From Machine Learning to Machine Intelligence

LÉON BOTTOU

META AI RESEARCH & NEW YORK UNIVERSITY

Conference on Data Science, Intelligence & Society

Context

THE STATISTICAL PROBLEM IS ONLY A PROXY

Machine Learning today

- An established academic discipline
  - that attracts bright students.
- Pervasive applications
  - all over the Internet
  - and sometimes in the real world.
- Massive investments.
- Massive returns.

And more…

Artificial intelligence
~~Machine Learning~~ today

- An established academic discipline
  - that attracts bright students.
- Pervasive applications
  - all over the Internet
  - and sometimes in the real world.
- Massive investments.
- Massive returns.

And more…

Artificial intelligence
~~Machine Learning~~ today

- An established academic di
  - that attracts bright
- Pervasive a
  - all
  - a
- Mass
- Massive

Ambitions→Promises→Investment→Expectations

Yandex

Google

Baidu 百度

Microsoft

amazon

And more…

Statistical machine learning

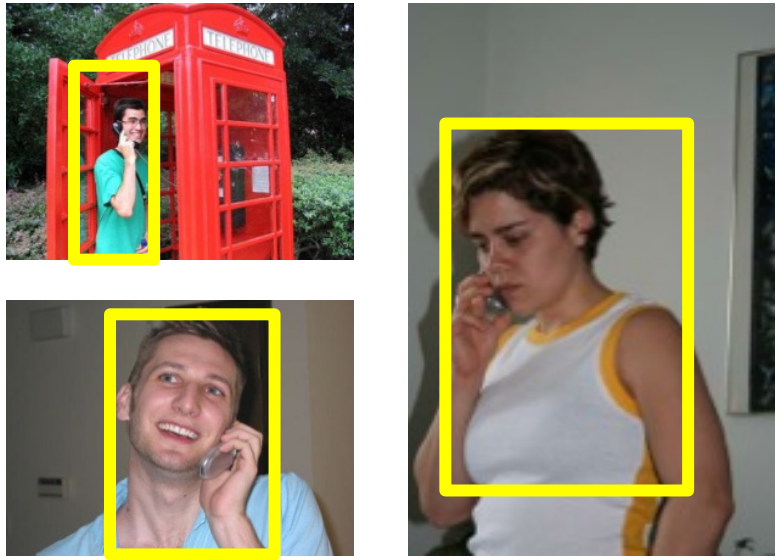
Absent a formal specification of what makes an image represent a mouse or a piece of cheese, we must

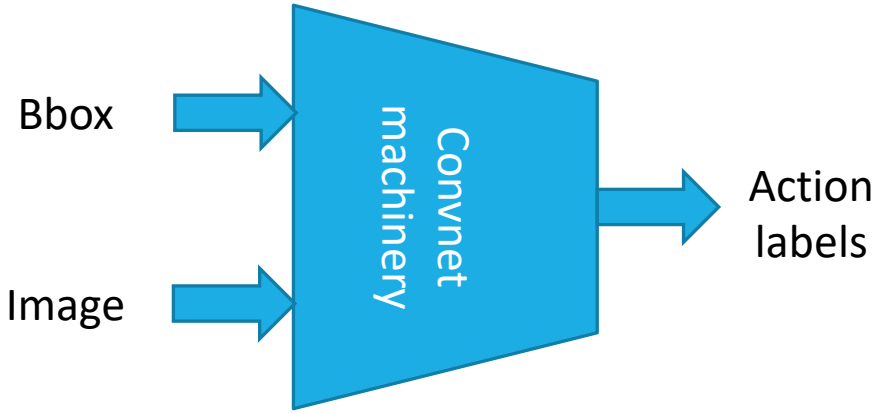
- either formulate heuristic specifications,
  and write a program that targets them.

- or rely on data, formulate a statistical proxy problem,
  and use a learning algorithm.

Observation 1
The statistical problem is only a proxy

Example: detection of the action "*giving a phone call*"





(Oquab et al., CVPR 2014)
~70% correct (SOTA in 2014)

Observation 1
The statistical problem is only a proxy

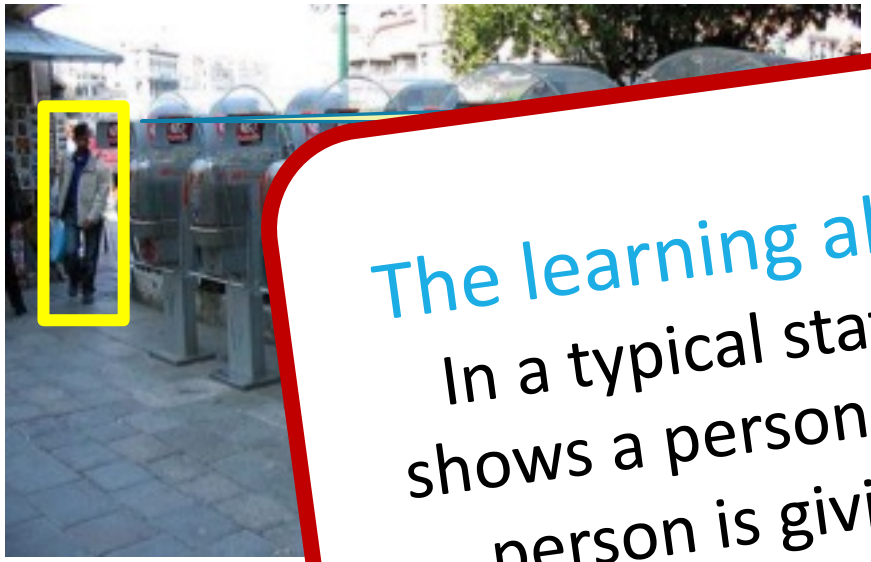
Example: detection of the action "*giving a phone call*"

Not giving a phone call.

Giving a phone call ????

Observation 1
The statistical problem is only a proxy

Example: detection of the action "*giving a phone call*"

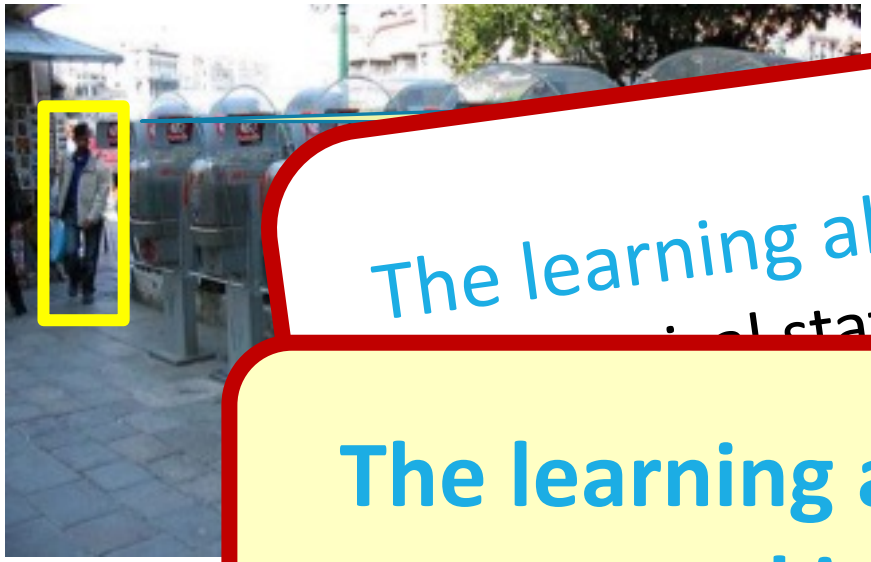


The learning algorithm is statistically correct!

In a typical static image dataset, when an image shows a person near a phone, chances are that the person is giving a phone call (a selection bias.)

Observation 1
The statistical problem is only a proxy

Example: detection of the action *"giving a phone call"*



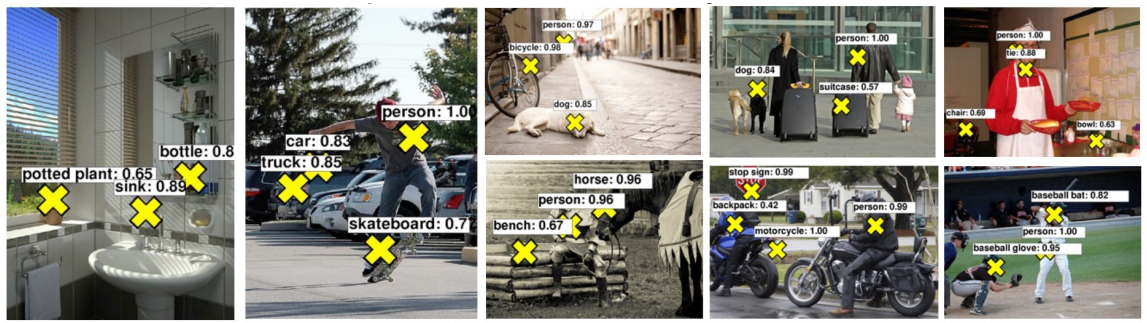
The learning algorithm is statistically correct!

static image dataset, when an image

The learning algorithm is statistically correct and is also missing the point!

Observation 2:
Statistical guarantees are too weak

"DeepVisotron™,℠,® detects 10000 object categories with less than 1% errors."



What is the nature of this statement?

- This does not mean that one rolls a dice for each picture.
- This statement refers to a specific testing set.
  The error guarantee is lost if the image distribution changes.

Observation 2: Statistical guarantees are too weak

- A smart programmer reasons that an inventive use of a sorting routine can solve a problem.
- The sorting routine fulfils its contract by sorting the data (however strange.)
- The code of the smart programmer does what was intended.

- A smart programmer reasons that an inventive use of a trained object recognizer can solve a problem.
- The object recognizer receives data that does not resemble the testing data and outputs nonsense.
- The code of the smart programmer does not work.

Observation 2: Statistical guarantees are too weak

- A smart programmer reasons that an inventive use of a sorting routine can solve a problem.
- The sorting routine fulfils its contract by sorting the data (however strange.)
- The code of the smart programmer does what was intended.

- A smart programmer reasons that an inventive use of a trained object recognizer can solve a problem.
- The object recognizer receives data that does not resemble the testing data and outputs nonsense.
- The code of the smart programmer does not work.

Collecting new data and retraining may help … or may not!

Observation 2: Statistical guarantees are too weak

- A smart programmer reasons that an inventive use of a sorting routine can solve a problem.
- The sorting routine fulfils its contract by sorting the data (however strange.)
- The code of the smart programmer does what was intended.

- A smart programmer reasons that an inventive use of a trained object recognizer can solve a problem.
- The object recognizer receives data that does not resemble the testing

A ⇒ B

does not ensure that

***almost* A ⇒ *almost* B**

retraining may help … or may not!

Further observations

- Adversarial examples
- Fairness questions in machine learning
- Spurious correlations
- Shortcut learning

We need to learn relations that are less tied to a particular data distribution.

Substantial departure from the statistical machine learning framework.

(Szegedy et al. 2014) (Kleinberg et al. 2018) (Geirhos et al. 2020) and many others

Out-of-distribution generalization

WHAT THEN CONNECTS THE TRAINING AND TESTING DATA?

Defining “out of distribution”

Testing data not distributed like the training data?

A very poor definition of a recurrent problem.

The real questions:

- Why do we expect a machine learning system to work?
- What distribution changes are acceptable?
- What unites training data and testing data?

