language model trained with vs. without curriculum on Wikipedia

"Error" is log of the rank of the next word (within 20k-word vocabulary)

1. The curriculum trained model skips examples with words outside of the 5k most frequent words
2. Then skips examples outside 10k most frequent words… (up to 20k)



Our first tutorial!
Due on April 23

Y. Bengio et al, "Curriculum Learning", ICML 2009

Universität Bremen

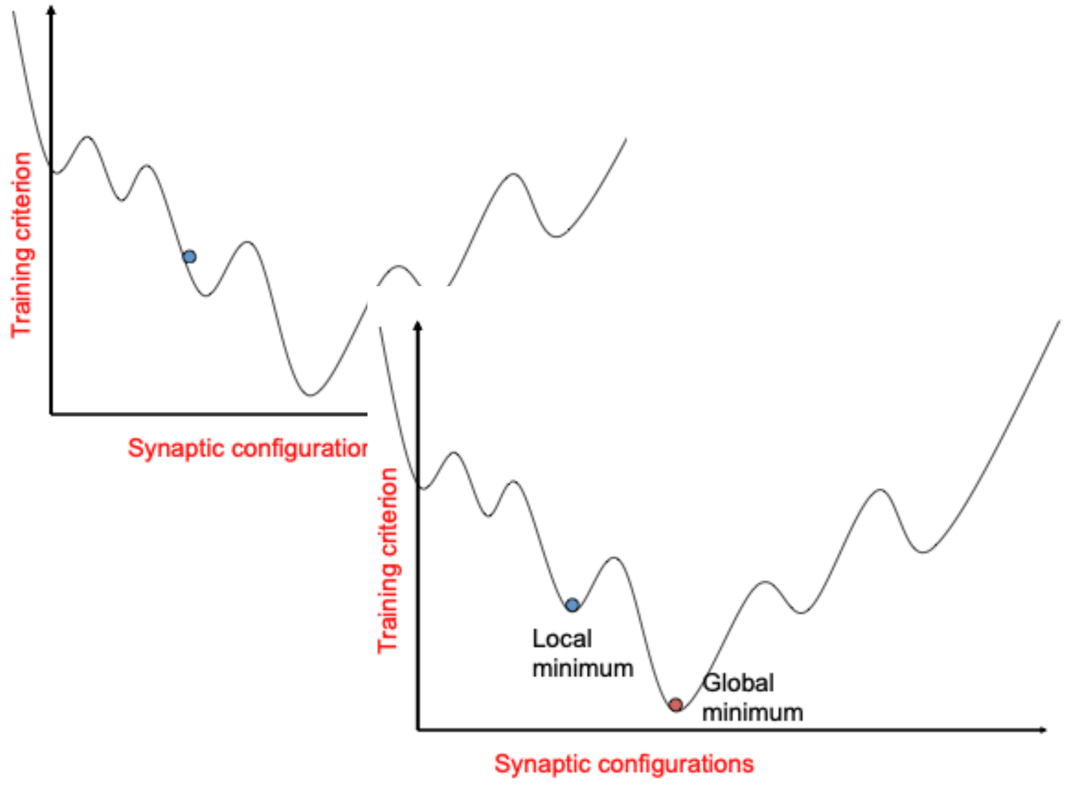
Question Time

Why did the curriculum (only) surpass the joint data training at the very end?

Universität Bremen

Computationally: “continuation” & “annealing” method

Local Descent Hypothesis: “When the brain of a single biological agent learns, it relies on approximate local descent in order to gradually improve itself”



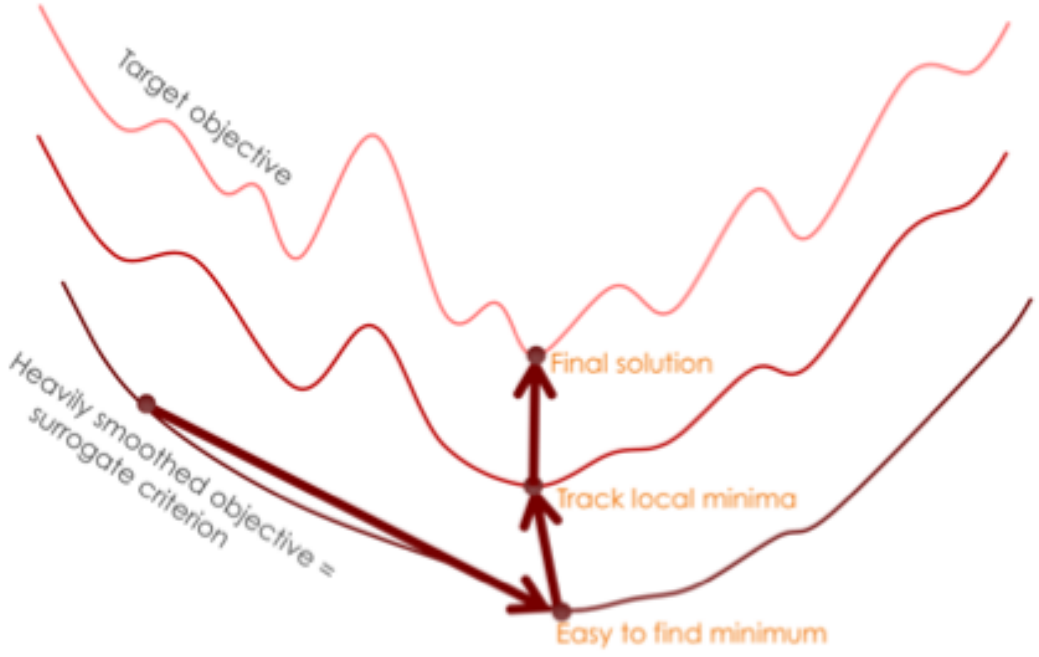
Y. Bengio, “Evolving culture versus local minima”, Growing Adaptive Machines, Springer, 2014

Universität Bremen

Computationally: “continuation” & “annealing” method

Local Descent Hypothesis: “When the brain of a single biological agent learns, it relies on approximate local descent in order to gradually improve itself”

Guided Learning Hypothesis: “A human brain can much more easily learn high-level abstractions if guided by the signals produced by other humans, which act as hints or indirect supervision for these high-level abstractions.



Y. Bengio, “Evolving culture versus local minima”, Growing Adaptive Machines, Springer, 2014

Curriculum learning: a formal definition

Definition: Original Curriculum Learning. (Bengio 2009)

A curriculum is a sequence of training criteria over T training steps: $C = < Q_1, \ldots, Q_t, \ldots, Q_T >$. Each criterion Q_t is a reweighing of the target training distribution $P(z)$:

$$Q_t(z) \propto W_t(z)P(z) \quad \forall\, \texttt{example}\; z \in \texttt{training set}\; D,$$

Bengio et al, "Curriculum Learning", ICML 2009

Universität Bremen

Curriculum learning: a formal definition

Definition: Original Curriculum Learning. (Bengio 2009)

A curriculum is a sequence of training criteria over T training steps: $C = < Q_1, \ldots, Q_t, \ldots, Q_T >$. Each criterion Q_t is a reweighing of the target training distribution $P(z)$:

$$Q_t(z) \propto W_t(z)P(z) \quad \forall\, \texttt{example}\; z \in \texttt{training set}\; D,$$

Such that the following three conditions are satisfied

1. The entropy of distributions gradually increases: $H(Q_t) < H(Q_{t+1})$.
2. The weight increases monotonically: $W_t(z) \leq W_{t+1}(z) \quad \forall z \in D$
3. $Q_T(z) = P(z)$

Bengio et al, “Curriculum Learning”, ICML 2009

Curriculum learning: a formal definition

Definition: Generalized Curriculum Learning. (Wang 2021)

Discarding the definition of Q_t and its three conditions in Definition 1, a curriculum is a sequence of training criteria over T training steps. Each criterion Q_t includes the design for all the elements in training a machine learning model, e.g., data/tasks, model capacity, learning objective, etc. Curriculum learning is the strategy that trains a model with such a curriculum.

(A bit of a tautology by) Wang et al, "A Survey on Curriculum Learning", TPAMI 2021, based on the original definition by Bengio on the slide before

Question time

Given these definitions: what are the central questions in curriculum learning?

Two key aspects

Scoring function (difficulty measurer):

Any function that provides us with an estimate of the difficulty of the instances in our dataset(s)

Two key aspects

Scoring function (difficulty measurer):

Any function that provides us with an estimate of the difficulty of the instances in our dataset(s)

Pacing function (training scheduler):

The function that tells us how to interleave samples into the training process over time.

Universität Bremen

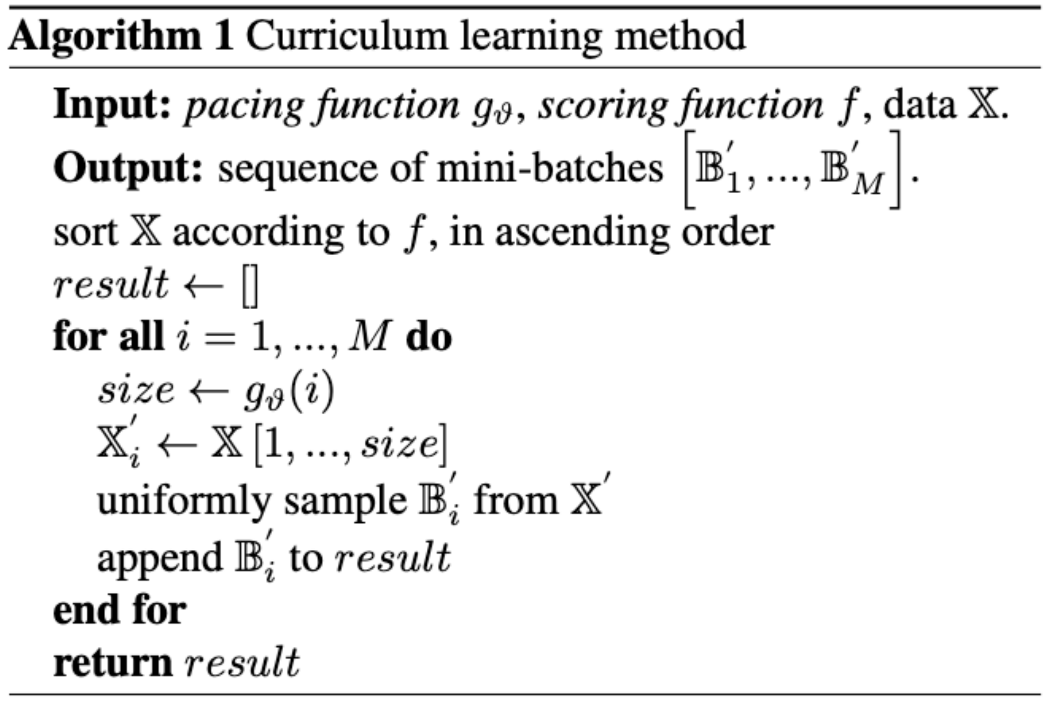
Two key aspects

Scoring function (difficulty measurer):

Any function that provides us with an estimate of the difficulty of the instances in our dataset(s)

Pacing function (training scheduler):

The function that tells us how to interleave samples into the training process over time.



Algorithm 1 Curriculum learning method

```
Input: pacing function $g_\vartheta$, scoring function $f$, data $\mathbb{X}$.
Output: sequence of mini-batches $\left[\mathbb{B}'_1, ..., \mathbb{B}'_M\right]$.
sort $\mathbb{X}$ according to $f$, in ascending order
$result \leftarrow []$
for all $i = 1, ..., M$ do
    $size \leftarrow g_\vartheta(i)$
    $\mathbb{X}'_i \leftarrow \mathbb{X}[1, ..., size]$
    uniformly sample $\mathbb{B}'_i$ from $\mathbb{X}'$
    append $\mathbb{B}'_i$ to $result$
end for
return $result$
```

Algorithm from Hacohen & Weinshall, “On the power of curriculum learning in deep networks”, ICML 2019

Universität Bremen

Curriculum learning: optimization intuition

$$\mathscr{L}(\theta) = 1/N \sum_{i=1}^{N} \mathscr{L}_\theta(x_i) \qquad \text{and} \qquad \theta = \mathtt{argmin}_\theta \mathscr{L}(\theta)$$

Instead, we can also rewrite it in terms of maximum likelihood estimation:

$$\theta = \mathtt{argmax}_\theta \exp(-\sum_{i=1}^{N} L_\theta(X_i)) = \mathtt{argmax}_\theta \prod_{i=1}^{N} e^{-L_\theta(x_i)}$$

With the probability defined as: $P(\theta \,|\, x) \propto e^{-L_\theta(x)}$

Hacohen & Weinshall, "On the power of curriculum learning in deep networks", ICML 2019

Curriculum learning: optimization intuition

With the probability defined as: $P(\theta | x) \propto e^{-L_\theta(x)}$

We can view this as choosing to maximize the average utility of the observed data: $\mathcal{U}(\theta) = \frac{1}{N}\sum_{i=1}^{N} e^{-L_\theta(x_i)}$ $\quad$ *and* $\quad$ $\mathtt{argmax}_\theta \mathcal{U}(\theta)$

Hacohen & Weinshall, "On the power of curriculum learning in deep networks", ICML 2019

Curriculum learning: optimization intuition

With the probability defined as: $P(\theta | x) \propto e^{-L_\theta(x)}$

We can view this as choosing to maximize the average utility of the observed data: $\mathcal{U}(\theta) = \frac{1}{N} \sum_{i=1}^{N} e^{-L_\theta(x_i)}$ $\quad$ *and* $\quad$ $\texttt{argmax}_\theta \mathcal{U}(\theta)$

The scoring function is then a (Bayesian) prior for data sampling:

$$\mathcal{U}_p(\theta) = \sum_{i=1}^{N} U_\theta(x_i)p(x_i) = \sum_{i=1}^{N} e^{-L_\theta(x_i)} p_i$$

Hacohen & Weinshall, "On the power of curriculum learning in deep networks", ICML 2019

Curriculum learning: optimization intuition

The scoring function is then a (Bayesian) prior for data sampling:

$$\mathcal{U}_p(\theta) = \sum_{i=1}^{N} U_\theta(x_i)p(x_i) = \sum_{i=1}^{N} e^{-L_\theta(x_i)}p_i$$

The prior probability $p(x_i)$ is determined by *scoring* and *pacing* functions

For instance, we could use $p(x_i) = \frac{1}{M}$ for training points above a certain difficulty threshold and $p(x_i) = 0$ otherwise

Hacohen & Weinshall, "On the power of curriculum learning in deep networks", ICML 2019

Curriculum learning: optimization intuition

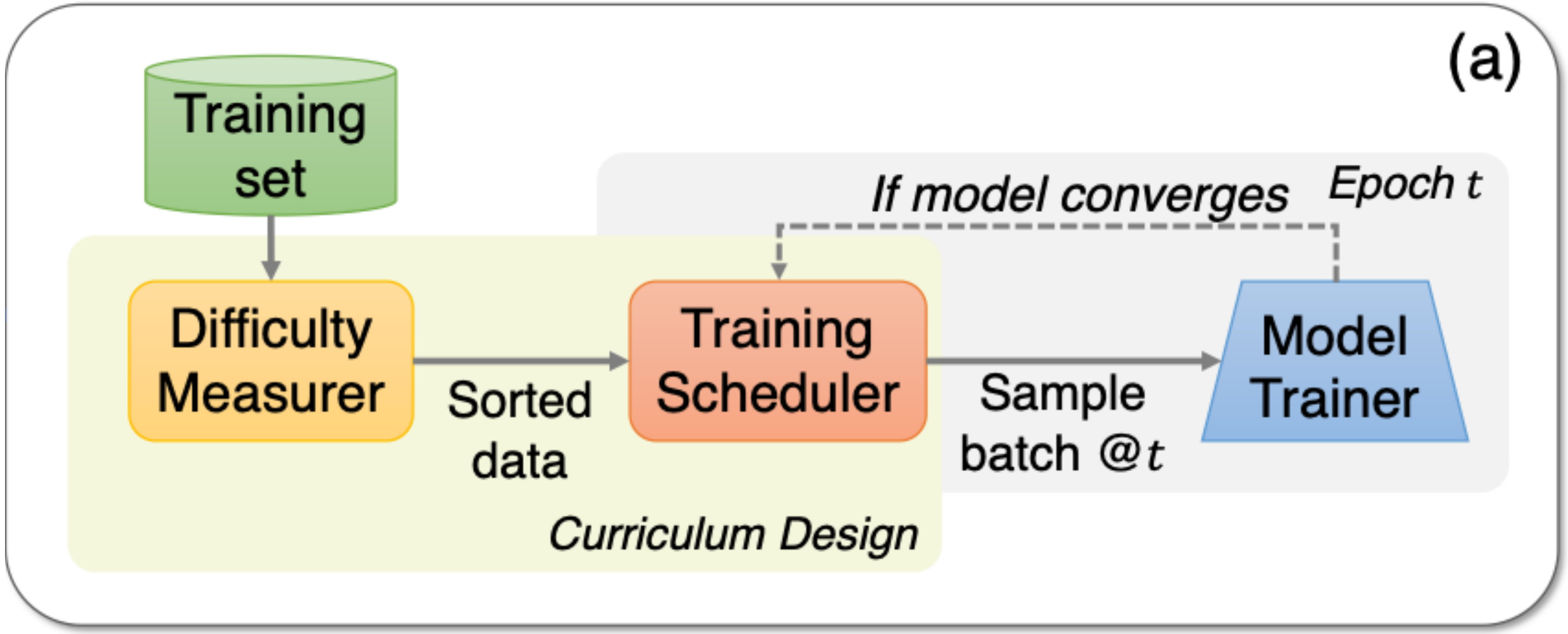
Hacohen & Weinshall (reference below) provide further theoretical analysis to quantify the difference between the expected utility when computed with and without prior p.

They arrive at an insight that this is related to the covariance between $U_\theta(x)$ and $p(x)$, and respectively that if the prior probability is positively correlated with an optimal utility, then the gradients in the new optimization landscape become overall steeper.

Let’s look at this from a slightly more practical & empirical point of view

Hacohen & Weinshall, “On the power of curriculum learning in deep networks”, ICML 2019

We start by considering a *pre-defined* curriculum



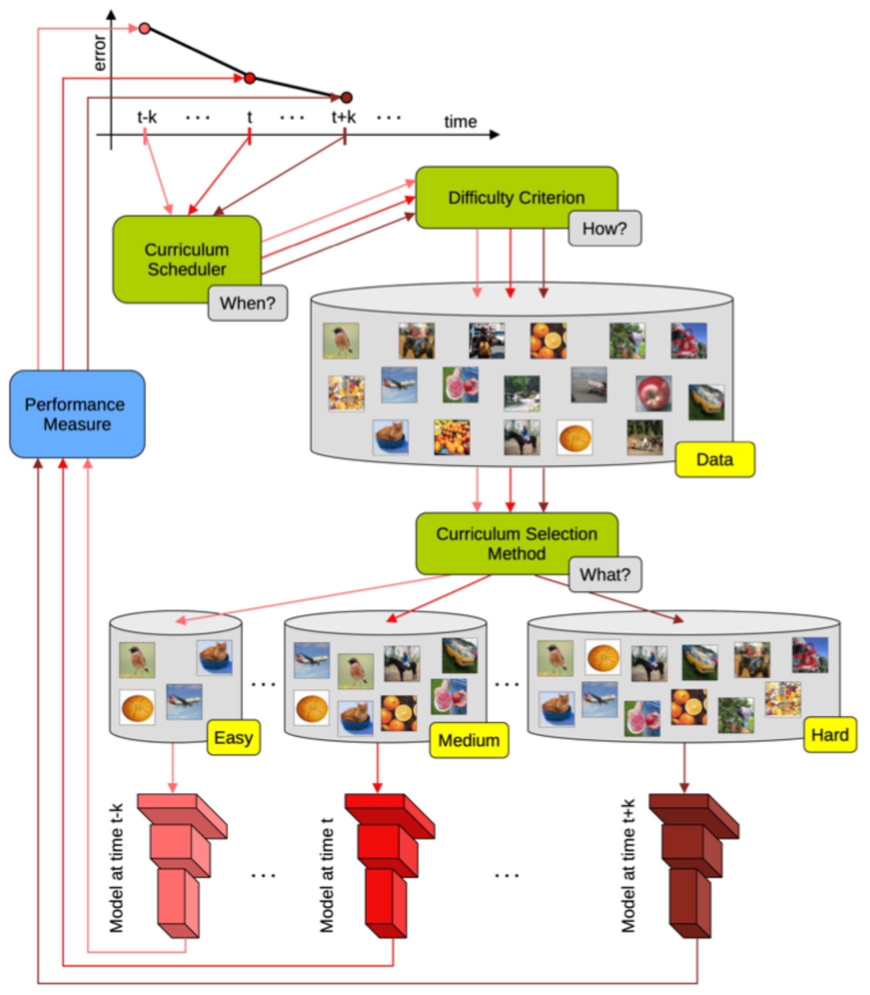
Wang et al, “A Survey on Curriculum Learning”, TPAMI 2021

"Data-level" curriculum

In a pre-defined, "data-level" curriculum, we first select a difficulty criterion.

Think of it as "learning from a textbook".

"Easier" concepts appear first, and "more difficulty" examples, that require prior knowledge are presented later



Soviany et al, "Curriculum Learning: A Survey", IJCV, 2022

Universität
Bremen

Question time

Can you think of ways to define “difficulty”?

Pre-defined difficulty

Many different "angles" to difficulty

Some notions of difficulty may be agnostic to data type, but many intuitive criteria are specific

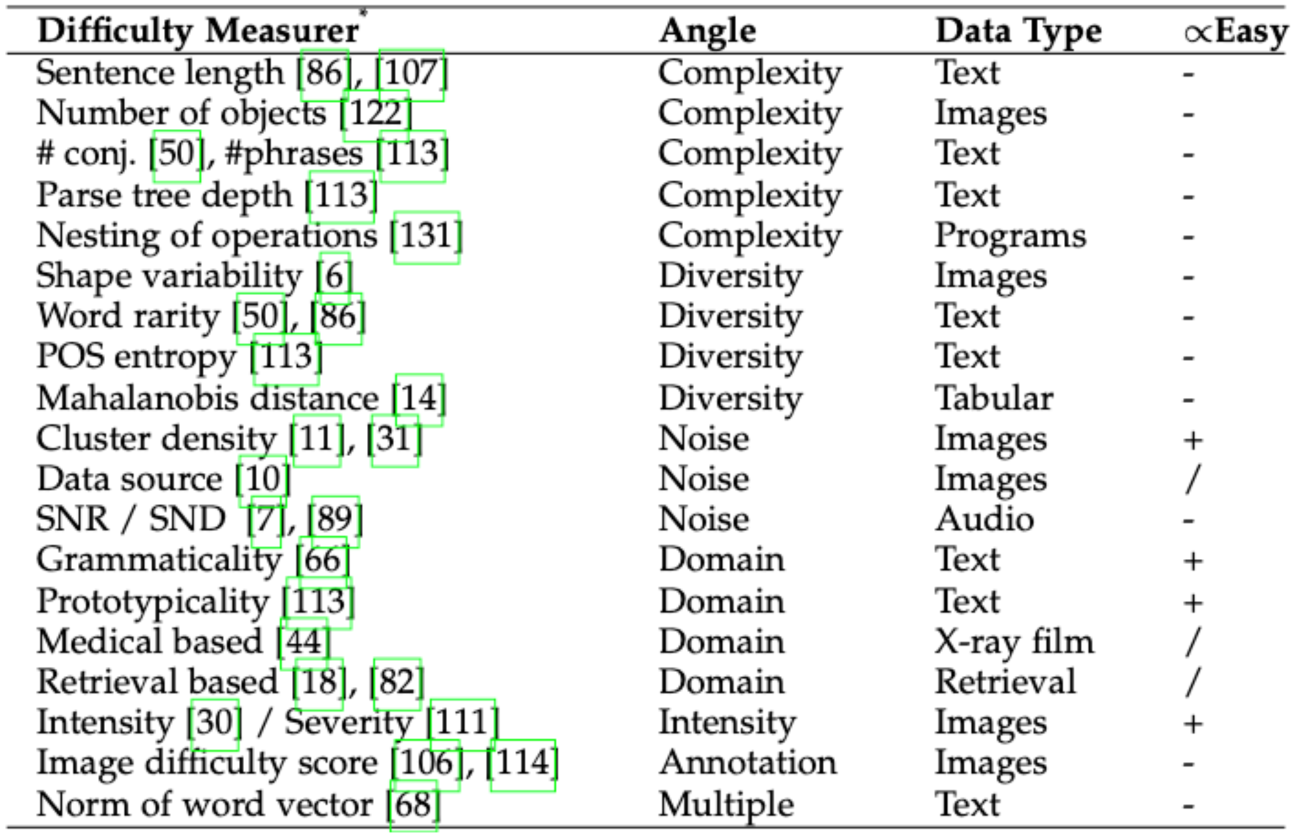


TABLE 2
Common types of predefined Difficulty Measurer. The "+" in ∝Easy means the higher the measured value, the easier the data example, and the "-" has the opposite meaning.

Difficulty Measurer*	**Angle**	**Data Type**	**∝Easy**
Sentence length [86], [107]	Complexity	Text	-
Number of objects [122]	Complexity	Images	-
# conj. [50], #phrases [113]	Complexity	Text	-
Parse tree depth [113]	Complexity	Text	-
Nesting of operations [131]	Complexity	Programs	-
Shape variability [6]	Diversity	Images	-
Word rarity [50], [86]	Diversity	Text	-
POS entropy [113]	Diversity	Text	-
Mahalanobis distance [14]	Diversity	Tabular	-
Cluster density [11], [31]	Noise	Images	+
Data source [10]	Noise	Images	/
SNR / SND [7], [89]	Noise	Audio	-
Grammaticality [66]	Domain	Text	+
Prototypicality [113]	Domain	Text	+
Medical based [44]	Domain	X-ray film	/
Retrieval based [18], [82]	Domain	Retrieval	/
Intensity [30] / Severity [111]	Intensity	Images	+
Image difficulty score [106], [114]	Annotation	Images	-
Norm of word vector [68]	Multiple	Text	-

Pre-defined difficulty

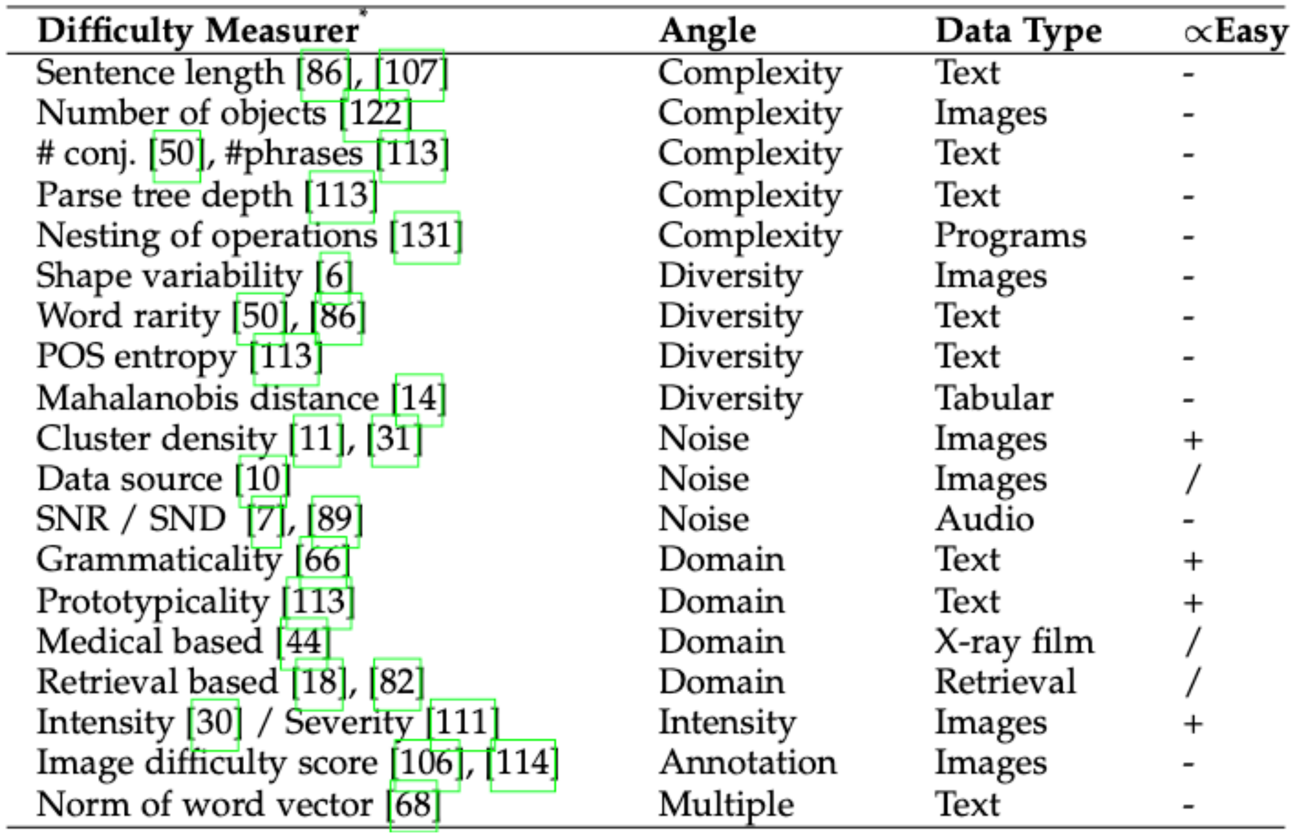
Many different “angles” to difficulty

Some notions of difficulty may be agnostic to data type, but many intuitive criteria are specific

Depending on the criterion, more difficulty may imply increase/ decrease of the measure

A curriculum is also often posed as “clear-to-ambiguous” samples

TABLE 2
Common types of predefined Difficulty Measurer. The “+” in $\propto$Easy means the higher the measured value, the easier the data example, and the “-” has the opposite meaning.



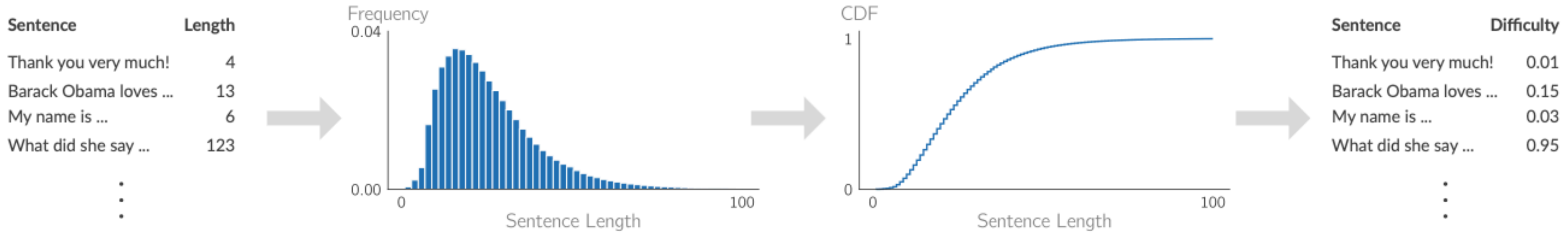
Difficulty Measurer*	**Angle**	**Data Type**	**$\propto$Easy**
Sentence length [86], [107]	Complexity	Text	-
Number of objects [122]	Complexity	Images	-
# conj. [50], #phrases [113]	Complexity	Text	-
Parse tree depth [113]	Complexity	Text	-
Nesting of operations [131]	Complexity	Programs	-
Shape variability [6]	Diversity	Images	-
Word rarity [50], [86]	Diversity	Text	-
POS entropy [113]	Diversity	Text	-
Mahalanobis distance [14]	Diversity	Tabular	-
Cluster density [11], [31]	Noise	Images	+
Data source [10]	Noise	Images	/
SNR / SND [7], [89]	Noise	Audio	-
Grammaticality [66]	Domain	Text	+
Prototypicality [113]	Domain	Text	+
Medical based [44]	Domain	X-ray film	/
Retrieval based [18], [82]	Domain	Retrieval	/
Intensity [30] / Severity [111]	Intensity	Images	+
Image difficulty score [106], [114]	Annotation	Images	-
Norm of word vector [68]	Multiple	Text	-

Universität
Bremen

Question time

What is the relationship between task difficulty and model choice?

Specific tasks -> specific difficulty definitions



Sentence	Length
Thank you very much!	4
Barack Obama loves ...	13
My name is ...	6
What did she say ...	123
⋮	

Sentence	Difficulty
Thank you very much!	0.01
Barack Obama loves ...	0.15
My name is ...	0.03
What did she say ...	0.95
⋮	

Figure 2: Example visualization of the preprocessing sequence used in the proposed algorithm. The histogram shown is that of sentence lengths from the WMT-16 En→De dataset used in our experiments. Here sentence lengths represent an example difficulty scoring function, *d*. "CDF" stands for the empirical "cumulative density function" obtained from the histogram on the left plot.

Platanios et al, "Competence based curriculum learning for neural machine translation", NAACL-HIT 2019

Universität Bremen

Specific tasks -> specific difficulty definitions

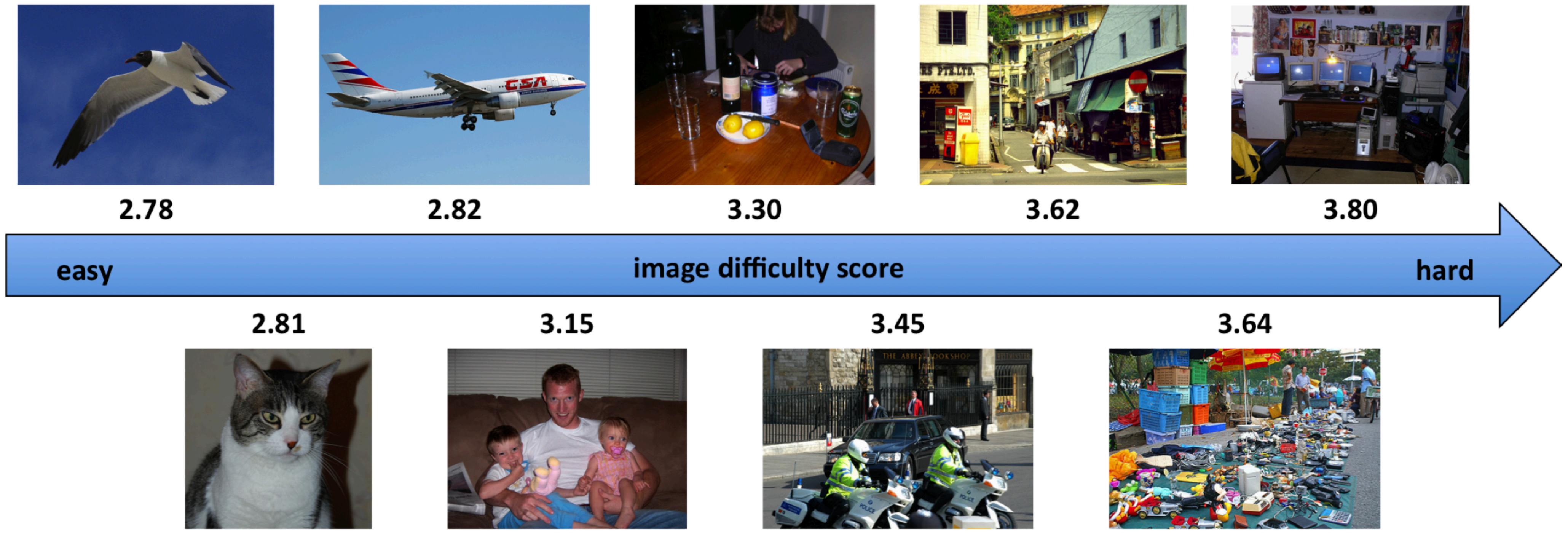


Figure 1. Images with difficulty scores predicted by our system in increasing order of their difficulty.

Ionescu et al, “How hard can it be? Estimating the difficulty of visual search in an image”, CVPR 2016

However, it’s not just about data statistics

Compositional factors:

Size

“A sail boat on the ocean.”

Location

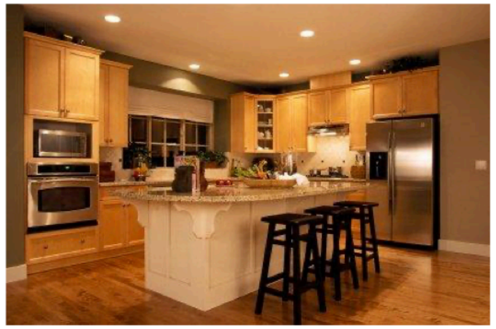
“Two men standing on beach.”

