
Rm Palaniappan, *Alien Planet-X-9*
Viscosity, pencil colour and ink on handmade paper

Exploring strange neu(ral network) worlds with well-worn tools

Anand D. Sarwate, Rutgers University
16 January 2025

BIRS Workshop 25w5389
Machine Learning and Statistics: From Theory to Practice
Chennai Mathematical Institute

Thanks to my collaborators/coauthors!

Most of this is their work, obviously

Sinjini Banerjee (Rutgers)

Reilly Cannon (PNNL)

Tony Chiang (ARPA-H)

Sutenay Choudhury (PNNL)

Ioana Dumitriu (UC San Diego)

Andrew Engel (Ohio State)

Natalie Frank (U Washington)

Tim Marrinan (PNNL)

Max Vargas (PNNL)

Zhichao Wang (UC Berkeley)

Papers:

[ArXiV] **Banerjee et al.** https://arxiv.org/abs/2406.08307
[NeurIPS 2023] **Wang et al.** https://openreview.net/forum?id=gpqBGyKeKH
[ICLR 2024] **Engel et al.** https://openreview.net/forum?id=yKksu38BpM
[ArXiV] **Vargas et al.** https://arxiv.org/abs/2408.10437

Image Credits

Rm. Palaniappan Prints:

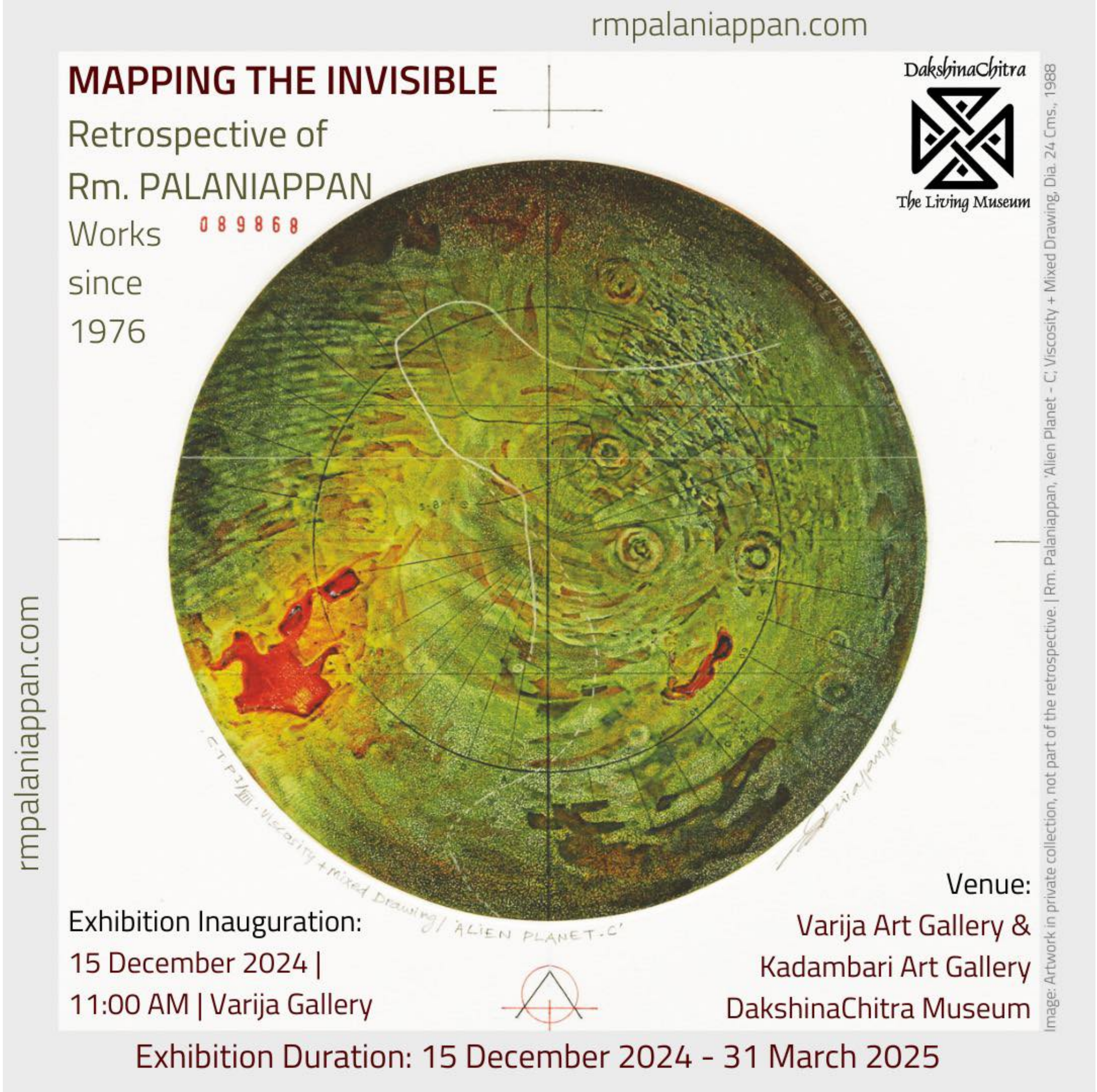
Alien Planet-X-9: DAG https://dagworld.com/palaniappanrm06.html
Center of International Modern Art: https://cimaartindia.com/artworks/p-571a-d/
MutualArt

TV images:

CBS/Getty and Paramount/CBS
Memory Alpha Wiki

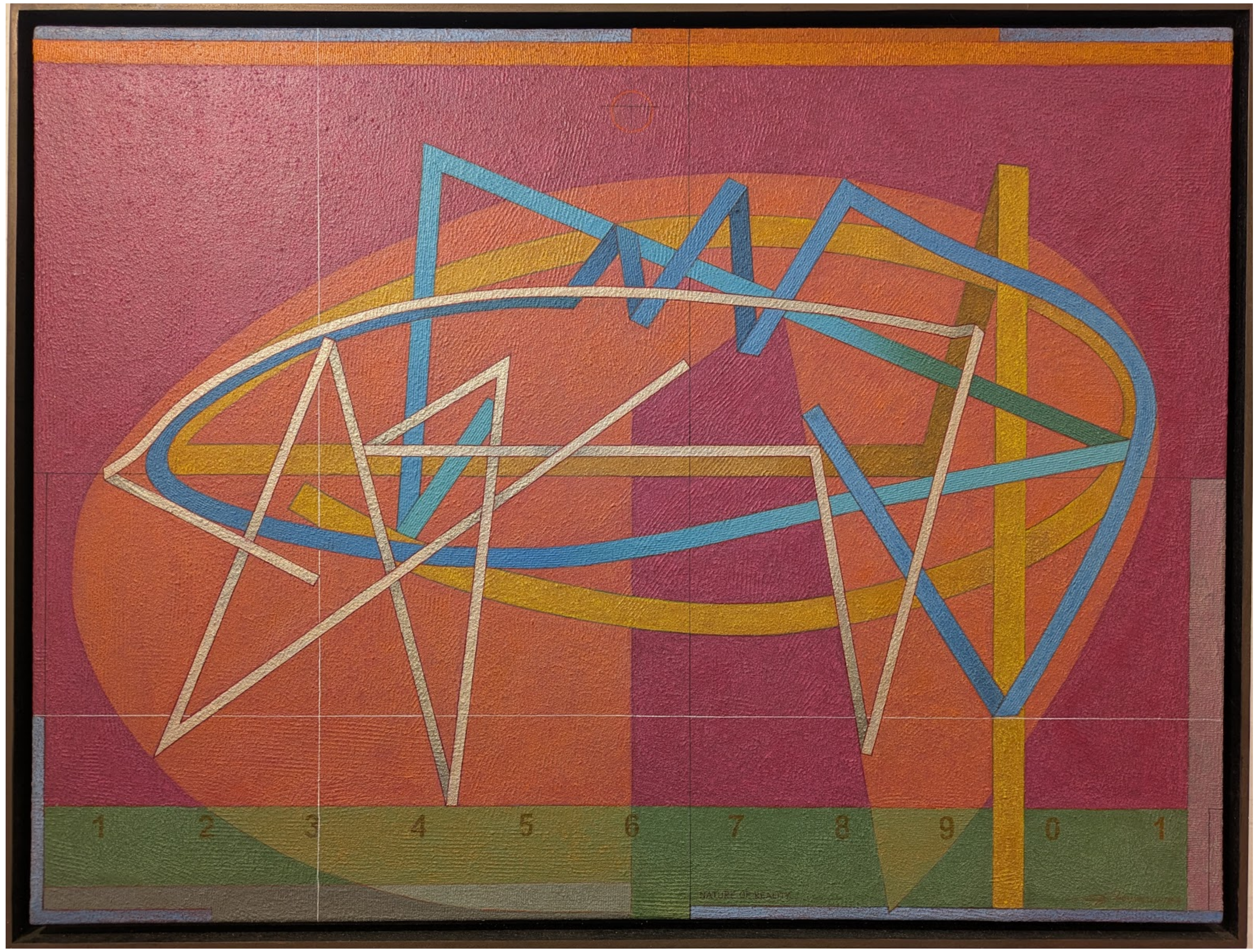
Misc:

AI Cat generator: https://www.basedlabs.ai/tools/ai-cat-generator
Foundation model: https://rehack.com/ai/what-are-foundation-models-in-generative-ai/
Data lake: https://databasetown.com
Wikimedia commons
OpenMoji: https://openmoji.org/



Ramanthan Palaniappan (b. 1957) is a Chennai-based artist who works in printmaking and mixed media.