Distributionally robust optimization

Inspired by robust statistics:

- Define a set $\mathcal{B}$ of distributions "close enough" to the training distribution
- Optimize the expected loss for the worst distribution in $\mathcal{B}$

$$\min_{w} \max_{p \in B} \mathbb{E}_{(x,y)\sim p}[\ell(x, y, w)]$$

- Question 1: How to express "close enough" ?
- Question 2: Does it matter ?

(Huber, 1981) (Bagnell. 2005) and many others

f-divergence balls

The Kullback-Leibler divergence is a common measure of distribution closeness, often considered attractive because it relates to information theory. This is a member of the broader family of f-divergences.

Let $\mathcal{B}$ be a f-divergences ball centered on the training distribution p_T

This this idea hits all the buzzwords, but (theorem) …

For a classification loss, solving the DRO problem

$$\min_{w} \max_{p \in B} \mathbb{E}_{(x,y)\sim p}[\ell(x, y, w)]$$

yields the same solution as the ordinary ERM problem

$$\min_{w} \mathbb{E}_{(x,y)\sim p_T}[\ell(x, y, w)]$$

(Hu et al., 2018)

Wasserstein balls

Using Wasserstein distance balls has proven more successful, with applications in finance and in making deep learning classifier more robust against adversarial examples.

Let $\mathcal{B}$ be a Wasserstein ball centered on the training distribution p_T …

Why is it different?

What does a Wasserstein ball look like anyway?

(Pflug et al., 2007) (Staib et al., 2017) and many others

Constructing a Wasserstein ball

- Let (x, y) be the training examples following distribution p_T
- Let $z = \varphi(x, \varepsilon_z)$ be patterns perturbed by an adversary

Perturbation function

Random noise

- Let p_φ represent the distribution of (z, y) for a given φ.

φ belongs to the set of all functions such that $\mathbb{E}[x - z] \leq M$

$\Leftrightarrow$ p_φ belongs to a Wasserstein ball centered on p_I.

(Bottou, in preparation)

Structural equation model (SEM)

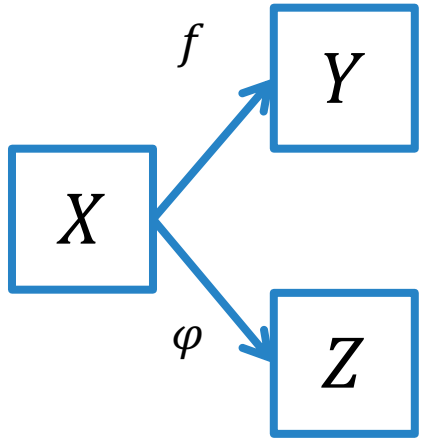
The Wasserstein ball $\mathcal{B}$ represents in fact the family of distributions generated by manipulating a causal model of the data:

$$x = g(\varepsilon_x)$$
$$z = \varphi(x, \varepsilon_z)$$
$$y = f(x, \varepsilon_y)$$

Genuine pattern distribution

Pattern perturbation

Classification function of interest



Some people like to think of SEMs as DAGs!

(Wright 1921)

Why do distribution change?

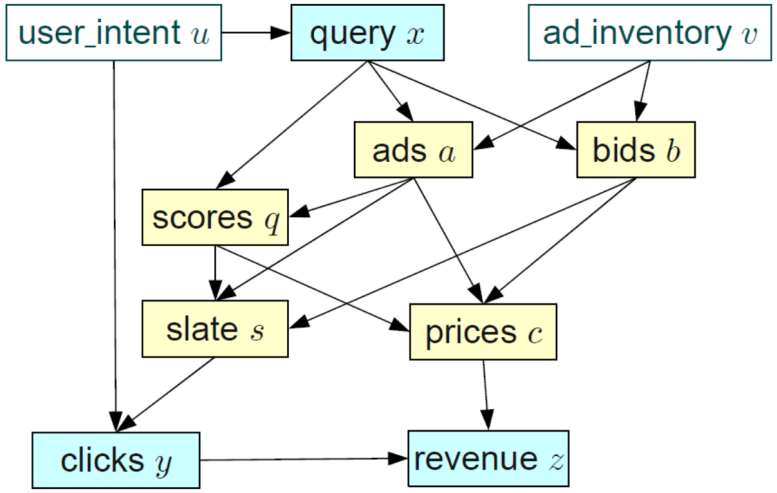
- Any learning process assumes a connection between training and testing data.

$\Rightarrow$ We must say how training and testing distributions are related.

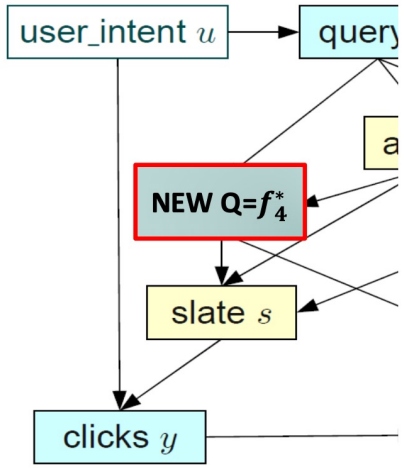
- An obvious answer in retrospect

Distributions change because the underlying causal process is perturbed for some reasons (a change of context, an adversary, …)

Digression
Causal inference, Pearl style.



$$
\begin{aligned}
x &= f_1(u, \varepsilon_1) \\
a &= f_2(x, v, \varepsilon_2) \\
b &= f_3(x, v, \varepsilon_3) \\
q &= \cancel{f_4(x, a, \varepsilon_4)} \quad f_4^*(x, a, \varepsilon_4) \\
s &= f_5(a, q, b, \varepsilon_5) \\
c &= f_6(a, q, b, \varepsilon_6) \\
y &= f_7(s, u, \varepsilon_7) \\
z &= f_8(y, c, \varepsilon_8)
\end{aligned}
$$



Given observed conditional distribution on this model, …

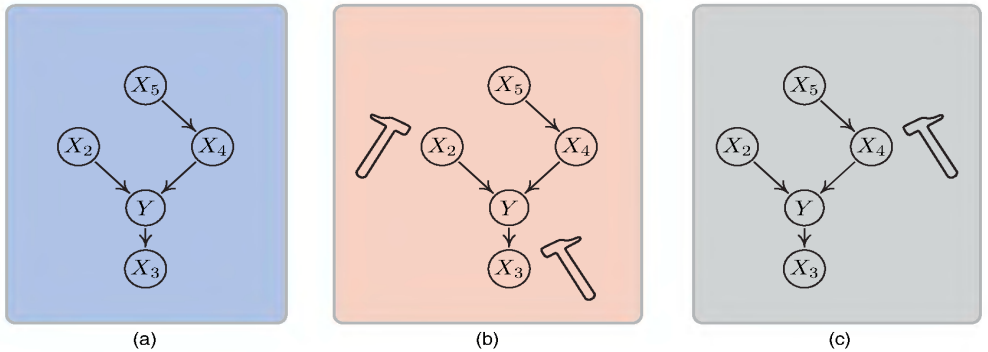
… can we predict conditional distributions on that model ?

(Pearl, 2009) (Bottou, Peters, et al. 2013)

Why do distribution change?

- An obvious answer in retrospect

  Distributions change because the underlying causal process is perturbed for some reasons (a change of context, an adversary, …)



Perturbations of simple causal models give Wasserstein balls. Perturbations of more complex causal models can give much more complicated $\mathcal{B}$.

(Peters, Bühlman, Meinshausen, 2016)

Questioning our goals…

" Is it true that we merely want to predict a predefined variable of interest Y from predefined variables Z in a manner that is robust under certain distribution changes?

We have just argued that, in order to solve such a problem, we need to understand the causal structure of our data. Knowing the causal structure of our data allows us to answer much more refined questions.

For instance, we might be able to construct a different prediction problem that serves our ultimate purpose equally well but is much less affected by distribution changes."

(Bottou, 2022, in preparation)