Semantic factors:

Object Type

“Girl in the street”

Scene Type & Depiction Strength


“kitchen in house”

Context factors:

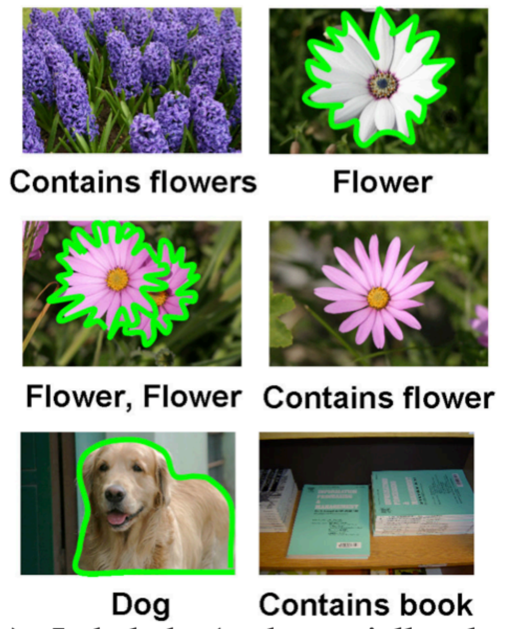
Unusual object-scene Pair

“A tree in water and a boy with a beard”

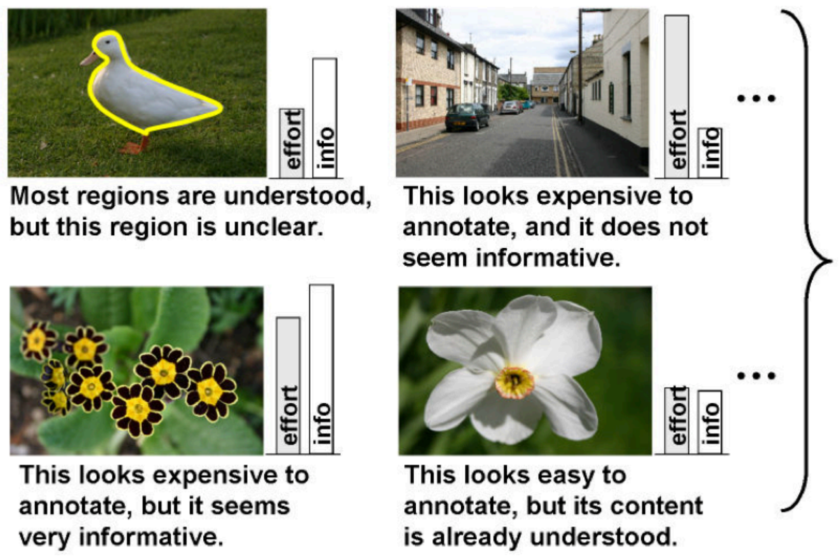
Berg et al, “Understanding and predicting importance in images”, CVPR 2012

Universität Bremen

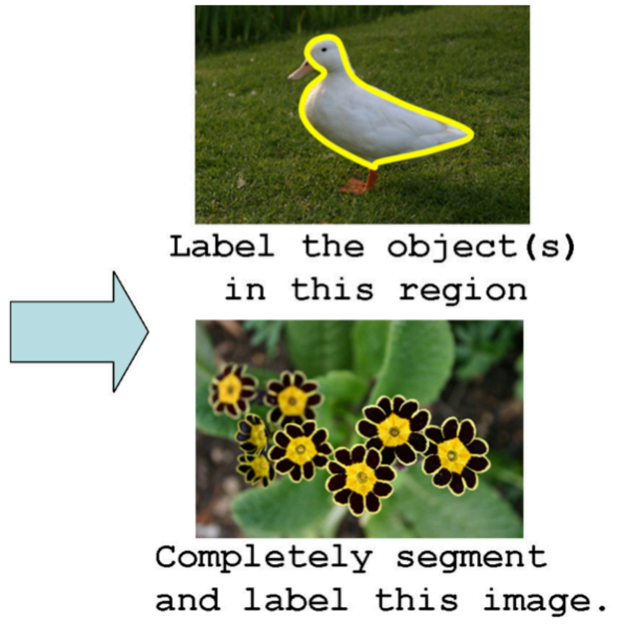
And curricula are different based on tasks: e.g. difficulty with respect to annotation cost



(a) Labeled (and partially labeled) examples to build models


effort
info
Most regions are understood, but this region is unclear.
This looks expensive to annotate, and it does not seem informative.
This looks expensive to annotate, but it seems very informative.
This looks easy to annotate, but its content is already understood.
...

(b) Unlabeled and partially labeled examples to survey


Label the object(s) in this region
Completely segment and label this image.

(c) Actively chosen queries sent to annotators

Vijayanarasimhan & Grauman, "What's It Going to Cost You?: Predicting Effort vs. Informativeness for Multi-Label Image Annotations", CVPR 2009

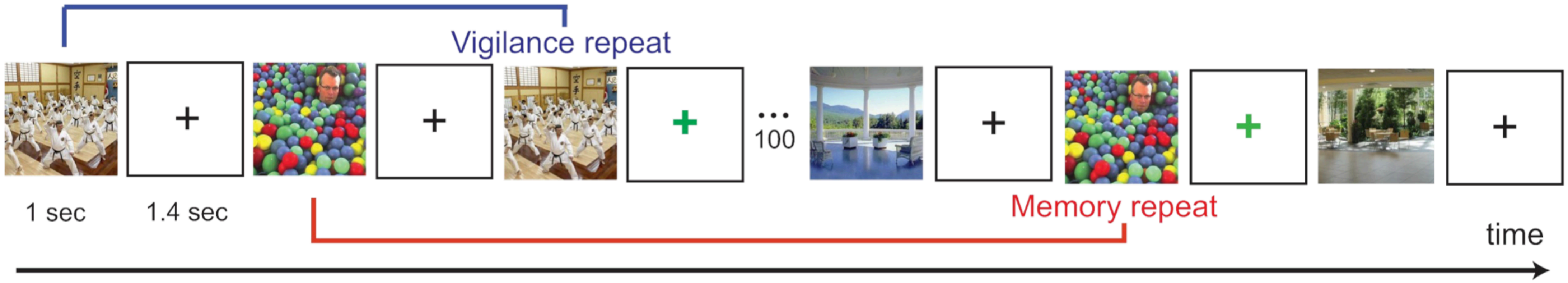
Question time

Do you expect your intuition of what is difficult to correspond to good "difficulty" scores in ML?

Universität Bremen

Study 1: what is memorable to humans?

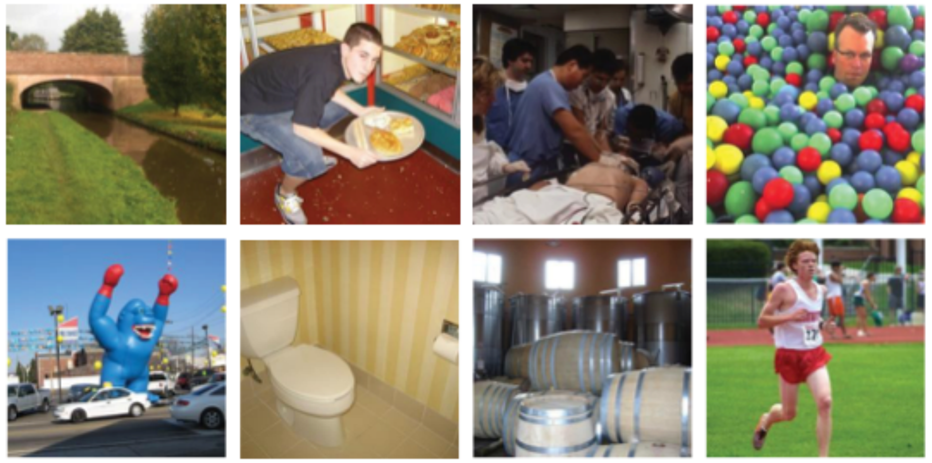
- Let people play a "game" in which they find repeats in a stream of images
- Each image shown for 1 sec, 1.4 sec pause, in a sequence of 120 images
- Press the space bar when a repeat of an identical image is noticed
- Unknown to participants, there were "targets" and "fillers"

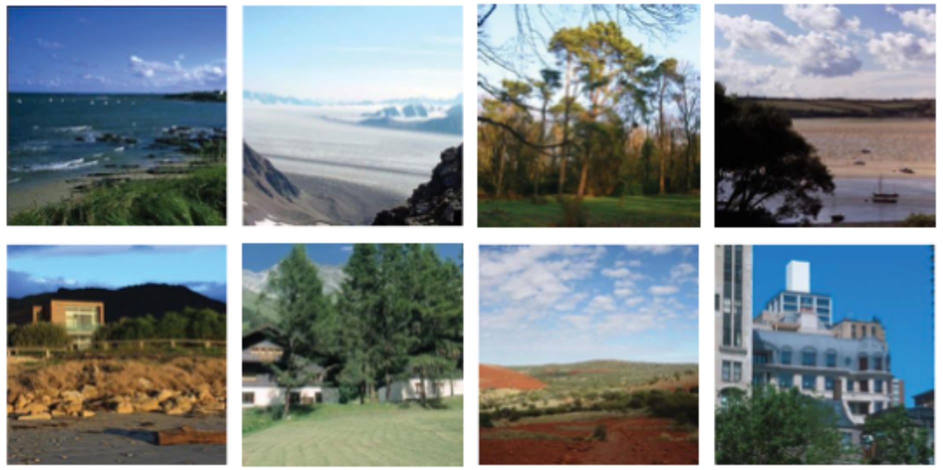


Isola et al, "What makes an image memorable", CVPR 2011

Study 1: what is memorable to humans?

- Strong correlations observed between the human participants


a) Most memorable images (86%)


c) Least memorable images (34%)

Isola et al, “What makes an image memorable”, CVPR 2011

Study 1: what is memorable to humans?

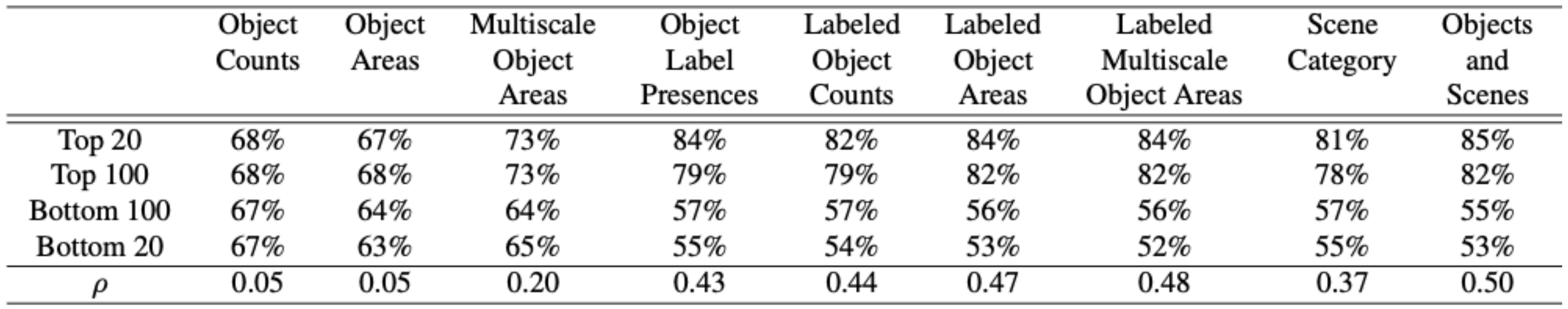
- An investigation has then checked correlations with several factors: color, simple image features, object stats, object semantics, scene semantics
- Train regressors on factors to predict memorability & compare to humans

Isola et al, “What makes an image memorable”, CVPR 2011

Universität Bremen

Study 1: what is memorable to humans?

- An investigation has then checked correlations with several factors: color, simple image features, object stats, object semantics, scene semantics
- Train regressors on factors to predict memorability & compare to humans
- Table: avg. empirically measured memorability for top & bottom images and Spearman ρ rank correlation between prediction & measurement



	Object Counts	Object Areas	Multiscale Object Areas	Object Label Presences	Labeled Object Counts	Labeled Object Areas	Labeled Multiscale Object Areas	Scene Category	Objects and Scenes
Top 20	68%	67%	73%	84%	82%	84%	84%	81%	85%
Top 100	68%	68%	73%	79%	79%	82%	82%	78%	82%
Bottom 100	67%	64%	64%	57%	57%	56%	56%	57%	55%
Bottom 20	67%	63%	65%	55%	54%	53%	52%	55%	53%
ρ	0.05	0.05	0.20	0.43	0.44	0.47	0.48	0.37	0.50

Isola et al, “What makes an image memorable”, CVPR 2011

Universität Bremen

Study 2: recording human response times

Collecting response times.
"We collected ground-truth difficulty annotations by human evaluators using the following protocol:

i) we ask each annotator a question of the type "Is there an {object class} in the next image?", where {object class} is one of the 20 classes included in the PASCAL VOC 2012

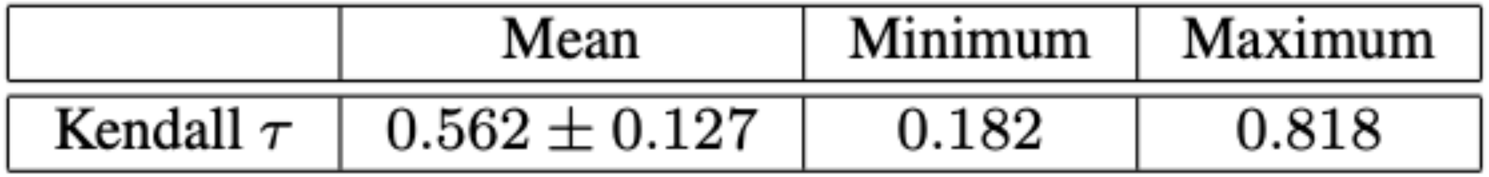
ii) We show the image to the annotator

iii) We record the time spent by the annotator to answer the question by "Yes" or "No"

Ionescu et al, "How hard can it be? Estimating the difficulty of visual search in an image", CVPR 2016

Study 2: recording human response times

- First: find correlation between humans (here, Kendall's τ)
- 0.562 means that the "average" human ranks about 80% of image pairs in the same order as given by the mean response time of all annotators



	Mean	Minimum	Maximum
Kendall τ	0.562 ± 0.127	0.182	0.818

Ionescu et al, "How hard can it be? Estimating the difficulty of visual search in an image", CVPR 2016

Universität Bremen

Study 2: recording human response times

- Check correlations across image properties

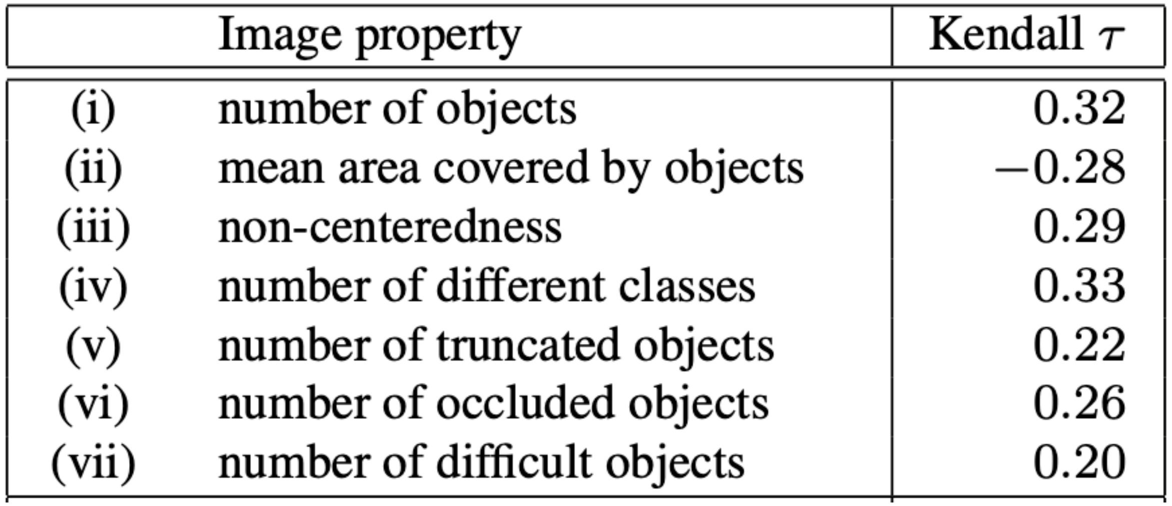


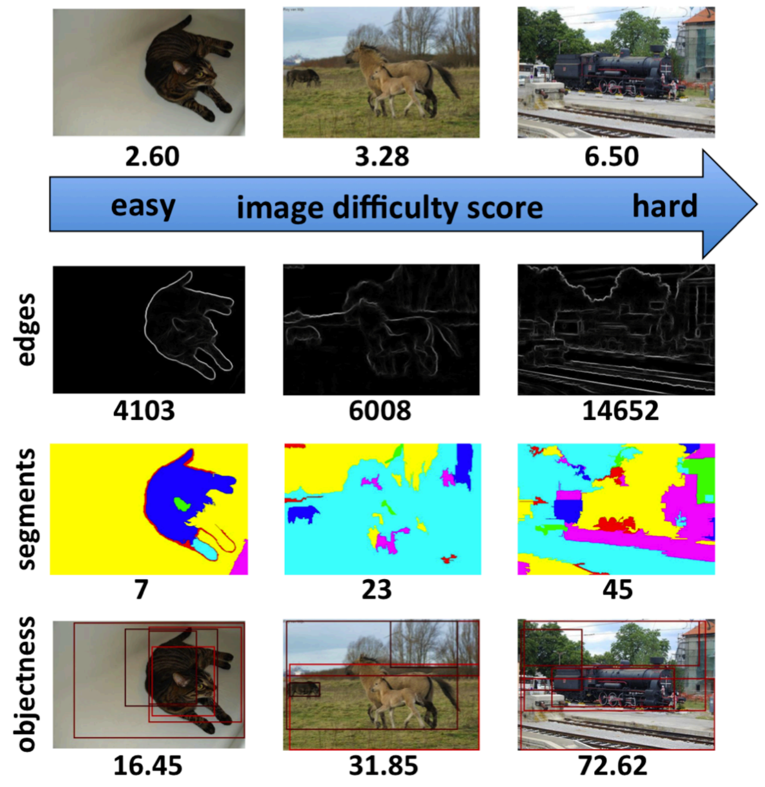
	Image property	Kendall τ
(i)	number of objects	0.32
(ii)	mean area covered by objects	-0.28
(iii)	non-centeredness	0.29
(iv)	number of different classes	0.33
(v)	number of truncated objects	0.22
(vi)	number of occluded objects	0.26
(vii)	number of difficult objects	0.20

Ionescu et al, “How hard can it be? Estimating the difficulty of visual search in an image”, CVPR 2016

Study 2: recording human response times

- Check correlations across image properties
- Train (deep) regressors on individual features and learning to predict from their combination

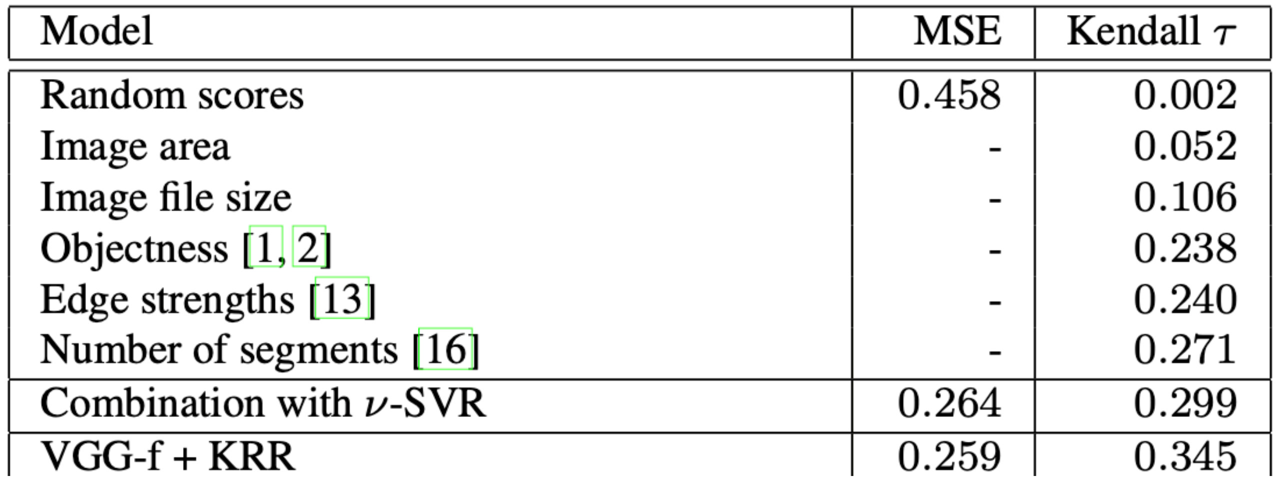
	Image property	Kendall τ
(i)	number of objects	0.32
(ii)	mean area covered by objects	−0.28
(iii)	non-centeredness	0.29
(iv)	number of different classes	0.33
(v)	number of truncated objects	0.22
(vi)	number of occluded objects	0.26
(vii)	number of difficult objects	0.20



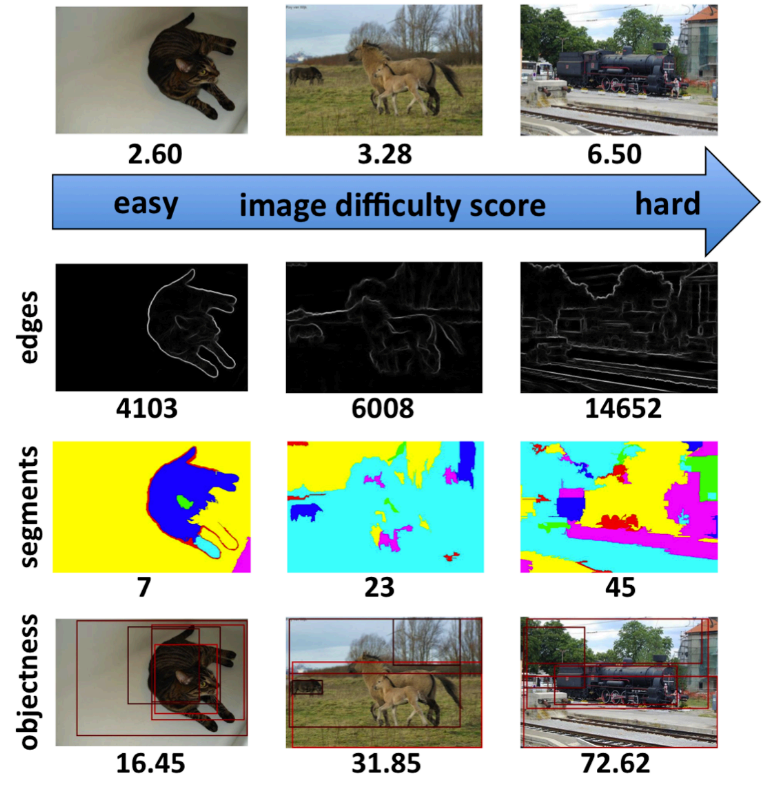
Ionescu et al, "How hard can it be? Estimating the difficulty of visual search in an image", CVPR 2016

Study 2: recording human response times

- Check correlations across image properties
- Train (deep) regressors on individual features and learning to predict from their combination



Model	MSE	Kendall τ
Random scores	0.458	0.002
Image area	-	0.052
Image file size	-	0.106
Objectness [1, 2]	-	0.238
Edge strengths [13]	-	0.240
Number of segments [16]	-	0.271
Combination with ν-SVR	0.264	0.299
VGG-f + KRR	0.259	0.345



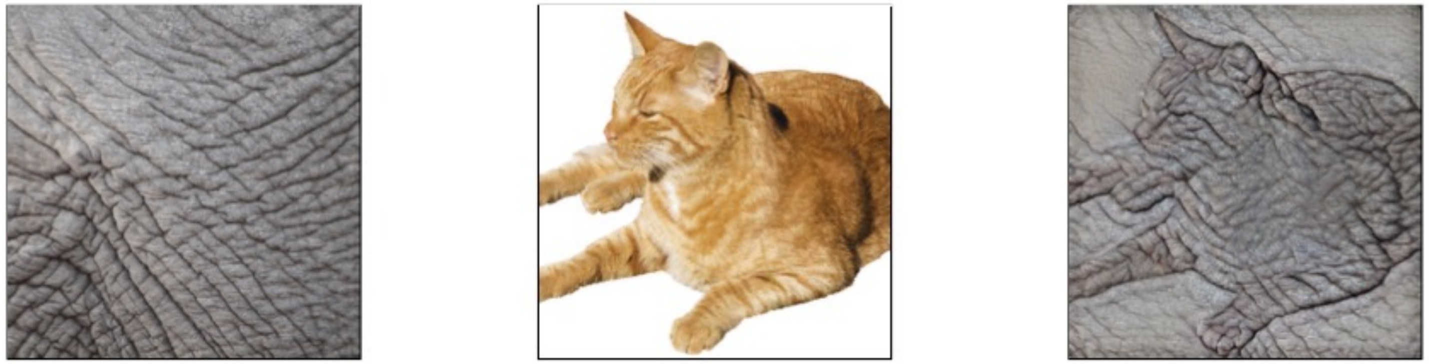
Ionescu et al, “How hard can it be? Estimating the difficulty of visual search in an image”, CVPR 2016

Question time

There are some observable correlations, but you might have expected more.
What do you think are the issues & pitfalls?

Representation bias

In ML models, especially the non-linear opaque like deep neural networks, representations can be biased in ways we don’t anticipate.
Example: decisions can be based on textures, ignoring geometry!



(a) Texture image

81.4%	**Indian elephant**
10.3%	indri
8.2%	black swan

(b) Content image

71.1%	**tabby cat**
17.3%	grey fox
3.3%	Siamese cat

(c) Texture-shape cue conflict

63.9%	**Indian elephant**
26.4%	indri
9.6%	black swan

“ImageNet-trained CNNs are biased towards texture”, Geirhos et al, ICLR 2019

Simplicity bias

In general, there is a phenomenon that can be referred to as simplicity bias.

For instance, neural networks tend to favor simpler solutions, even if another feature would be equally predictive!

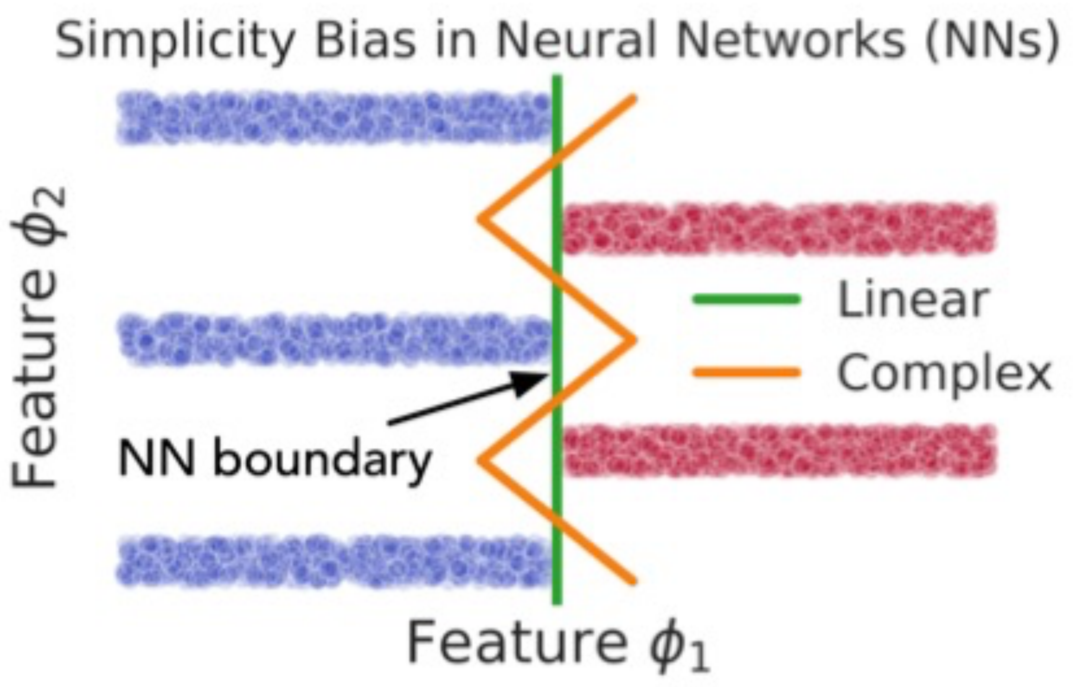
Universität
Bremen

Simplicity bias

In general, there is a phenomenon that can be referred to as simplicity bias.

For instance, neural networks tend to favor simpler solutions, even if another feature would be equally predictive!

In the right-hand example, the "orange" solution would actually be favorable, but is not the one generally learned



Shah et al, "The Pitfalls of Simplicity Bias in Neural Networks", NeurIPS 2020

Confounders

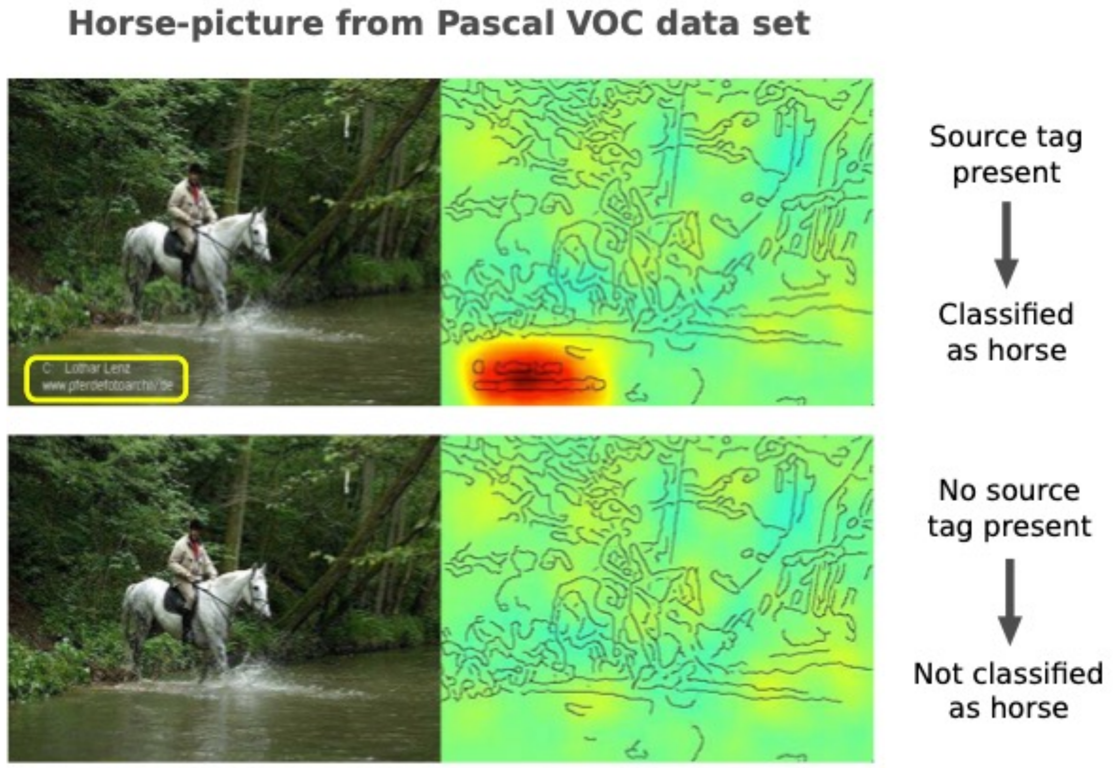
We rarely anticipate are confounders: features that *can be relied on* for prediction due to correlations in the dataset, *without real predictive power*

Universität Bremen

Confounders

We rarely anticipate are confounders: features that *can be relied on* for prediction due to correlations in the dataset, *without real predictive power*

Pascal is a famous example of this, where the “Clever Hans” (related to a famous horse in history) phenomenon was exposed: most to all images of the dataset of horses contain photographer’s tags



“Unmasking Clever Hans Predictors”, Lapuschkin et al, Nature Communications 2019

Adversarial features

And finally, some confounders are really hard to spot, because they are adversarial to the decision making.

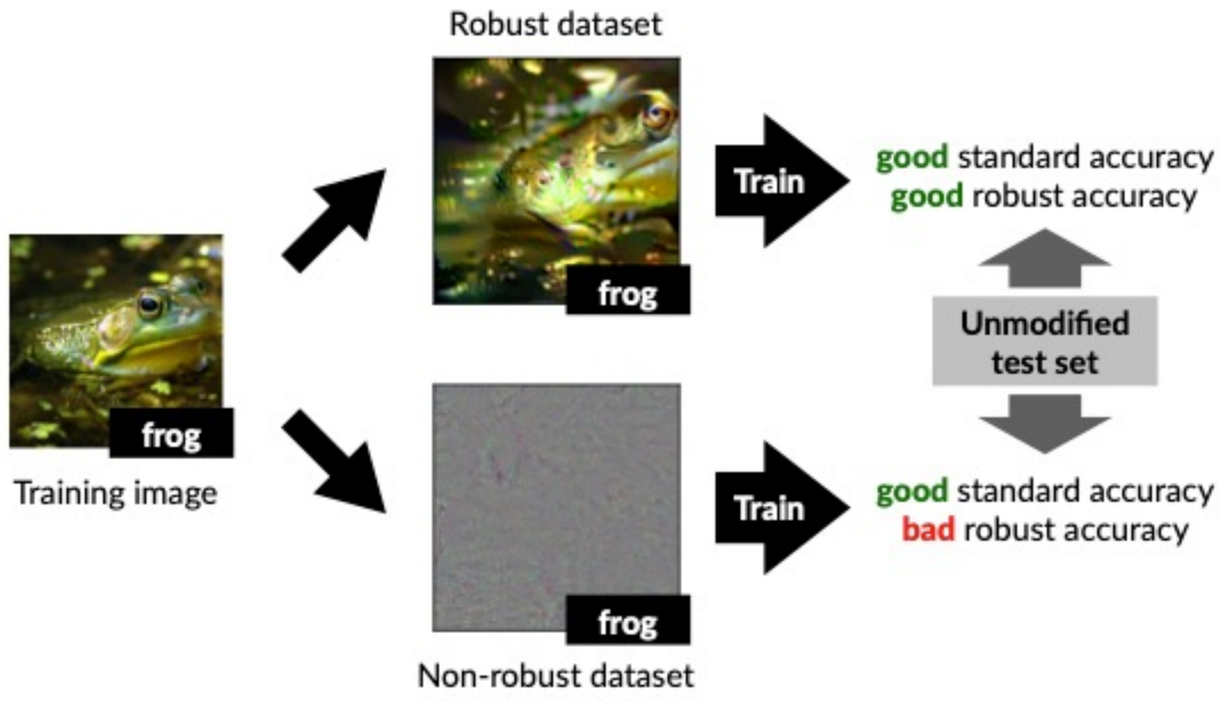
A most popular instance are frequency-based perturbations that are minor in the sense that they are hard/impossible to spot for humans.

Adversarial features

And finally, some confounders are really hard to spot, because they are adversarial to the decision making.

A most popular instance are frequency-based perturbations that are minor in the sense that they are hard/impossible to spot for humans.

It has been shown that training on these perturbations *alone can* perform well on benchmark test sets.