The Dakshina Chitra museum (very close to CMI/the hotel!) has a restrospective of his works, some of which incorporate elements from architectural and engineering diagrams. Check it out!



Modern ML/AI practice owes a lot to theory!

Frameworks, algorithms, etc.

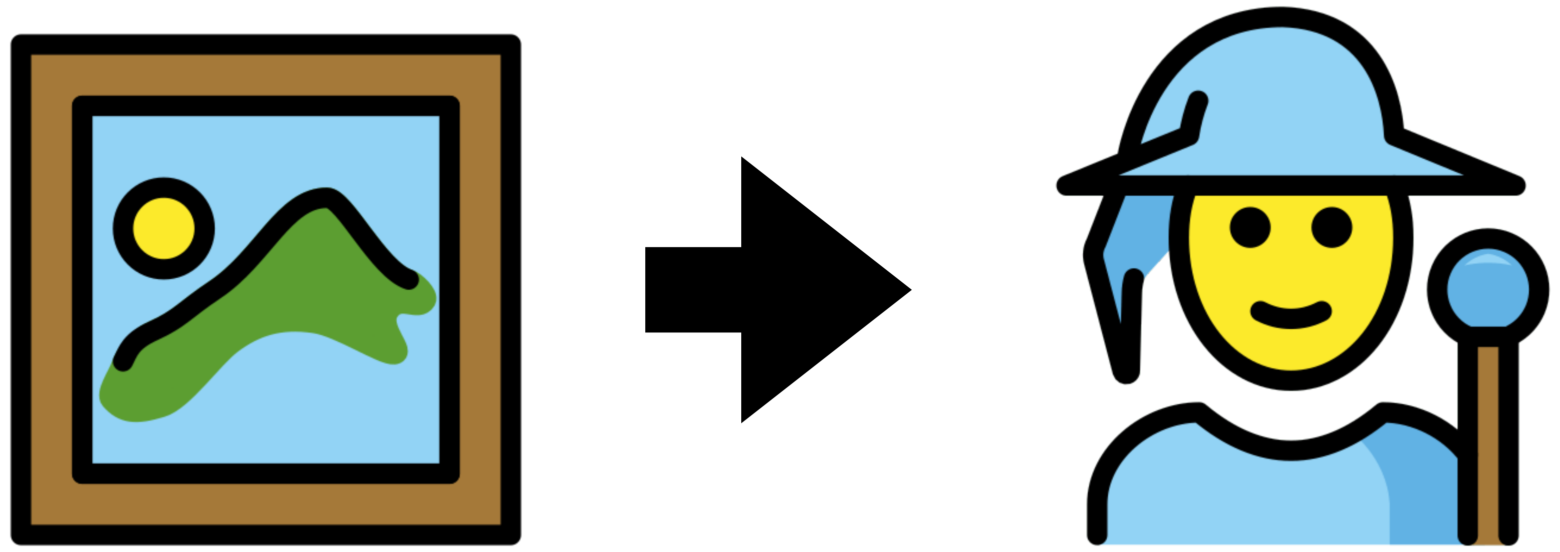
Modern ML/AI practice owes a lot to theory!

Frameworks, algorithms, etc.

Frameworks/abstractions for learning problems are fundamentally a theoretical contribution.

Modern ML/AI practice owes a lot to theory!

Frameworks, algorithms, etc.

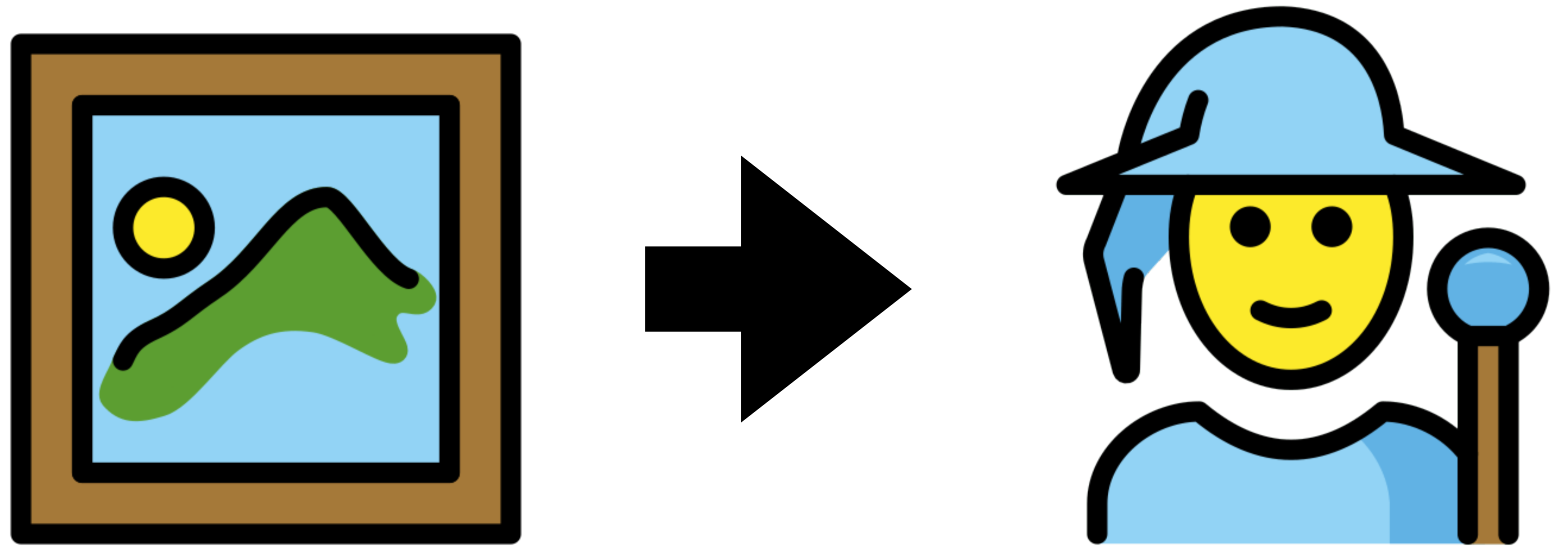


Frameworks/abstractions for learning problems are fundamentally a theoretical contribution.

Taks/objectives for AI systems are less clear.

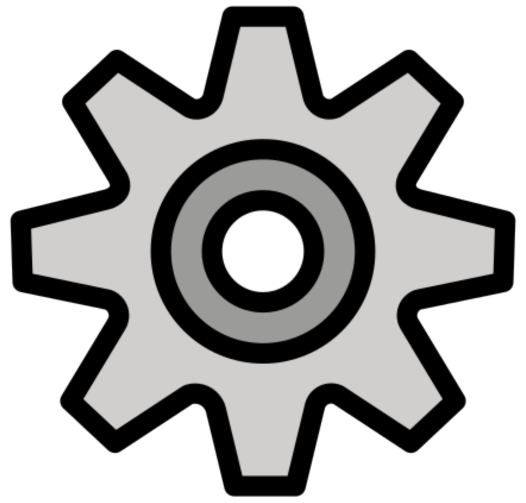
Modern ML/AI practice owes a lot to theory!

Frameworks, algorithms, etc.



Frameworks/abstractions for learning problems are fundamentally a theoretical contribution.