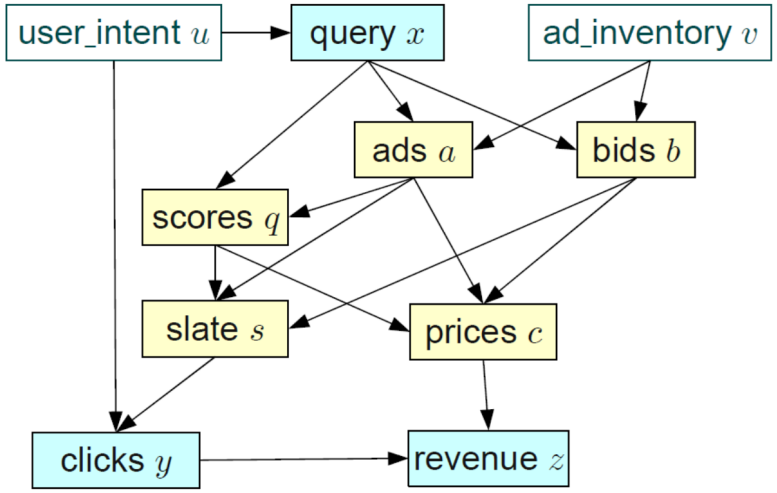
3 Causal invariance

Martin Arjovsky

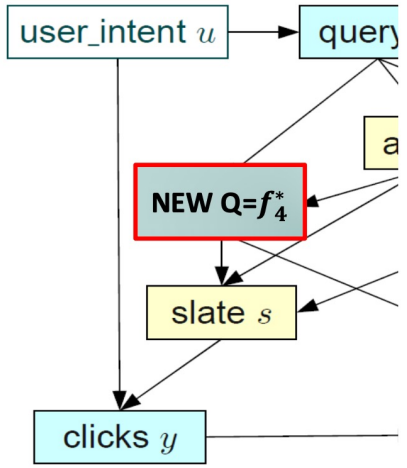
Ishaan Gulrajani

David Lopez-Paz

Causal inference with DAGs



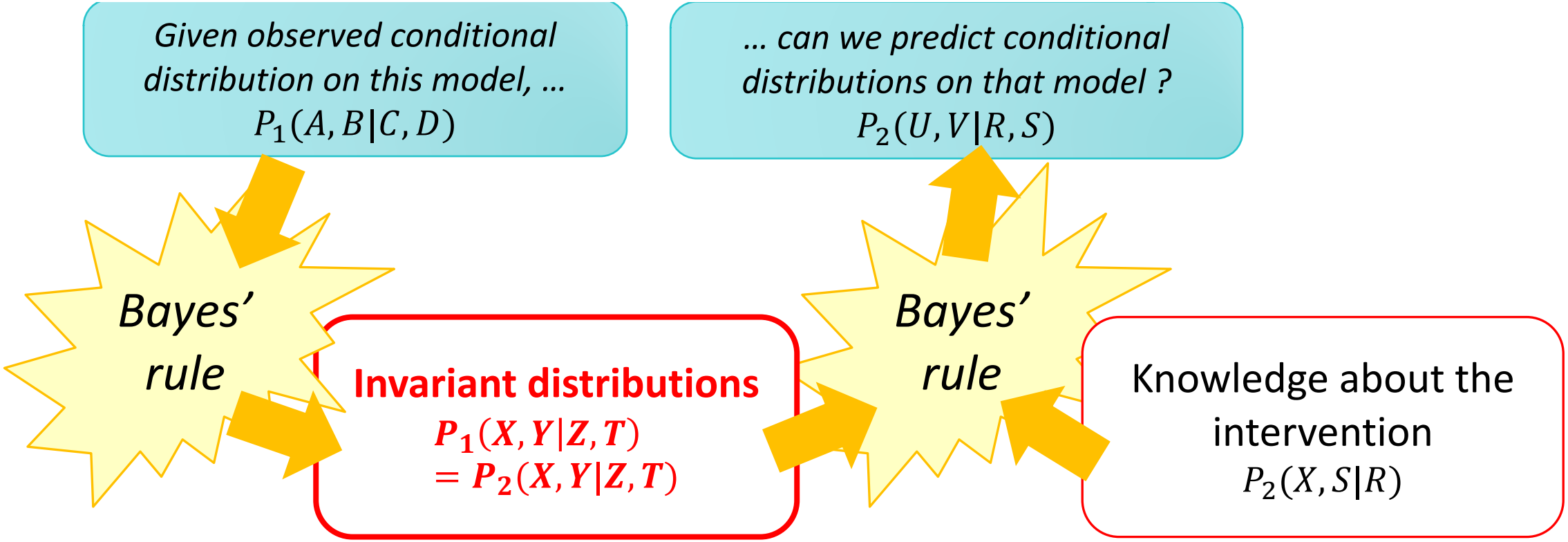
$$
\begin{aligned}
x &= f_1(u, \varepsilon_1) \\
a &= f_2(x, v, \varepsilon_2) \\
b &= f_3(x, v, \varepsilon_3) \\
q &= \cancel{f_4(x, a, \varepsilon_4)} \quad f_4^*(x, a, \varepsilon_4) \\
s &= f_5(a, q, b, \varepsilon_5) \\
c &= f_6(a, q, b, \varepsilon_6) \\
y &= f_7(s, u, \varepsilon_7) \\
z &= f_8(y, c, \varepsilon_8)
\end{aligned}
$$



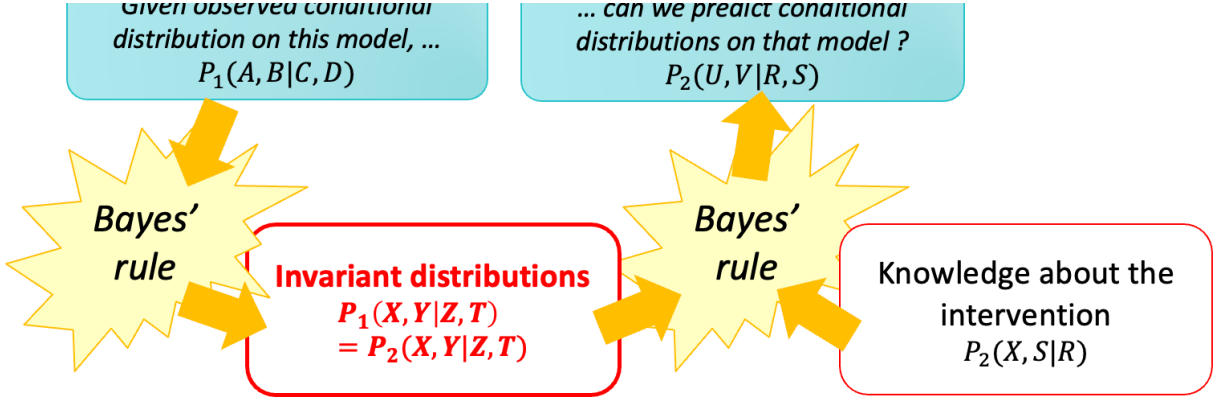
Given observed conditional distribution on this model, …

… can we predict conditional distributions on that model ?

Causal inference with DAGs

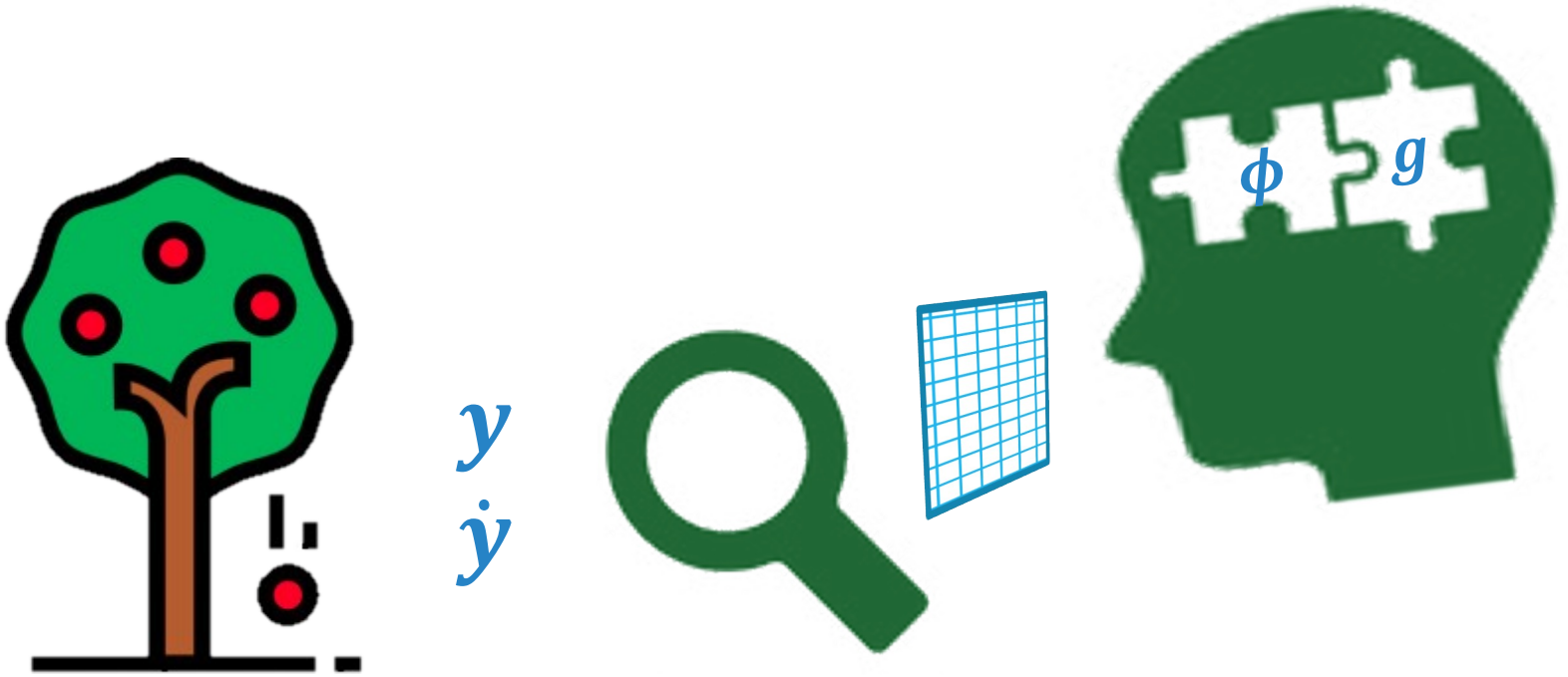


Invariance is the key



- This process is formalized by Pearl's *do-calculus*.
  The *assumed graph structure* informs us about the invariant conditionals
- Rubin's *methods of potential outcomes* is less formal, yet
  "ignorability" assumptions inform us about the invariant conditionals

DAG issues : (1) the variables



DAG issues : (2) the structure

- Even if we know what the variables are …
- Even if we observe all relevant variables …
- Even if all causal edges have a trace in the joint distribution (*"faithfulness"*) …

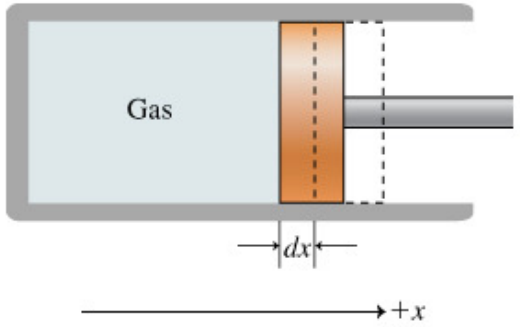
Reconstructing a causal graph is a messy business.

- Multiple DAGs are equivalent (strong) or compatible with the data (weak)
- Each reconstruction decision depends on statistical tests that may fail.
- Reconstructing a useful graph is often an all-or-nothing affair.

(Spirtes, Glymour, et al., 2000) (Chickering, 2003) (review: Glymour et al., 2019)

DAG issues : (3) directionless causation

$$PV = nRT$$



Does P cause V and T? Or does T cause V and P? Or …

- If we push a piston, we change V and cause a change in P and T.
- If we heat the mixture, we change T and possibly cause changes in P and V.

The direction of the arrow depends on the intervention

(von Wright, 1971, 1974) (Dawid 2010)

How do we solve causal inference problems in physics

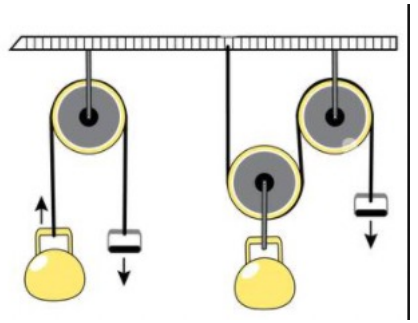
Method 1: ODE -> DAG -> invariants

- Numerical integration scheme defines a DAG
- Must predict all the trajectory to determine the final state.

$$m\ddot{x} = f(x, \dot{x})$$

Method 2: Use invariants directly

- Write equations that describe the intervention
- Write equations that describe invariants (local or universal)
- Solve!



How do we solve causal inference problems in physics

Method 1: ODE -> DAG -> invariants

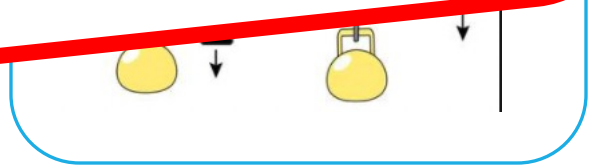
- Numerical i
- M

Met

- Wr
- Writ
- Solv

Invariance ⇔ Causation
(Cartwright 2003) (Woodward 2005)

Can we discover invariances more easily than DAGs?



Multiple environments

Following Peters et al. (2016), we consider that data from each environment e comes with a different distribution P_e.