“Adversarial Examples are not Bugs, they are Features”, Ilyas et al, NeurIPS 2019

Universität Bremen

Lack of understanding of what is learned

We still require more understanding of what these (deep) models learn

However, there are some interesting curriculum results between ML models

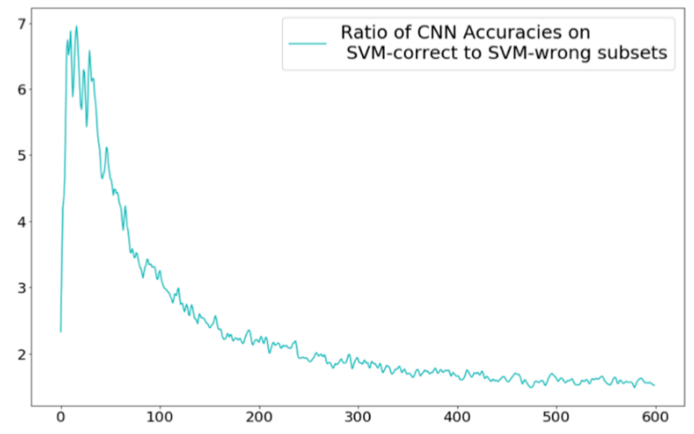
Universität Bremen

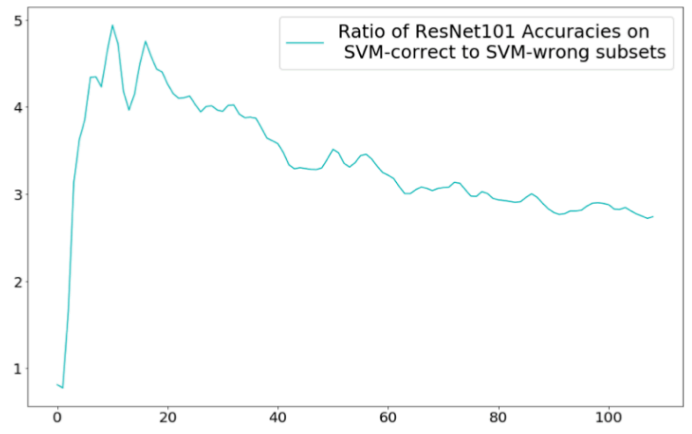
Lack of understanding of what is learned

We still require more understanding of what these (deep) models learn

However, there are some interesting curriculum results between ML models

For instance, it has been shown that deep networks learn the same instances first, that shallow networks (SVM, random forests) are able to learn



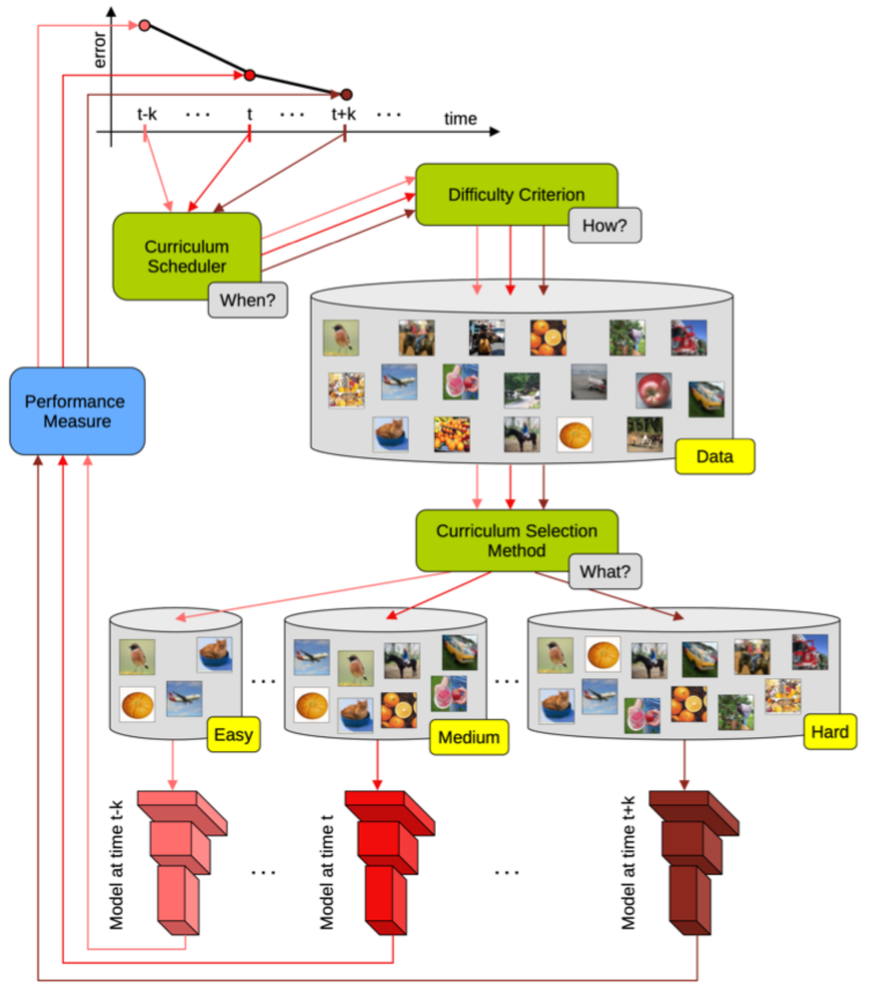


Mangalam & Prabhu, "Do deep neural networks learn shallow learnable examples first?", ICML-W 2019

"Data-level" curriculum

We will encounter many of these concerns and how to spot them again later in the course

For now, let's move back to curriculum learning, and concentrate on the second crucial part - *the pacing*



Soviany et al, "Curriculum Learning: A Survey", IJCV, 2022

Universität
Bremen

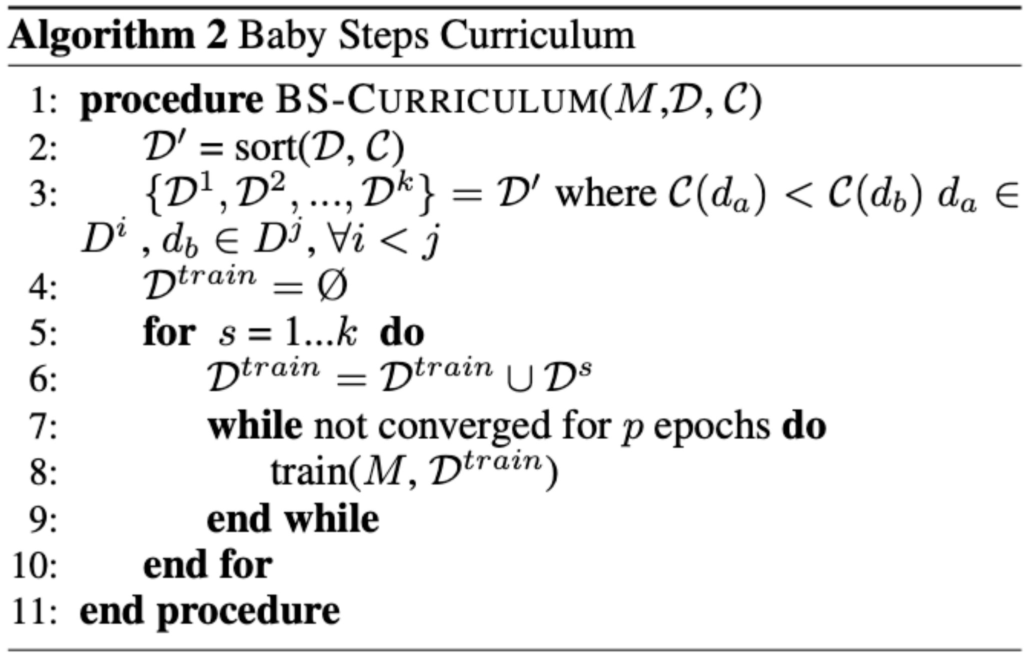
Question time

Assume a difficulty scorer that ranks all instances, how would you introduce the data into training?

“Baby-steps” curriculum

The most intuitive curriculum?

- Sort the data according to difficulty
- Create k bins
- Optimize on one bin until convergence
- Add a new bin in a *concatenated*, *growing* dataset



Algorithm 2 Baby Steps Curriculum

```
1:  procedure BS-CURRICULUM($M$,$\mathcal{D}$, $\mathcal{C}$)
2:      $\mathcal{D}'$ = sort($\mathcal{D}$, $\mathcal{C}$)
3:      $\{\mathcal{D}^1, \mathcal{D}^2, ..., \mathcal{D}^k\} = \mathcal{D}'$ where $\mathcal{C}(d_a) < \mathcal{C}(d_b)$ $d_a \in D^i$, $d_b \in D^j, \forall i < j$
4:      $\mathcal{D}^{train} = \emptyset$
5:      for $s = 1...k$ do
6:          $\mathcal{D}^{train} = \mathcal{D}^{train} \cup \mathcal{D}^s$
7:          while not converged for $p$ epochs do
8:              train($M$, $\mathcal{D}^{train}$)
9:          end while
10:     end for
11: end procedure
```

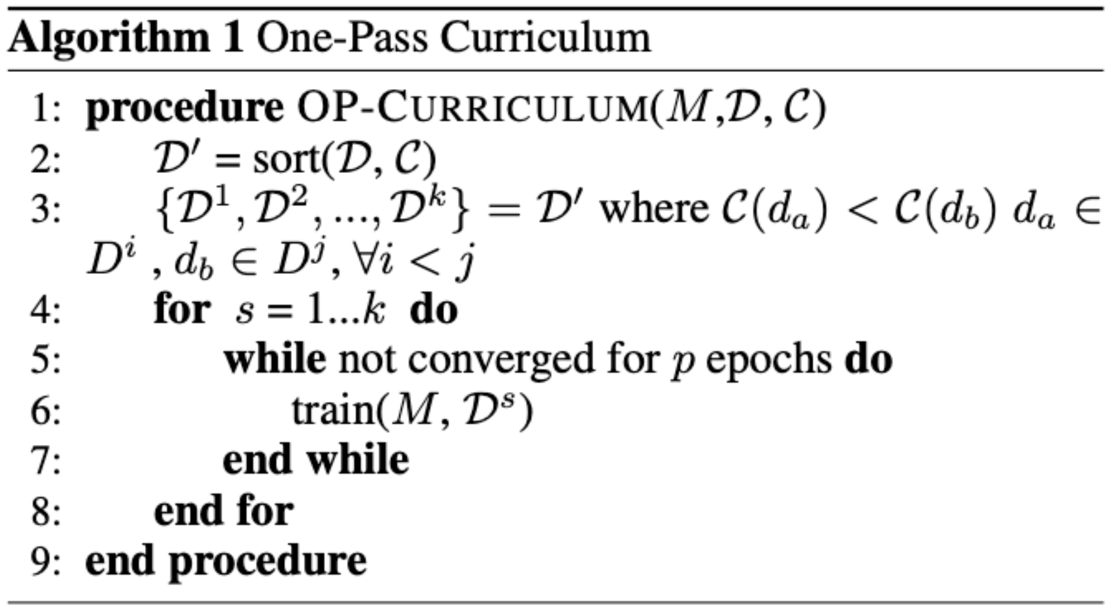
Algorithm from Cirik et al, “Visualizing and understanding curriculum learning for long short-term memory networks”, arXiv, 2016
Procedure from Spitkovsky et al, “From baby steps to leapfrogs: how less is more in unsupervised dependency parsing”, NAACL-HLT, 2010

Universität Bremen

“One-pass” curriculum

The intuitive alternative

- Sort the data according to difficulty
- Create k bins
- Optimize each bin *sequentially* until convergence - *do not accumulate* data



Algorithm 1 One-Pass Curriculum

```
1: procedure OP-CURRICULUM($M$,$\mathcal{D}$, $\mathcal{C}$)
2:     $\mathcal{D}' = \text{sort}(\mathcal{D}, \mathcal{C})$
3:     $\{\mathcal{D}^1, \mathcal{D}^2, ..., \mathcal{D}^k\} = \mathcal{D}'$ where $\mathcal{C}(d_a) < \mathcal{C}(d_b)$ $d_a \in D^i$ , $d_b \in D^j, \forall i < j$
4:     for $s = 1...k$ do
5:         while not converged for $p$ epochs do
6:             train($M, \mathcal{D}^s$)
7:         end while
8:     end for
9: end procedure
```

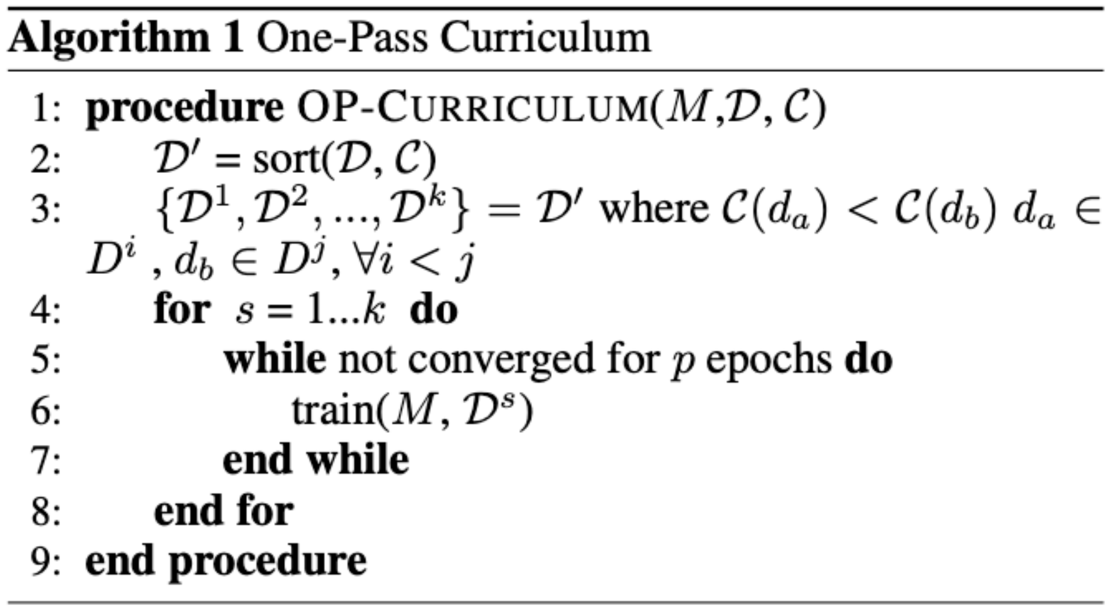
Algorithm from Cirik et al, “Visualizing and understanding curriculum learning for long short-term memory networks”, arXiv, 2016
Procedure from Bengio et al, “Curriculum Learning”, ICML 2009

“One-pass” curriculum

The intuitive alternative

- Sort the data according to difficulty
- Create k bins
- Optimize each bin *sequentially* until convergence - *do not accumulate* data

While intuitive, we will later see that this kind of pacing is very hard with several problems (like forgetting) without care. However, it is what is done in e.g. large-scale language models in practice



Algorithm 1 One-Pass Curriculum

```
procedure OP-CURRICULUM($M$,$\mathcal{D}$, $\mathcal{C}$)
    $\mathcal{D}' = \text{sort}(\mathcal{D}, \mathcal{C})$
    $\{\mathcal{D}^1, \mathcal{D}^2, ..., \mathcal{D}^k\} = \mathcal{D}'$ where $\mathcal{C}(d_a) < \mathcal{C}(d_b)$ $d_a \in D^i$ , $d_b \in D^j, \forall i < j$
    for $s = 1...k$ do
        while not converged for $p$ epochs do
            train($M, \mathcal{D}^s$)
        end while
    end for
end procedure
```

Algorithm from Cirik et al, “Visualizing and understanding curriculum learning for long short-term memory networks”, arXiv, 2016
Procedure from Bengio et al, “Curriculum Learning”, ICML 2009

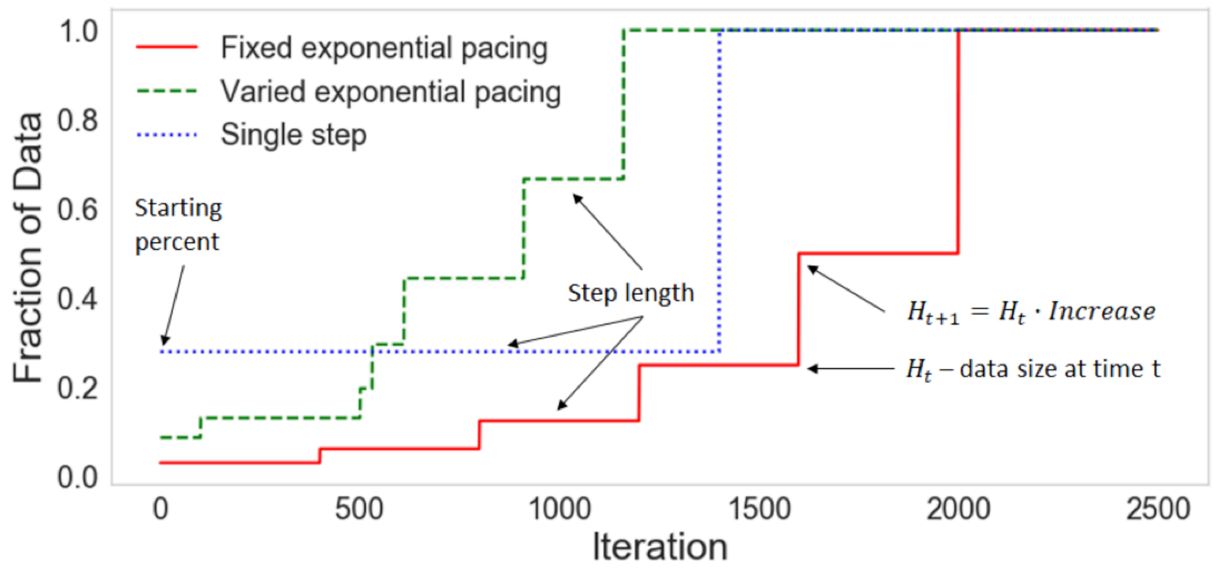
Universität Bremen

Pacing is often heuristic

Pacing functions are often heuristic

In pre-defined or "data-level" curriculum learning, they often follow a notion we think is intuitive

For example:
- "Build a strong foundation to learn the hard examples later (red)"



Hacohen & Weinshall, "On the power of curriculum learning in deep networks", ICML 2019

Universität Bremen

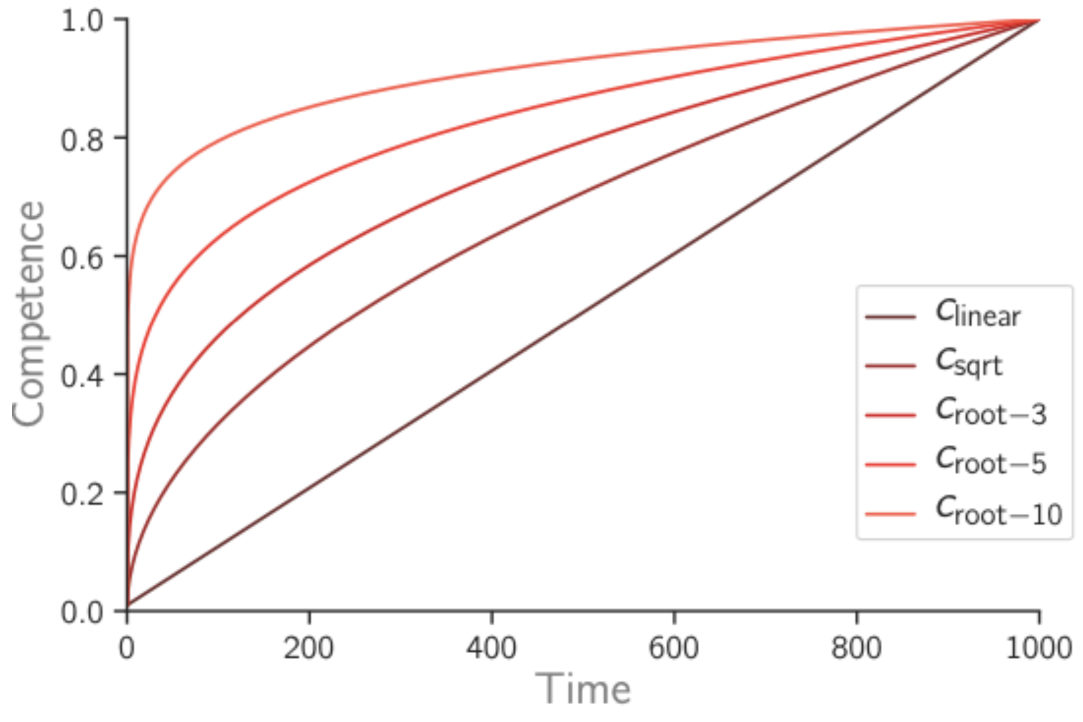
Pacing is often heuristic

Pacing functions are often heuristic

In pre-defined or “data-level” curriculum learning, they often follow a notion we think is intuitive

For example:

- “Build a strong foundation to learn the hard examples later (red)”
- “Easy examples can be learned quickly”



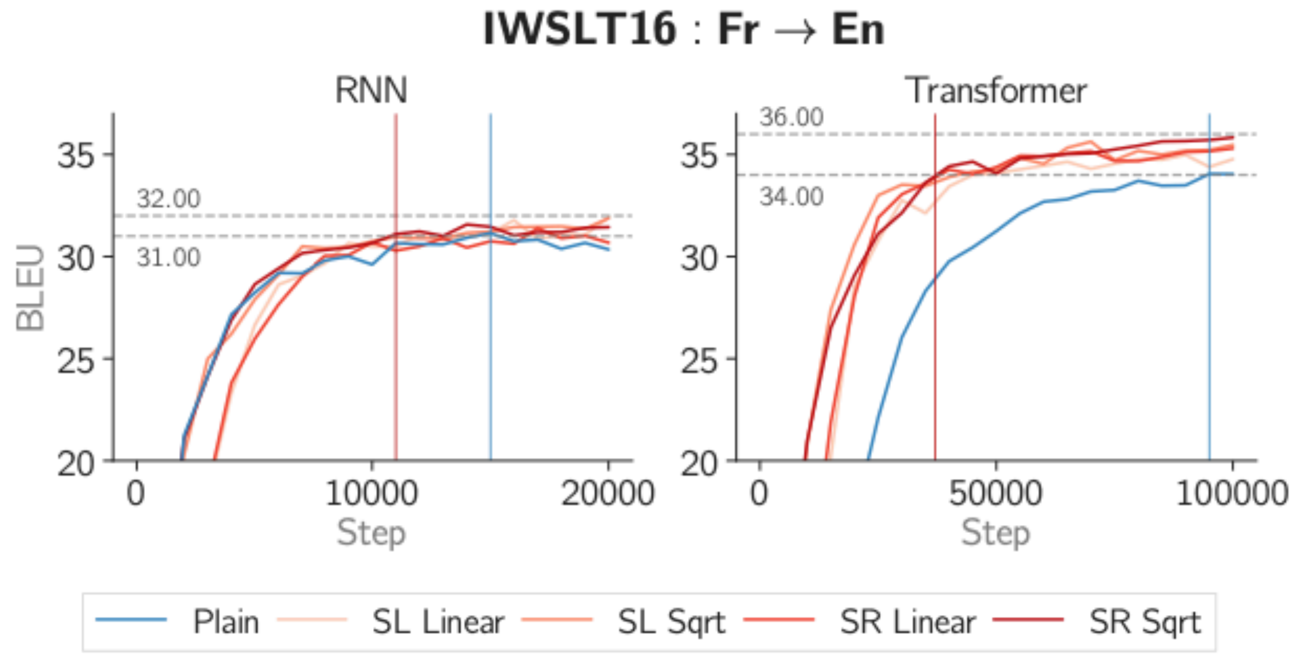
Platanios et al, “Competence based curriculum learning for neural machine translation”, NAACL-HLT 2019

Universität Bremen

Pacing is model and task dependent

The same pacing & difficulty may work for one model, but not another

For instance, in the French to English language translation examples different pacing makes no difference to "plain", but in transformers all of them work well



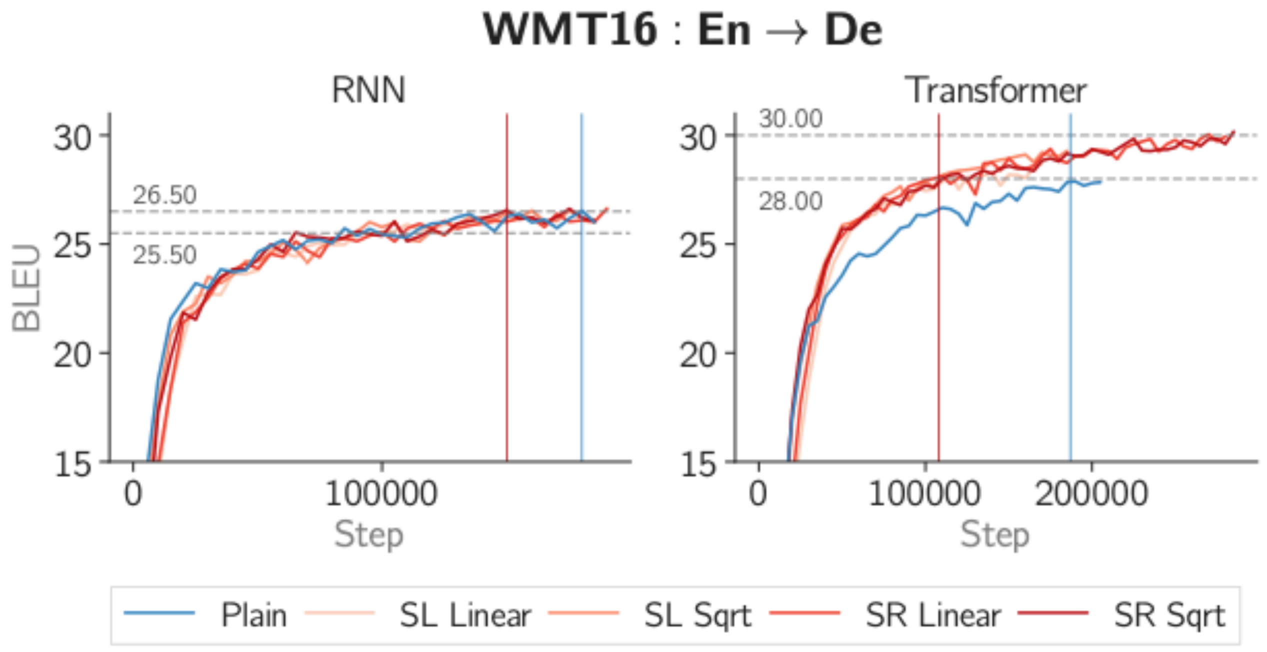
Platanios et al, "Competence based curriculum learning for neural machine translation", NAACL-HLT 2019

Pacing is model and task dependent

The same pacing & difficulty may work for one model, but not another

For instance, in the French to English language translation examples different pacing makes no difference to “plain”, but in transformers all of them work well

But for English to German there seems to much less gain

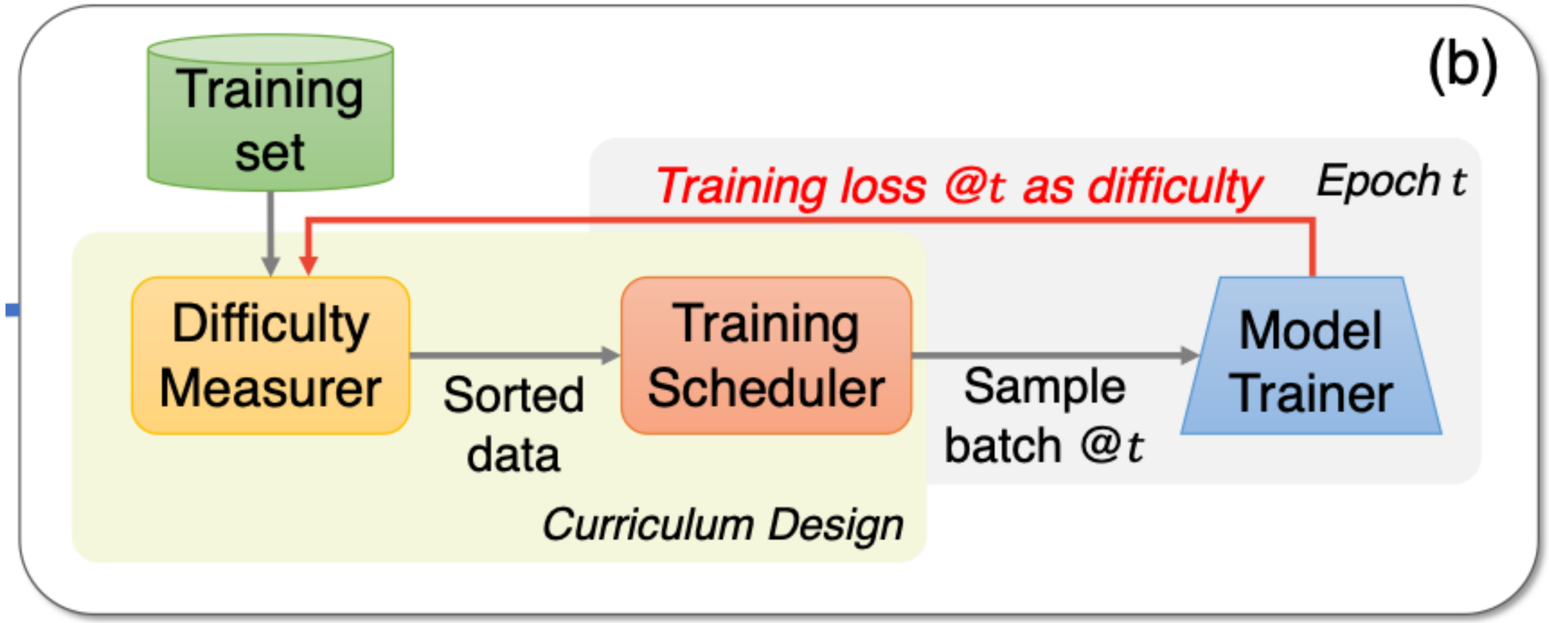


Platanios et al, “Competence based curriculum learning for neural machine translation”, NAACL-HLT 2019

Question time

Could we determine the pacing dynamically?

We refer to this as a *self-paced* curriculum



Wang et al, “A Survey on Curriculum Learning”, TPAMI 2021

Self-paced curriculum

Self-paced curriculum learning jointly learns the model parameters w and a latent weight variable $\mathbf{v} = [v_1, \ldots, v_n]^T$ by minimizing:

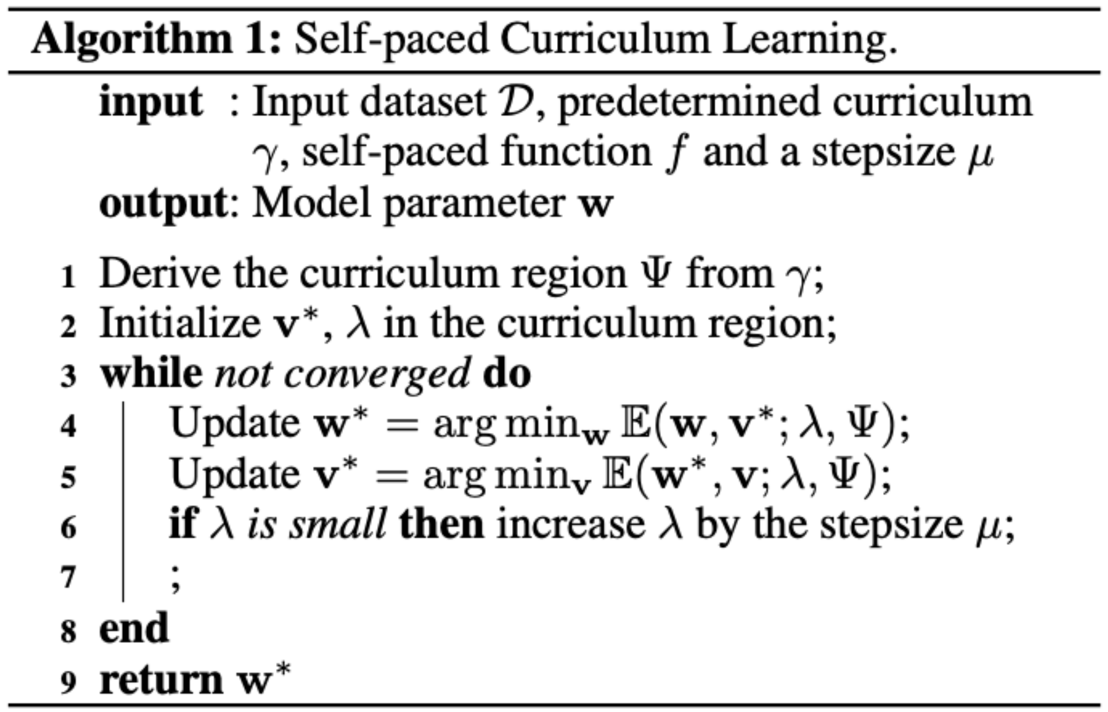
$$\mathtt{min}_{w,v}\mathbb{E}(\mathbf{w}, \mathbf{v}; \lambda) = \sum_{i=1}^{n} v_i \mathscr{L}(y_i, f(\mathbf{x}_i, \mathbf{w})) - \lambda \sum_{i=1}^{n} v_i$$

Self-paced curriculum

Self-paced curriculum learning jointly learns the model parameters w and a latent weight variable $\mathbf{v} = [v_1, \ldots, v_n]^T$ by minimizing:

$$\mathtt{min}_{w,v}\mathbb{E}(\mathbf{w}, \mathbf{v}; \lambda) = \sum_{i=1}^{n} v_i \mathcal{L}(y_i, f(\mathbf{x}_i, \mathbf{w})) - \lambda \sum_{i=1}^{n} v_i$$

In practice, we could threshold by "model age" or some other function:
$v_i = 1 \texttt{ if } \mathcal{L}(y_i, f(x_i, w)) < \lambda \;\&\; 0 \texttt{ otherwise}$



Algorithm 1: Self-paced Curriculum Learning.

```
input  : Input dataset $\mathcal{D}$, predetermined curriculum
         $\gamma$, self-paced function $f$ and a stepsize $\mu$
output: Model parameter $\mathbf{w}$

1 Derive the curriculum region $\Psi$ from $\gamma$;
2 Initialize $\mathbf{v}^*$, $\lambda$ in the curriculum region;
3 while not converged do
4     Update $\mathbf{w}^* = \arg\min_{\mathbf{w}} \mathbb{E}(\mathbf{w}, \mathbf{v}^*; \lambda, \Psi)$;
5     Update $\mathbf{v}^* = \arg\min_{\mathbf{v}} \mathbb{E}(\mathbf{w}^*, \mathbf{v}; \lambda, \Psi)$;
6     if $\lambda$ is small then increase $\lambda$ by the stepsize $\mu$;
7     ;
8 end
9 return $\mathbf{w}^*$
```

Jiang et al, "Self-Paced Curriculum Learning", AAAI 2015

Self-paced curriculum

We are going to again face many choices: this time how to use the loss.

We could make a “hard” choice to yield binary weights (Kumar et al, 2010):

$$f(\mathbf{v}; \lambda) = -\lambda ||v||_1 = -\lambda \sum_{i=1}^{n} v_i$$

We could discriminative linearly with respect to loss:

$$f(\mathbf{v}; \lambda) = \frac{1}{2}\lambda \sum_{i=1}^{n} (v_i^2 - 2v_i), \quad \texttt{with } \lambda > 0$$

Or lots of more complicated hybrids ….

Jiang et al, “Self-Paced Curriculum Learning”, AAAI 2015

Universität Bremen

Question time

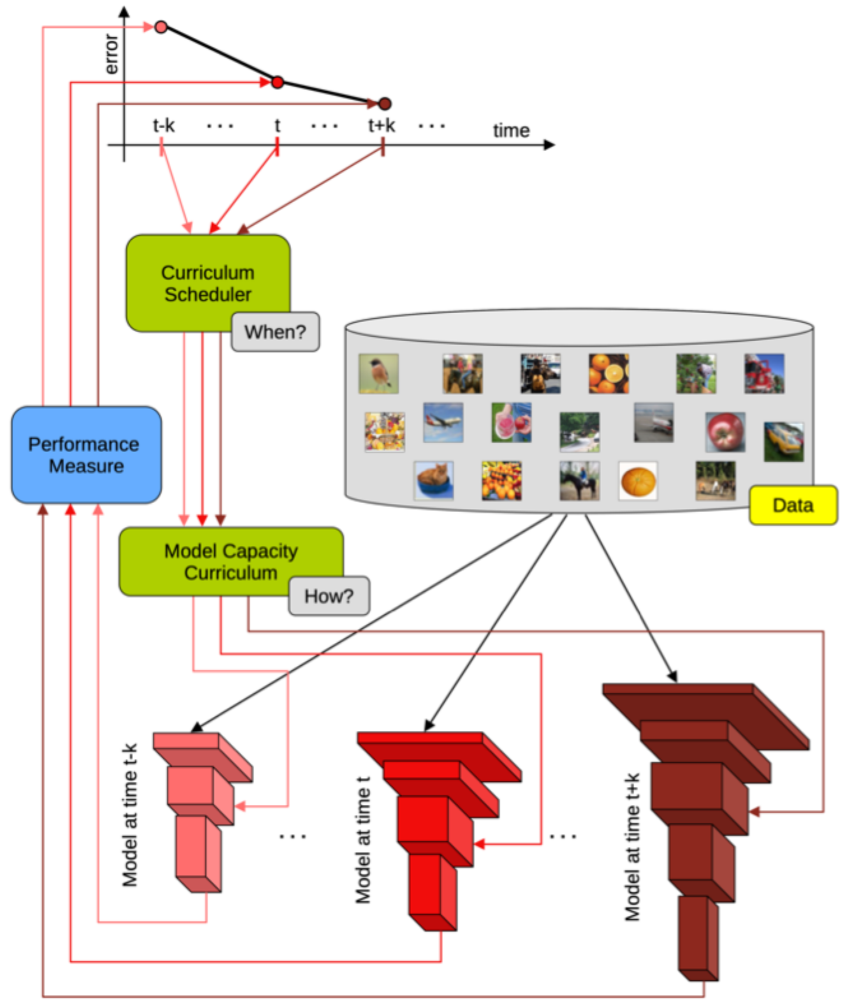
Could we also define the difficulty dynamically?

“Model-level” curriculum

We have seen that our intuition on difficulty and what is hard to train and learned in practice in ML is challenging to match!

Is it possible to instead rely on other means to define difficulty & pacing?

What if we relied on models to adaptively define and measure both difficulty and pacing functions?



Soviany et al, “Curriculum Learning: A Survey”, IJCV, 2022

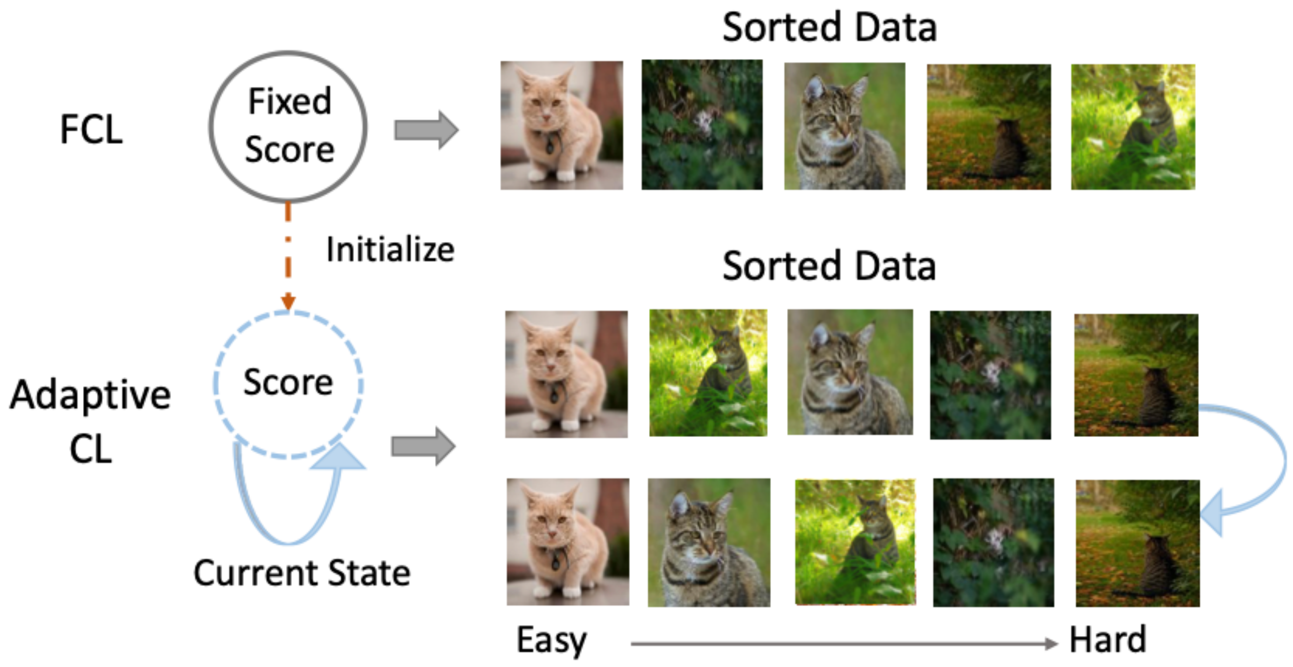
Universität Bremen

Adaptive curriculum

The key differences to pre-defined curriculum learning are:

1. We gauge the difficulty of a sample based on our model's capabilities. The most intuitive would be to take the loss/prediction confidence
2. The difficulty of each data point can get adjusted in every step

Attention: terms are overloaded, often also simply called "self-pacing"



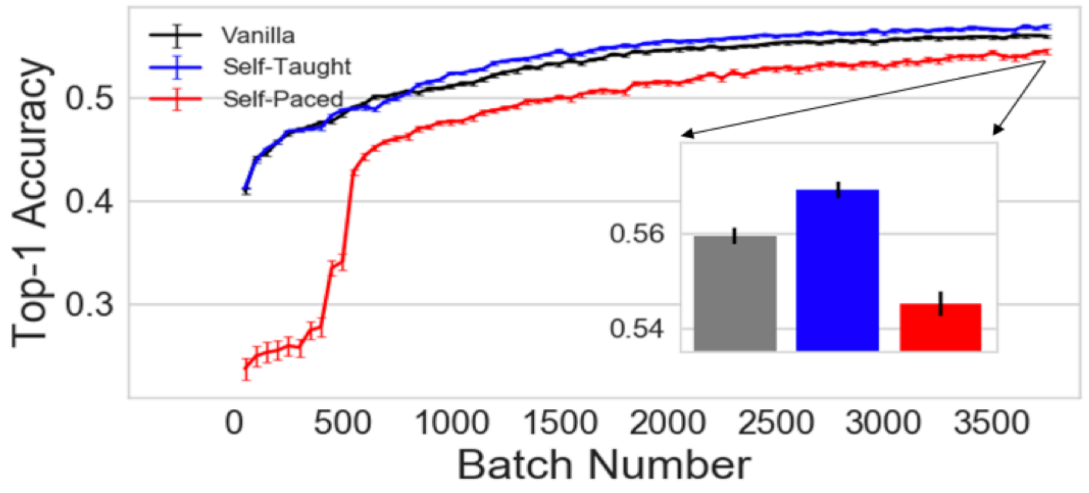
Kong et al, "Adaptive Curriculum Learning", ICCV 2022

Universität Bremen

Self-pacing is hard

Example: image classification

Self-pacing can start out very poorly and get stuck way below the desired performance (red curve).