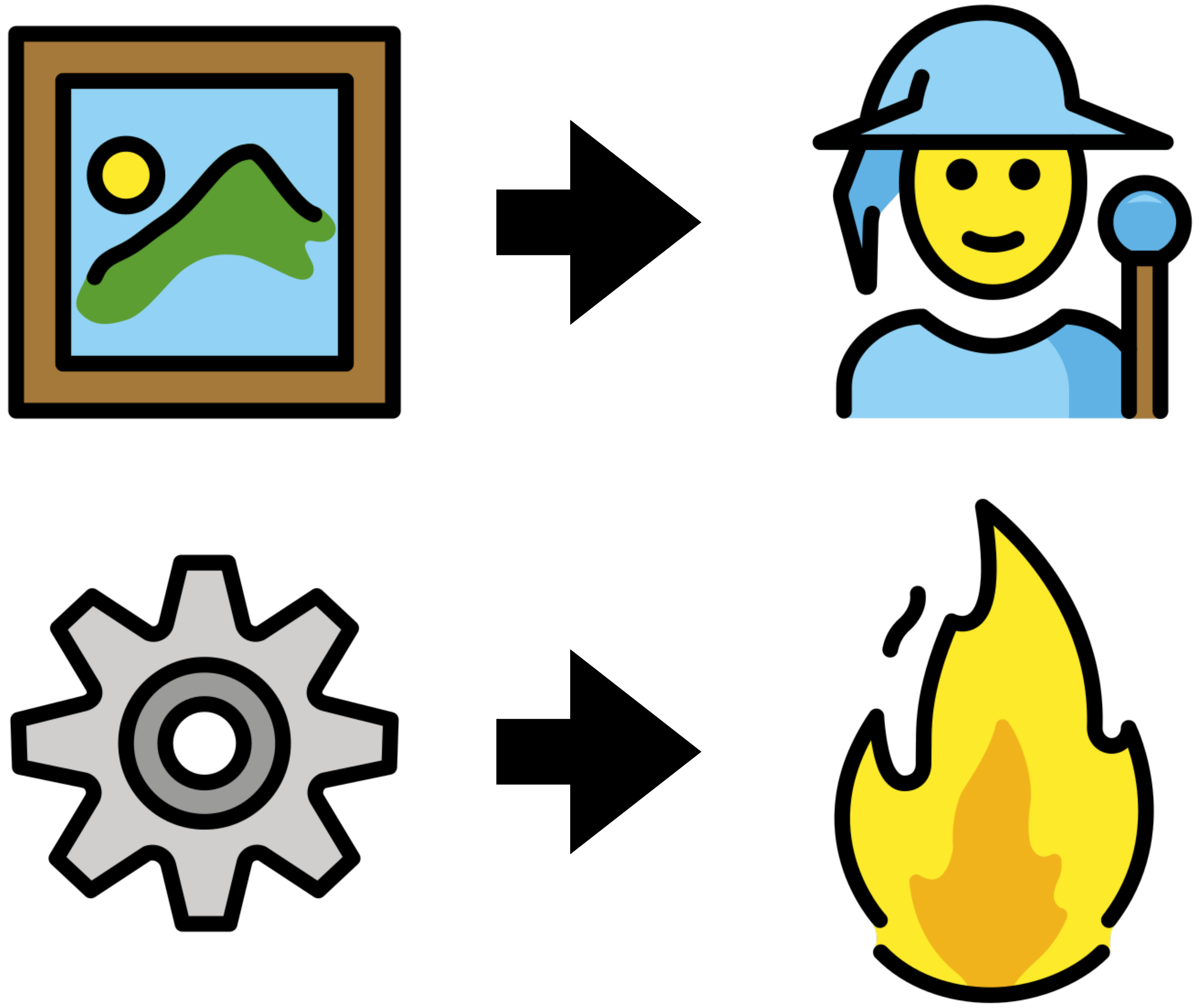
Taks/objectives for AI systems are less clear.



Algorithms used for optimization are grounded in a solid understanding of the mathematics.

Modern ML/AI practice owes a lot to theory!

Frameworks, algorithms, etc.



Frameworks/abstractions for learning problems are fundamentally a theoretical contribution.

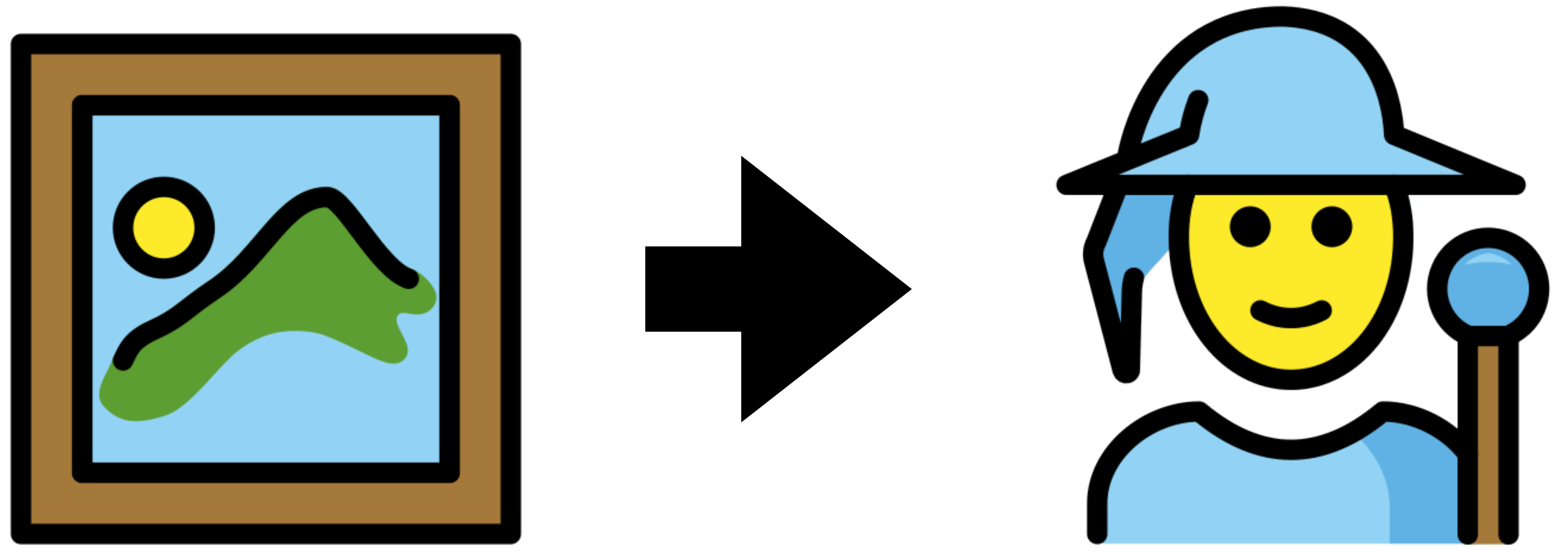
Taks/objectives for AI systems are less clear.

Algorithms used for optimization are grounded in a solid understanding of the mathematics.

Pseudocode may not reflect actual code.