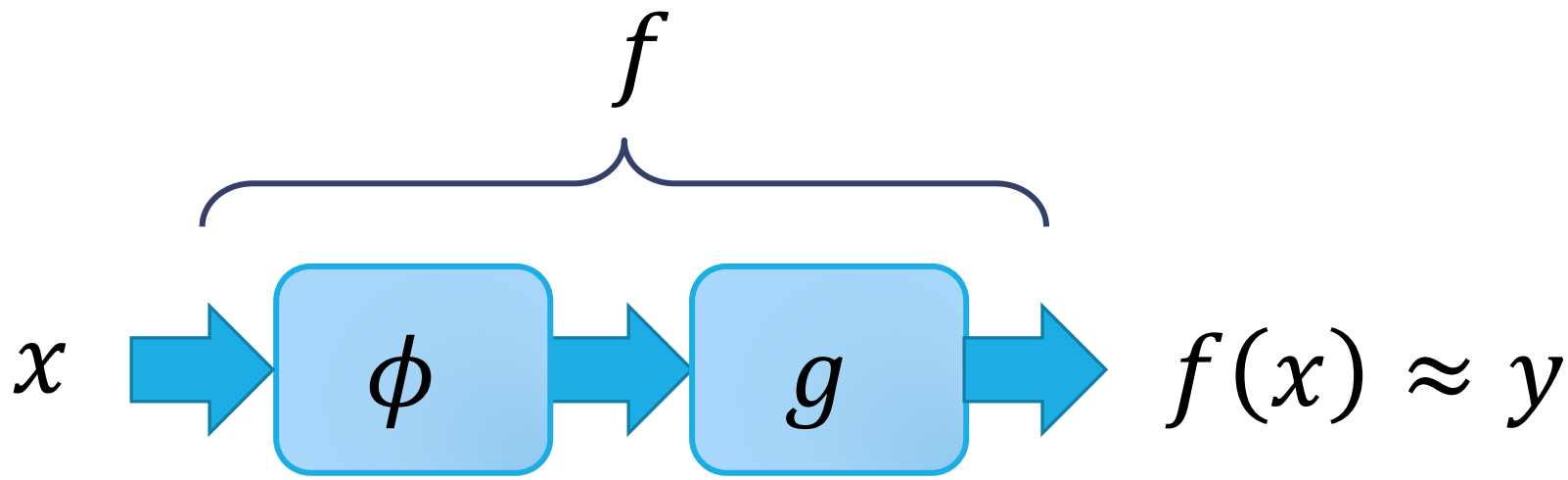
$$P_e = P(X_e, Y_e) \text{ for } e = 1,2,3 \ldots$$

- Training sets $D_e = \{(x_i^e, y_i^e) \sim P_e\}$ are provided for some e.
- We want a predictor $f(x) \approx y$ that works for many e.

Intuition

"If we find a representation in which
all falling objects obey the same laws,
then we possibly understand something useful."

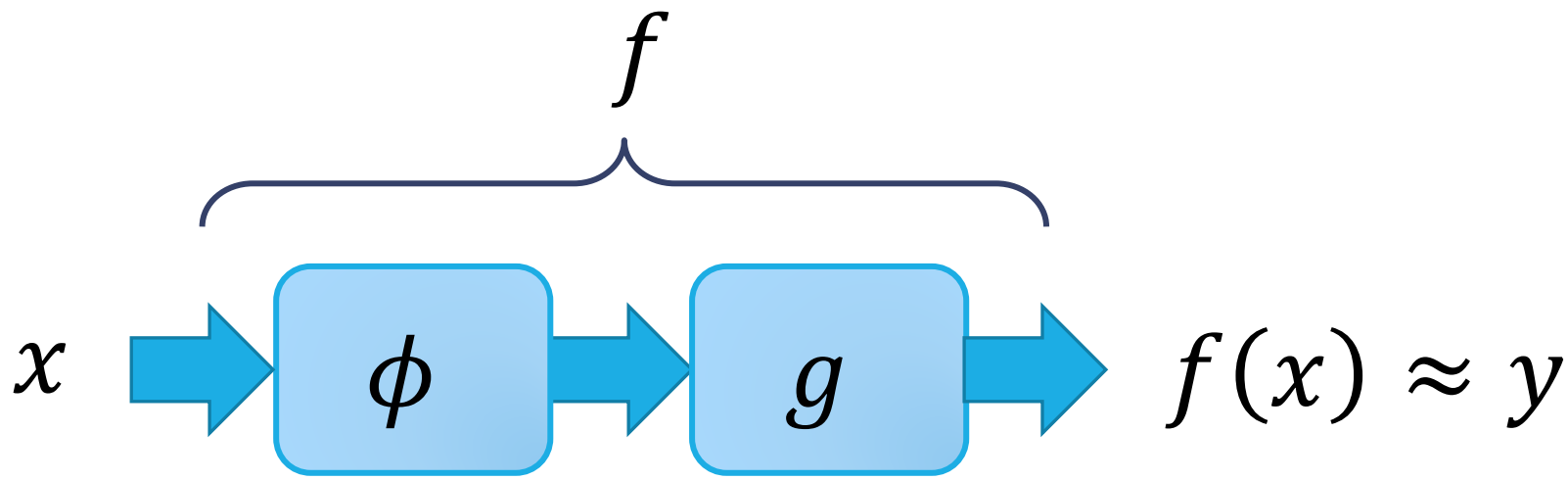
Invariant representation



Find a representation $\phi(x)$

Such that the regression from $\phi(x)$ to y is invariant across environments

Invariant representation



Find a representation $\phi(x)$

Such that the regression f
is invariant
ents

And also has low error…

Invariant risk minimization (IRM)

Minimize a regularized cost

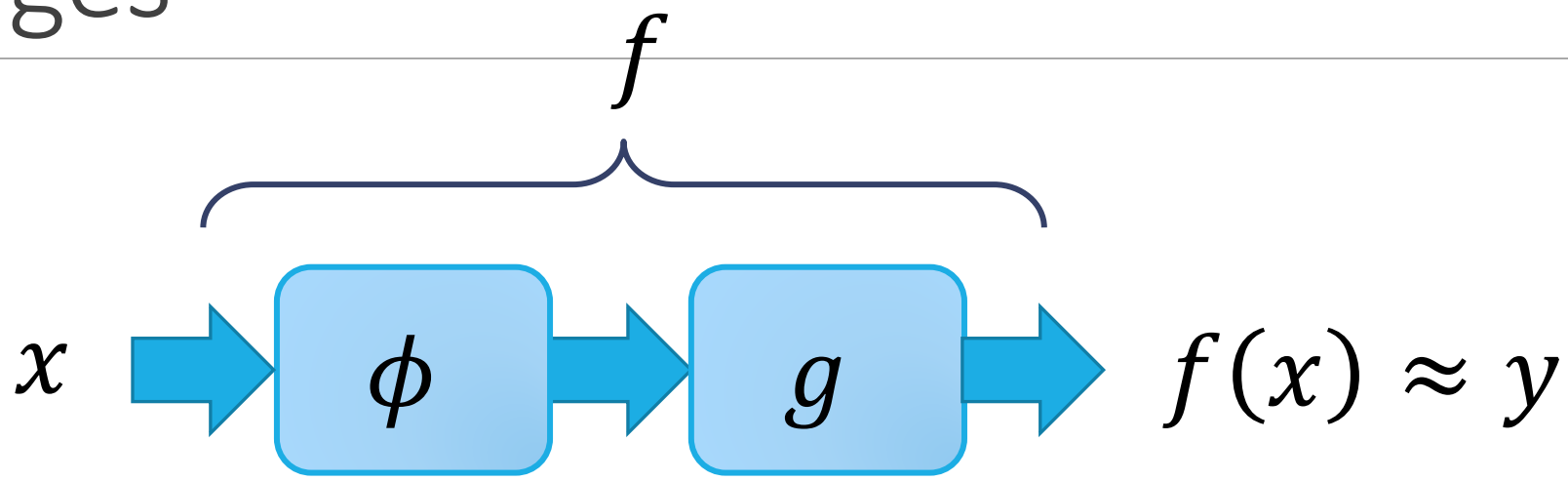
Average error across all environments.

Penalty that favors invariant solutions

$$\overbrace{\sum_e \frac{1}{n_e} \|Y_e - f_w(X_e)\|^2}^{C_e(w)} + \kappa \sum_e \Omega_e(w)$$

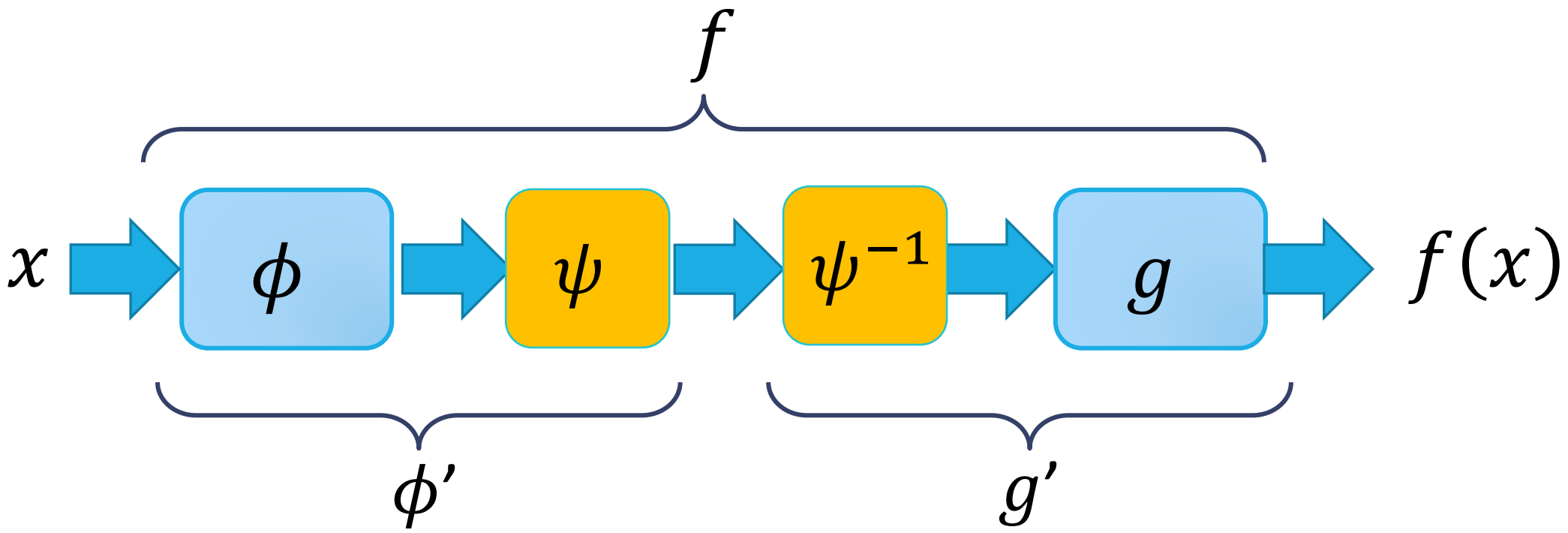
What's in this term?

Challenges

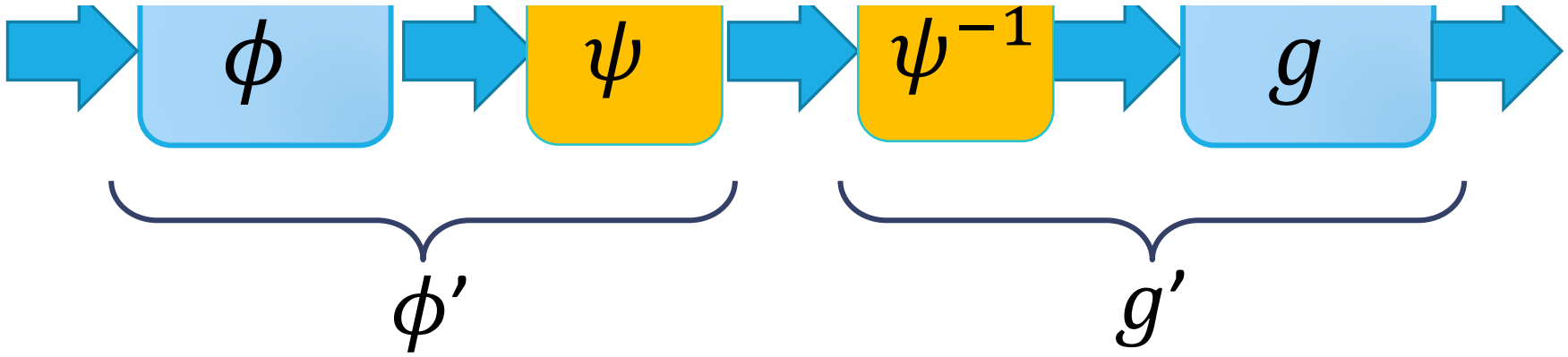


- Suppose x can be split as $(x_{relevant}, x_{spurious})$
- Invariance is achieved when ϕ suppresses $x_{spurious}$.
- Noncontinuity : If ϕ does not fully suppress $x_{spurious}$, then g can restore it.