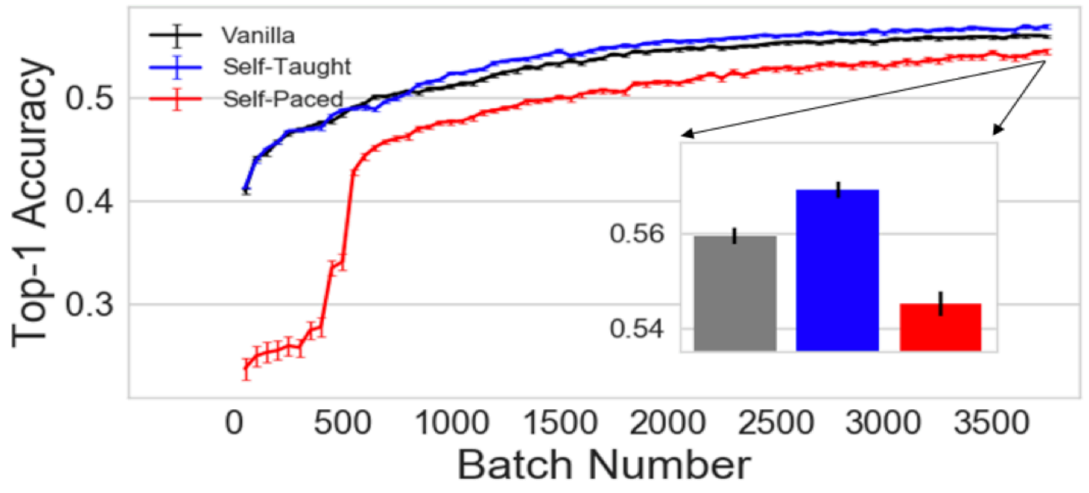
Hacohen & Weinshall, “On the power of curriculum learning in deep networks”, ICML 2019

Self-pacing is hard

Example: image classification

Self-pacing can start out very poorly and get stuck way below the desired performance (red curve).

This can have several reasons, false initial predictions, local minima, and as we will see later in the course: model loss & predictions being poor proxies for data acquisition



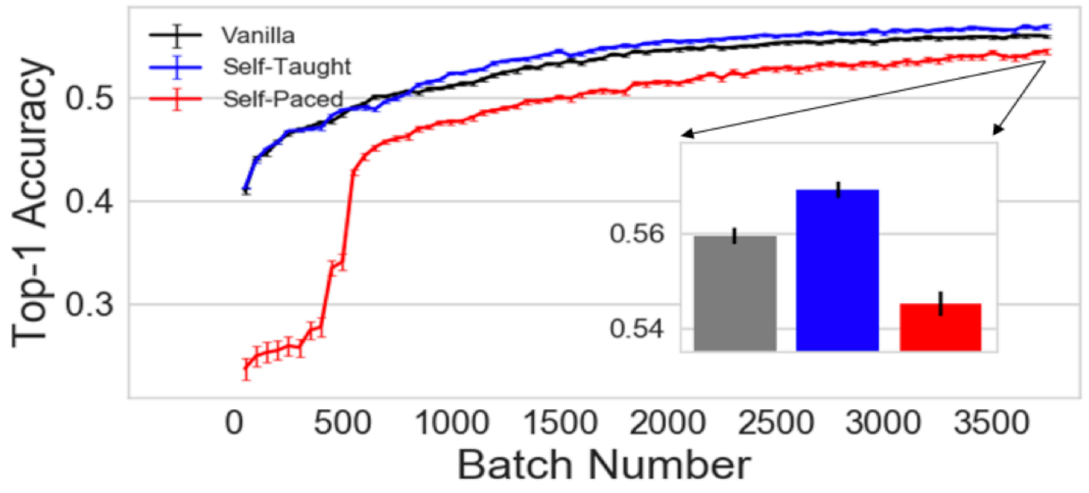
Hacohen & Weinshall, “On the power of curriculum learning in deep networks”, ICML 2019

Self-taught curriculum

Example: image classification

An alternative is to use the converged model itself as a “teacher”

That is: train a model fully, measure each instance according to the final model, assign difficulty score and start over with the curriculum -> repeat (Related to the ideas in *boosting*)

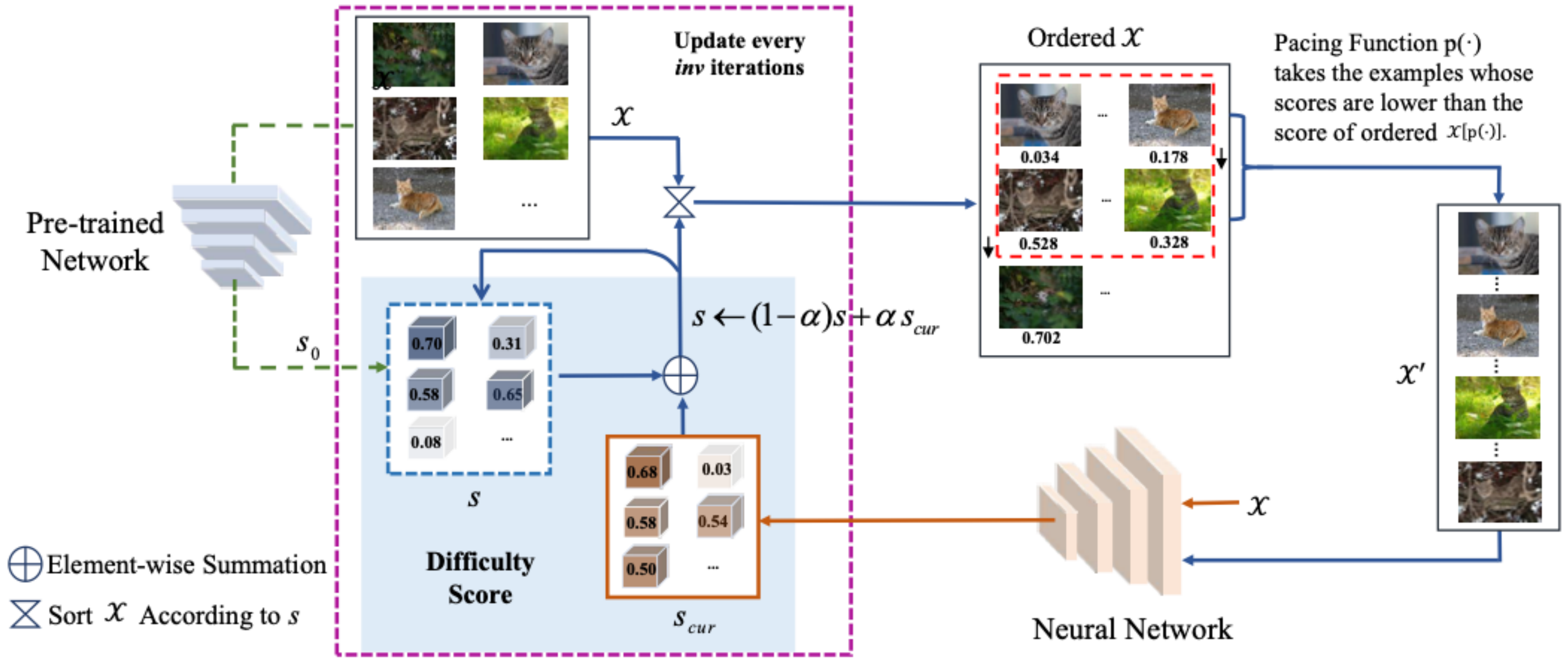


Hacohen & Weinshall, “On the power of curriculum learning in deep networks”, ICML 2019

Universität Bremen

OWLML

Teacher + self-pacing can be combined to avoid initial self-pacing pitfalls



Kong et al, “Adaptive Curriculum Learning”, ICCV 2022

Question time

Why does teacher-based curriculum learning work well? (A question that is challenging to answer)

Universität
Bremen

Hypothesis: do intrinsic curricula exist?

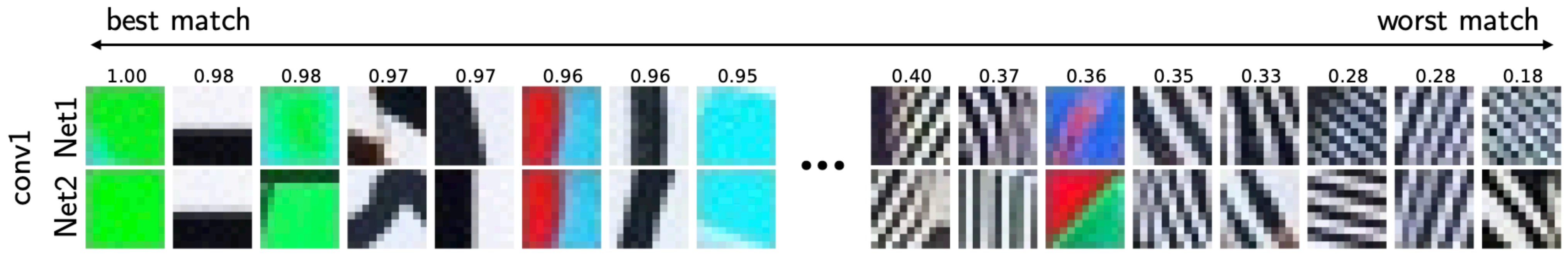
An experiment: train *multiple models* & check *representation similarity*

Why is this interesting? Recall that we use randomly shuffled mini-batches, randomly initialized models, and stochastic gradient descent

Hypothesis: do intrinsic curricula exist?

An experiment: train *multiple models* & check *representation similarity*

Why is this interesting? Recall that we use randomly shuffled mini-batches, randomly initialized models, and stochastic gradient descent



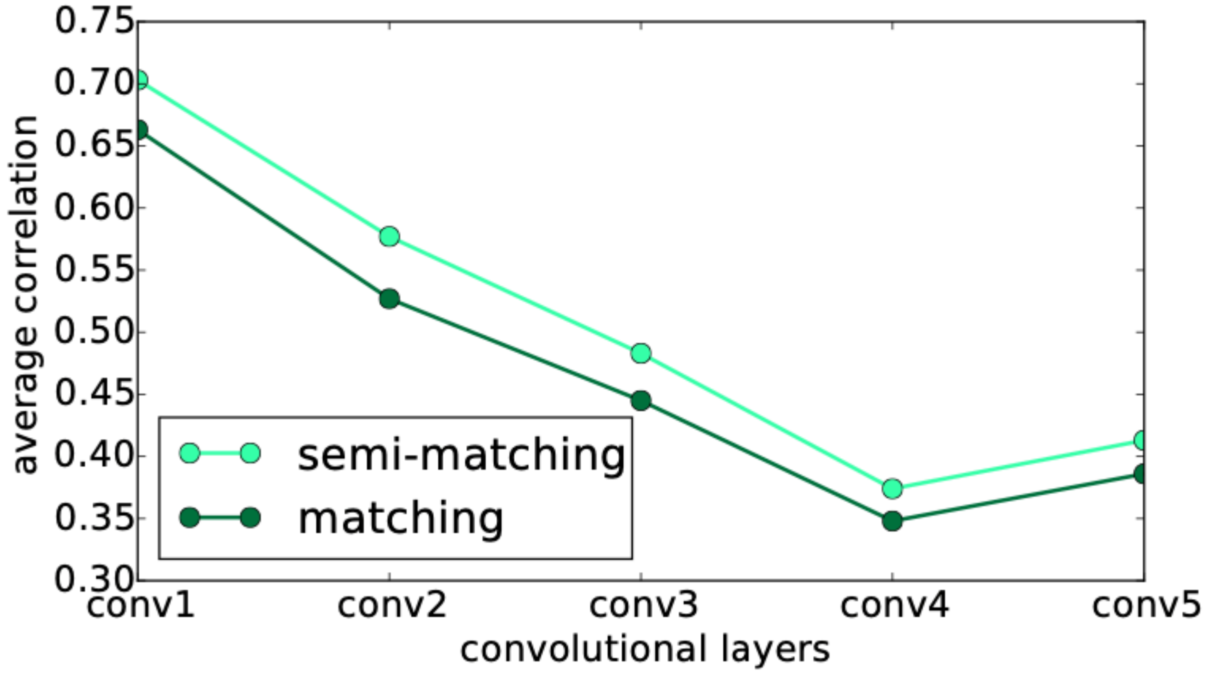
Li & Yosinski et al, “Convergent learning: do different neural networks learn the same representations”, ICLR 2016

Universität Bremen

Representation similarity

We can try a bi-partite matching of the representations in each neural network layer of the different models and see if they are similar
(Note, why do we need to match?)

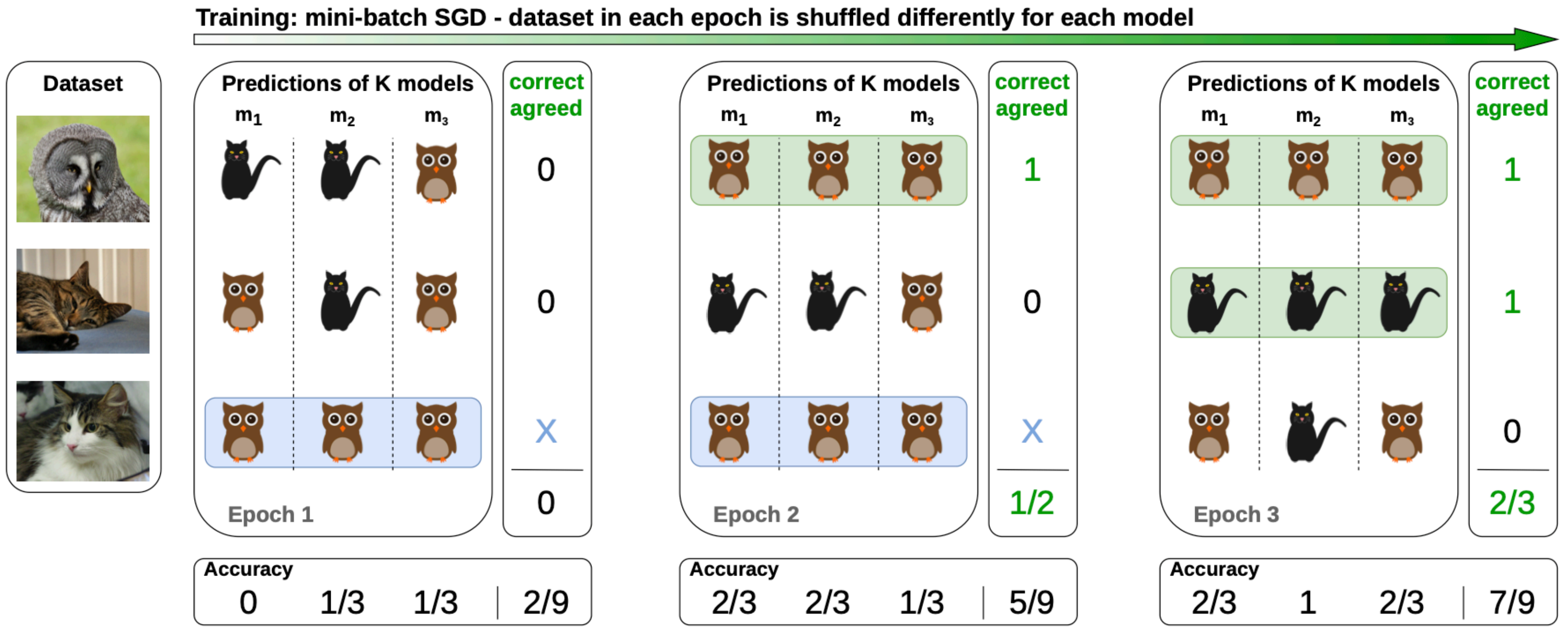
We can observe strong correlations, especially in early, “generic”, features
(Note, these will be useful for “transfer", which we will transition to soon)



Li & Yosinski et al, “Convergent learning: do different neural networks learn the same representations”, ICLR 2016

Do similar representations imply similar outcome

An experiment repeated: train *multiple models* & check *prediction similarity*



Pliushch et al, “When Deep Classifiers Agree: Analyzing correlations between learning order and image statistics”, ECCV 2022

Universität Bremen

Quantifying neural network agreement

We can define true positive (and conversely negative) agreement as an exact match of correctly predicted instances in an epoch t by all classifiers, normalized by the sum of instances classified correctly by any classifier at that point in time:

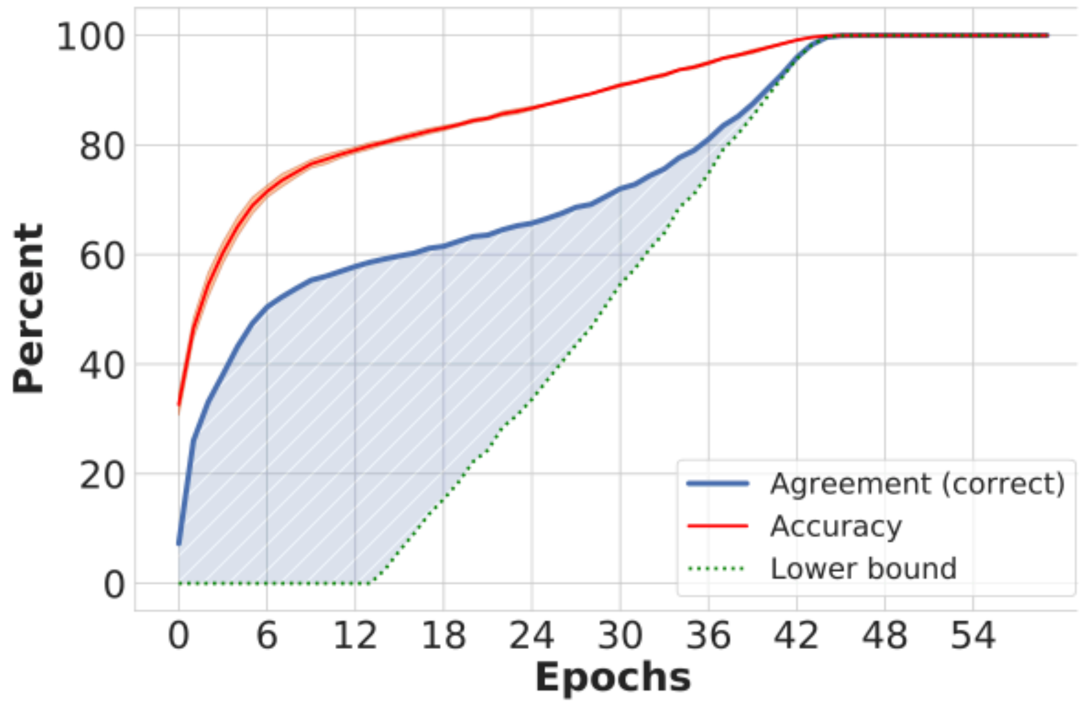
$$TPA^{(t)}(x, y) = \frac{\sum_{n \in N} \prod_{k \in K} \mathbb{1}_{f^{(t)}{}_k(x_n) = y_n}}{\sum_{n \in N} \max_{k \in K} \mathbb{1}_{f^{(t)}{}_k(x_n) = y_n}}$$

Intuitively, models do not have to agree. For instance, 5 models that each have 80% accuracy, could false predict a different 20% subset.

Pliushch et al, “When Deep Classifiers Agree: Analyzing correlations between learning order and image statistics”, ECCV 2022

Neural network agreement

- Several neural networks "agree" in terms of learning to classify the same instances correctly at a similar time

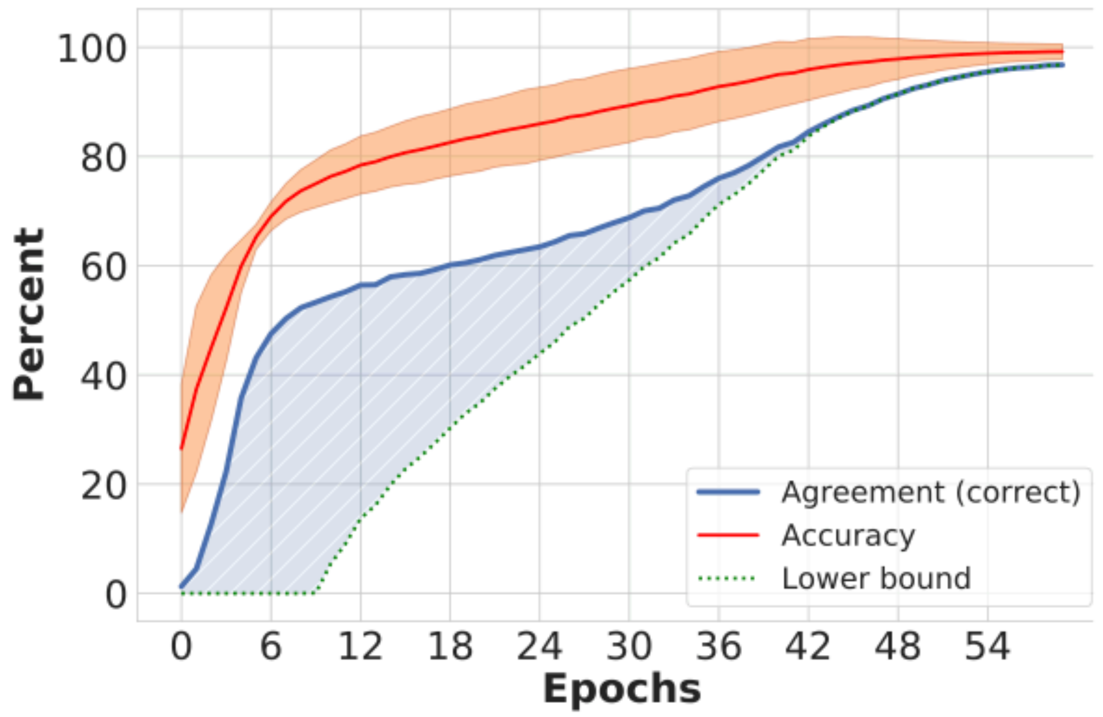


Pliushch et al, "When Deep Classifiers Agree: Analyzing correlations between learning order and image statistics", ECCV 2022

Universität Bremen

Neural network agreement

- Several neural networks “agree” in terms of learning to classify the same instances correctly at a similar time
- Several neural networks of different architecture agree

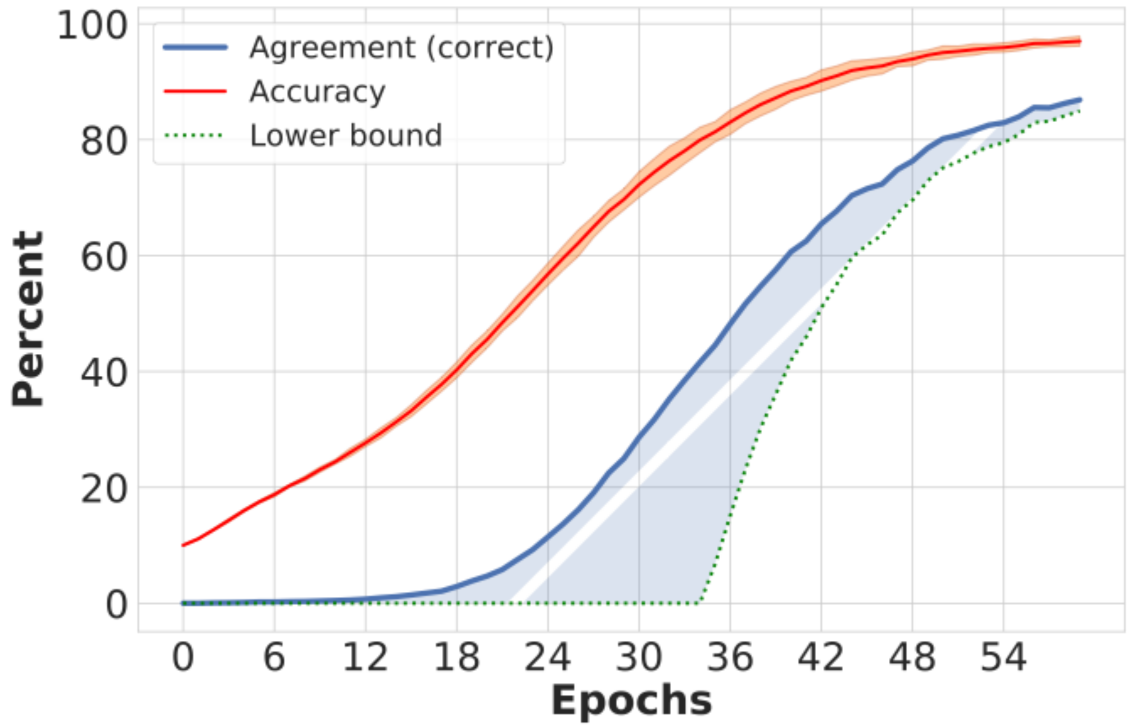


Pliushch et al, “When Deep Classifiers Agree: Analyzing correlations between learning order and image statistics”, ECCV 2022

Universität Bremen

Neural network agreement

- Several neural networks “agree” in terms of learning to classify the same instances correctly at a similar time
- Several neural networks of different architecture agree
- Several neural networks agree on “correctly” classified examples when labels are randomized -> they overfit in similar ways



Pliushch et al, “When Deep Classifiers Agree: Analyzing correlations between learning order and image statistics”, ECCV 2022

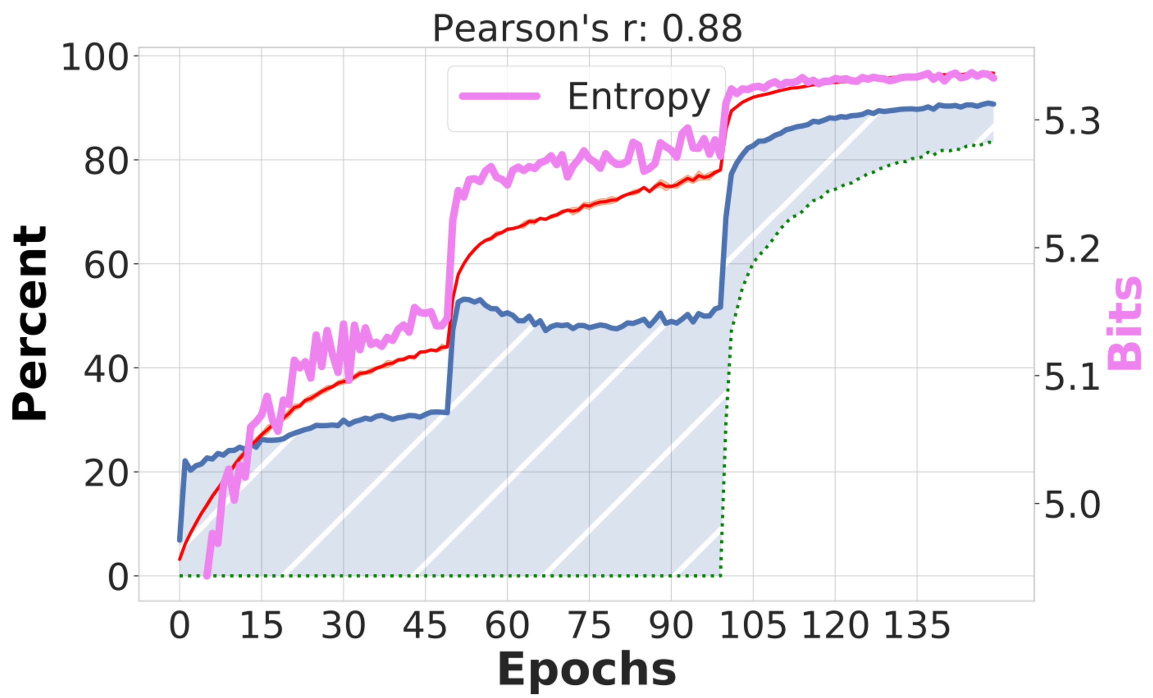
Question time

If agreement is not due to the architecture, nor a factor of the labels, what drives it?

Universität Bremen

Back to square one?

Turns out, that there is a very strong correlation between agreement and data statistics - the point where we originally started



Pliushch et al, “When Deep Classifiers Agree: Analyzing correlations between learning order and image statistics”, ECCV 2022

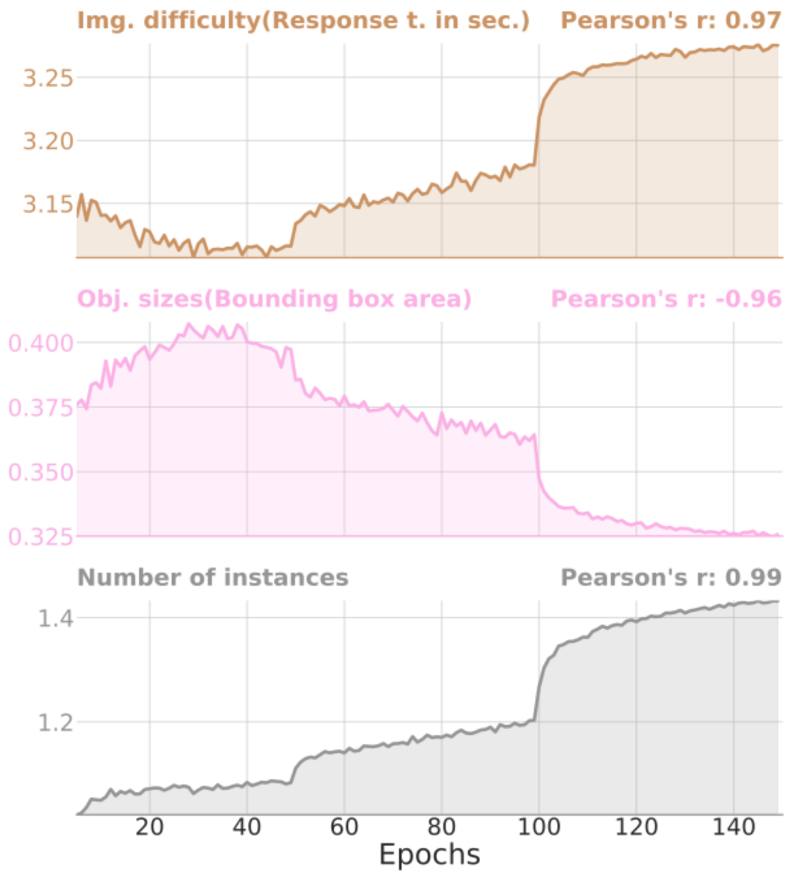
Universität Bremen

Back to square one?

Turns out, that there is a very strong correlation between agreement and data statistics - the point where we originally started

However, models seem to be making use of this intrinsically, and correlations can be flipped depending on data set

Another factor that seems to make curriculum learning “hard to get to work”



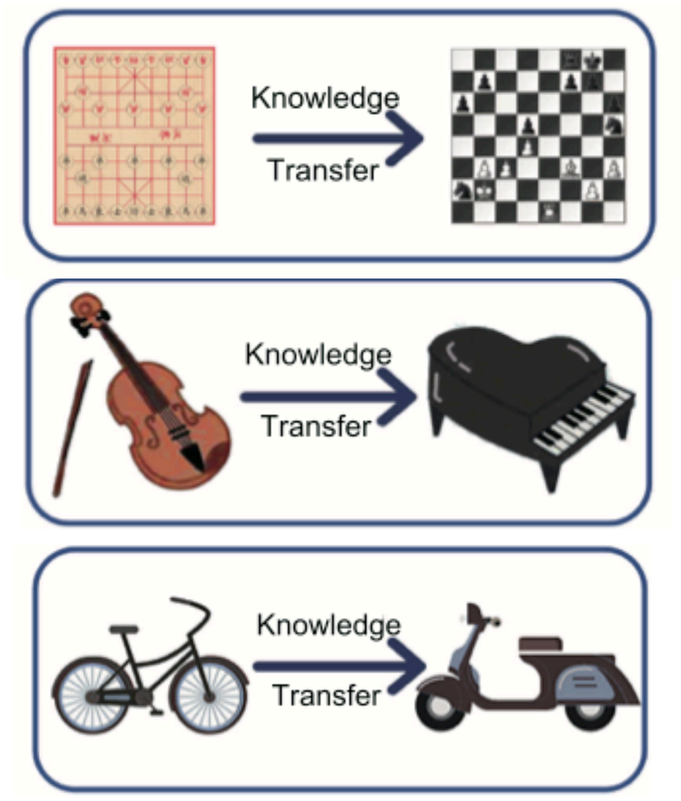
Pliushch et al, “When Deep Classifiers Agree: Analyzing correlations between learning order and image statistics”, ECCV 2022

Universität Bremen

Question time

We have seen a model can serve as a teacher. Does that mean we can transfer its knowledge?

Universität Bremen



“A Comprehensive Survey on Transfer Learning”, Zhuang et al, Proceedings of IEEE, 2020

Part 1 cont. - Learning in the Present

Transfer Learning & Domain Adaptation

Lifelong Machine Learning
Summer 2025

Prof. Dr. Martin Mundt

Early Definition

Thrun 1996:

“*The system has performed N tasks. When faced with the (N+1)th task, it uses the knowledge gained from the N tasks to help the (N+1)th task.*”

“Is Learning The n-th Thing Any Easier Than Learning the First?” (NeurIPS 1996) & “Explanation based Neural Network Learning A Lifelong Learning Approach”, Springer US, 1996

Early Definition

Thrun 1996:

"*The system has <u>performed N</u> tasks. When faced with the <u>(N+1)th</u> task, it uses the <u>knowledge</u> gained from the N tasks to <u>help</u> the (<u>N+1)th task</u>.*"

"Is Learning The n-th Thing Any Easier Than Learning the First?" (NeurIPS 1996) & "Explanation based Neural Network Learning A Lifelong Learning Approach", Springer US, 1996

Early Definition

Thrun 1996:

“*The system has <u>performed N</u> tasks. When faced with the <u>(N+1)th</u> task, it uses the <u>knowledge</u> gained from the N tasks to <u>help</u> the (<u>N+1)th task</u>.*”

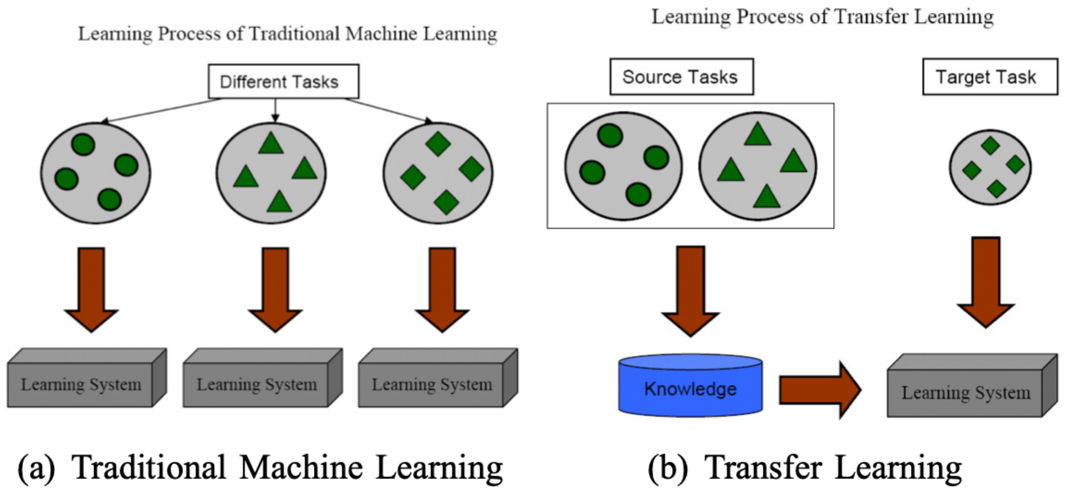
“Is Learning The n-th Thing Any Easier Than Learning the First?” (NeurIPS 1996) & “Explanation based Neural Network Learning A Lifelong Learning Approach”, Springer US, 1996

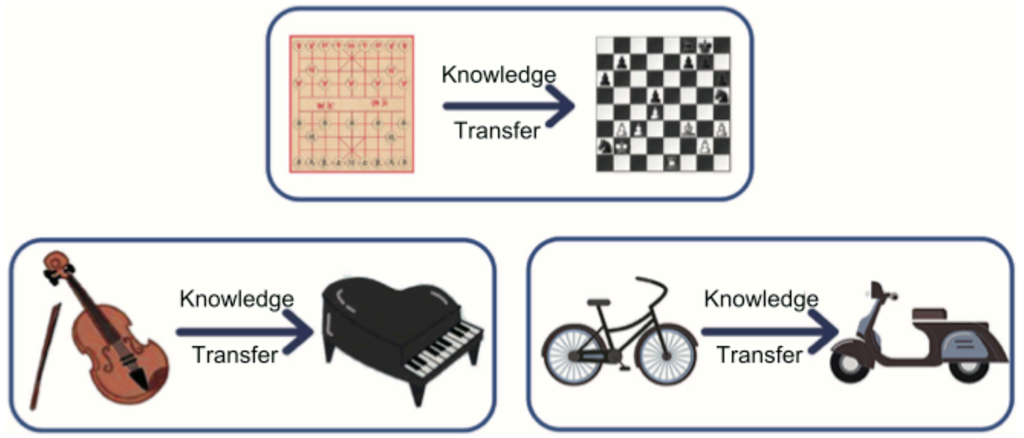
- What is a *task*?
- What is *knowledge*?
- What exactly is *help*?