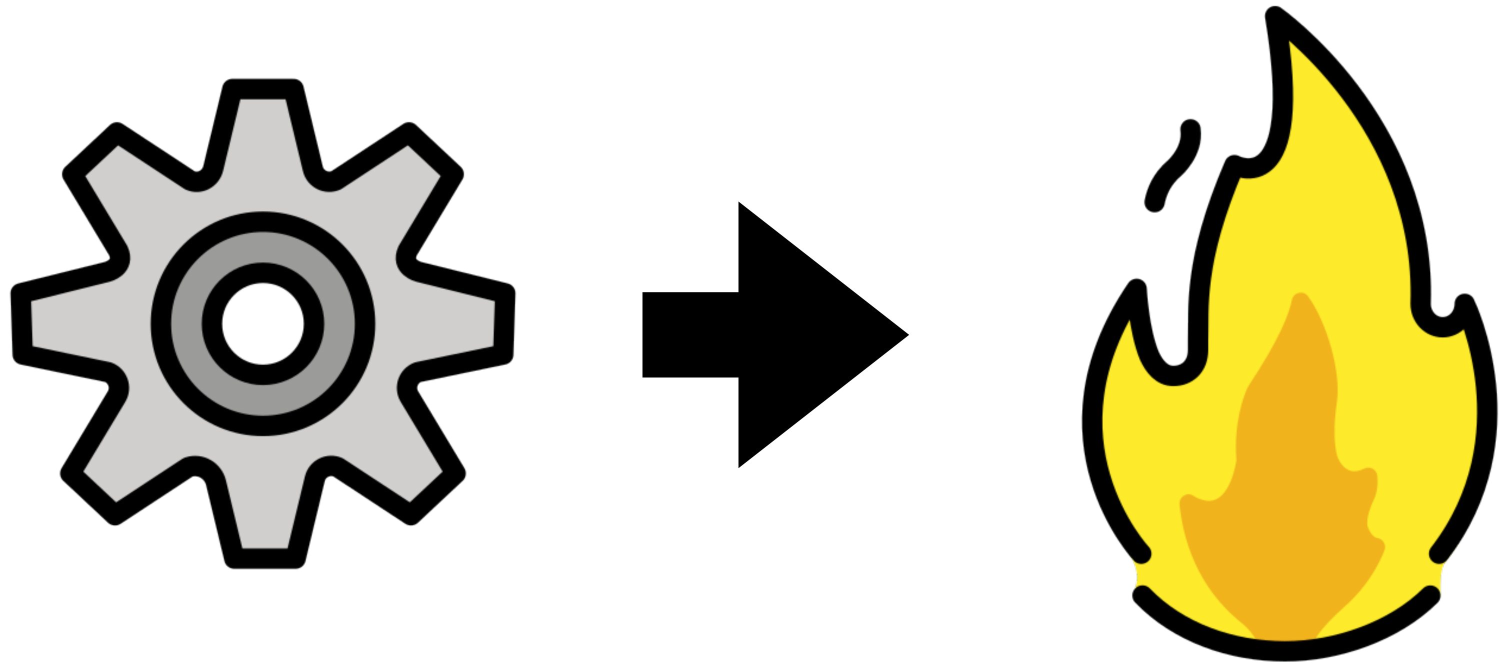
Modern ML/AI practice owes a lot to theory!

Frameworks, algorithms, etc.



Frameworks/abstractions for learning problems are fundamentally a theoretical contribution.

Taks/objectives for AI systems are less clear.



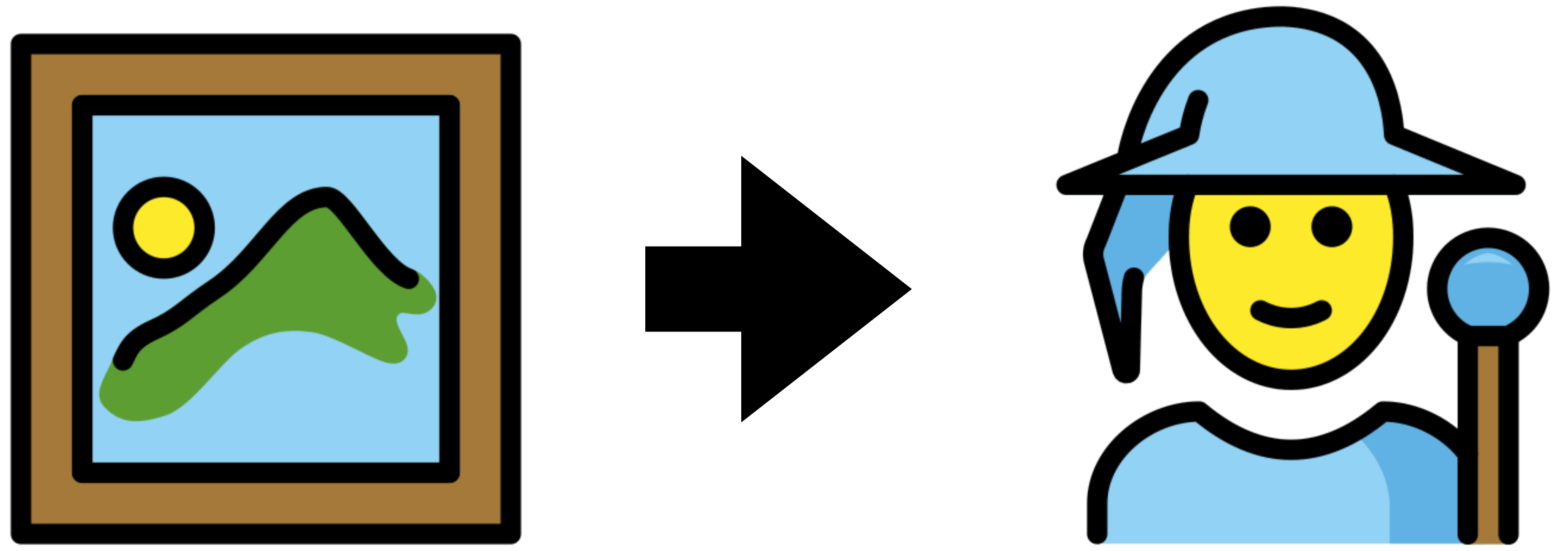
Algorithms used for optimization are grounded in a solid understanding of the mathematics.

Pseudocode may not reflect actual code.

Probabilistic analyses led credence to what people do in practice.

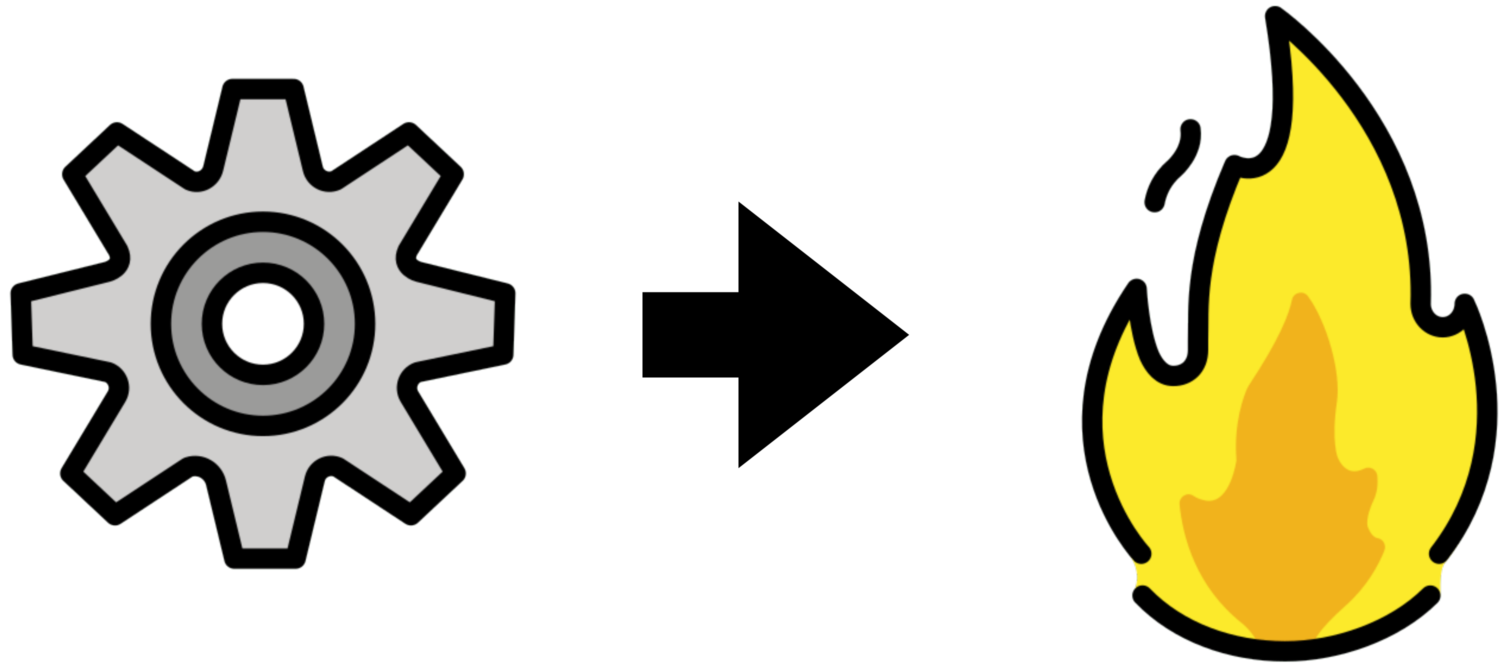
Modern ML/AI practice owes a lot to theory!

Frameworks, algorithms, etc.