Equivalence classes

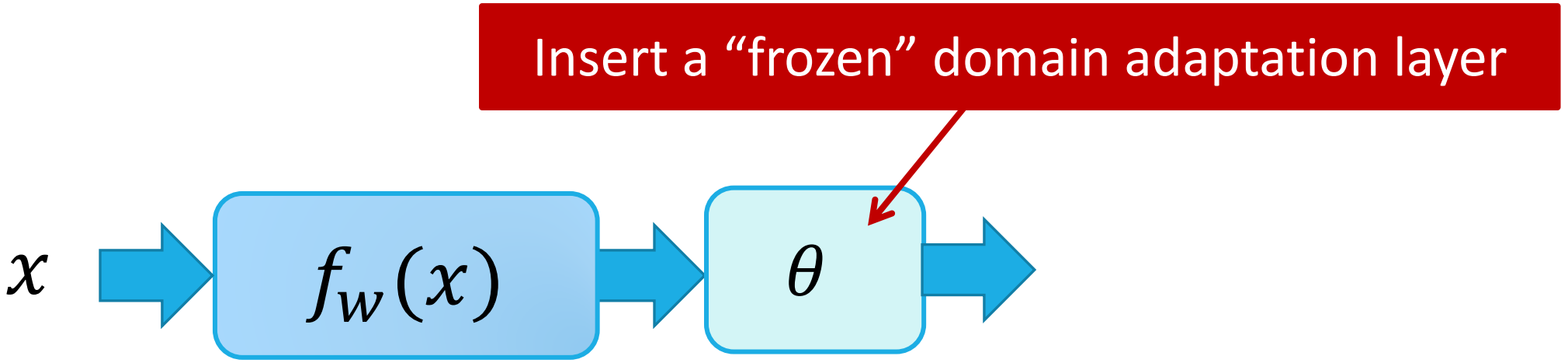


Simplifying our problem



If g' can be anything we want,
then we can choose it a priori (e.g., the identity)
and we do not need to optimize its parameters.

IRM v1



$$\Omega_e(w) = \left(\frac{\partial C_e}{\partial \theta} \Big|_{\theta=1} \right)^2$$

Regularization favors weights w such that no environments would benefit from $\theta \neq 1$.

(Arjovsky, Bottou, Gulrajani, Lopez-Paz, 2019)

Colored MNIST

Digits with misleading colors

	Y=0	Y=1
{0,1,2,3,4}	0.75	0.25
{5,6,7,8,9}	0.25	0.75

The optimal classification rate on the basis of the shape only is 75%.

Random guess is 50%.

	Red	Green
Y=0	$1-e$	e
Y=1	e	$1-e$

During the training $e \in \{0.1, 0.2\}$. The color is a better indicator than the shape, but not a stable one.

Then we test with $e = 0.9$.

(Arjovsky, Bottou, Gulrajani, Lopez-Paz, 2019)

Colored MNIST

Training with $e \in \{0.1, 0.2\}$	Testing with $e \in \{0.1, 0.2\}$	Testing with $e = 0.9$
Minimize empirical risk after mixing data from both environments	84.3%	10.1%
Minimize empirical risk with invariant regularization	70.8%	66.9%

- Network is a MLP with 256 hidden units on 14x14 images.
- Invariant regularization tuned high : regularization term is nearly zero.

Follow up work

Alternate algorithms and constraints

- IGA (Koyama et al., 2020); Invariant Games (Ahuja et al. 2020);
- vREX (Krueger at al. 2020); FISH (Shi et al., 2021); FISHr (Rame et al, 2021);
- SD; RSC, LfF, CLOvE, MAML-IRM, …

- A couple applications to real data
  - Wildcam dataset (https://ff13.fastforwardlabs.com)
  - Toxic language classification (Adragna et al., 2020)
- Benchmarking studies
  - Domainbed (Gulrajani et al., 2020)

Painful lesson

Trying invariant learning on real OOD problems:

- Invariant learning sometimes yields a small improvement.
- But these results do not measure with our hopes…
- And always come after a finicky optimization…

4 Invariance, optimization, and features

Jianyu Zhang

David Lopez-Paz

(Zhang, Bottou et al., arXiv 2022)

Algorithms for OoD learning

- IRMv1 (Arjovsky et al., 2018)
- IGA (Koyama et al., 2020); Invariant Games (Ahuja et al. 2020);
- vREX (Krueger at al. 2020); FISH (Shi et al., 2021); FISHr (Ramé et al, 2021);
- SD, RSC, LfF, CLOvE, MAML-IRM, …

All these algorithms work by defining rather complicated penalty terms $\Omega_e(w)$.

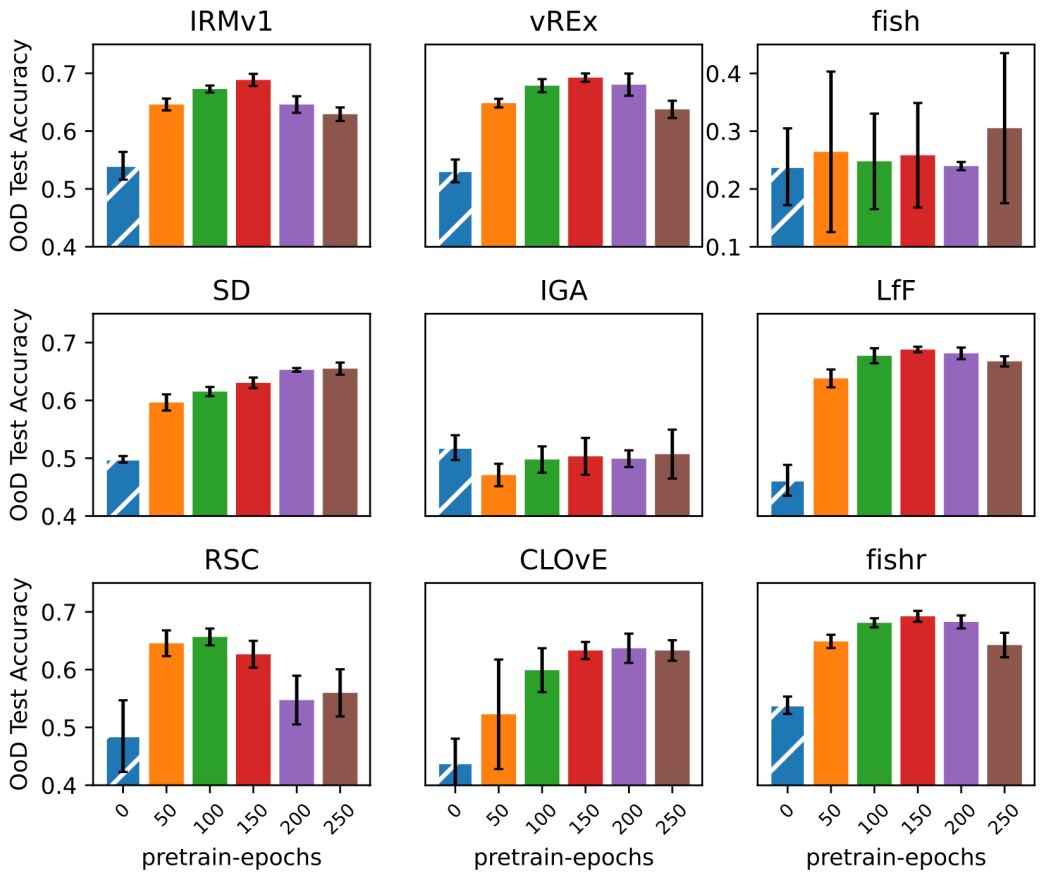
It is usually necessary to pre-train without the penalty ($\kappa = 0$), then continue with with a well-chosen weight κ.

$$\overbrace{\sum_e \frac{1}{n_e} \|Y_e - f_w(X_e)\|^2}^{C_e(w)} + \kappa \sum_e \Omega_e(w)$$

Optimization troubles

Using the final κ recommended by the authors, how does the OoD performance depends on the number of pre-training iterations ($\kappa = 0$) ?

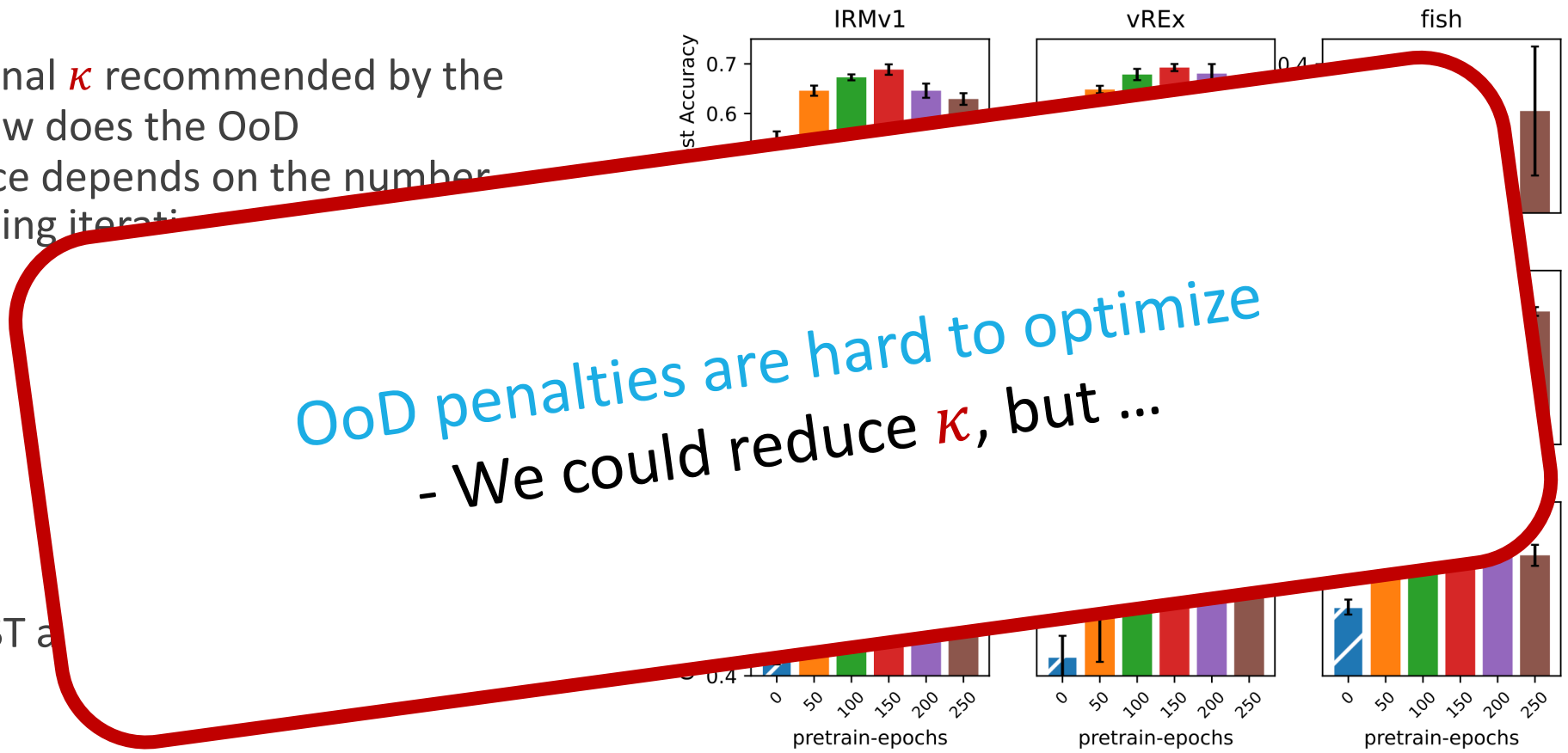
(ColorMNIST again)