Universität Bremen

Early Definition

- Different data
- Different objectives





A Survey on Transfer Learning", Pan and Yang, IEEE Transactions on Knowledge & Data Engineering, 2010

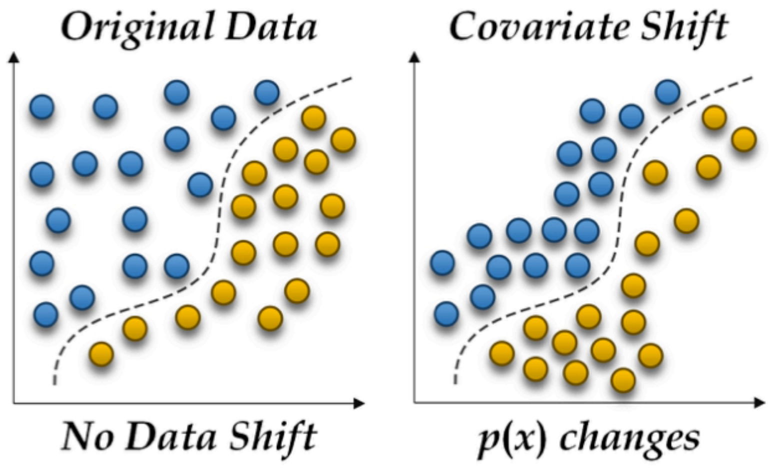
"A Comprehensive Survey on Transfer Learning", Zhuang et al, Proceedings of IEEE, 2020

Question time

In what ways can data be different?
What kind of shifts can you think of?

Dataset shift

Covariate shift: distribution of input features (covariates) changes (while the labeling function stays the same)



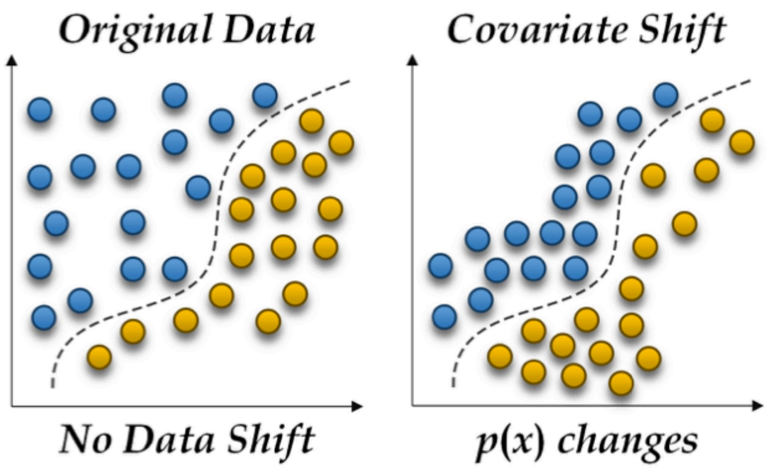
(a) Original data (b) Covariate shift

Figure from “Understanding Dataset Shift and Potential Remedies”, Vector Institute Tech Report, 2021
Also: “Dataset Shift in Machine Learning”, MIT Press 2009

Dataset shift

Covariate shift: distribution of input features (covariates) changes (while the labeling function stays the same)

Specifically: marginal distribution p(x) changes, while conditional probability of an output p(y|x) remains the same

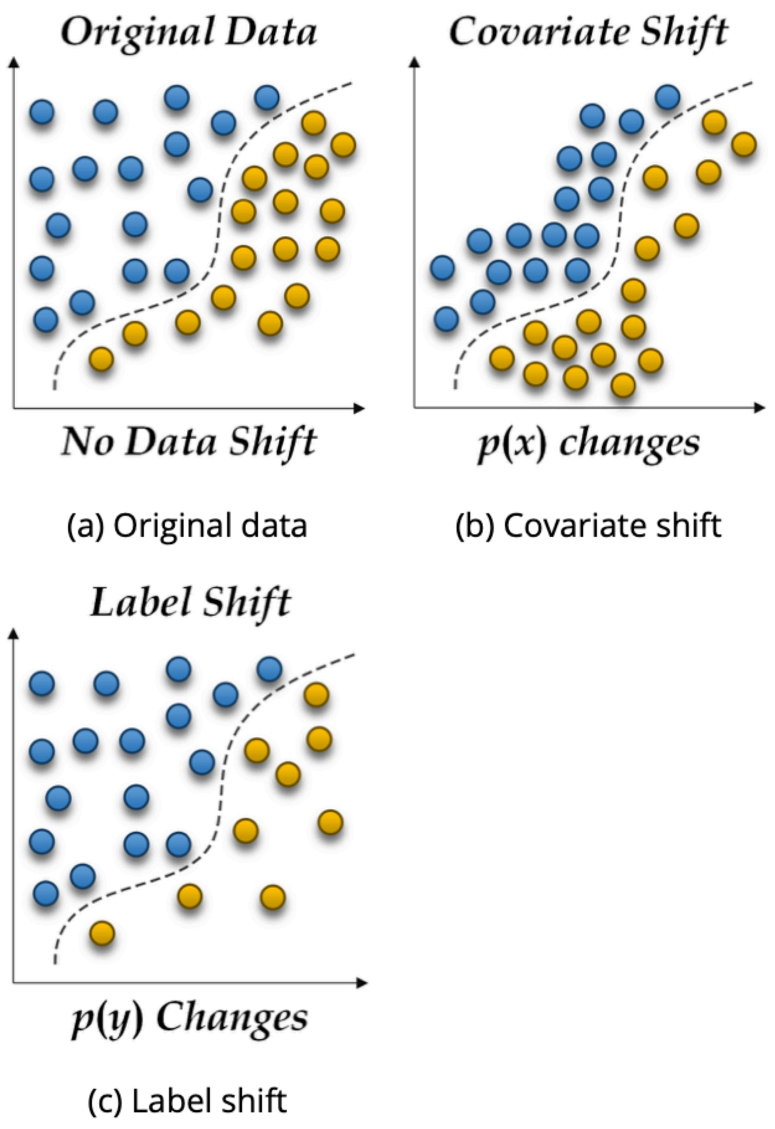


(a) Original data (b) Covariate shift

Figure from "Understanding Dataset Shift and Potential Remedies", Vector Institute Tech Report, 2021
Also: "Dataset Shift in Machine Learning", MIT Press 2009

Dataset shift

Label shift: class proportions vary between source/target (train/test), but feature distributions of each class do not



(a) Original data (b) Covariate shift (c) Label shift

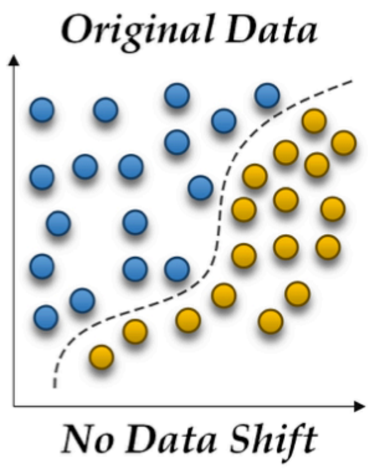
Figure from “Understanding Dataset Shift and Potential Remedies”, Vector Institute Tech Report, 2021
Also: “Dataset Shift in Machine Learning”, MIT Press 2009

Dataset shift

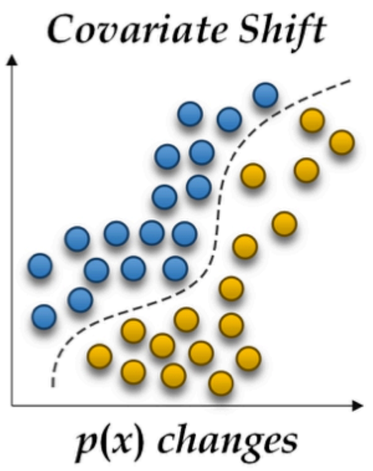
Label shift: class proportions vary between source/target (train/test), but feature distributions of each class do not

Specifically: the label distribution p(y) changes, but conditional p(x|y) remains the same. (Prior probability shift)

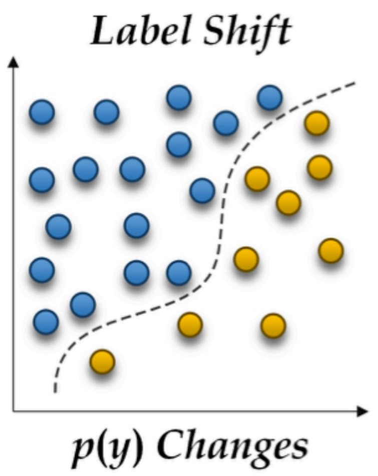
Figure from “Understanding Dataset Shift and Potential Remedies”, Vector Institute Tech Report, 2021
Also: “Dataset Shift in Machine Learning”, MIT Press 2009



(a) Original data



(b) Covariate shift



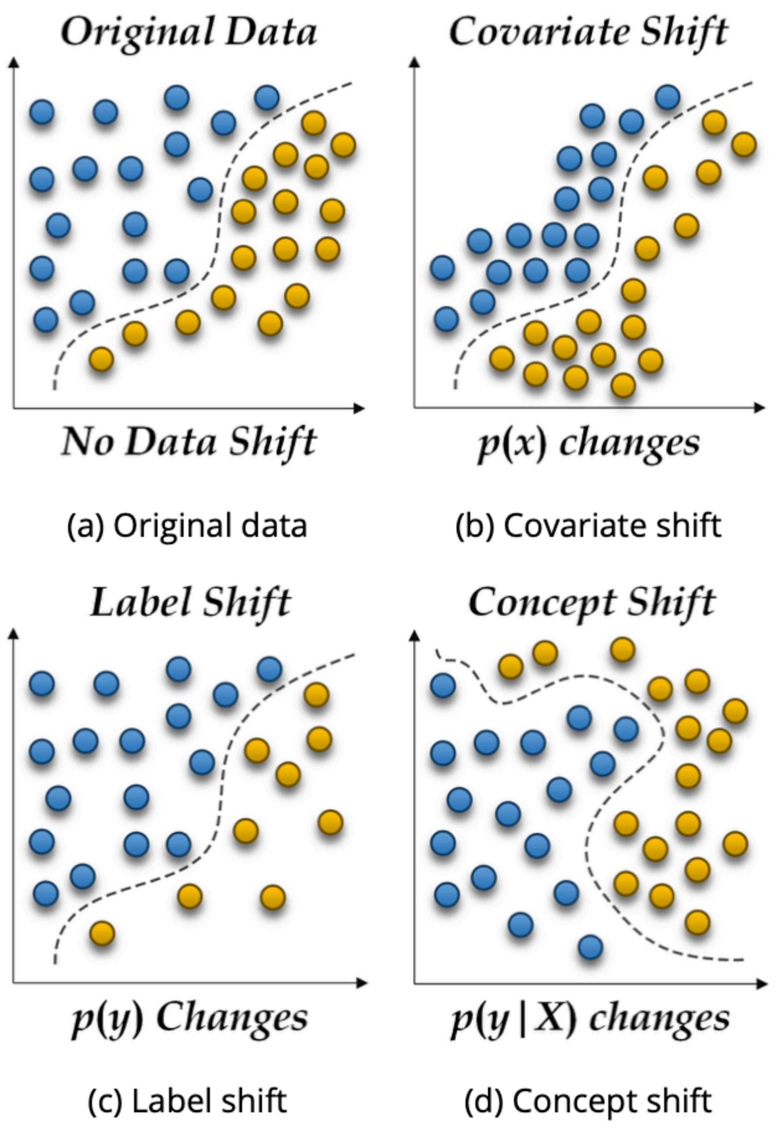
(c) Label shift

Universität Bremen

Dataset shift

Concept shift: distribution of input features (covariates) changes (while the labeling function stays the same)

Figure from "Understanding Dataset Shift and Potential Remedies", Vector Institute Tech Report, 2021
Also: "Dataset Shift in Machine Learning", MIT Press 2009

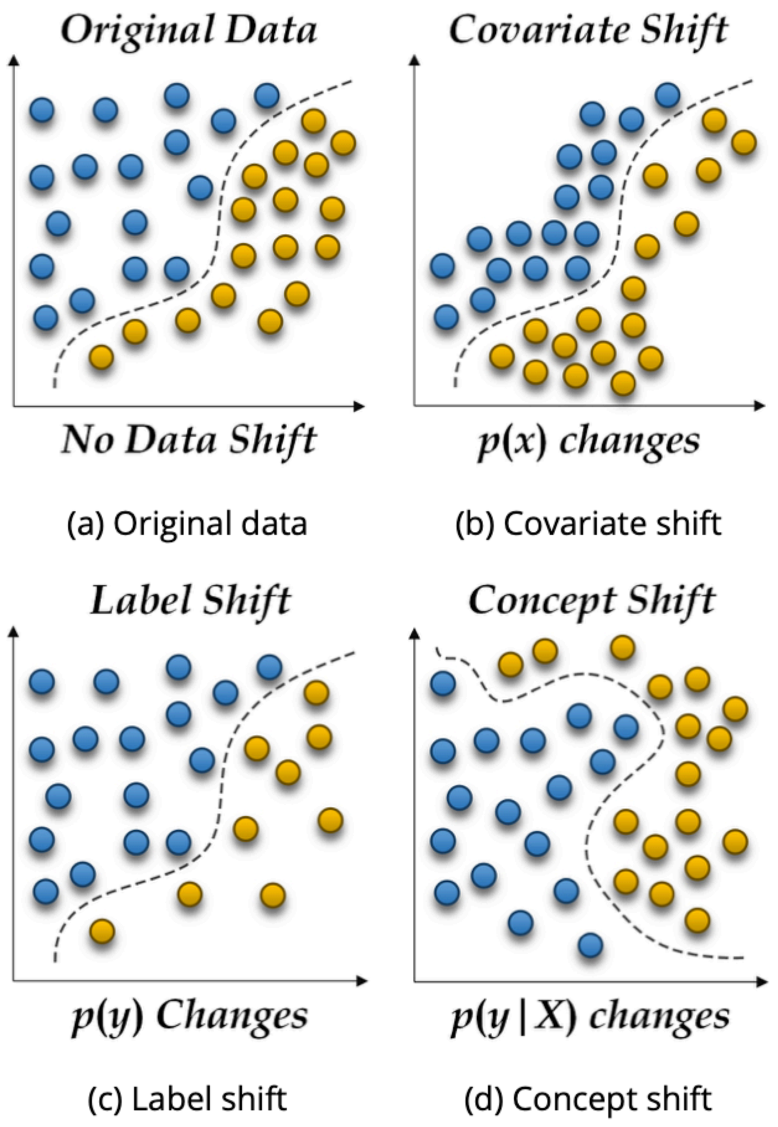


Dataset shift

Concept shift: distribution of input features (covariates) changes (while the labeling function stays the same)

Specifically: conditional p(y|x) changes, p(x), the feature distribution, remains the same. (Posterior probability shift)

Figure from “Understanding Dataset Shift and Potential Remedies”, Vector Institute Tech Report, 2021
Also: “Dataset Shift in Machine Learning”, MIT Press 2009



Universität
Bremen

Formalizing transfer learning: definition

Definition - Transfer Learning - Pan & Yang 2009:
"*Given a source domain* D_S *and learning task* $\mathcal{T}_S$, *a target domain* D_T *and learning task* $\mathcal{T}_T$, *transfer learning aims to help improve the learning of the target predictive function* $f_T(\,.\,)$ *in* D_T *using the knowledge in* D_S *and* $\mathcal{T}_S$, *where* $D_S \neq D_T$ *or* $\mathcal{T}_s \neq \mathcal{T}_T$."

Formalizing transfer learning: definition

Definition - Transfer Learning - Pan & Yang 2009:

"Given a source domain D_S *and learning task* $\mathcal{T}_S$, *a target domain* D_T *and learning task* $\mathcal{T}_T$, *transfer learning aims to help improve the learning of the target predictive function* $f_T(\,.\,)$ *in* D_T *using the knowledge in* D_S *and* $\mathcal{T}_S$, *where* $D_S \neq D_T$ or $\mathcal{T}_s \neq \mathcal{T}_T$.*"*

- Domain D
- Task $\mathcal{T}$
- Source S
- Target T

Universität Bremen

Formalizing transfer learning: definition

Definition - Domain & Task - Pan & Yang 2009:
"*Given a specific domain,* $D = \{\mathscr{X}, p(x)\}$*, a task consists of two components: a label space Y and an objective predictive function* $f()$ *(denoted by* $T = \{Y, f()\}$*, which is not observed but can be learned from the training data, which consist of pairs* $\{x^{(n)}, y^{(n)}\}$*, where* $x^{(n)} \in X$ and $y^{(n)} \in Y$."

Formalizing transfer learning: definition

Definition - Domain & Task - Pan & Yang 2009:
"Given a specific domain, $\mathcal{D} = \{\mathcal{X}, p(x)\}$, a task consists of two components: a label space Y and an objective predictive function $f()$ (denoted by $T = \{Y, f()\}$, which is not observed but can be learned from the training data, which consist of pairs $\{x^{(n)}, y^{(n)}\}$, where $x^{(n)} \in X$ and $y^{(n)} \in Y$."

- Domain D: pair of data distribution $p(x)$ and feature space $\mathcal{X}$
- Task $\mathcal{T}$: find a function f() (to map to labels for supervision)
- Where generally $\mathcal{X}_S \neq \mathcal{X}_T$ or $p_S(x) \neq p_T(x)$

Formalizing transfer learning: definition

Definition - Transductive Transfer Learning - Pan & Yang 2009:
"*Given a source domain* D_S *and learning task* $\mathcal{T}_S$, *a target domain* D_T *and learning task* $\mathcal{T}_T$, *transductive transfer learning aims to help improve the learning of the target predictive function* $f_T(\,.\,)$ *in* D_T *using the knowledge in* D_S *and* $\mathcal{T}_S$, *where* $D_S \neq D_T$ & $\mathcal{T}_s = \mathcal{T}_T$."

Formalizing transfer learning: definition

Definition - Transductive Transfer Learning - Pan & Yang 2009:
"*Given a source domain* D_S *and learning task* $\mathcal{T}_S$, *a target domain* D_T *and learning task* $\mathcal{T}_T$, *transductive transfer learning aims to help improve the learning of the target predictive function* $f_T(\,.\,)$ *in* D_T *using the knowledge in* D_S *and* $\mathcal{T}_S$, *where* $D_S \neq D_T$ & $\mathcal{T}_s = \mathcal{T}_T$."

- Feature spaces of source & target are different $\mathcal{X}_S \neq \mathcal{X}_T$
- Same feature spaces, but $p_S(x) \neq p_T(x)$
- Often called: **domain adaptation** or **sample selection bias**

Question time

Can you name a practical example where the domain is different in terms of either data distribution or feature space?

“Homogeneous” transfer

$D_S \neq D_T$ & $\mathcal{T}_s = \mathcal{T}_T$
and specifically $p_S(x) \neq p_T(x)$ & $\mathcal{X}_S = \mathcal{X}_T$

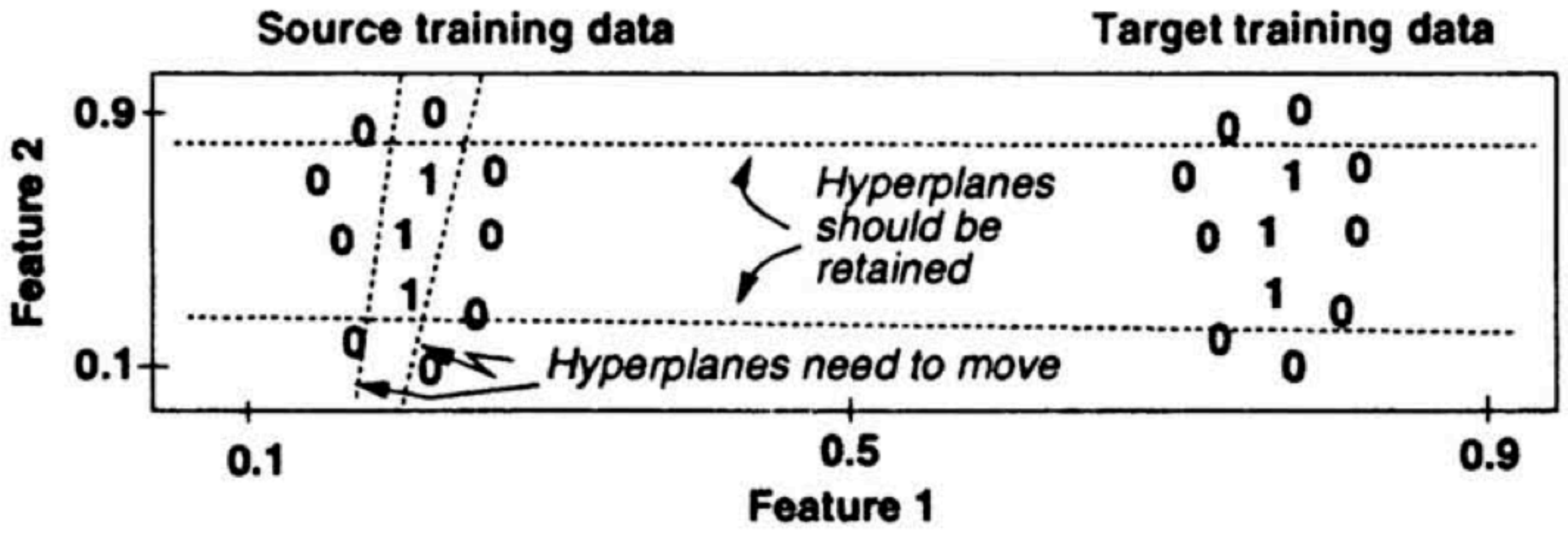
Example 1:
Classifying objects in natural images and e.g. drawings

Example 2:
Disease detection with different patients with diverse backgrounds

Universität Bremen

Transductive (homogeneous) intuition

Early approaches transfer by identifying the amount that a specific hyperplane helps to separate the data into different classes



“Discriminability-Based Transfer between Neural Networks”, L. Y. Pratt, NeurIPS 1992

“Heterogeneous” transfer

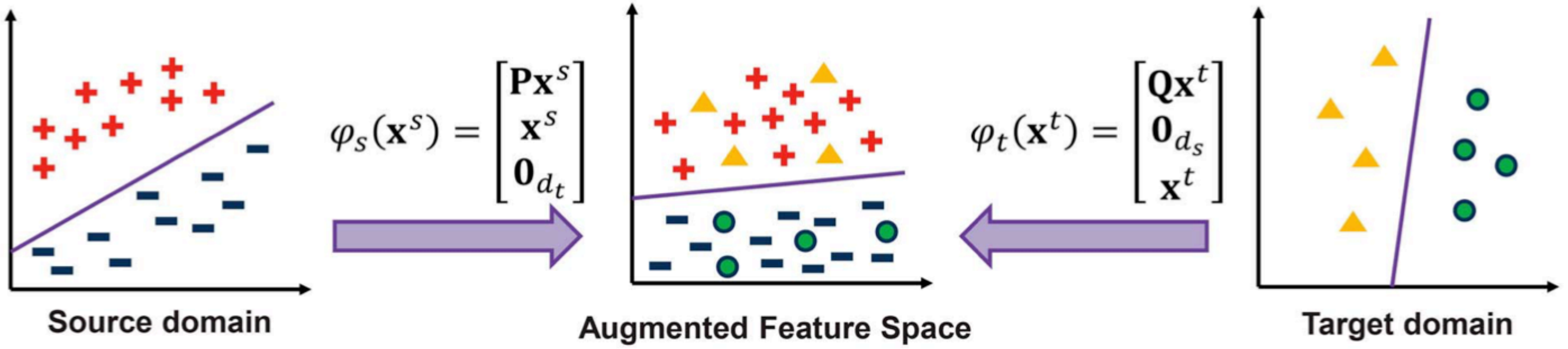
$D_S \neq D_T$ & $\mathscr{T}_s = \mathscr{T}_T$
and specifically $p_S(x) = p_T(x)$ and $\mathscr{X}_S \neq \mathscr{X}_T$

Example 1:
Classifying objects based on text versus images

Example 2:
Disease detection with the same patients, but different sensors

Transductive (heterogeneous) intuition

Domain adaptation through feature transformation of source and target features to an augmented latent feature space with projection matrices P and Q


Source domain
$\varphi_s(\mathbf{x}^s) = \begin{bmatrix} \mathbf{Px}^s \\ \mathbf{x}^s \\ \mathbf{0}_{d_t} \end{bmatrix}$
Augmented Feature Space
$\varphi_t(\mathbf{x}^t) = \begin{bmatrix} \mathbf{Qx}^t \\ \mathbf{0}_{d_s} \\ \mathbf{x}^t \end{bmatrix}$
Target domain

“Discriminability-Based Transfer between Neural Networks”, L. Y. Pratt, NeurIPS 1992

Formalizing transfer learning: definition

Definition - Inductive Transfer Learning - Pan & Yang 2009:
"*Given a source domain* D_S *and learning task* $\mathcal{T}_S$*, a target domain* D_T *and learning task* $\mathcal{T}_T$*, inductive transfer learning aims to help improve the learning of the target predictive function* $f_T(\,.\,)$ *in* D_T *using the knowledge in* D_S *and* $\mathcal{T}_S$*, where* $\mathcal{T}_s \neq \mathcal{T}_T$."

Formalizing transfer learning: definition

Definition - Inductive Transfer Learning - Pan & Yang 2009:
"*Given a source domain* D_S *and learning task* $\mathcal{T}_S$, *a target domain* D_T *and learning task* $\mathcal{T}_T$, *inductive transfer learning aims to help improve the learning of the target predictive function* $f_T(\,.\,)$ *in* D_T *using the knowledge in* D_S *and* $\mathcal{T}_S$, *where* $\mathcal{T}_s \neq \mathcal{T}_T$."

- We will learn that (a few) (labeled) data points are required to "induce" the target predictive function

Question time

What are the key questions in transfer learning?

(Some) central questions in transfer learning

1. **What to transfer**
   some knowledge is domain or task specific or may be more general/transferable

2. **When to transfer**
   does transfer learning always help or when could it even hurt?

3. **How to transfer**
   algorithms to include, carry over, and combine knowledge

Question time

What are the key measures of success of transfer?

(Some) central goals of transfer learning

1. **Improve**
   obtain a more accurate function in direct comparison to learning exclusively on the target

2. **Accelerate**
   learn to fine-tune faster to the target in comparison to learning from scratch

3. **Use less data**
   reduce data dependency of the target or required amount of labels in the target

Question time

What strategies for transfer can you think of?

Families of practical approaches

Instance transfer: re-weigh some labeled data in the source domain for use in the target domain

Families of practical approaches

Instance transfer: re-weigh some labeled data in the source domain for use in the target domain

Feature-representation transfer: find a “good” representation that reduces difference between the source & target domains

“A Survey on Transfer Learning”, Pan & Yang, IEEE Transactions on Knowledge and Data Engineering 22(10), 2009

Families of practical approaches

Instance transfer: re-weigh some labeled data in the source domain for use in the target domain

Feature-representation transfer: find a “good” representation that reduces difference between the source & target domains

Parameter-transfer: discover shared parameters or priors between the source domain and target domain models

“A Survey on Transfer Learning”, Pan & Yang, IEEE Transactions on Knowledge and Data Engineering 22(10), 2009

Families of practical approaches

Instance transfer: re-weigh some labeled data in the source domain for use in the target domain

Feature-representation transfer: find a “good” representation that reduces difference between the source & target domains

Parameter-transfer: discover shared parameters or priors between the source domain and target domain models

Relational-knowledge transfer: build mapping of relational knowledge between the source domain and the target domains

Instance transfer: re-weighing source data

A simple idea if domains differ only in marginal distribution p(x), but conditionals p(y|x) remain the same :

$$\mathbb{E}_{(x,y)\sim p^T}[\mathscr{L}(x,y;\theta)] = \mathbb{E}_{(x,y)\sim p^S}[\frac{p^T(x)}{p^S(x)}\mathscr{L}(x,y;\theta)]$$

Universität Bremen

Instance transfer: re-weighing source data

A simple idea if domains differ only in marginal distribution p(x), but conditionals p(y|x) remain the same :

$$\mathbb{E}_{(x,y)\sim p^T}[\mathscr{L}(x,y;\theta)] = \mathbb{E}_{(x,y)\sim p^S}[\frac{p^T(x)}{p^S(x)}\mathscr{L}(x,y;\theta)]$$

We could then use a ratio (that is typically unknown & often difficulty to practically obtain, but there are ways) in a minimization objective

$$\mathtt{min}_\theta \frac{1}{n^S}\sum_{i=1}^{n^S}\beta_i \mathscr{L}(f_\theta(x_i^S), y_i^S) \quad \text{with} \quad \beta_i = \frac{p^T(x_i)}{p_S(x_i)}$$

Feature representation transfer

A key goal of feature transformation based transfer is to reduce the distribution difference of source and target domain instances

Feature representation transfer

A key goal of feature transformation based transfer is to reduce the distribution difference of source and target domain instances

There are various metrics to minimize.
A popular one is maximum-mean discrepancy, which frames distribution differences as distances between mean embeddings of features for some feature map $\Phi : \mathscr{X} \to \mathscr{H}$:

$$MMD(x^S, x^T) = \left|\left|\frac{1}{n^S}\sum_{i=1}^{n^S}\Phi(x_i^S) - \frac{1}{n^T}\sum_{j=1}^{n^T}\Phi(x_j^T)\right|\right|_{\mathscr{H}}^2$$

Parameter transfer

For parameter transfer, we are now subject to the vast space of model choices and model-based interpretations.

There exist various prior techniques, but parameter based transfer has really become vastly popular with the advent of deep neural networks.

Let us do a quick revisit of some deep neural network fundamentals and discuss them in the context of transfer learning

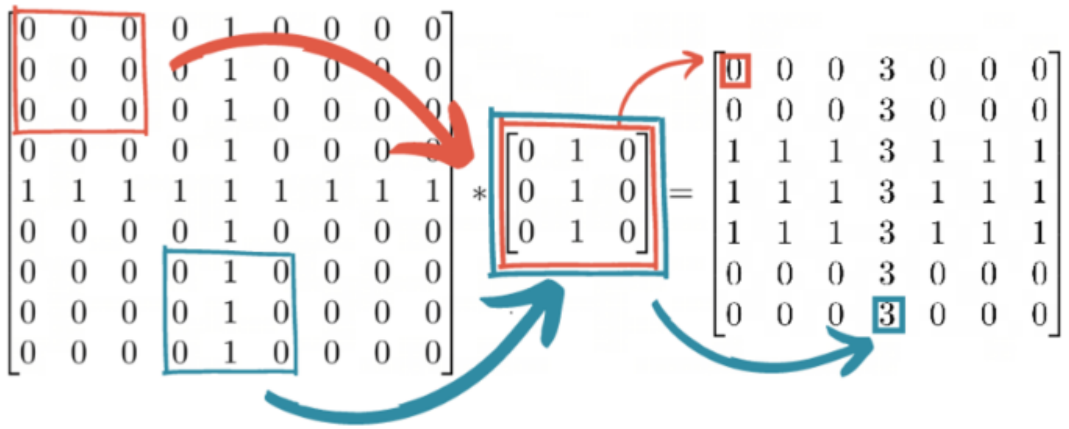
“A Survey on Transfer Learning”, Pan & Yang, IEEE Transactions on Knowledge and Data Engineering 22(10), 2009

Question time

Would you expect the representations of a deep neural network to be useful for transfer? Why?