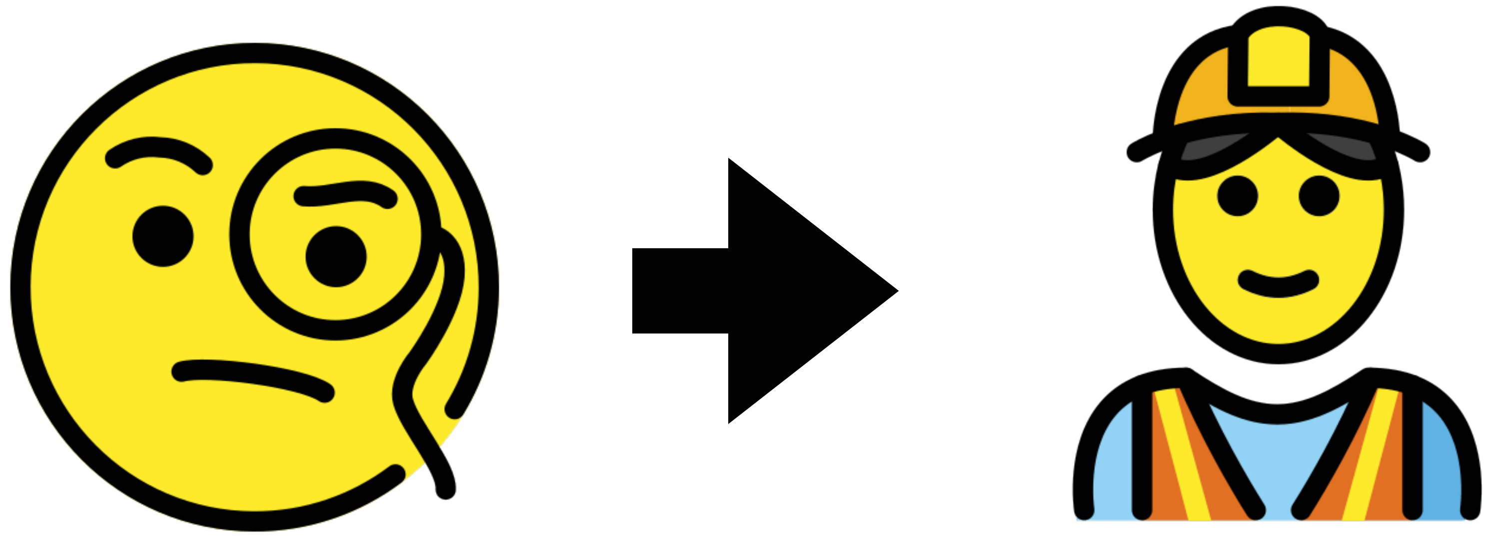
Frameworks/abstractions for learning problems are fundamentally a theoretical contribution.

Taks/objectives for AI systems are less clear.



Algorithms used for optimization are grounded in a solid understanding of the mathematics.

Pseudocode may not reflect actual code.

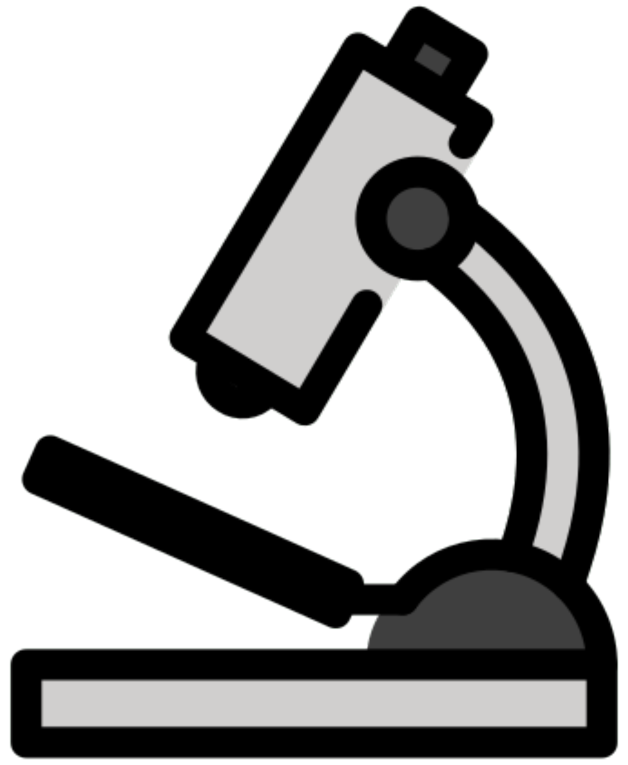


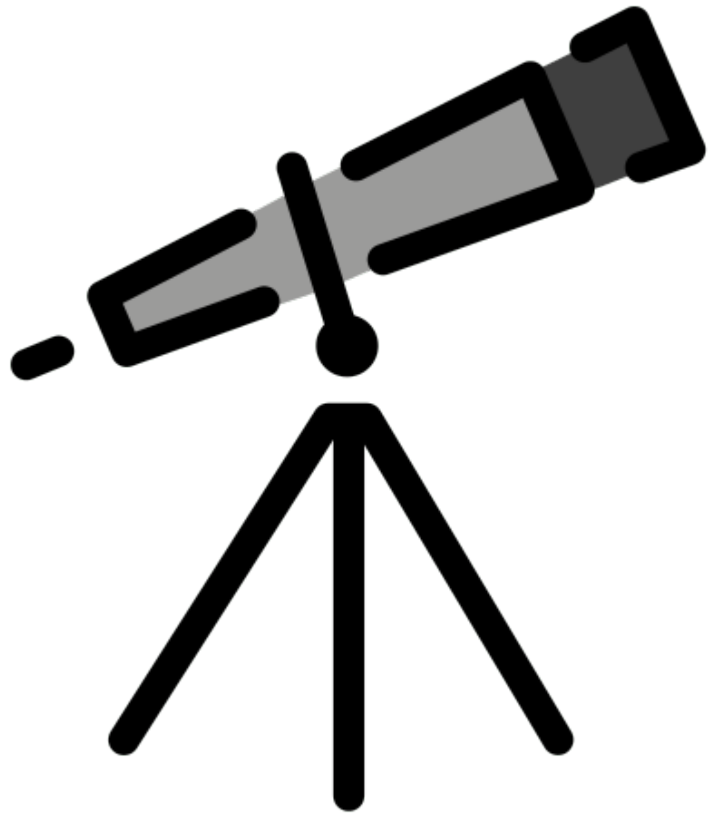
Probabilistic analyses led credence to what people do in practice.

Sometimes feels “after the fact.”

This is not particular to ML

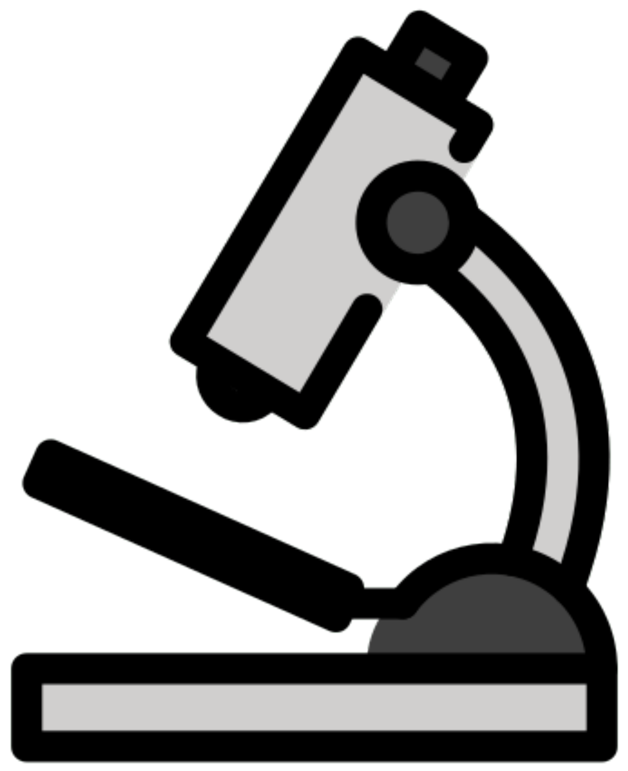
Almost the natural evolution of technologies?