Optimization troubles

Using the final κ recommended by the authors, how does the OoD performance depends on the number of pre-training iterati

(ColorMNIST a

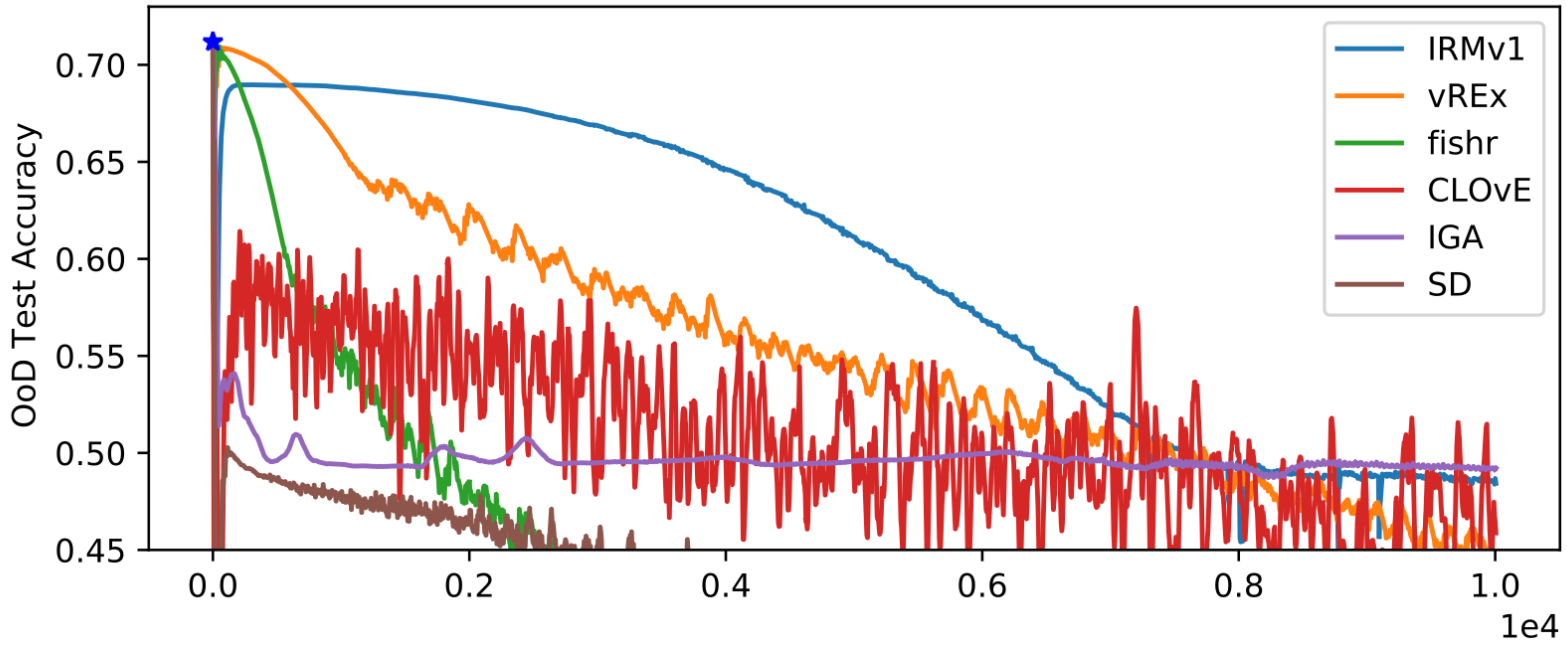


OoD penalties are hard to optimize

- We could reduce κ, but …

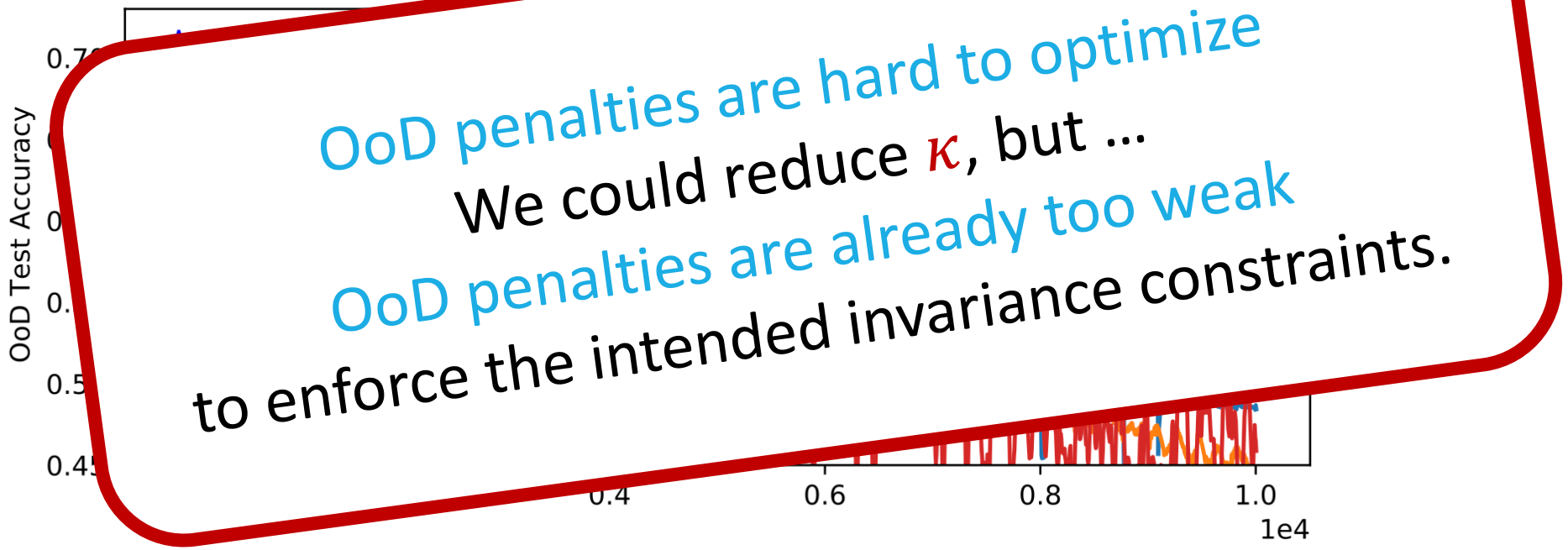
Optimization troubles

- Instead of pre-training, what if we initialize the network with a correct solution?
  (in the case of ColorMnist, with weights that only depend on the shape, not the color…)



Optimization troubles

- Instead of pre-training, what if we initialize the network with the correct solution?
  (in the case of ColorMnist, with weights that only depend on



OoD penalties are hard to optimize

We could reduce κ, but …

OoD penalties are already too weak

to enforce the intended invariance constraints.

Deep learning optimization

- Finding features that work for in-distribution testing has proven to be easier than expected.

  Training overparametrized deep nets works remarkably well.

- Finding the correct solution —a solution that also works out-of-distribution— might be far harder.

Gradient starvation

- The network associates the class “cow” with the presence of grassy texture in the image.
- This association leads to high accuracy, leaving only a few training examples unaccounted for.
- Not enough training examples to give a strong enough gradient.

(Pezeshki et al., 2020)

Gradient starvation = ill-conditioning

- T
"c
te
- T
ac
ex
- I
gi

There is a descent path towards the "right features" (e.g the cow shape) but this path is ill-conditioned.

- *Shallow descent in the direction that elicits the right features (few examples benefit from this)*
- *Strong curvature in the directions that reduce the network dependency on the shortcut features (many examples could suffer from an inexact change.)*

Initializing with a rich set of features

Difficult optimization can often be helped by a good initialization

- Initialize the network with a rich features set.
- Let the learning algorithm pick the one it likes

Rich Feature Construction (RFC) - basic idea

- Train a network

$$\min_{\Phi,g} \mathbb{E}_p[\ell(y, g(\Phi(x))]$$

- Freeze features and find pessimal data reweighting

$$\max_{q \in Q} \min_{g} \mathbb{E}_q[\ell(y, g(\Phi(x))]$$

- Training again forces discovering new features Φ'

$$\min_{\Phi\prime,g} \mathbb{E}_q[\ell(y, g(\Phi'(x))]$$

- Gather old and new features

$$\Phi \cup \Phi' \rightarrow \Phi_{new}$$

- Repeat

Rich Feature Construction (RFC) - more details

- Train a network

$$\min_{\Phi, g}$$

- Freeze features and find pessi… data reweighting

$$\max_{q \in \mathcal{Q}} \min_{g} \mathbb{E}_q[\ell(y, g(\Phi(x))]$$

- Training again forces discovering new features Φ'