A small recap of convolutional neural nets

Convolutions: multiple learnable patterns - “weight-sharing”

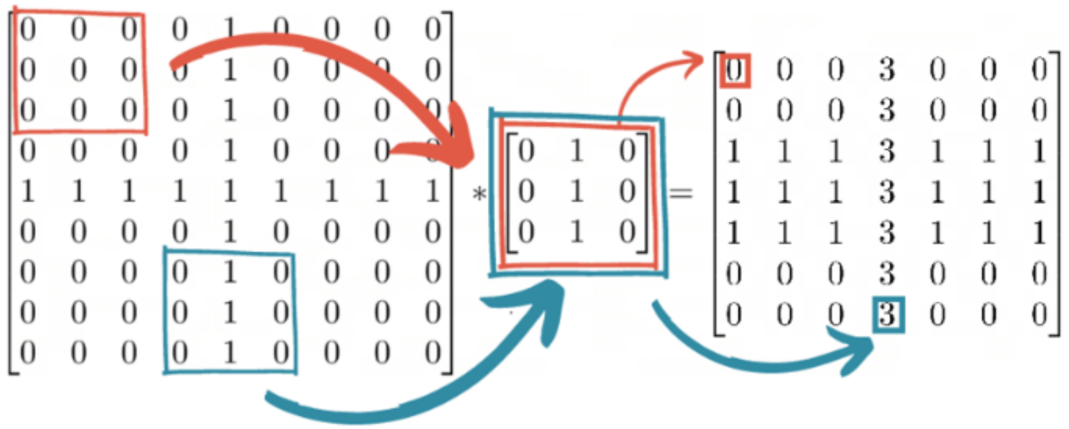


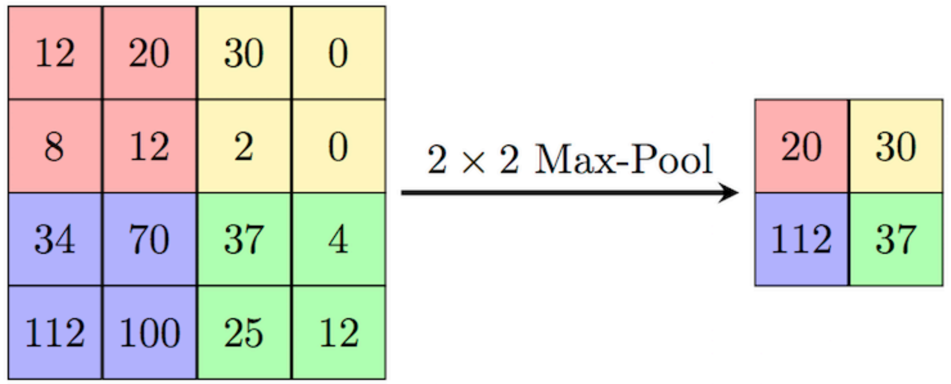
$$\begin{bmatrix}0&0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0&0\\1&1&1&1&1&1&1&1&1\\0&0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0&0\end{bmatrix} * \begin{bmatrix}0&1&0\\0&1&0\\0&1&0\end{bmatrix} = \begin{bmatrix}\boxed{0}&0&0&3&0&0&0\\0&0&0&3&0&0&0\\1&1&1&3&1&1&1\\1&1&1&3&1&1&1\\1&1&1&3&1&1&1\\0&0&0&3&0&0&0\\0&0&0&\boxed{3}&0&0&0\end{bmatrix}$$

https://deepai.org/machine-learning-glossary-and-terms/convolutional-neural-network

A small recap of convolutional neural nets

Convolutions: multiple learnable patterns - “weight-sharing”
Pooling: not learned dimensionality reduction - “local invariance”





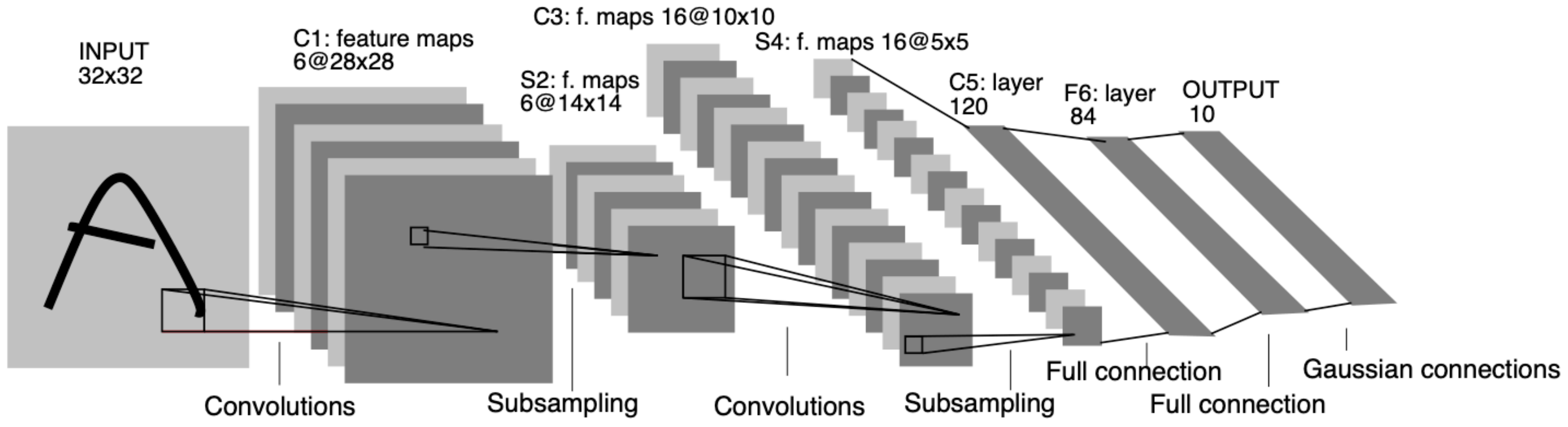
https://deepai.org/machine-learning-glossary-and-terms/convolutional-neural-network

https://computersciencewiki.org/index.php/Max-pooling_/_Pooling

Universität Bremen

A small recap of convolutional neural nets

Convolutions: multiple learnable patterns - “weight-sharing”
Pooling: not learned dimensionality reduction - “local invariance”
Stacked: with dropout, batch-normalization, other motivated “tricks”

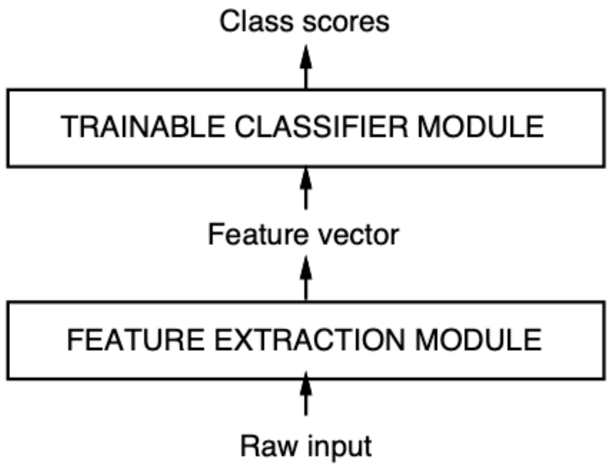


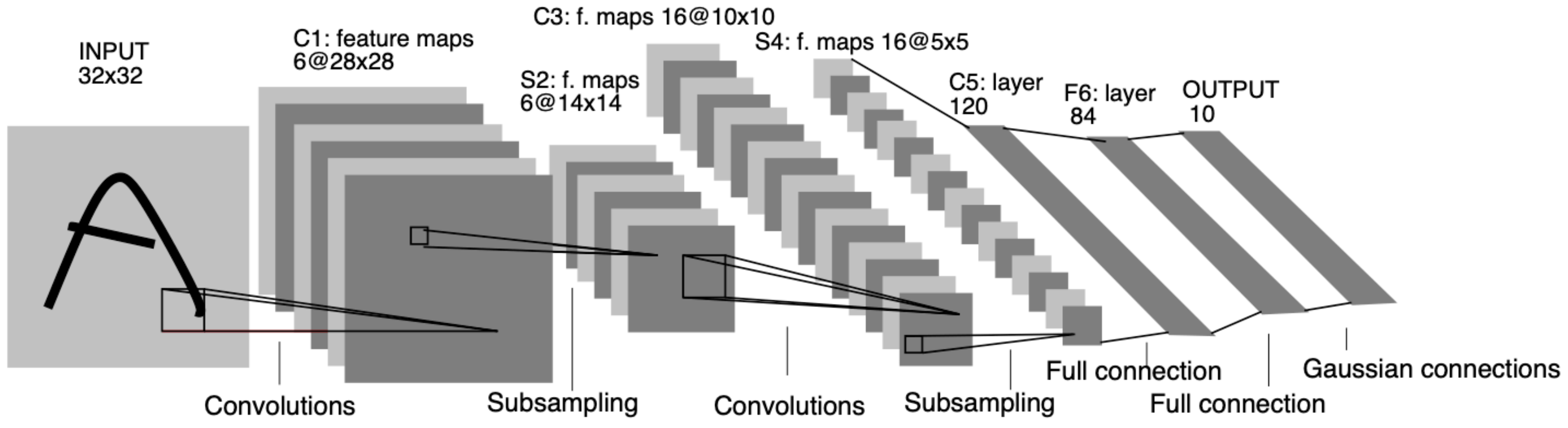
“Gradient-Based Learning Applied to Document Recognition”, LeCun et al, Proceedings of the IEEE, 1998

Universität Bremen

A small recap of convolutional neural nets

Convolutions: multiple learnable patterns - "weight-sharing"
Pooling: not learned dimensionality reduction - "local invariance"
Stacked: with dropout, batch-normalization, other motivated "tricks"
Don't worry about training, we'll revisit this again later when needed.
For now, it is important we have feature extraction + a classifier





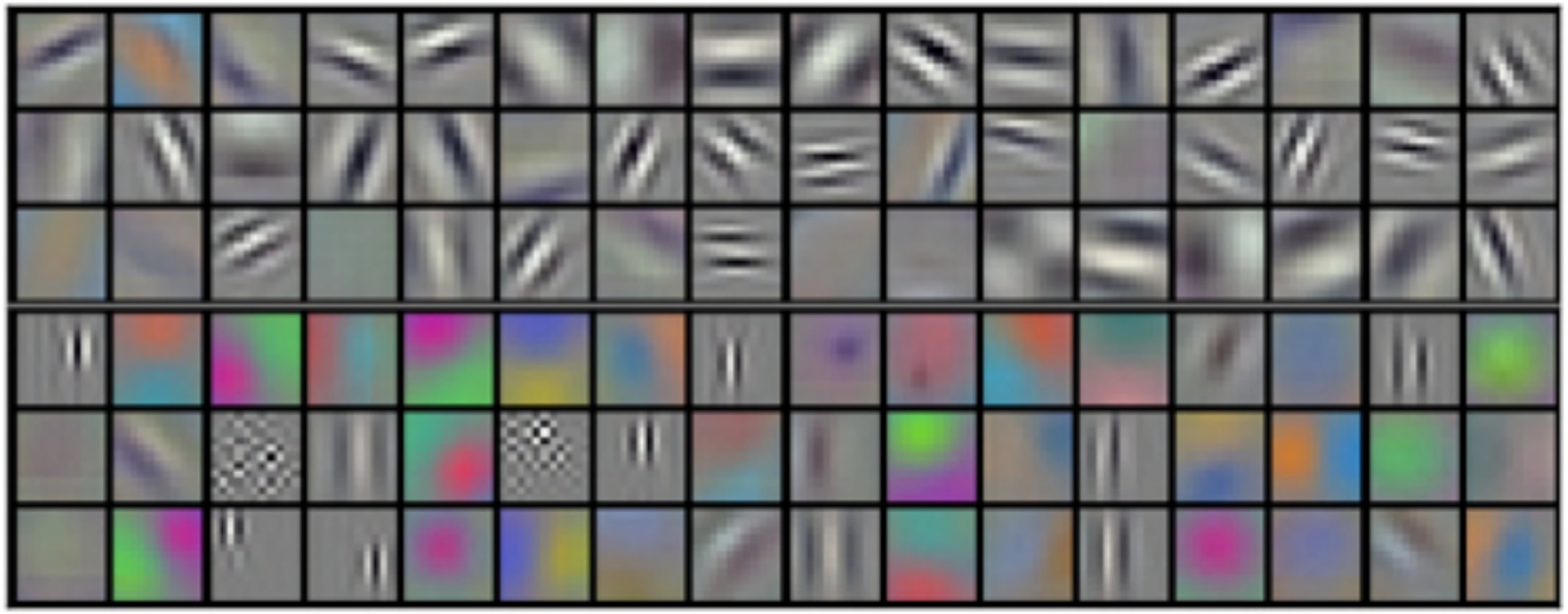
"Gradient-Based Learning Applied to Document Recognition", LeCun et al, Proceedings of the IEEE, 1998

Question time

What makes the learned “filters” of deep networks supposedly generic? What do they extract?

Neural transfer learning key hypothesis

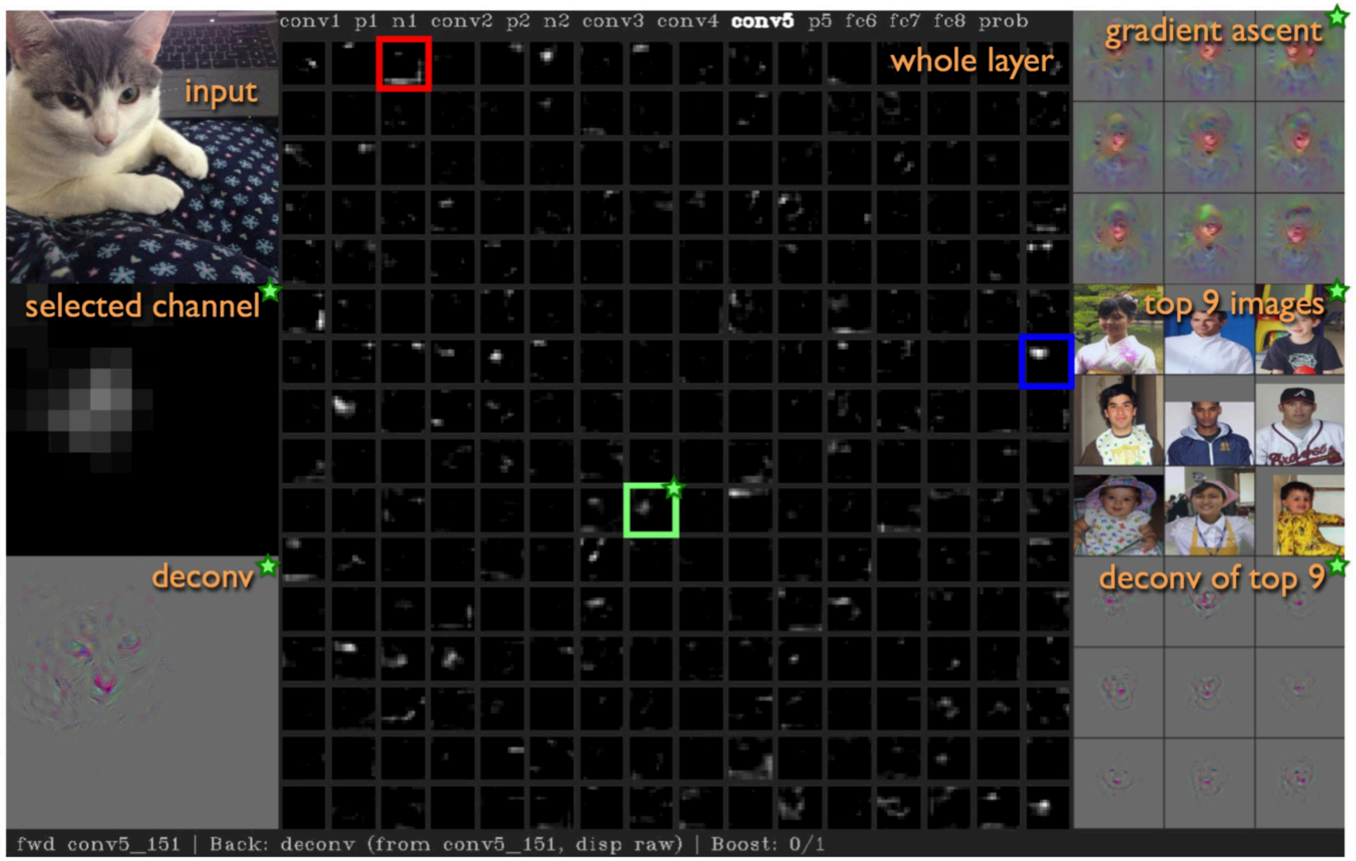
earlier/lower NN layers learn more “generic” patterns,
later/deeper layers become increasingly more task-specific



“ImageNet Classification with Deep Convolutional Neural Networks”, Krizhevsky et al, NeurIPS 2012

Universität Bremen

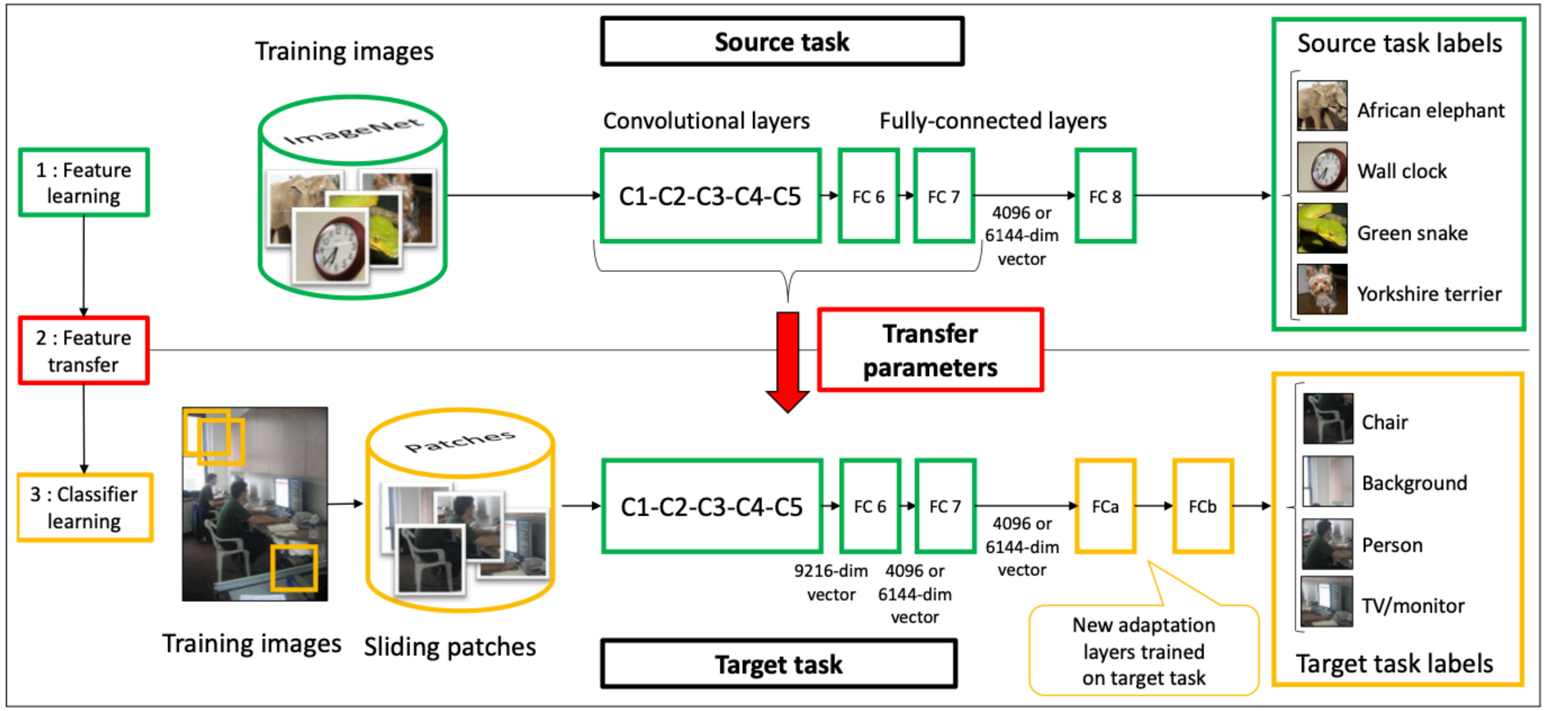
Neural transfer learning key hypothesis


conv1 p1 n1 conv2 p2 n2 conv3 conv4 conv5 p5 fc6 fc7 fc8 prob
input
whole layer
gradient ascent
selected channel
top 9 images
deconv
deconv of top 9
fwd conv5_151 | Back: deconv (from conv5_151, disp raw) | Boost: 0/1

“Understanding Neural Networks Through Deep Visualization”, Yosinski et al, ICML Deep Learning Workshop, 2015

Neural transfer learning key hypothesis

The historically common way is to copy the entire feature extractor



"Learning and Transferring Mid-Level Image Representations using Convolutional Neural Networks", Oquab et al, CVPR 2014

Universität
Bremen

Example: (inductive) ImageNet transfer

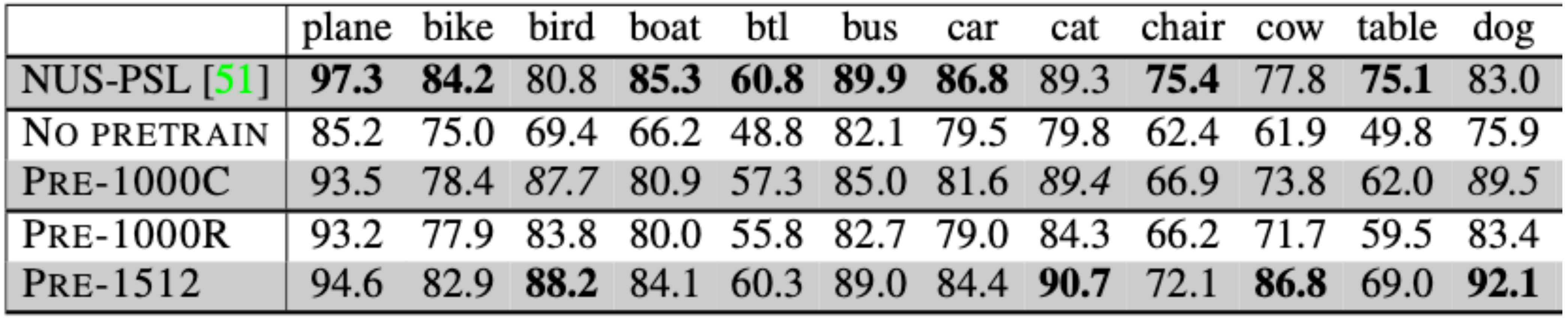
Pre-training on ImageNet (i.e. 59 bird species and 120 dog breeds) for the task on Pascal VOC 2012 (bird and dog class)

"Learning and Transferring Mid-Level Image Representations using Convolutional Neural Networks", Oquab et al, CVPR 2014

Universität Bremen

Example: (inductive) ImageNet transfer

Pre-training on ImageNet (i.e. 59 bird species and 120 dog breeds) for the task on Pascal VOC 2012 (bird and dog class)



	plane	bike	bird	boat	btl	bus	car	cat	chair	cow	table	dog
NUS-PSL [51]	**97.3**	**84.2**	80.8	**85.3**	**60.8**	**89.9**	**86.8**	89.3	**75.4**	77.8	**75.1**	83.0
NO PRETRAIN	85.2	75.0	69.4	66.2	48.8	82.1	79.5	79.8	62.4	61.9	49.8	75.9
PRE-1000C	93.5	78.4	*87.7*	80.9	57.3	85.0	81.6	*89.4*	66.9	73.8	62.0	*89.5*
PRE-1000R	93.2	77.9	83.8	80.0	55.8	82.7	79.0	84.3	66.2	71.7	59.5	83.4
PRE-1512	94.6	82.9	**88.2**	84.1	60.3	89.0	84.4	**90.7**	72.1	**86.8**	69.0	**92.1**

“Learning and Transferring Mid-Level Image Representations using Convolutional Neural Networks”, Oquab et al, CVPR 2014

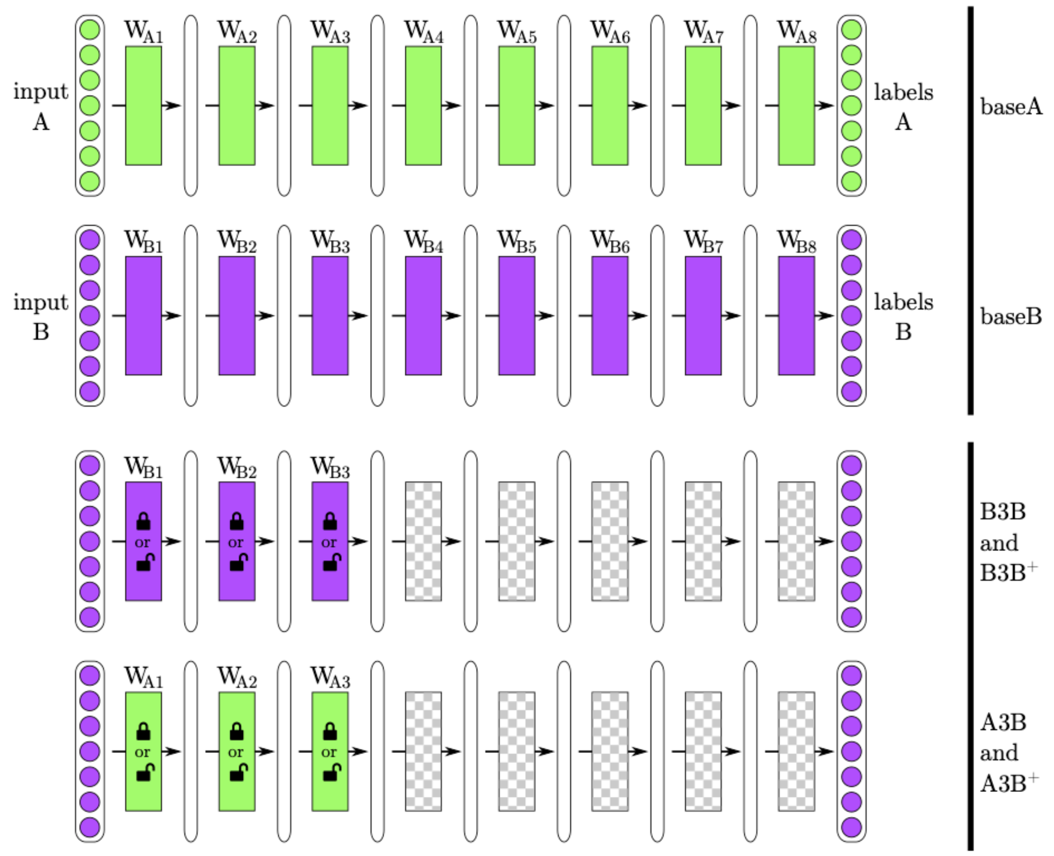
The next tutorial!

It will be your next tutorial to figure out how the different layers of a deep neural network relate to transferability.

The next tutorial!

It will be your next tutorial to figure out how the different layers of a deep neural network relate to transferability.

You will transfer weights, freeze weights, reset & fine-tune different configurations to test this in practice, inspired by prior work



"How transferable are features in deep neural networks", Yosinski et al, NeurIPS 2014

Question time

Are (deep) neural networks only good for parameter transfer, or can we also leverage the other approaches to transfer? How?

Families of practical approaches

Instance transfer: re-weigh some labeled data in the source domain for use in the target domain

Feature-representation transfer: find a “good” representation that reduces difference between the source & target domains

Parameter-transfer: discover shared parameters or priors between the source domain and target domain models

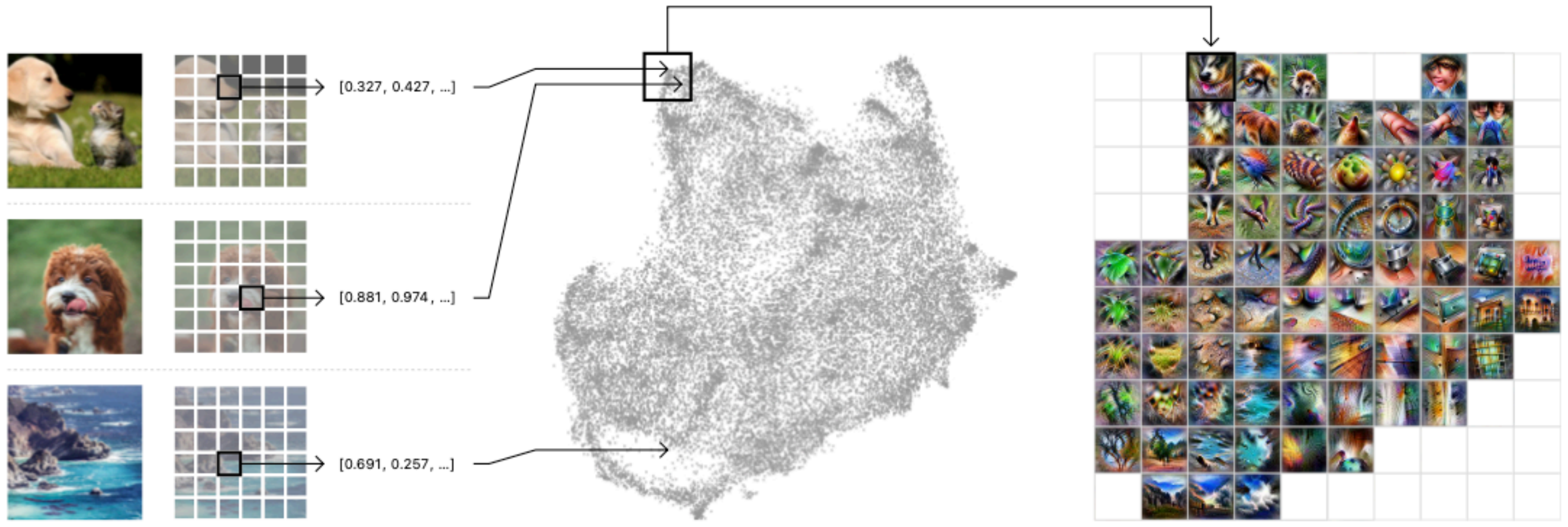
Relational-knowledge transfer: build mapping of relational knowledge between the source domain and the target domains

“A Survey on Transfer Learning”, Pan & Yang, IEEE Transactions on Knowledge and Data Engineering 22(10), 2009

Universität Bremen

Feature representation transfer in NNs

Embeddings should be semantically related -> measure discrepancy



A randomized set of one million images is fed through the network, collecting one random spatial activation per image.

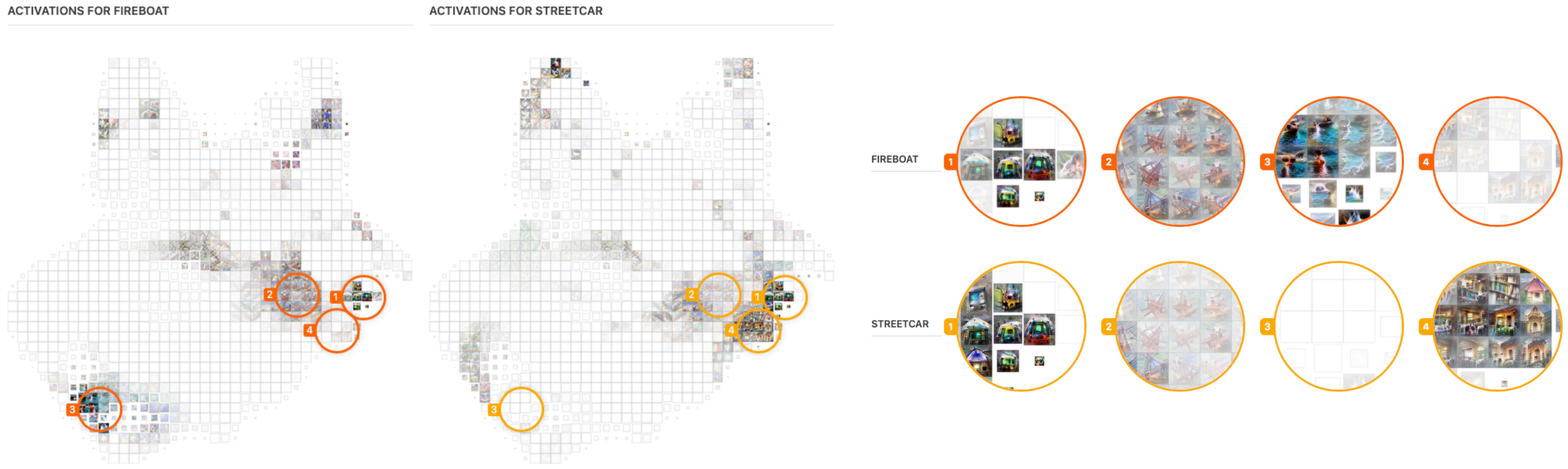
The activations are fed through UMAP to reduce them to two dimensions. They are then plotted, with similar activations placed near each other.

We then draw a grid and average the activations that fall within a cell and run feature inversion on the averaged activation. We also optionally size the grid cells according to the density of the number of activations that are averaged within.

“Activation Atlas”, Carter et al, Distill 2019

Feature representation transfer in NNs

Embeddings should be semantically related -> measure discrepancy



“Activation Atlas”, Carter et al, Distill 2019

Feature representation transfer in NNs

Neural embeddings are useful because they allow:

- To also measure discrepancies, e.g. MMD, in addition to only transferring by copying parameters
- “Align” representations due to transformations being complex (recall our earlier motivating example of projection matrices)
- Infer labels from potential unlabelled target domain data or learn with only few labelled data points

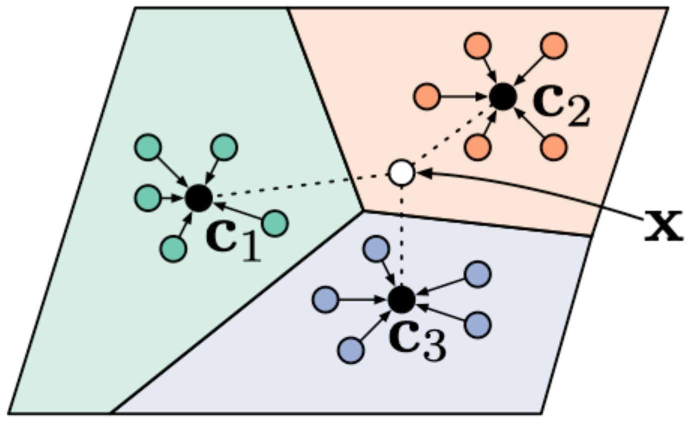
Formalizing transfer learning: a last definition

Definition - Unsupervised Transfer Learning - Pan & Yang 2009:
"*Given a source domain* D_S *and learning task* $\mathcal{T}_S$, *a target domain* D_T *and learning task* $\mathcal{T}_T$, *unsupervised transfer learning aims to help improve the learning of the target predictive function* $f_T(\,.\,)$ *in* D_T *using the knowledge in* D_S *and* $\mathcal{T}_S$, *where* $\mathcal{T}_s \neq \mathcal{T}_T$ *and* $\mathcal{Y}_S \,\&\, \mathcal{Y}_T$ *are not observable.*"

"A Survey on Transfer Learning", Pan & Yang, IEEE Transactions on Knowledge and Data Engineering 22(10), 2009

A relaxation: few-shot

Few shot learning is a specific instance of transfer, where we assume only a few data points in the target domain to be labelled

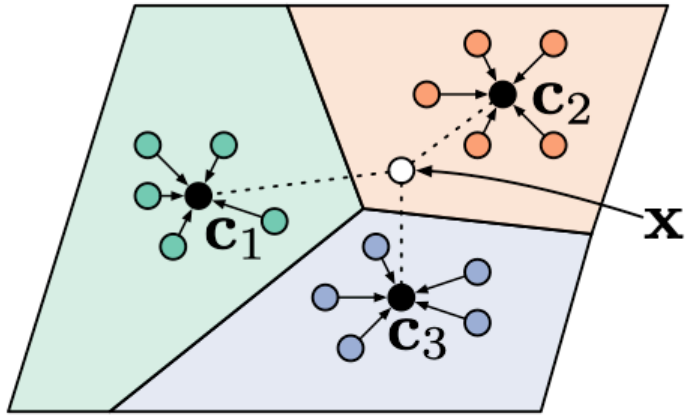


(a) Few-shot

"Prototypical Networks for Few-shot Learning", Snell et al, NeurIPS 2017
See also "Object Classification from a Single Example Utilizing Class relevance Metrics", M. Fink, NeurIPS 2004 & "One-shot Learning of Object Categories", Fei-Fei et al, TPAMI 2006

A relaxation: few-shot

Few shot learning is a specific instance of transfer, where we assume only a few data points in the target domain to be labelled



(a) Few-shot

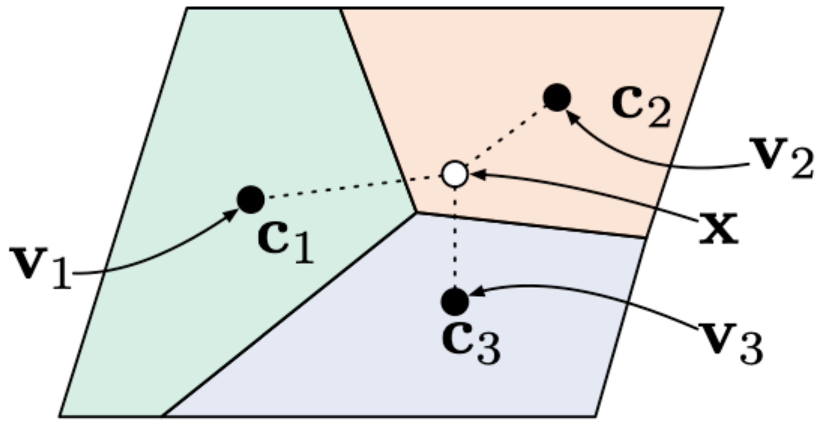
- Compute "few-shot prototypes" c_k as the mean of the embedded support examples for each class/concept
- Classify via a Softmax over distances d (of choice) to prototypes:

$$p_\theta(y = k \,|\, x) \propto \exp(-d(f_\theta(x), c_k))$$

"Prototypical Networks for Few-shot Learning", Snell et al, NeurIPS 2017
See also "Object Classification from a Single Example Utilizing Class relevance Metrics", M. Fink, NeurIPS 2004 & "One-shot Learning of Object Categories", Fei-Fei et al, TPAMI 2006

A relaxation: one/zero-shot

One shot learning is a specific instance of transfer based on one example. Zero-shot is related to few-shot, but instead embedding additional meta-data v_k



(b) Zero-shot

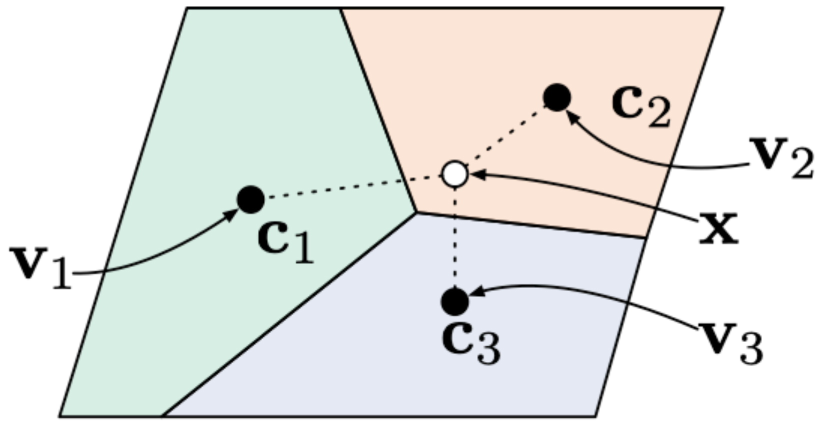
“Prototypical Networks for Few-shot Learning”, Snell et al, NeurIPS 2017
See also “Object Classification from a Single Example Utilizing Class relevance Metrics”, M. Fink, NeurIPS 2004 & “One-shot Learning of Object Categories”, Fei-Fei et al, TPAMI 2006

Universität Bremen

OWL ML

A relaxation: one/zero-shot

One shot learning is a specific instance of transfer based on one example. Zero-shot is related to few-shot, but instead embedding additional meta-data v_k



(b) Zero-shot

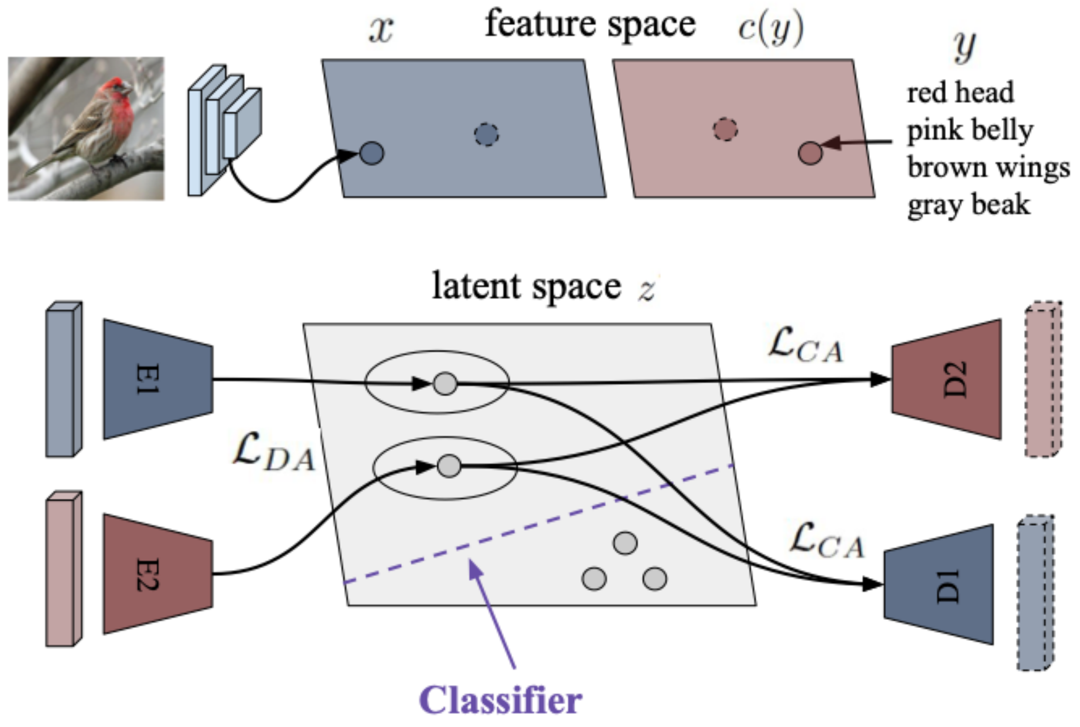
"*We say that a set of classes is* $\gamma > 0$ *separated with respect to a distance function d if for any pair of examples belonging to the same class* $\{(x_1, c), (x_1', c)\}$*, the distance* $d(x_1, x_1')$ *is smaller than the distance between any pair of examples from different classes* $\{(x_2, e), (x_2', g)\}$ *by at least* γ*:* $d(x_1, x_1') \leq d(x_2, x_2') - \gamma$*."*