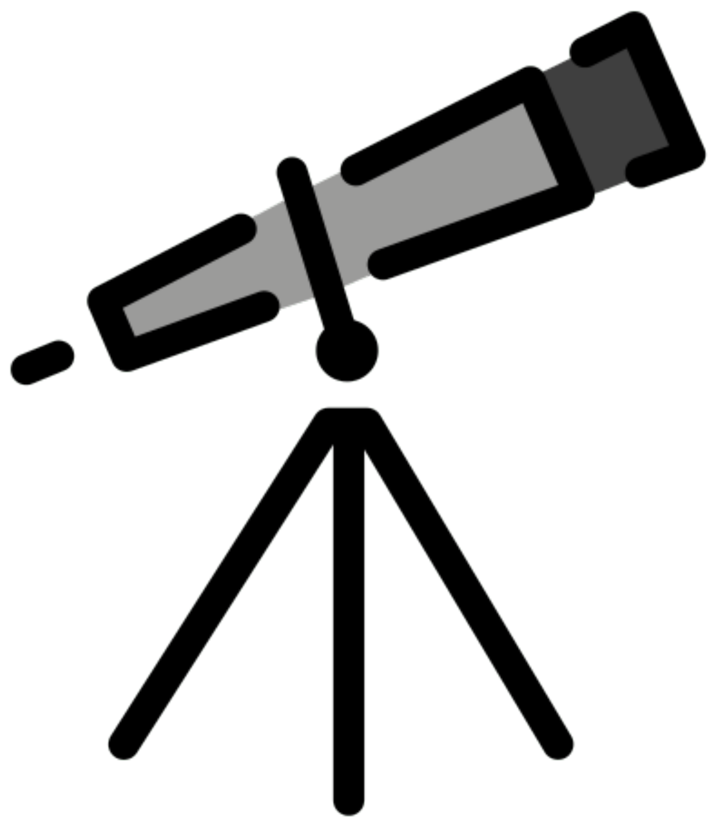
This is not particular to ML

Almost the natural evolution of technologies?



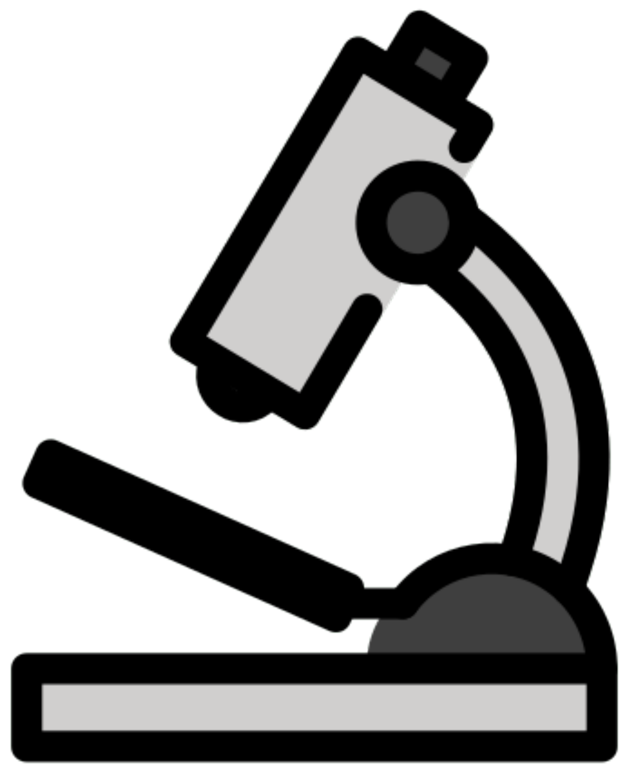
There's a huge push to bring AI into scientific research:

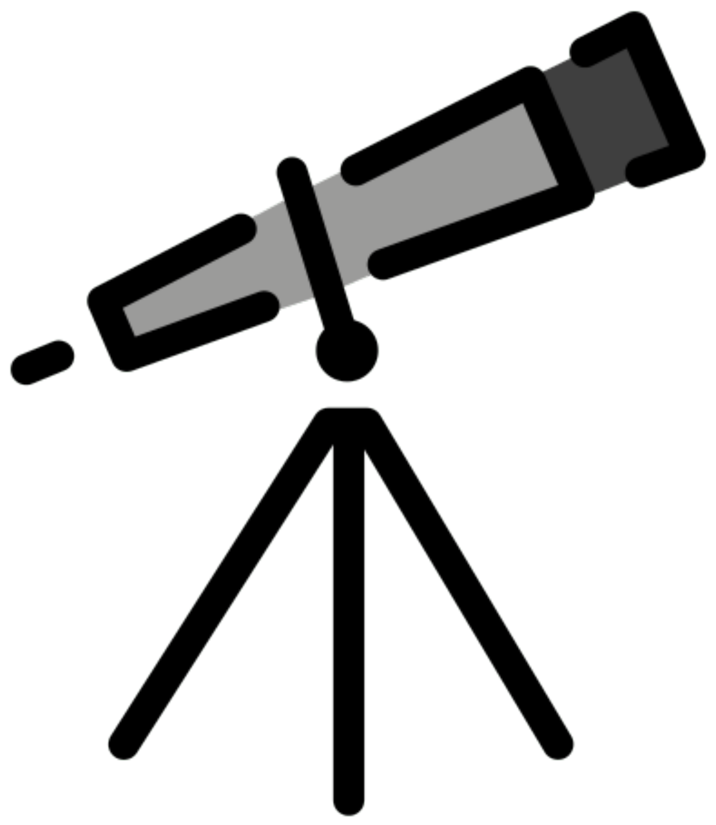
- Framed as a new data analysis tool.
- Supposed to break intractable barriers.



This is not particular to ML

Almost the natural evolution of technologies?





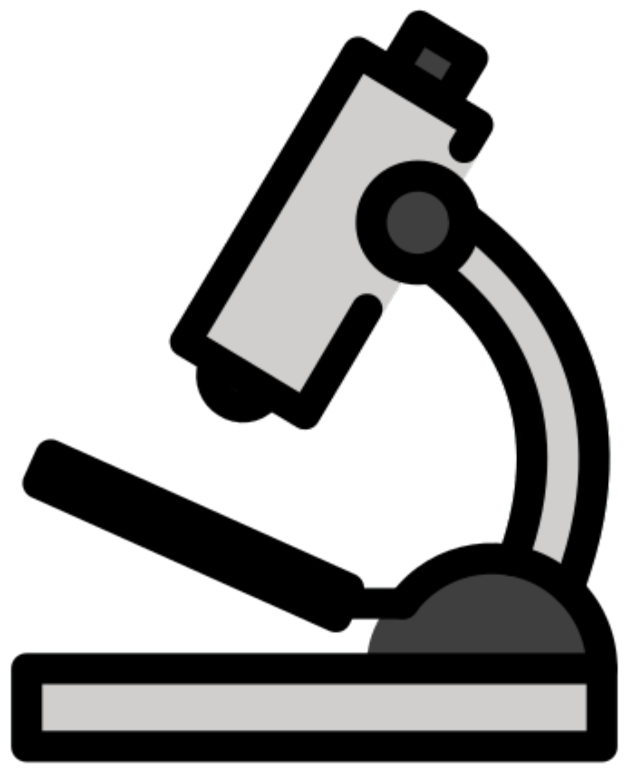
There's a huge push to bring AI into scientific research:

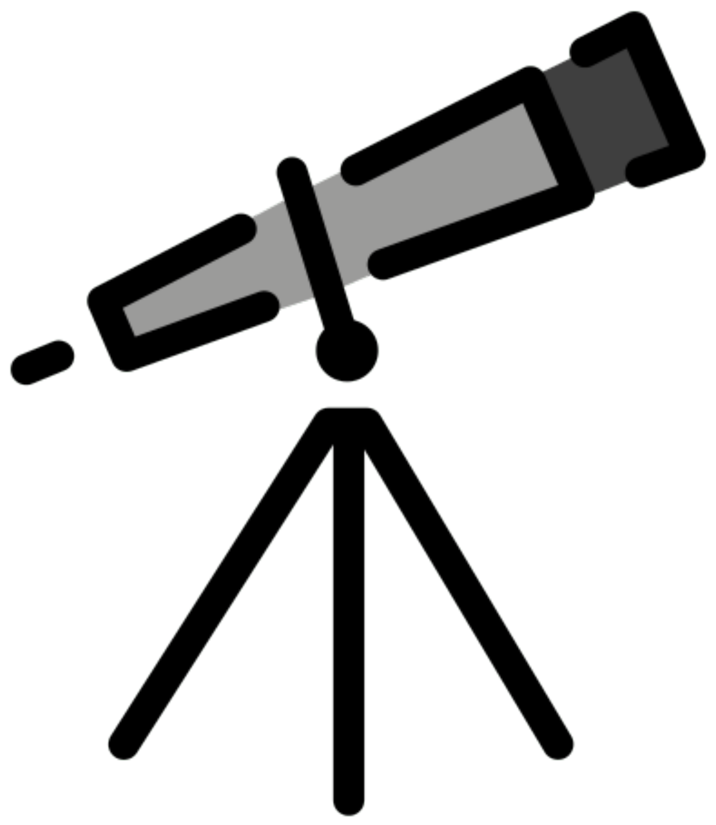
- Framed as a new data analysis tool.
- Supposed to break intractable barriers.