$$\min_{\Phi', g} \mathbb{E}_q[\ell(y, g(\Phi'(x))$$

- Gather old and new features

$$\Phi \cup \Phi' \rightarrow \Phi_{new}$$

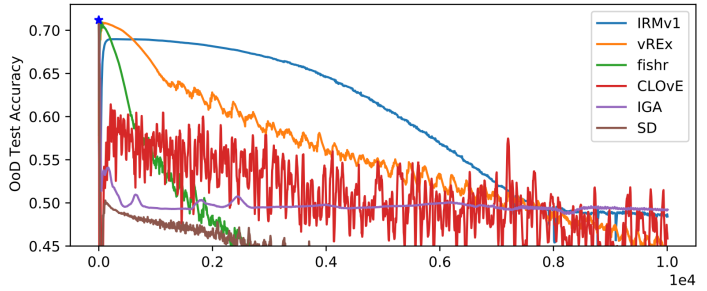
- Repeat

Weight on subsets of examples

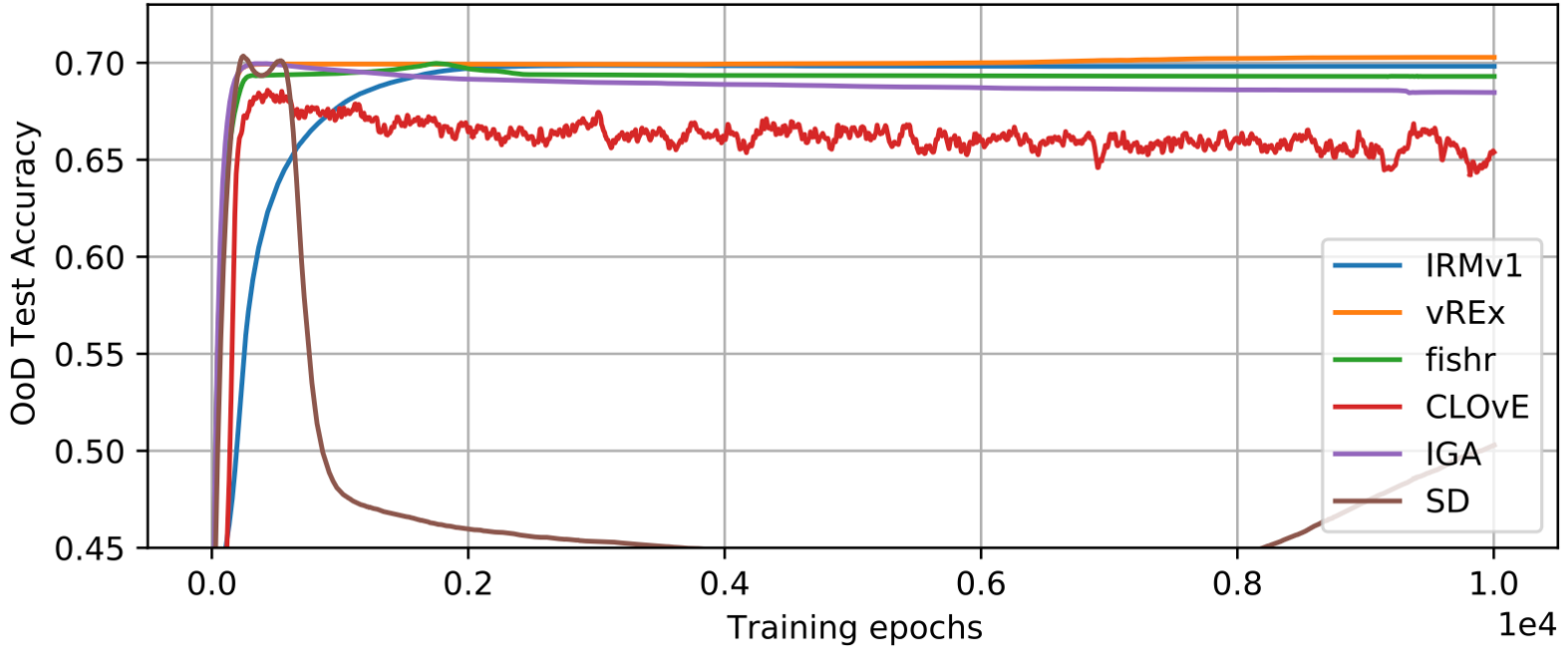
Distillation

Duality trick gives a DRO formulation that saves distillation time

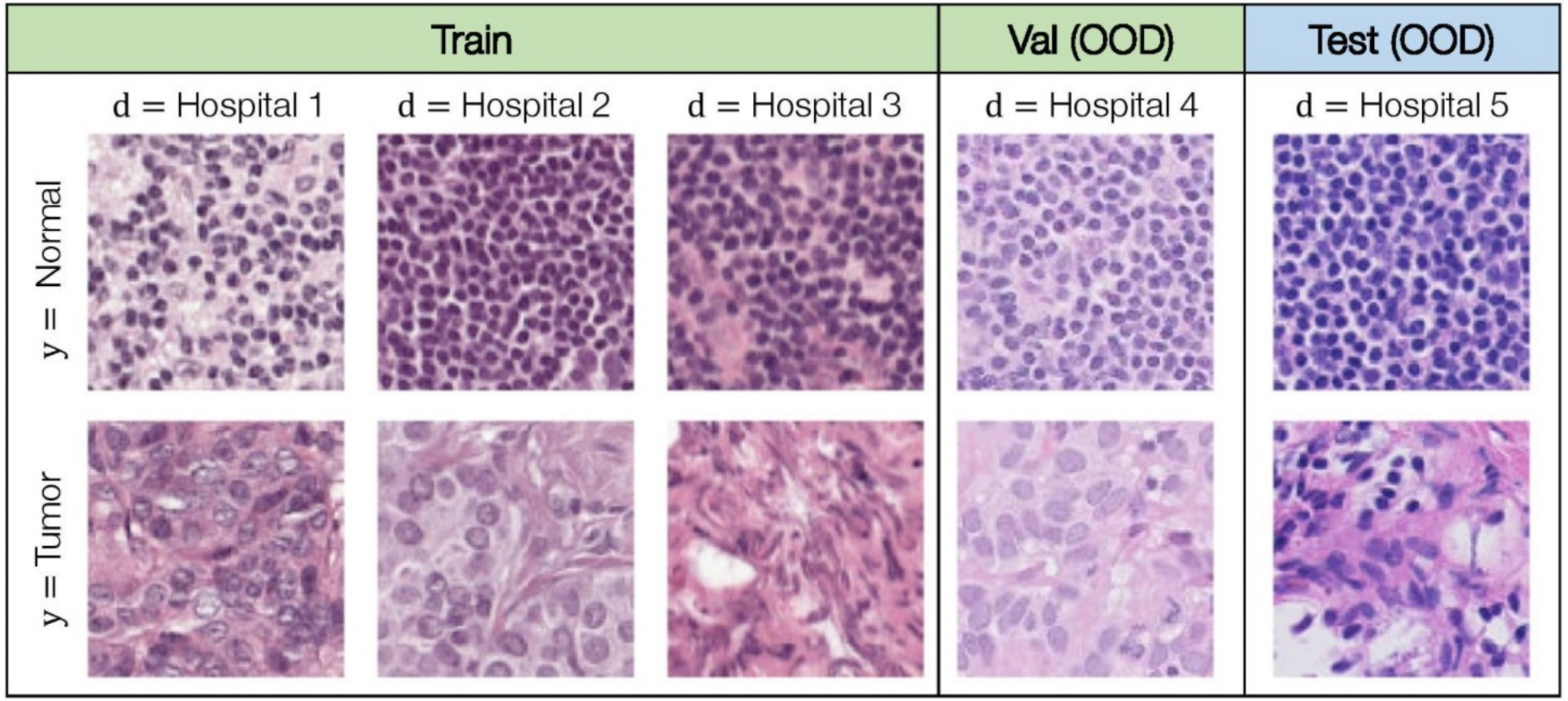
Initializing with RFC



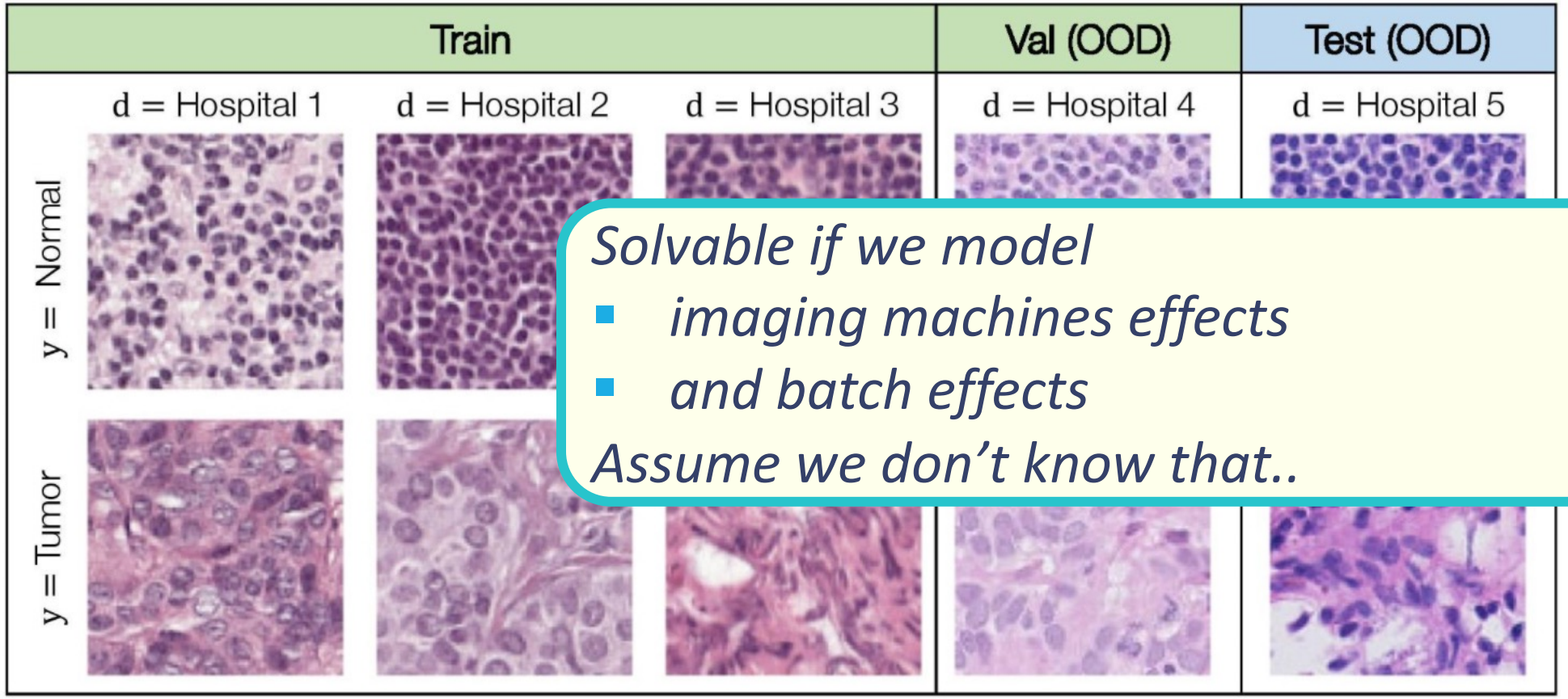
Color MNIST



Wilds / Camelyon17



Wilds / Camelyon17



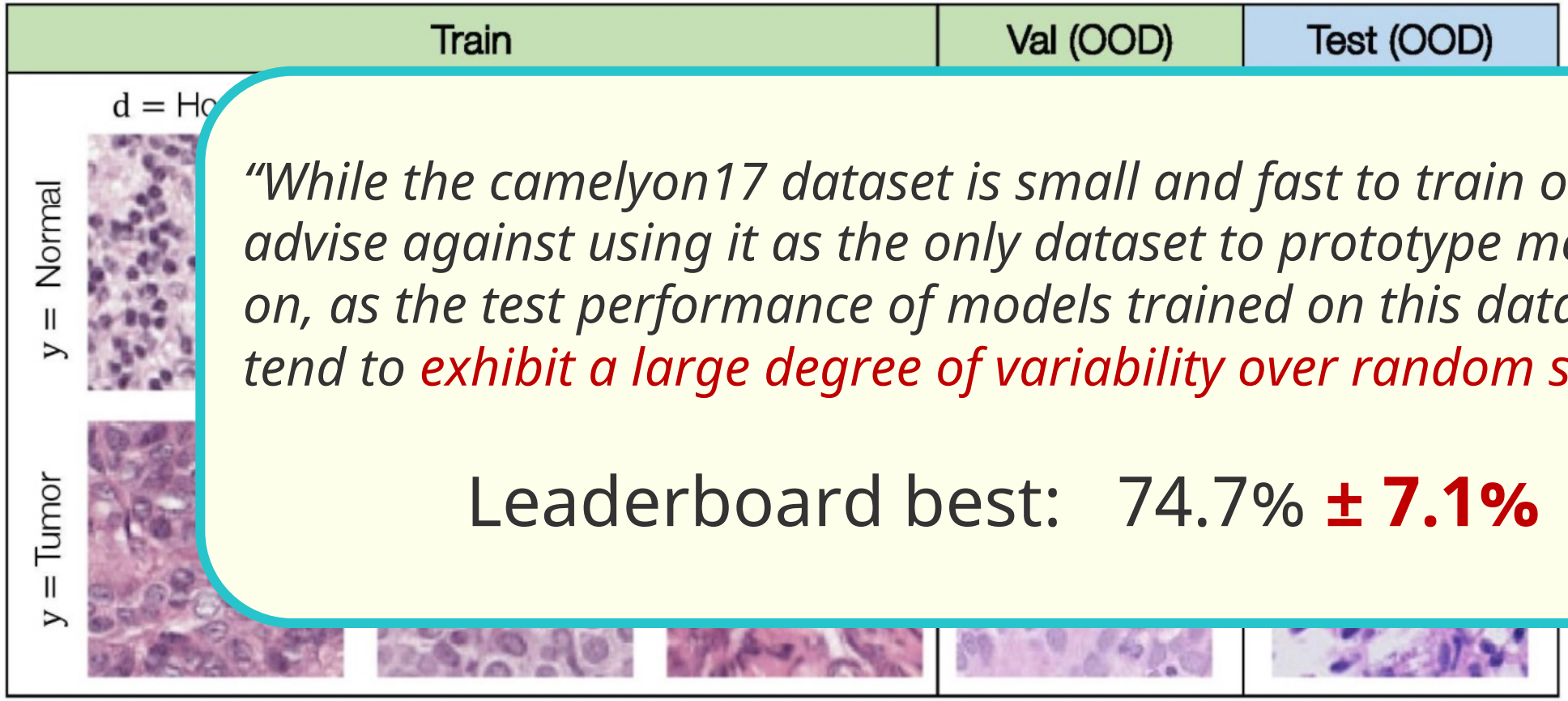
Solvable if we model

- *imaging machines effects*
- *and batch effects*

Assume we don't know that..

https://wilds.stanford.edu/datasets/#camelyon17

Wilds / Camelyon17



"While the camelyon17 dataset is small and fast to train on, we advise against using it as the only dataset to prototype methods on, as the test performance of models trained on this dataset tend to exhibit a large degree of variability over random seeds."

Leaderboard best: 74.7% **± 7.1%**

https://wilds.stanford.edu/datasets/#camelyon17

Network Initialization	Methods	Test Acc	
		IID Tune	OOD Tune
×	ERM	66.6±9.8	70.2±8.7
ERM	IRMv1	68.6±6.8	68.5±6.2
ERM	vREx	69.1±8.1	69.1±13.2
ERM	CLOvE	71.7±10.2	69.0±12.1