"Prototypical Networks for Few-shot Learning", Snell et al, NeurIPS 2017
See also "Object Classification from a Single Example Utilizing Class relevance Metrics", M. Fink, NeurIPS 2004 & "One-shot Learning of Object Categories", Fei-Fei et al, TPAMI 2006

Universität Bremen

An intuitive multi-modal example for few/zero shots

Train two variational auto-encoder models (composed of an encoder projecting into a latent embedding and a decoder), one for images and one for text describing these images

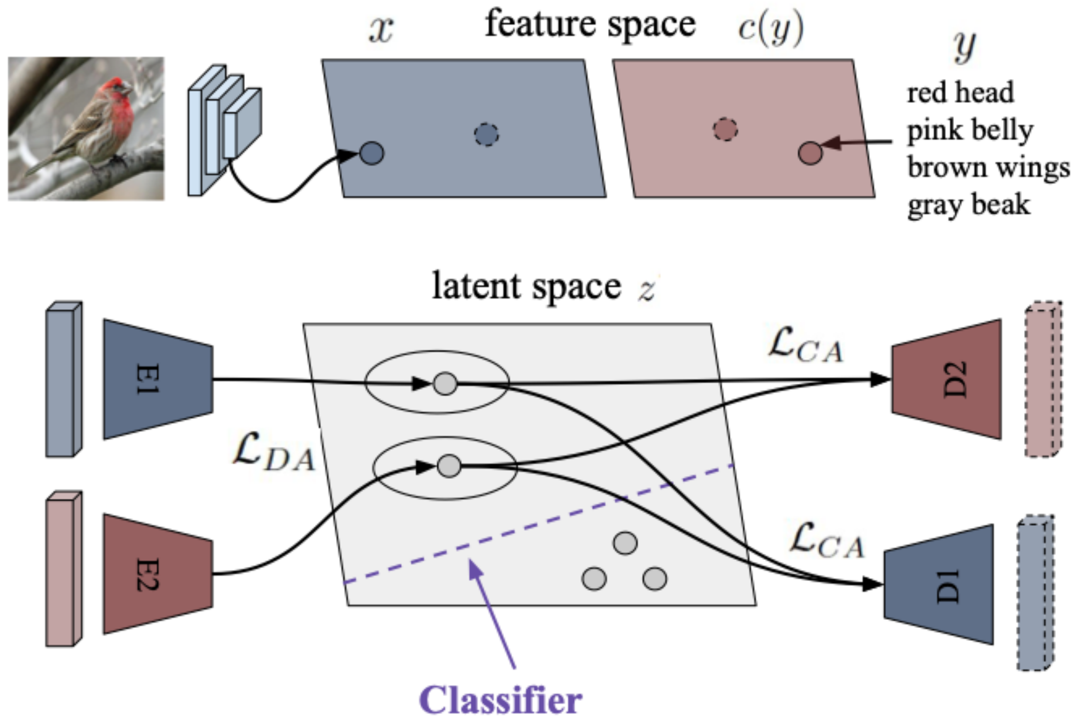


“Generalized Zero- and Few-Shot Learning via Aligned Variational Autoencoders”, Schoenfeld et al, CVPR 2019

An intuitive multi-modal example for few/zero shots

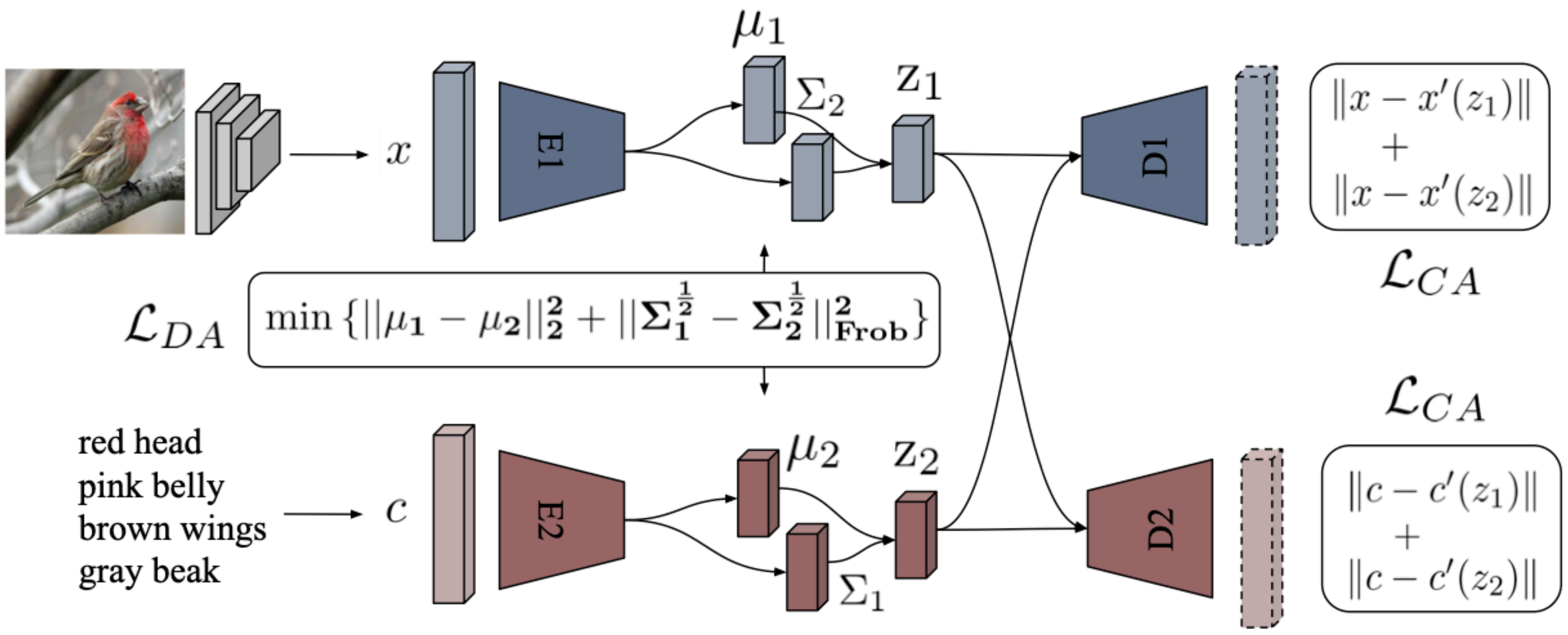
Train two variational auto-encoder models (composed of an encoder projecting into a latent embedding and a decoder), one for images and one for text describing these images

In this example, assume representations follow latent multivariate Gaussian distributions and minimize discrepancy



"Generalized Zero- and Few-Shot Learning via Aligned Variational Autoencoders", Schoenfeld et al, CVPR 2019

An intuitive multi-modal example for few/zero shots


x
E1
μ_1
Σ_2
z_1
D1
$\|x - x'(z_1)\| + \|x - x'(z_2)\|$
$\mathcal{L}_{CA}$
$\mathcal{L}_{DA}$
$\min \{||\mu_1 - \mu_2||_2^2 + ||\Sigma_1^{\frac{1}{2}} - \Sigma_2^{\frac{1}{2}}||^2_{Frob}\}$
red head
pink belly
brown wings
gray beak
c
E2
μ_2
Σ_1
z_2
D2
$\mathcal{L}_{CA}$
$\|c - c'(z_1)\| + \|c - c'(z_2)\|$

"Generalized Zero- and Few-Shot Learning via Aligned Variational Autoencoders", Schoenfeld et al, CVPR 2019

Question time

Zero-shot learning sounds too good to be true? When do we need to transfer?

Transfer in context of learning from “hints”

“A hint is any piece of information about the function f. As a matter of fact, an input-output example is a special case of a hint. A hint may take the form of a global constraint on f, such as a symmetry property or an invariance.”

Abu-Mostafa, “Learning from Hints in Neural Networks”
Journal of Complexity 6, 1990

Transfer in context of learning from “hints”

“A hint is any piece of information about the function f. As a matter of fact, an input-output example is a special case of a hint. A hint may take the form of a global constraint on f, such as a symmetry property or an invariance.”

Abu-Mostafa, “Learning from Hints in Neural Networks”
Journal of Complexity 6, 1990

<u>Remember: feature transfer and augmented latent spaces!</u>
Symmetries and (quasi-)invariances play a big role in answering when we even need to transfer, or when we can “get-away” with re-using the representations that are already learned fully

Symmetries & Invariance

We could directly include in- or equivariance into our representations (e.g. rotation)

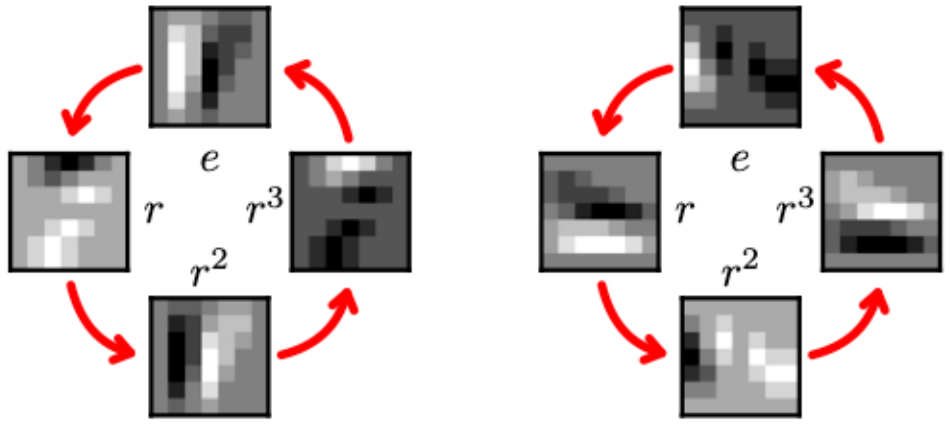


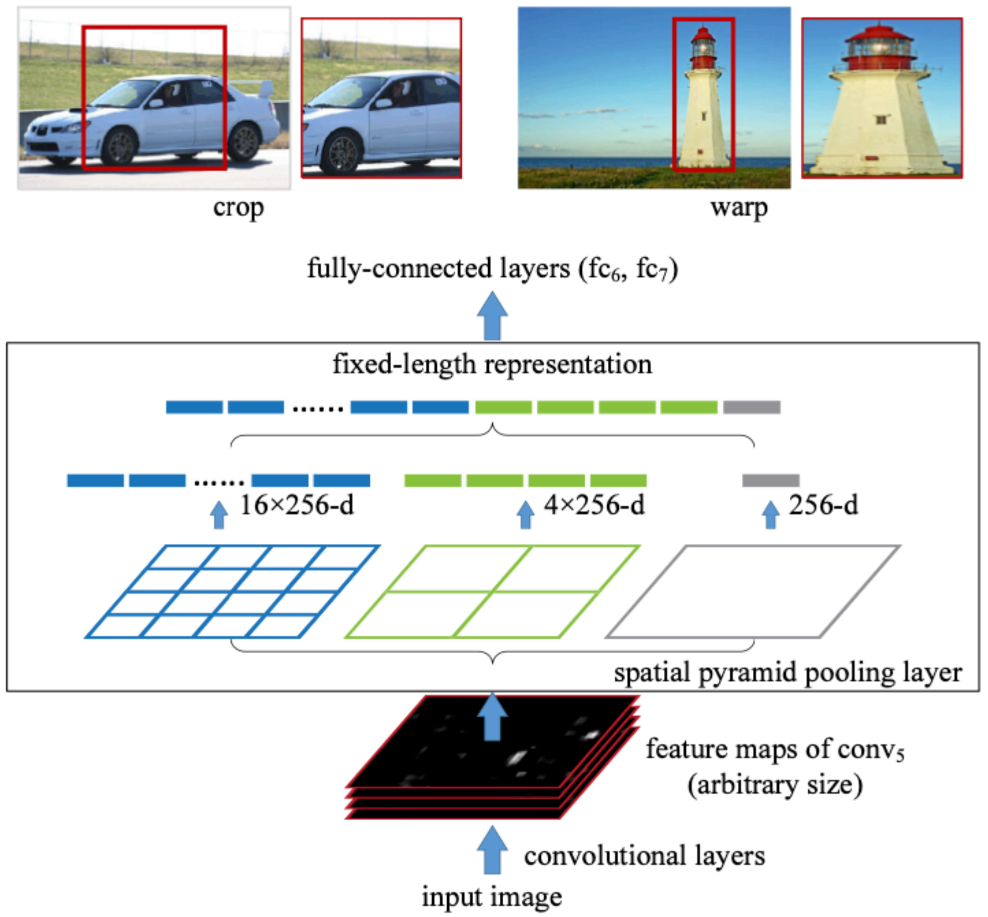
Figure 1. A p4 feature map and its rotation by r.

“Group Equivariant Convolutional Networks”, Cohen & Welling, ICML 2016

Symmetries & Invariance

We could directly include in- or equivariance into our representations (e.g. rotation)

We could also incorporate a degree of scale invariance, or even try to learn it



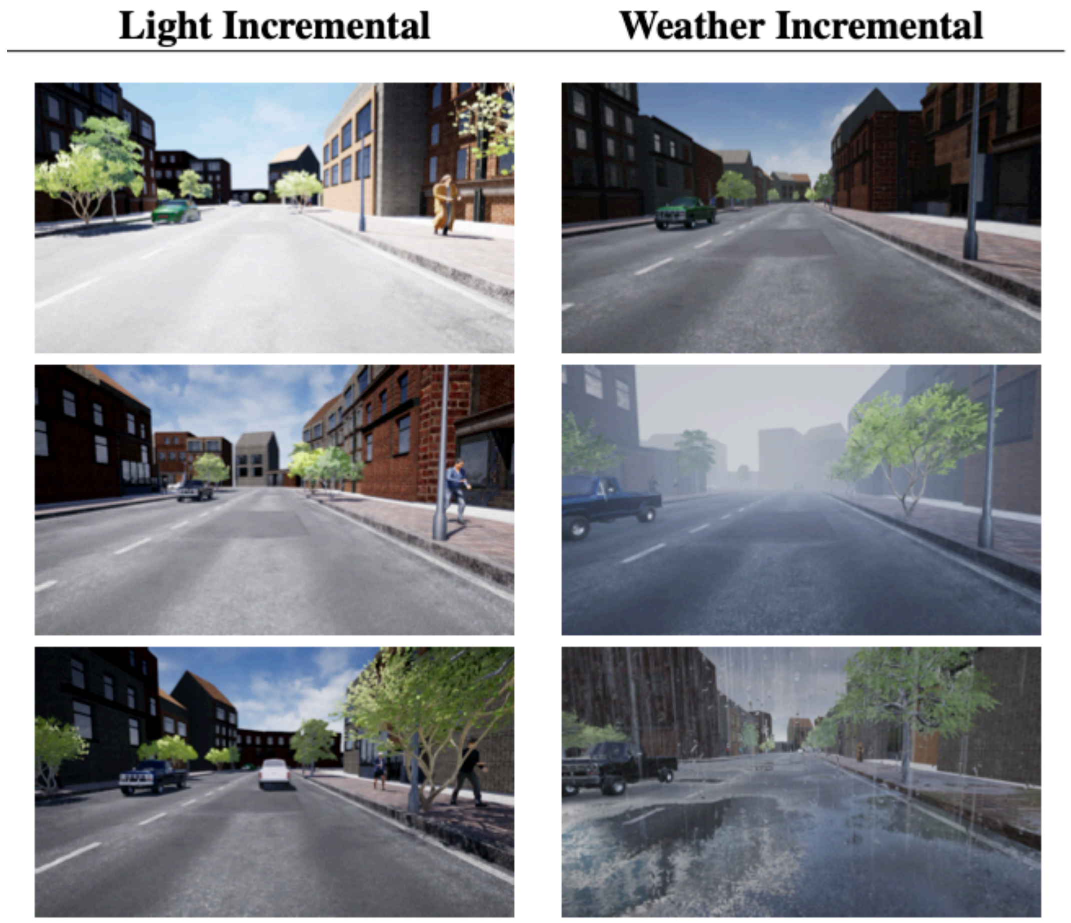
“Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition”, K. He et al, TPAMI 2015

Symmetries & Invariance

We could directly include in- or equivariance into our representations (e.g. rotation)

We could also incorporate a degree of scale invariance, or even try to learn it

We could use other knowledge about the world in pre-processing, for instance, accounting for homogeneous illumination changes



“A Procedural World Generation Framework for Systematic Evaluation of Continual Learning”, Hess et al, NeurIPS 2021

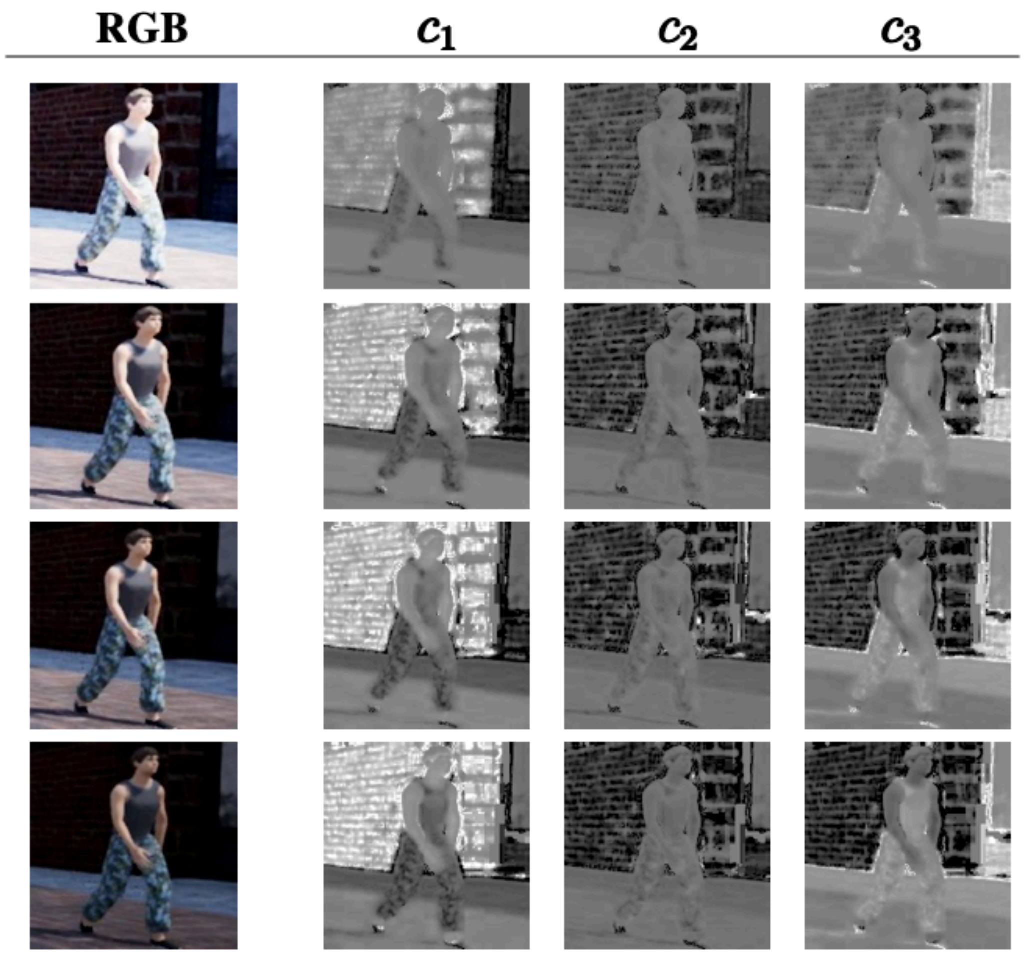
Universität Bremen

Symmetries & Invariance

Using the illumination example: we could assume that RGB color ratios are quasi-invariant with white illumination:

$$c_1 = \mathtt{arctan}\left(\frac{R}{\mathtt{max}(G, B)}\right)$$

(Gevers & Smeulders, “Color Based Object Recognition”, Pattern Recognition 32:3, 1999)



“A Procedural World Generation Framework for Systematic Evaluation of Continual Learning”, Hess et al, NeurIPS 2021

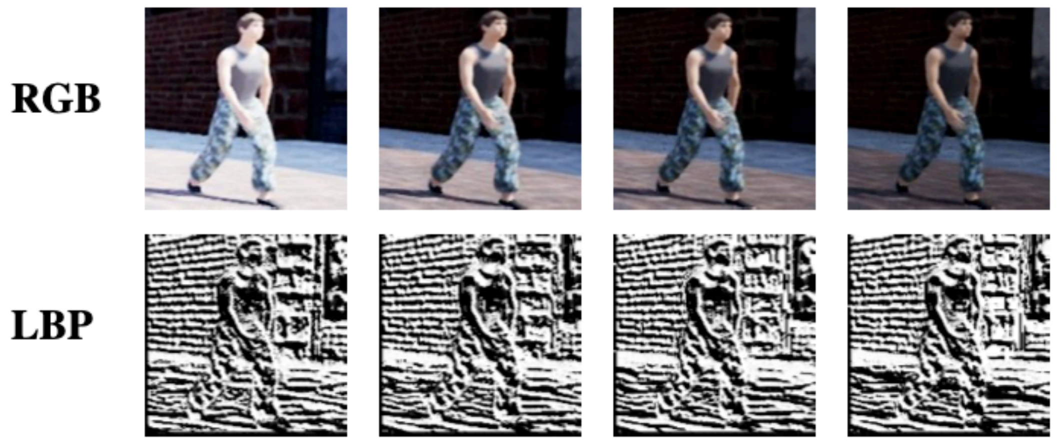
Symmetries & Invariance

Alternatively, we could make use of local binary patterns.

In a simplified form:

- mark all nearest neighbors for a pixel with 0 if greater and 1 otherwise
- compute histogram over values to create features

(He & Wang, Pattern Recognition 23:8, 1990)



“A Procedural World Generation Framework for Systematic Evaluation of Continual Learning”, Hess et al, NeurIPS 2021

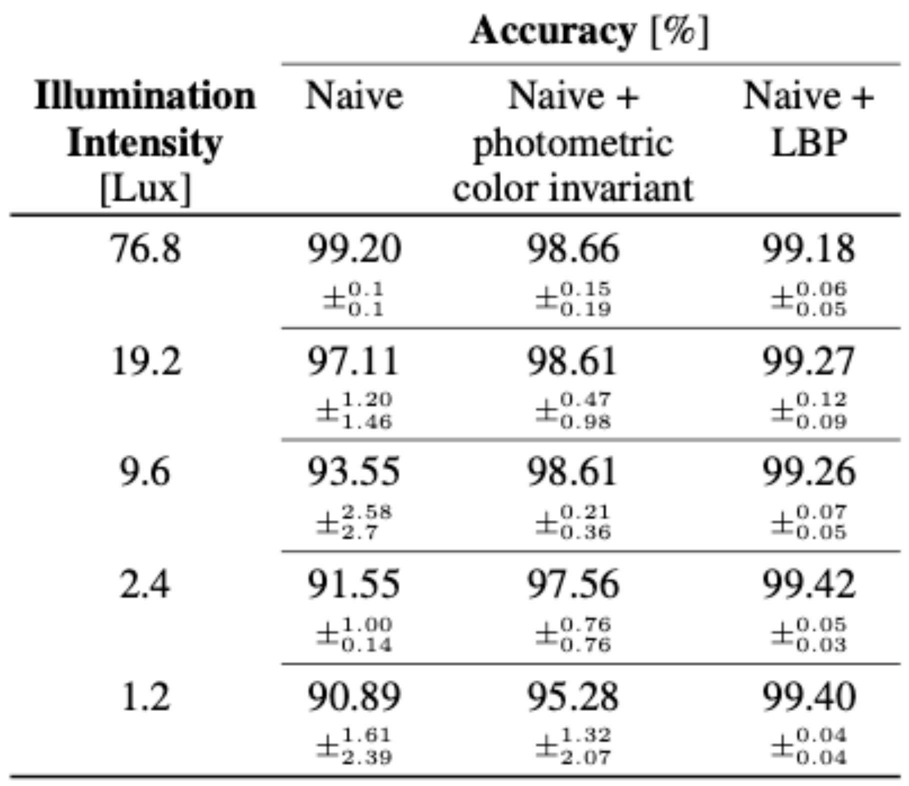
Symmetries & Invariance

Alternatively, we could make use of local binary patterns.

In a simplified form:

- mark all nearest neighbors for a pixel with 0 if greater and 1 otherwise
- compute histogram over values to create features

(He & Wang, Pattern Recognition 23:8, 1990)



	Accuracy [%]		
Illumination Intensity [Lux]	Naive	Naive + photometric color invariant	Naive + LBP
76.8	99.20 $\pm^{0.1}_{0.1}$	98.66 $\pm^{0.15}_{0.19}$	99.18 $\pm^{0.06}_{0.05}$
19.2	97.11 $\pm^{1.20}_{1.46}$	98.61 $\pm^{0.47}_{0.98}$	99.27 $\pm^{0.12}_{0.09}$
9.6	93.55 $\pm^{2.58}_{2.7}$	98.61 $\pm^{0.21}_{0.36}$	99.26 $\pm^{0.07}_{0.05}$
2.4	91.55 $\pm^{1.00}_{0.14}$	97.56 $\pm^{0.76}_{0.76}$	99.42 $\pm^{0.05}_{0.03}$
1.2	90.89 $\pm^{1.61}_{2.39}$	95.28 $\pm^{1.32}_{2.07}$	99.40 $\pm^{0.04}_{0.04}$

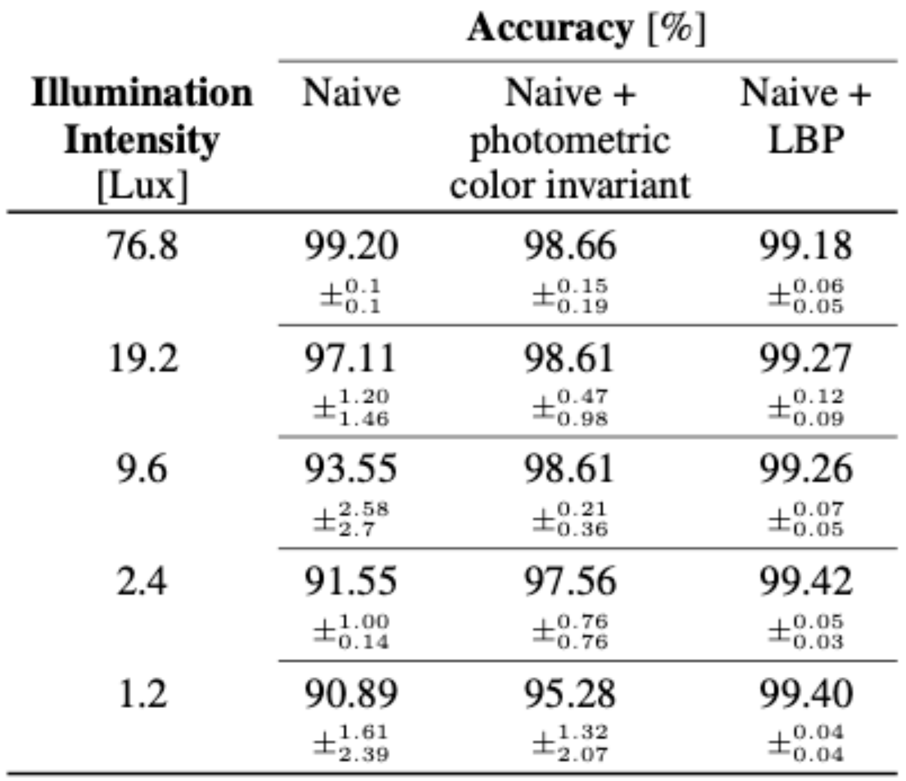
"A Procedural World Generation Framework for Systematic Evaluation of Continual Learning", Hess et al, NeurIPS 2021

Symmetries & Invariance

Independently of using a neural network, in this scenario, there is no need to overcomplicate approaches.

Pre-processing in the form of a feature augmentation can very well do the trick!

Remember that sometimes the solution is more simple than involved mechanisms in some big neural network



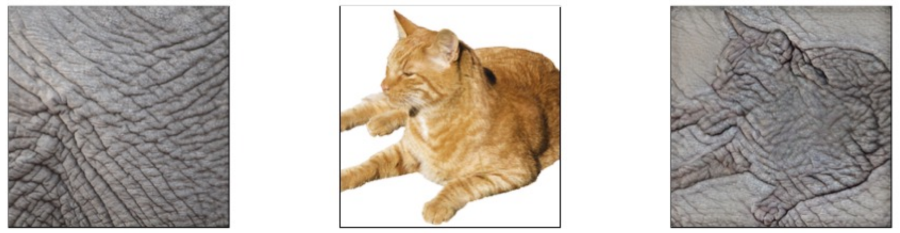
Illumination Intensity [Lux]	Accuracy [%]		
	Naive	Naive + photometric color invariant	Naive + LBP
76.8	99.20 $\pm^{0.1}_{0.1}$	98.66 $\pm^{0.15}_{0.19}$	99.18 $\pm^{0.06}_{0.05}$
19.2	97.11 $\pm^{1.20}_{1.46}$	98.61 $\pm^{0.47}_{0.98}$	99.27 $\pm^{0.12}_{0.09}$
9.6	93.55 $\pm^{2.58}_{2.7}$	98.61 $\pm^{0.21}_{0.36}$	99.26 $\pm^{0.07}_{0.05}$
2.4	91.55 $\pm^{1.00}_{0.14}$	97.56 $\pm^{0.76}_{0.76}$	99.42 $\pm^{0.05}_{0.03}$
1.2	90.89 $\pm^{1.61}_{2.39}$	95.28 $\pm^{1.32}_{2.07}$	99.40 $\pm^{0.04}_{0.04}$

"A Procedural World Generation Framework for Systematic Evaluation of Continual Learning", Hess et al, NeurIPS 2021

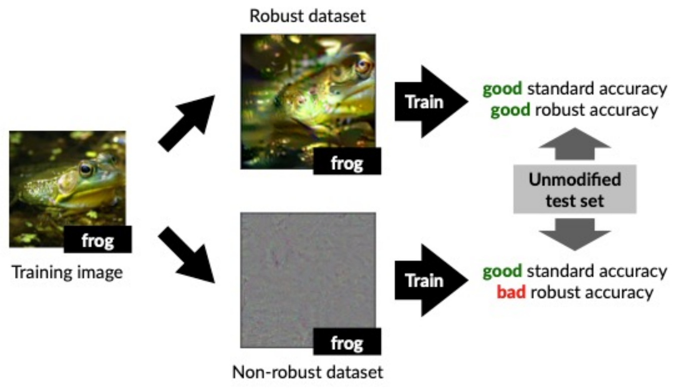
Question time

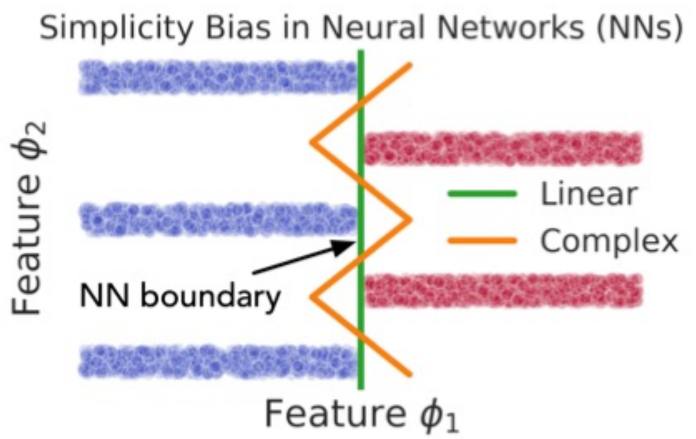
Before proceeding to the last transfer approach based on "relational-knowledge": is transfer in NNs really that simple? What other pitfalls do you see?

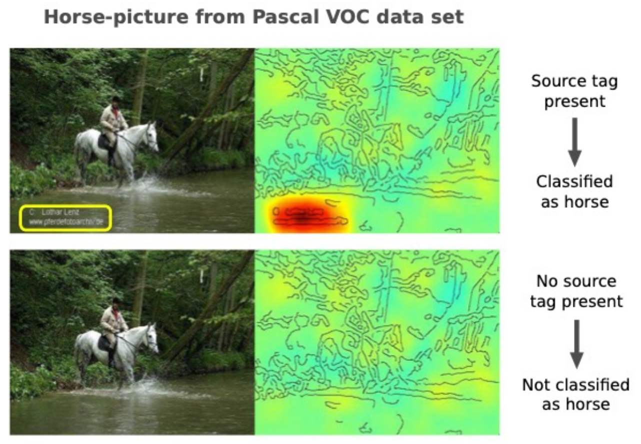
First recall what we discussed for curricula



(a) Texture image		(b) Content image		(c) Texture-shape cue conflict	
81.4%	**Indian elephant**	71.1%	**tabby cat**	63.9%	**Indian elephant**
10.3%	indri	17.3%	grey fox	26.4%	indri
8.2%	black swan	3.3%	Siamese cat	9.6%	black swan





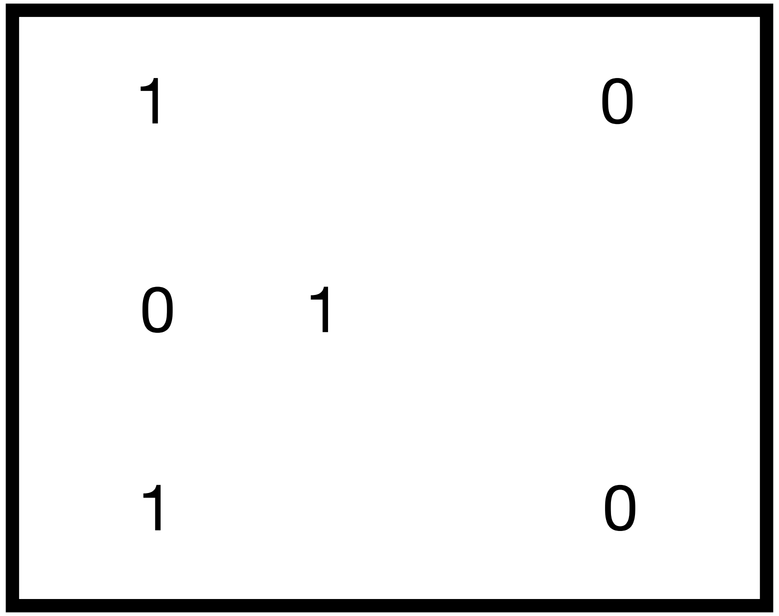


"The Pitfalls of Simplicity Bias in Neural Networks", Shah et al, NeurIPS 2020. "Unmasking Clever Hans Predictors", Lapuschkin et al, Nature Communications 2019. "ImageNet-trained CNNS are biased towards texture", Geirhos et al, ICLR 2019. "The Pitfalls of Simplicity Bias in Neural Networks", Shah et al, NeurIPS 2020

Universität Bremen

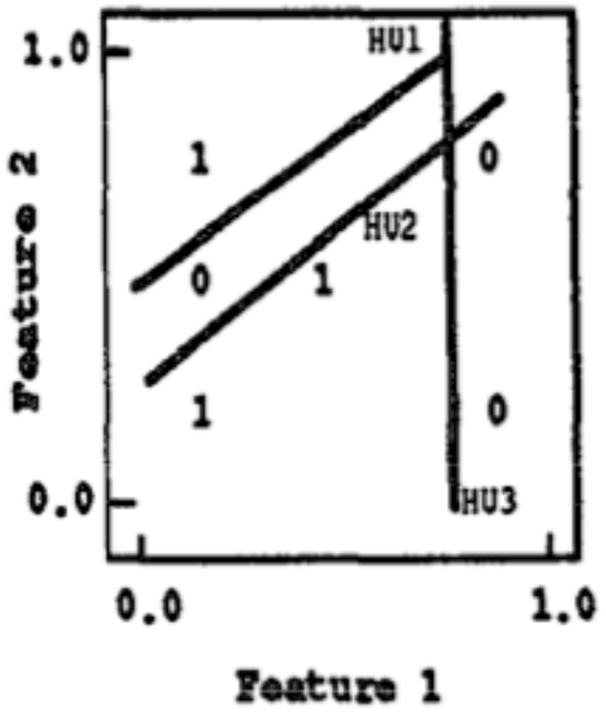
Even when ignoring these pitfalls, it is challenging!

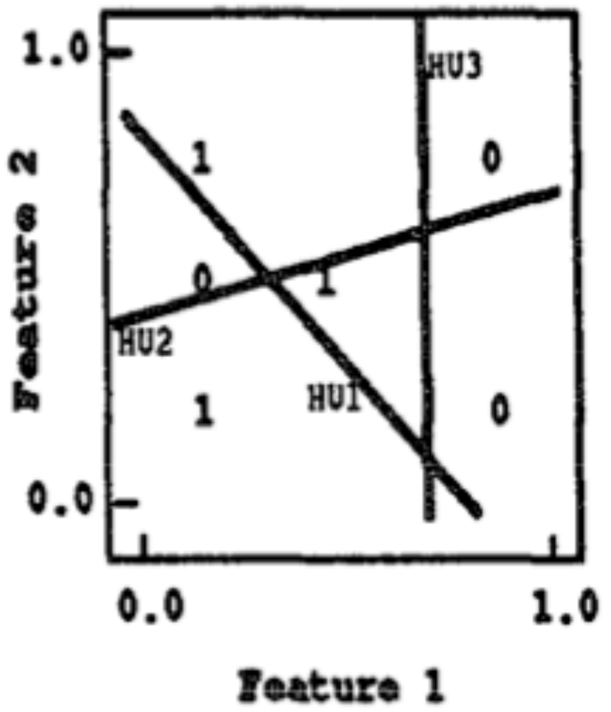
How would you separate this data with a set of hyperplanes? (Try 3)



Even when ignoring these pitfalls, it is challenging!