A thought experiment: what if we think of ML/AI models as scientific instruments? Instruments need to be:

This is not particular to ML

Almost the natural evolution of technologies?





There's a huge push to bring AI into scientific research:

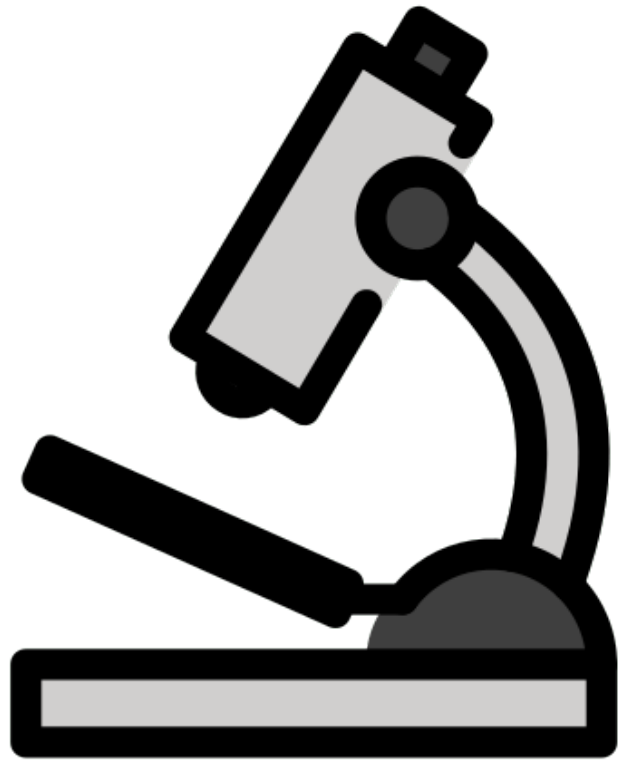
- Framed as a new data analysis tool.
- Supposed to break intractable barriers.

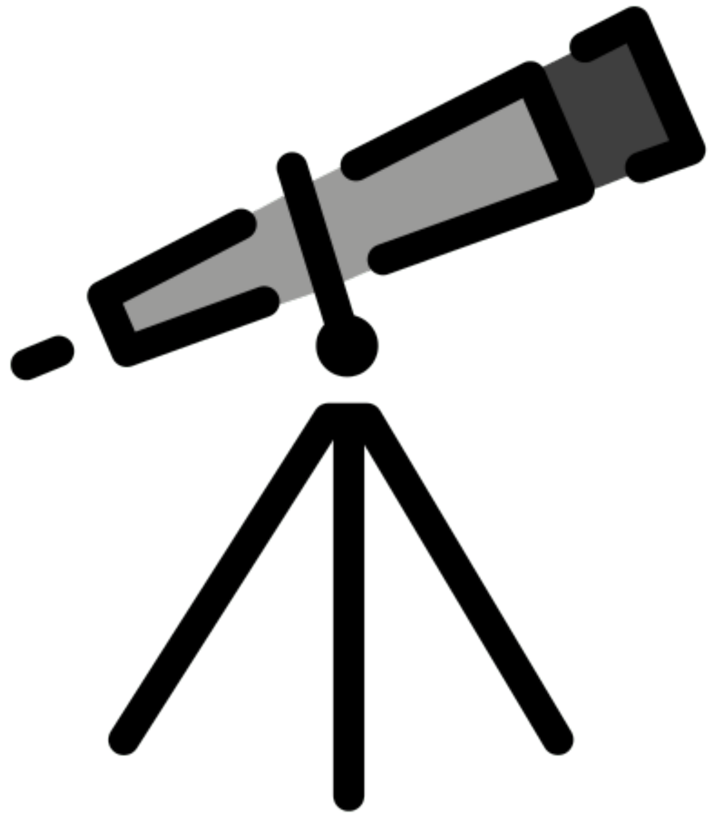
A thought experiment: what if we think of ML/AI models as scientific instruments? Instruments need to be:

- Characterized

This is not particular to ML

Almost the natural evolution of technologies?





There’s a huge push to bring AI into scientific research:

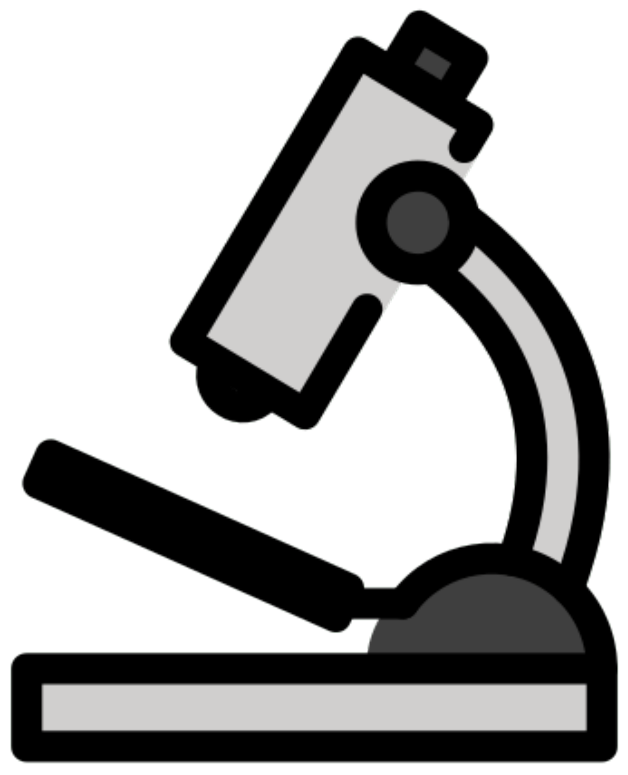
- Framed as a new data analysis tool.
- Supposed to break intractable barriers.

A thought experiment: what if we think of ML/AI models as scientific instruments? Instruments need to be:

- Characterized
- Calibrated

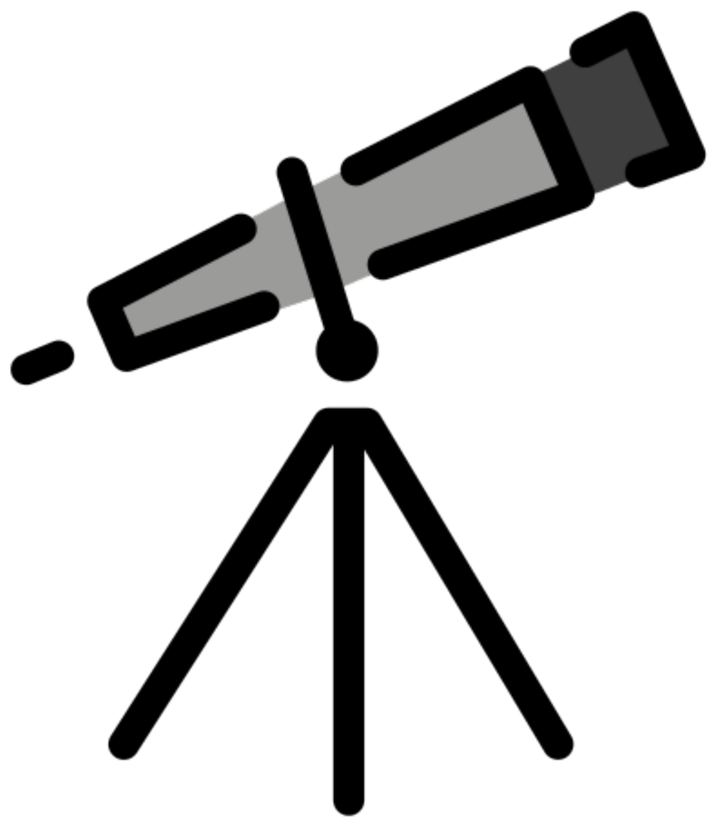
This is not particular to ML

Almost the natural evolution of technologies?



There's a huge push to bring AI into scientific research:

- Framed as a new data analysis tool.
- Supposed to break intractable barriers.