2-RFC	ERM	72.8±3.2	74.7±4.3
2-RFC	IRMv1	71.6±4.2	75.3±4.8
2-RFC	vREx	73.4±3.3	76.4±5.3
2-RFC	CLOvE	74.0±4.6	76.6±5.3
2-RFC	ERM(cf)	78.2±2.6	78.6±2.6
2-RFC	IRMv1(cf)	78.0±2.1	79.1±2.1
2-RFC	vREx(cf)	77.9±2.7	79.5±2.7
2-RFC	CLOvE(cf)	77.8±2.2	78.6±2.6

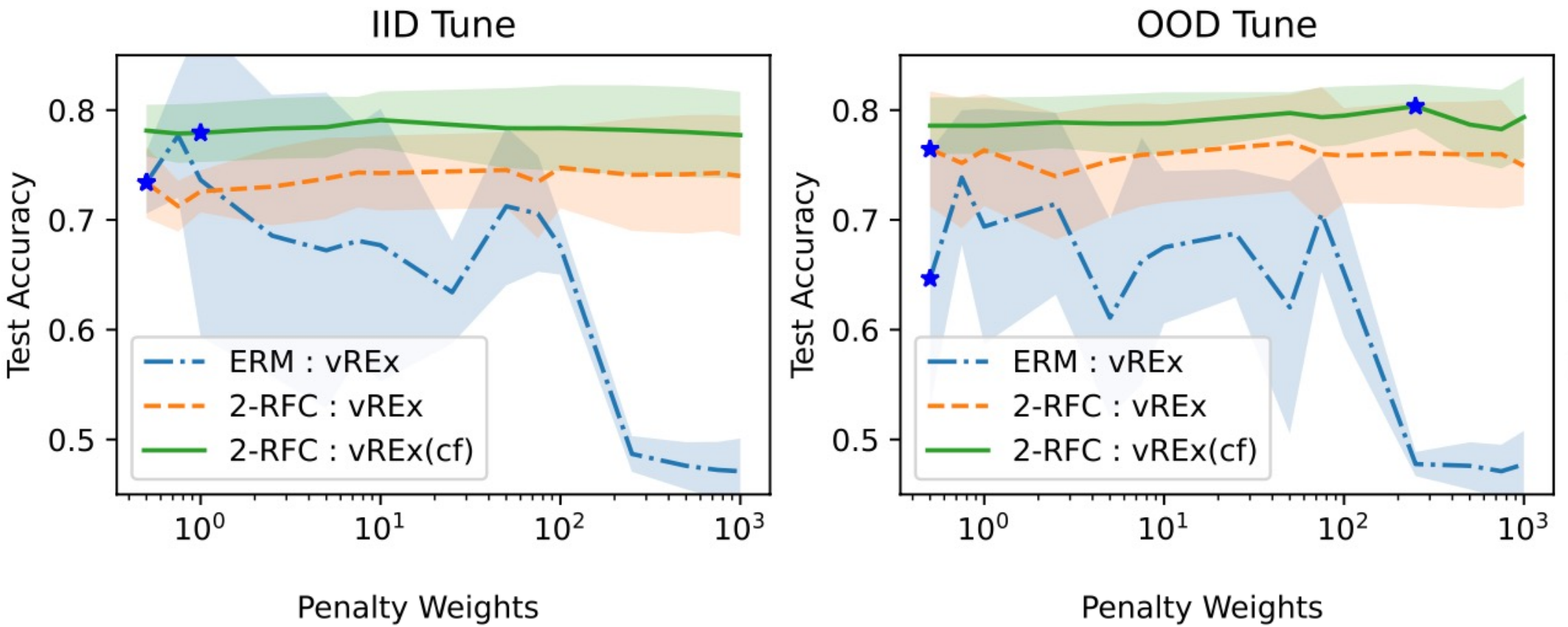
Leaderboard best:
74.7 ± 7.1 %

2RFC + ERM !

Frozen features !

Everything works!

Wilds/Camelyon17



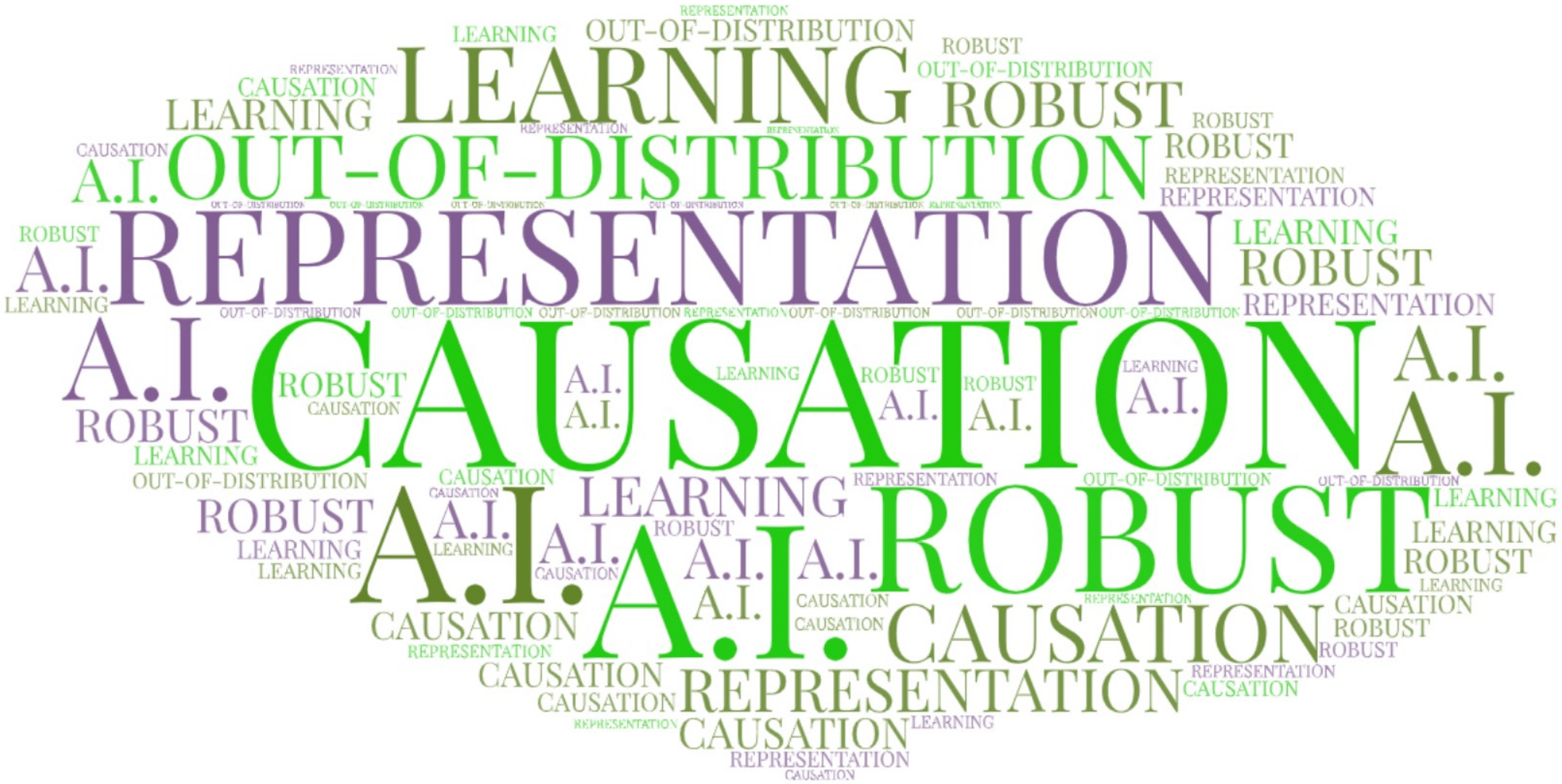
Rich feature initialization

- Everything works robustly after RFC!
- RFC needs no additional data but is computationally cumbersome
- RFC is just one way to construct rich features
- Other ways include self-supervised methods
  - usually with additional data
  - maybe they do not need so much data, in fact…

Conclusion

Recap

- We need learning algorithms that are less dependent on the data distribution
- Out-of-distribution generalization is tied to causation
- There are lots of causation hints out there
- Causation can be approached through invariances
- Discovering invariant features can be formulated mathematically
- In order to find them in practice, we must start with a good set of features


REPRESENTATION
LEARNING
OUT-OF-DISTRIBUTION
ROBUST
CAUSATION
A.I.