How would you separate this data with a set of hyperplanes? (Try 3)





“Direct Transfer of Learned Information Among Neural Networks” , L. Y. Pratt et al, AAAI 1991

It remains unintuitive when transfer works or hurts

Consider an example of training a deep neural network to predict defects in infrastructure, e.g. concrete bridges

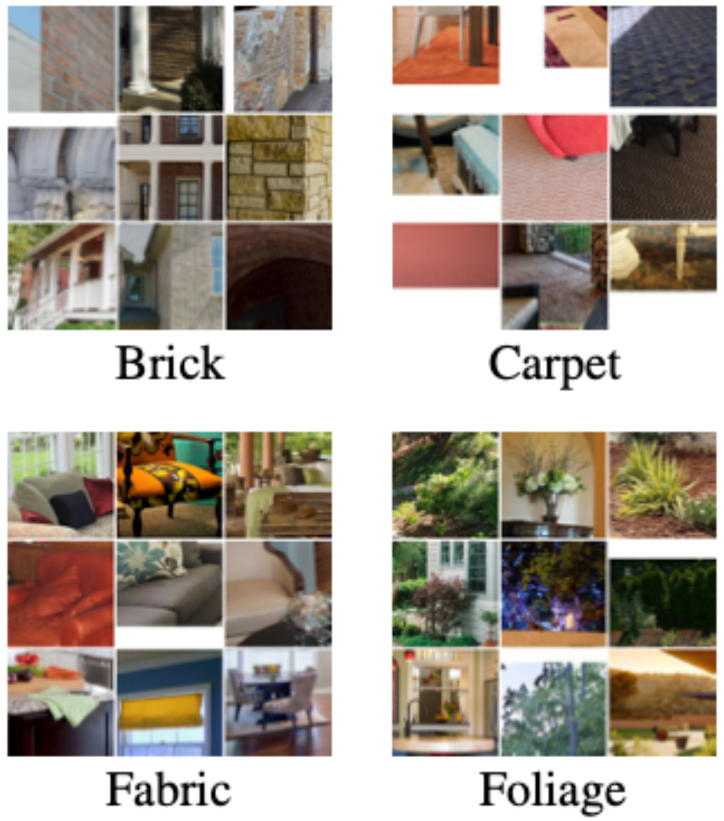
“Meta-learning Convolutional Neural Architectures for Multi-target Concrete Defect Classification with the Concrete Defect Bridge Image Dataset”, Mundt et al, CVPR 2019

It remains unintuitive when transfer works or hurts

Consider an example of training a deep neural network to predict defects in infrastructure, e.g. concrete bridges

Would we expect transfer to work based on pre-training on object-centric datasets like ImageNet?

Would we think it helps to pre-train on a material recognition task?



"Material Recognition in the Wild with the Materials in Context Database, CVPR 2015"

Universität
Bremen

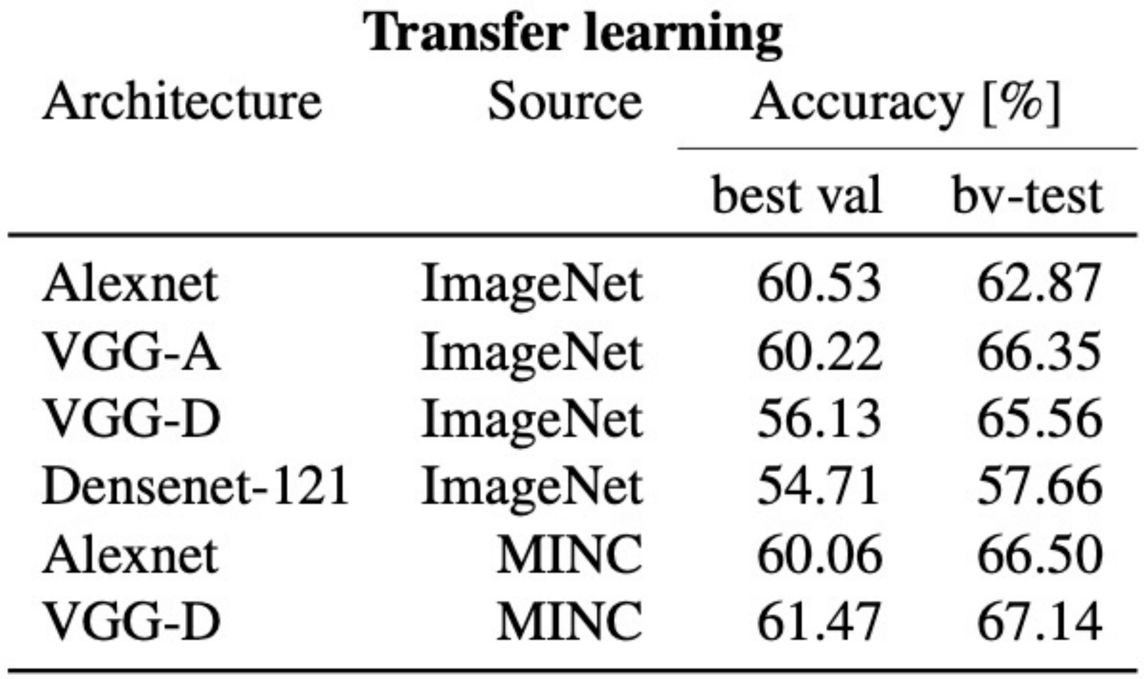
It remains unintuitive when transfer works or hurts

Training from scratch:

- Alexnet: 66.98 %
- VGG-A: 70.45%
- VGG-D: 70.61%

Pre-training on object recognition hurts

Unfortunately, pre-training on materials doesn't seem to help either



Transfer learning

Architecture	Source	Accuracy [%]	
		best val	bv-test
Alexnet	ImageNet	60.53	62.87
VGG-A	ImageNet	60.22	66.35
VGG-D	ImageNet	56.13	65.56
Densenet-121	ImageNet	54.71	57.66
Alexnet	MINC	60.06	66.50
VGG-D	MINC	61.47	67.14

"Meta-learning Convolutional Neural Architectures for Multi-target Concrete Defect Classification with the Concrete Defect Bridge Image Dataset", Mundt et al, CVPR 2019

Selective feature transfer

Selectively transferring features, rather than copying or freezing the parameters of entire layers, is beneficial

Universität Bremen

Selective feature transfer

Selectively transferring features, rather than copying or freezing the parameters of entire layers, is beneficial

We can try to identify features that are “representative” of new tasks, e.g. the subset with the highest set of activation values. The remainder could be reset.

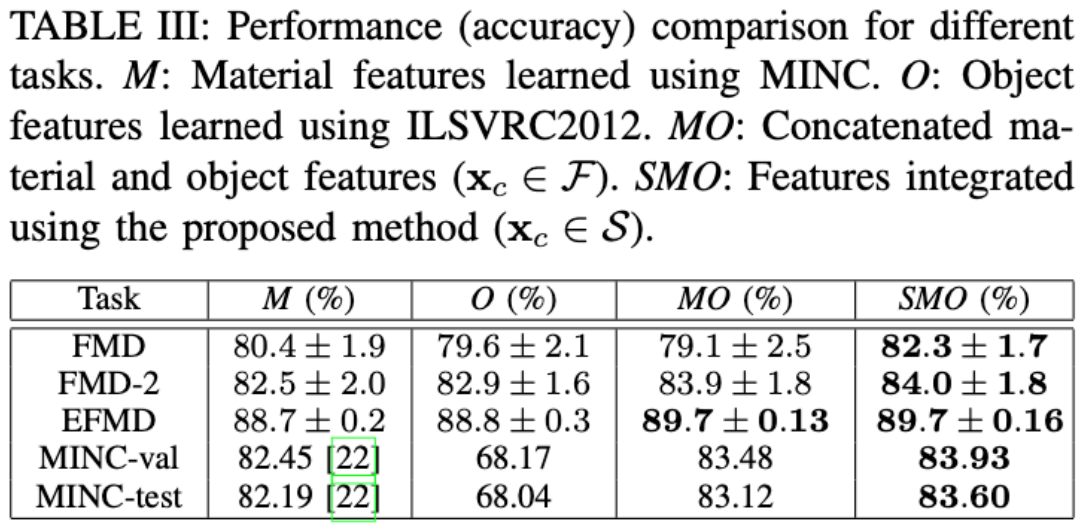


TABLE III: Performance (accuracy) comparison for different tasks. *M*: Material features learned using MINC. *O*: Object features learned using ILSVRC2012. *MO*: Concatenated material and object features ($\mathbf{x}_c \in \mathcal{F}$). *SMO*: Features integrated using the proposed method ($\mathbf{x}_c \in \mathcal{S}$).

Task	*M* (%)	*O* (%)	*MO* (%)	*SMO* (%)
FMD	80.4 ± 1.9	79.6 ± 2.1	79.1 ± 2.5	$\mathbf{82.3 \pm 1.7}$
FMD-2	82.5 ± 2.0	82.9 ± 1.6	83.9 ± 1.8	$\mathbf{84.0 \pm 1.8}$
EFMD	88.7 ± 0.2	88.8 ± 0.3	$\mathbf{89.7 \pm 0.13}$	$\mathbf{89.7 \pm 0.16}$
MINC-val	82.45 [22]	68.17	83.48	**83.93**
MINC-test	82.19 [22]	68.04	83.12	**83.60**

“Integrating Deep Features for Material Recognition”, Zhang et al, ICPR 2016

Selective feature transfer

Selectively transferring features, rather than copying or freezing the parameters of entire layers, is beneficial

We can try to identify features that are “representative” of new tasks, e.g. the subset with the highest set of activation values. The remainder could be reset.

We will revisit selection in greater detail when moving to continual learning

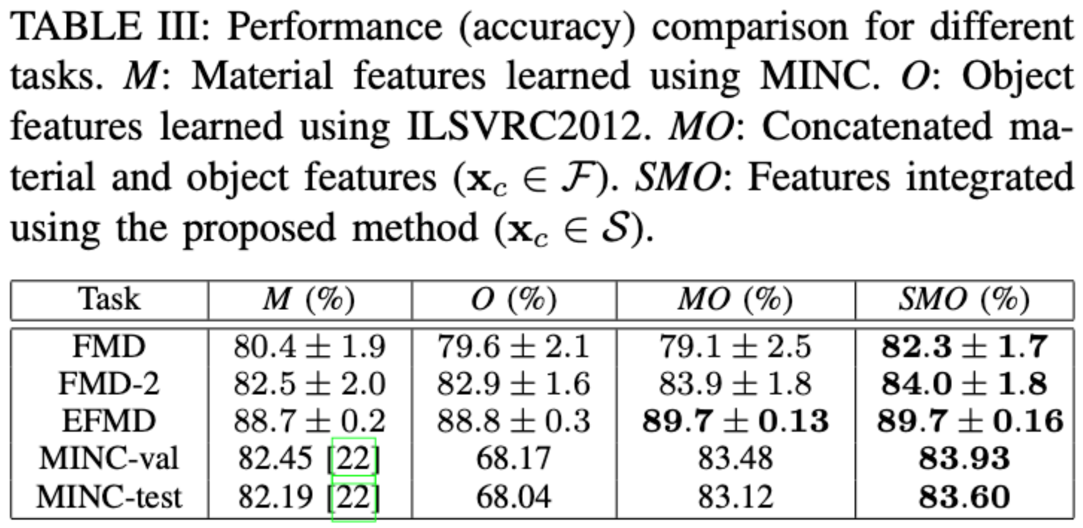


TABLE III: Performance (accuracy) comparison for different tasks. *M*: Material features learned using MINC. *O*: Object features learned using ILSVRC2012. *MO*: Concatenated material and object features ($\mathbf{x}_c \in \mathcal{F}$). *SMO*: Features integrated using the proposed method ($\mathbf{x}_c \in \mathcal{S}$).

Task	*M* (%)	*O* (%)	*MO* (%)	*SMO* (%)
FMD	80.4 ± 1.9	79.6 ± 2.1	79.1 ± 2.5	$\mathbf{82.3 \pm 1.7}$
FMD-2	82.5 ± 2.0	82.9 ± 1.6	83.9 ± 1.8	$\mathbf{84.0 \pm 1.8}$
EFMD	88.7 ± 0.2	88.8 ± 0.3	$\mathbf{89.7 \pm 0.13}$	$\mathbf{89.7 \pm 0.16}$
MINC-val	82.45 [22]	68.17	83.48	**83.93**
MINC-test	82.19 [22]	68.04	83.12	**83.60**

“Integrating Deep Features for Material Recognition”, Zhang et al, ICPR 2016

Universität Bremen

Let us first complete discussing approaches

Instance transfer: re-weigh some labeled data in the source domain for use in the target domain

Feature-representation transfer: find a “good” representation that reduces difference between the source & target domains

Parameter-transfer: discover shared parameters or priors between the source domain and target domain models

Relational-knowledge transfer: build mapping of relational knowledge between the source domain and the target domains

“A Survey on Transfer Learning”, Pan & Yang, IEEE Transactions on Knowledge and Data Engineering 22(10), 2009

Question time

What is (relational) knowledge, in particular in machine learning based models?

Universität Bremen

Knowledge is generally more than model parameters

“We define visual knowledge as any information that can be useful for improving vision tasks such as image understanding and object/scene recognition. One form of visual knowledge would be labeled examples of different categories or labeled segments/boundaries. Another example would be relationships.”

“NEIL: Extracting Visual Knowledge form Web Data”,
X. Chen et al, ICCV 2013

Universität
Bremen

Knowledge is generally more than model parameters

“We define visual knowledge as any information that can be useful for improving vision tasks such as image understanding and object/scene recognition. One form of visual knowledge would be labeled examples of different categories or labeled segments/boundaries. Another example would be relationships.”

Knowledge base consists of labeled examples of objects, scenes, attributes & relationships of four types:

1. Object-object
2. Object-attribute
3. Scene-object
4. Scene-attribute

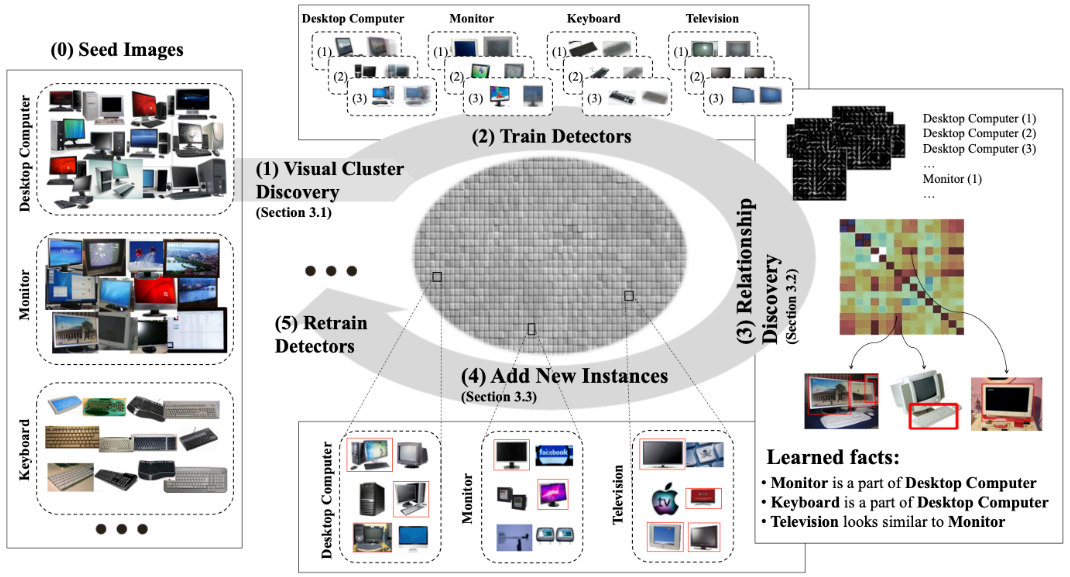
“NEIL: Extracting Visual Knowledge form Web Data”,
X. Chen et al, ICCV 2013

Universität Bremen

Knowledge is generally more than model parameters

Example: Never-ending image learner (NEIL)

- Object categories with bounding boxes
- Labeled examples of scenes
- Examples of attributes
- Visual subclasses of object categories
- Common sense relationships

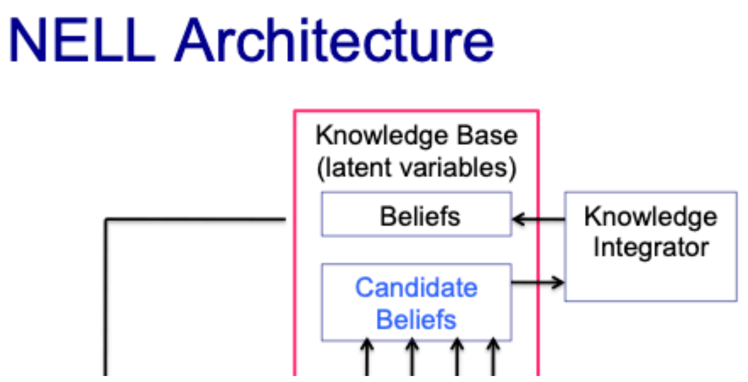

(0) Seed Images
Desktop Computer
Monitor
Keyboard
Television
(1) Visual Cluster Discovery (Section 3.1)
(2) Train Detectors
(3) Relationship Discovery (Section 3.2)
(4) Add New Instances (Section 3.3)
(5) Retrain Detectors
Desktop Computer (1)
Desktop Computer (2)
Desktop Computer (3)
Monitor (1)
Learned facts:
• Monitor is a part of Desktop Computer
• Keyboard is a part of Desktop Computer
• Television looks similar to Monitor

"NEIL: Extracting Visual Knowledge form Web Data", X. Chen et al, ICCV 2013

Knowledge is generally more than model parameters

Example: NELL (language counterpart)
Ran 24/7 from 2010 to 2018

- Relational database
- Facts: “barley is a grain”
- Beliefs: sportUsesEquip(soccer, ball)



“Never-Ending Learning”, T. Mitchell et al, AAAI 2015

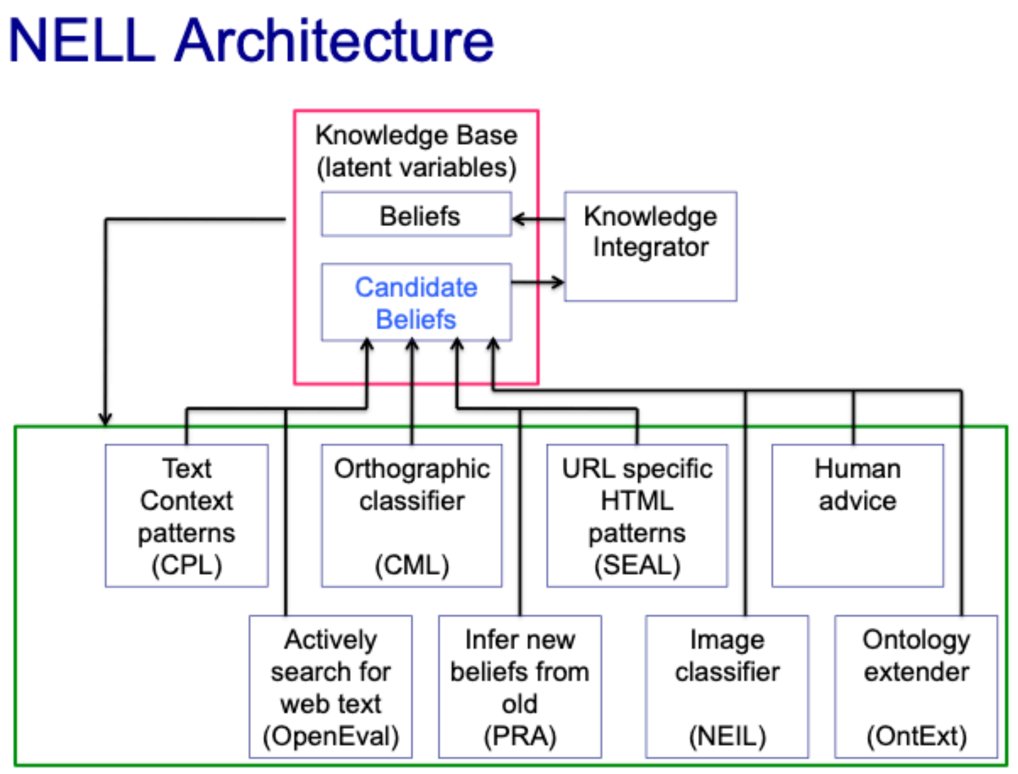
Universität Bremen

Knowledge is generally more than model parameters

Example: NELL (language counterpart)
Ran 24/7 from 2010 to 2018

- Relational database
- Facts: “barley is a grain”
- Beliefs: sportUsesEquip(soccer, ball)

Accumulated over 50 million candidate beliefs by reading the web, using active search, contextual pattern identification, classifiers, ontologies, reasoning etc.

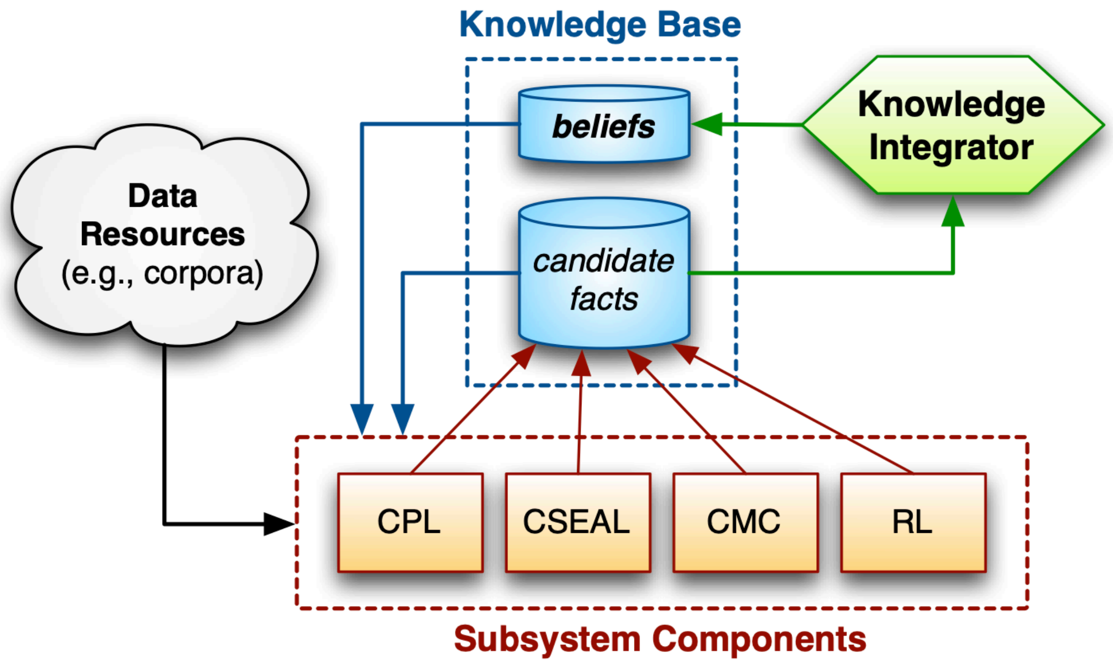

NELL Architecture
Knowledge Base (latent variables)
Beliefs
Candidate Beliefs
Knowledge Integrator
Text Context patterns (CPL)
Orthographic classifier (CML)
URL specific HTML patterns (SEAL)
Human advice
Actively search for web text (OpenEval)
Infer new beliefs from old (PRA)
Image classifier (NEIL)
Ontology extender (OntExt)

“Never-Ending Learning”, T. Mitchell et al, AAAI 2015

Knowledge is generally more than model parameters

NELL consists of (a brief overview):

- Coupled Pattern Learner (CPL)
- Coupled Set Expander for Any Language (CSEAL)
- Coupled Morphological Classifier (CMC)
- Rule Learner (RL)
- Knowledge Integrator (KI)
- + NEIL for images (in the second version)



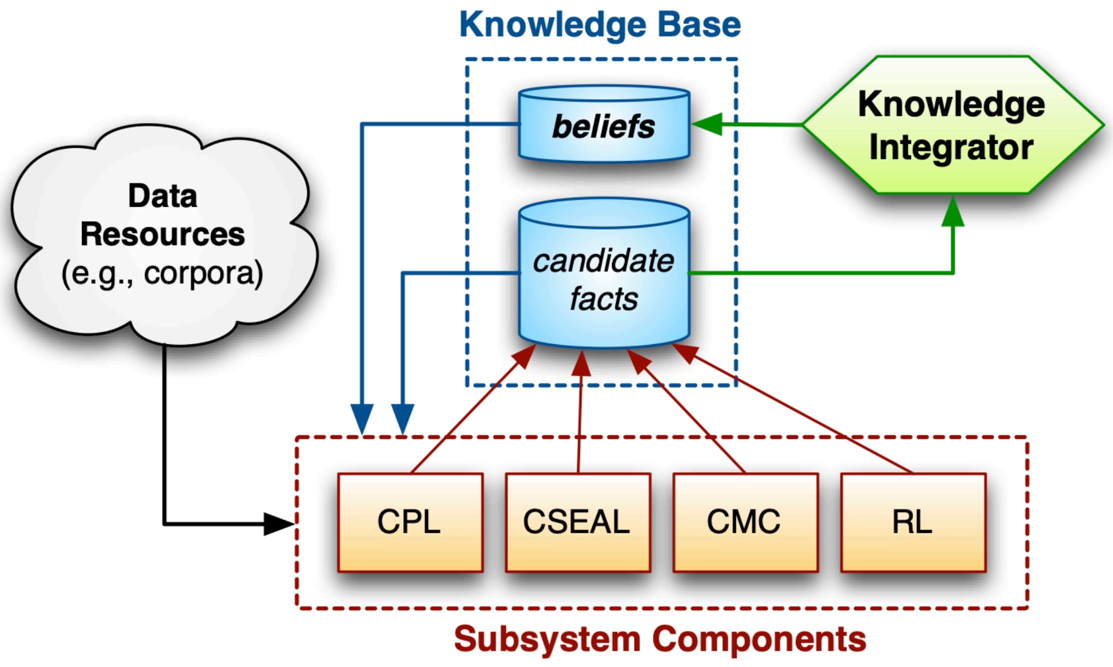
“Never-Ending Learning”, T. Mitchell et al, AAAI 2015

Universität Bremen

Knowledge is generally more than model parameters

Coupled Pattern Learner (CPL):

- Learns <u>contextual patterns</u> like "mayor of X" and "X plays for Y" to extract categories/relations
- Uses <u>co-occurrence statistics</u> between noun-phrases and contextual patterns
- Relationships are used to filter out patterns that are too general



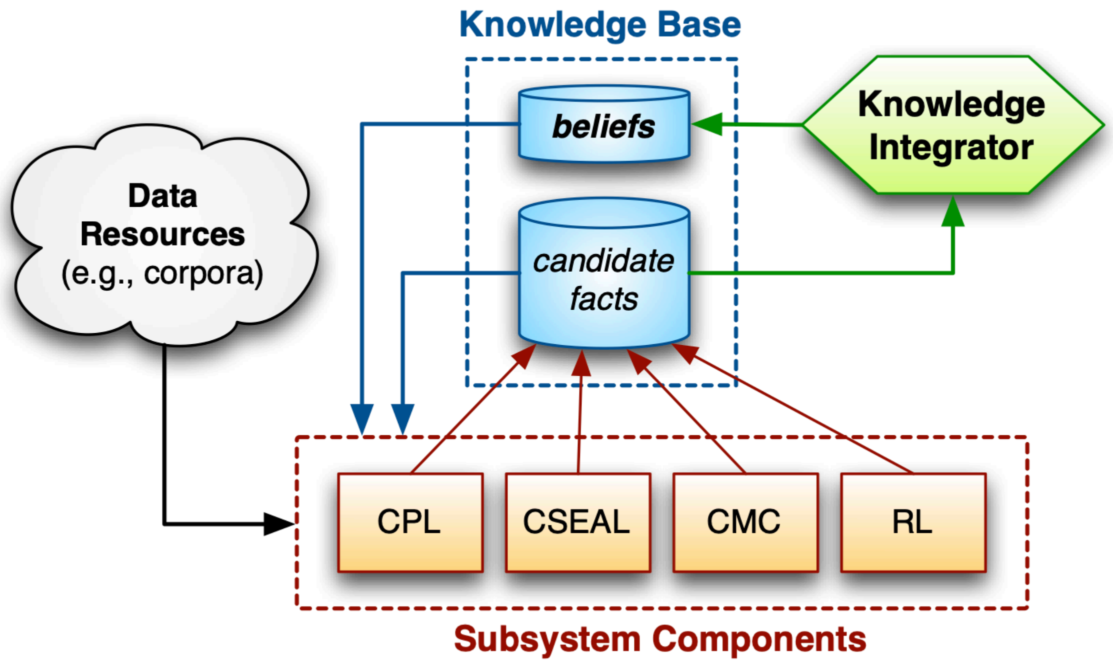
"Never-Ending Learning", T. Mitchell et al, AAAI 2015

Universität Bremen

Knowledge is generally more than model parameters

Coupled Set Expander for Any Language (CSEAL):

- Queries internet with sets of beliefs from categories/relations + mines list & tables to extract novel instances
- Uses mutual exclusion relationships to provide negative examples, used to filter out overly general lists and tables



"Never-Ending Learning", T. Mitchell et al, AAAI 2015

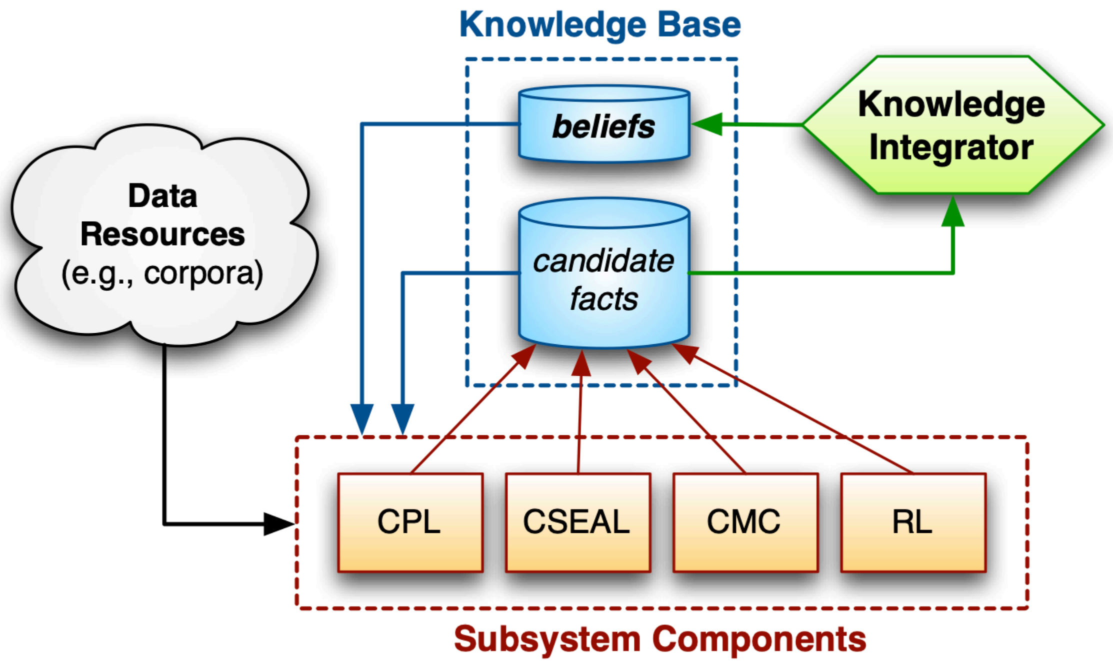
Knowledge is generally more than model parameters

Coupled Morphological Classifier (CMC):

- Set of binary logistic regression models to classify noun phrases based on morphological features (words, affixes, capitalization,…)

Rule Learner (RL):

- First order relational learning to learn probabilistic Horn clauses. Used to infer new relation instances from other relation instances in the KB



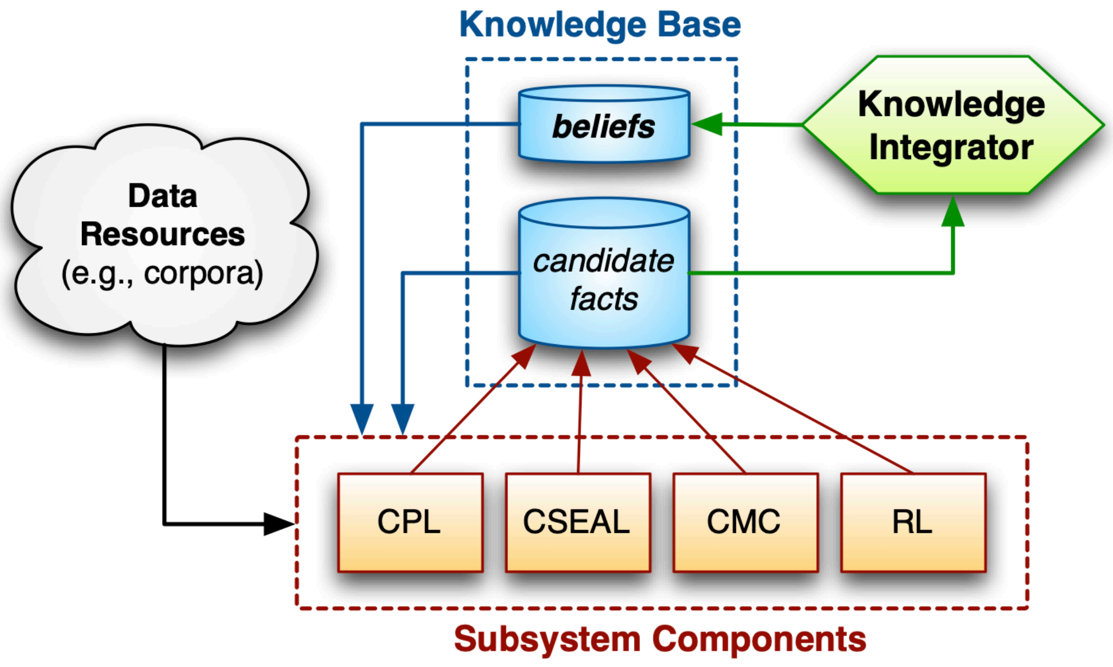
“Never-Ending Learning”, T. Mitchell et al, AAAI 2015

Universität Bremen

Knowledge is generally more than model parameters

Knowledge Integrator (KI) + <u>coupling constraints</u>

- Single source confidence > 0.9
- Moderate confidence if alternate classifiers agree
- Respects mutual exclusion (disjoint categories)
- Subsets/supersets are coupled & Horn clause coupling (learned mappings are consistent)
- Once included, never removed



"Never-Ending Learning", T. Mitchell et al, AAAI 2015

More than transfer - accumulating knowledge!

"We will never truly understand machine or human learning until we can build computer programs that, like people,

- *learn many different types of **knowledge** or **functions**,*
- *from **years of diverse** mostly self-supervised **experience**,*
- *in a **staged curricular** fashion, where **previously learned knowledge enables learning** further types of knowledge,*
- *Where self-reflection and the ability to formulate new representations and new learning tasks enable the learner to **avoid stagnation and performance plateaus**."*

Quote from "Never-Ending Learning", T. Mitchell et al, AAAI 2015

Early Definition

Definition - ~~Lifelong Machine Learning~~ - Thrun 1996:

“*The system has performed N tasks. When faced with the (N+1)th task, it uses the knowledge gained from the N tasks to help the (N+1)th task.*”

“Is Learning The n-th Thing Any Easier Than Learning the First?” (NeurIPS 1996) & “Explanation based Neural Network Learning A Lifelong Learning Approach”, Springer US, 1996

- The definition we’ve seen was actually called “lifelong learning”
- In practice, it corresponds to a narrow one for transfer learning

Universität Bremen

Early Definition

Definition - ~~Lifelong Machine Learning~~ - Thrun 1996:

“*The system has <u>performed N</u> tasks. When faced with the <u>(N+1)th</u> task, it uses the <u>knowledge</u> gained from the N tasks to <u>help</u> the (<u>N+1)th task</u>.*”

“Is Learning The n-th Thing Any Easier Than Learning the First?” (NeurIPS 1996) & “Explanation based Neural Network Learning A Lifelong Learning Approach”, Springer US, 1996

- The definition we’ve seen was actually called “lifelong learning”
- In practice, it corresponds to a narrow one for transfer learning
- It discounts what happens to the first N tasks (as is typical of transfer learning). However, there is also <u>positive/negative backward transfer</u>, which is critical for <u>continual learning</u>!

Early Definition

This concludes "learning in the present" - let's learn to deal with the past next!

Definition - ~~Lifelong Machine Learning~~ - Thrun 1996:

"*The system has <u>performed N</u> tasks. When faced with the <u>(N+1)th</u> task, it uses the <u>knowledge</u> gained from the N tasks to <u>help</u> the (<u>N+1)th task</u>.*"

"Is Learning The n-th Thing Any Easier Than Learning the First?" (NeurIPS 1996) & "Explanation based Neural Network Learning A Lifelong Learning Approach", Springer US, 1996

- The definition we've seen was actually called "lifelong learning"
- In practice, it corresponds to a narrow one for transfer learning
- It discounts what happens to the first N tasks (as is typical of transfer learning). However, there is also <u>positive/negative backward transfer</u>, which is critical for <u>continual learning</u>!