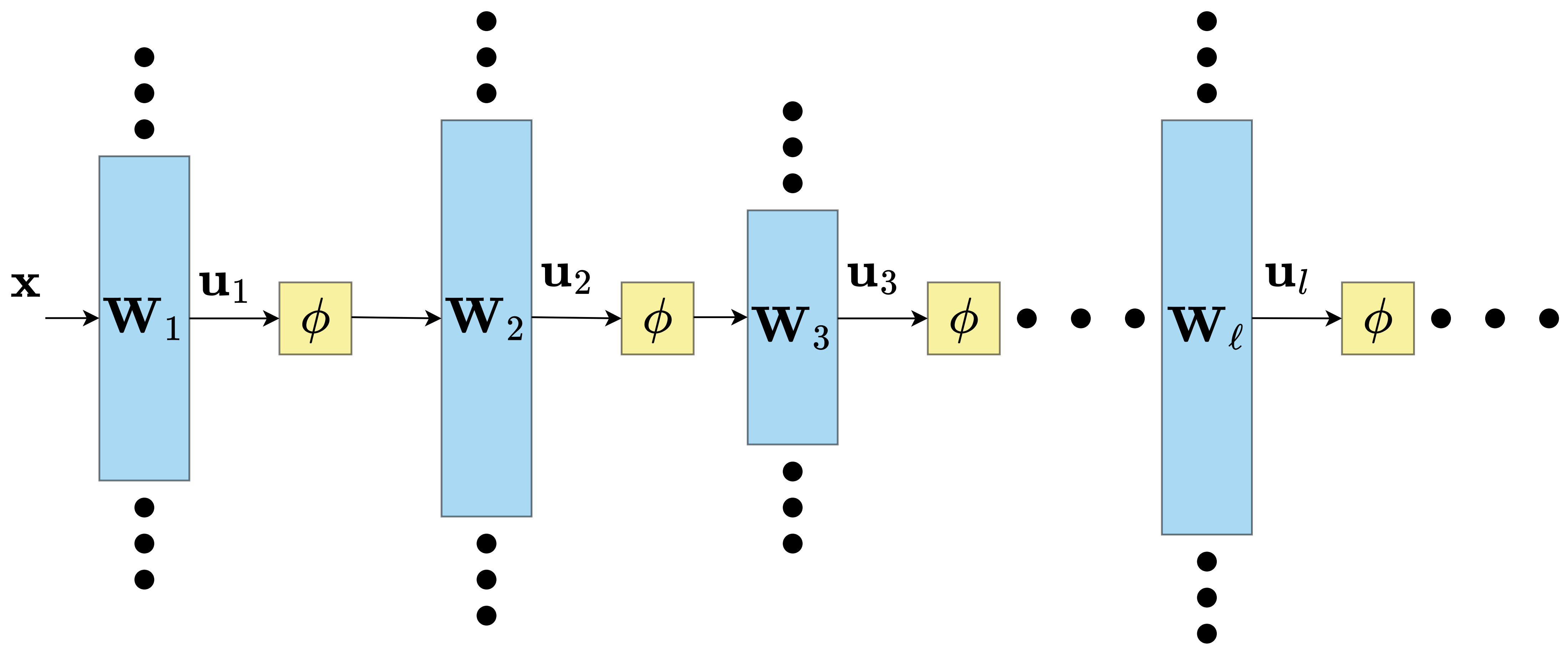


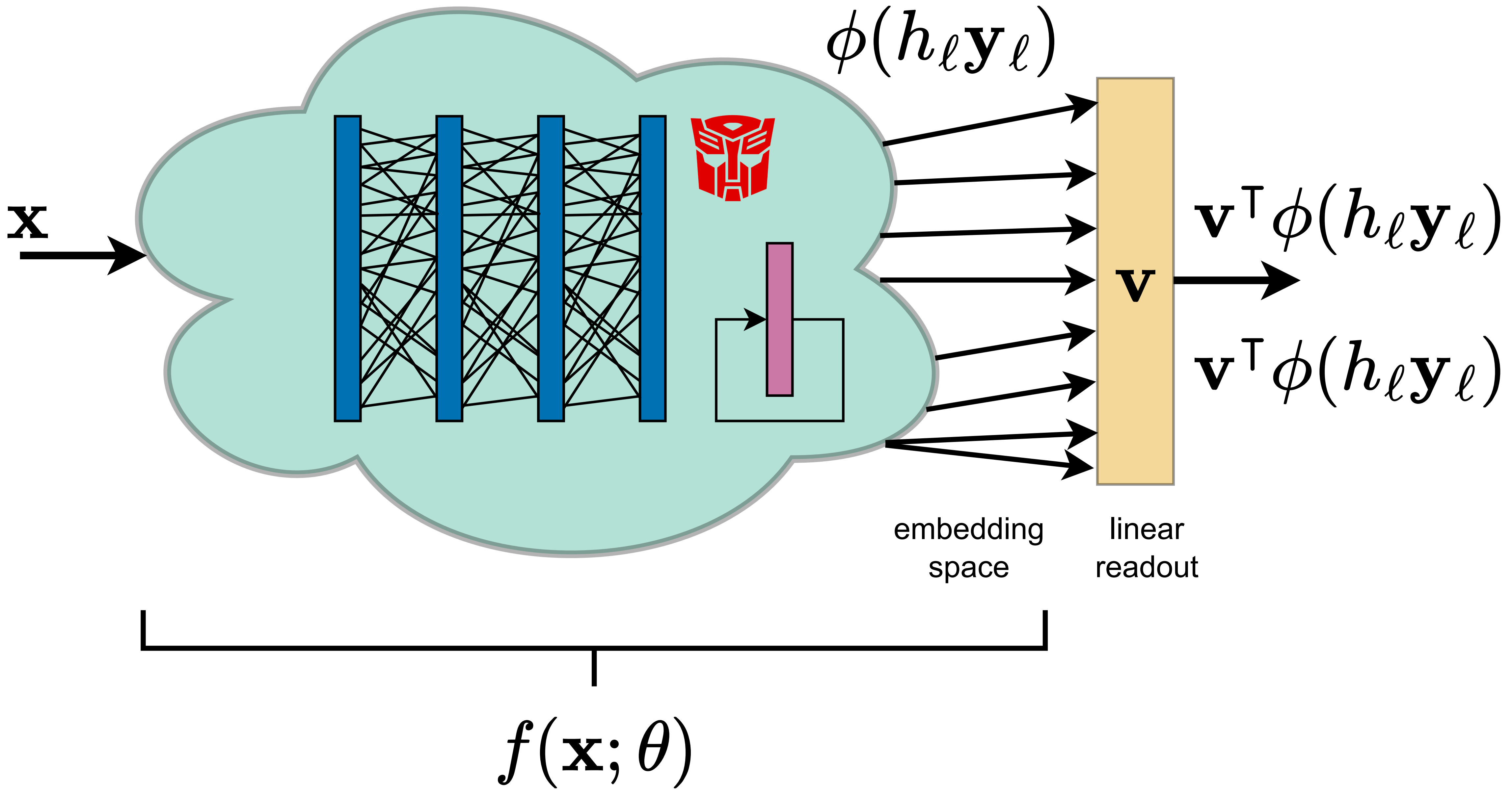
A thought experiment: what if we think of ML/AI models as scientific instruments? Instruments need to be:

- Characterized
- Calibrated
- Comparable (or interoperable)

Scientific instruments are very complex!

Or: architecture-schmarchitecture





MLPs and other architectures for which the "mechanism of action" feels tractable are one way of abstracting it

Treating a model like an instrument can mean "be a bit agnostic to the internals"

The big underlying question

A bit speculative but hopefully not too fictional

The big underlying question

A bit speculative but hopefully not too fictional

The fundamental question is:

How can/should we compare two different models?

The big underlying question

A bit speculative but hopefully not too fictional

The fundamental question is:

How can/should we compare two different models?

This is challenging because what it means for models to be similar is not clear.

The big underlying question

A bit speculative but hopefully not too fictional

The fundamental question is:

How can/should we compare two different models?

This is challenging because what it means for models to be similar is not clear.

- We often ask: “are these two models the same”?

The big underlying question

A bit speculative but hopefully not too fictional

The fundamental question is:

How can/should we compare two different models?

This is challenging because what it means for models to be similar is not clear.

- We often ask: “are these two models the same”?
- Maybe we should ask: “are these two models sufficiently different?”

A thought experiment

You land on an alien planet and discover some artifacts…

A thought experiment

You land on an alien planet and discover some artifacts…

Databases of measurements!

A thought experiment

You land on an alien planet and discover some artifacts…

Databases of measurements!

Strange alien technology!

A thought experiment

You land on an alien planet and discover some artifacts…

Databases of measurements!

Strange alien technology!

Cute fuzzy animals?

Looking at things today…

Maybe it’s not so far-fetched?

Looking at things today…

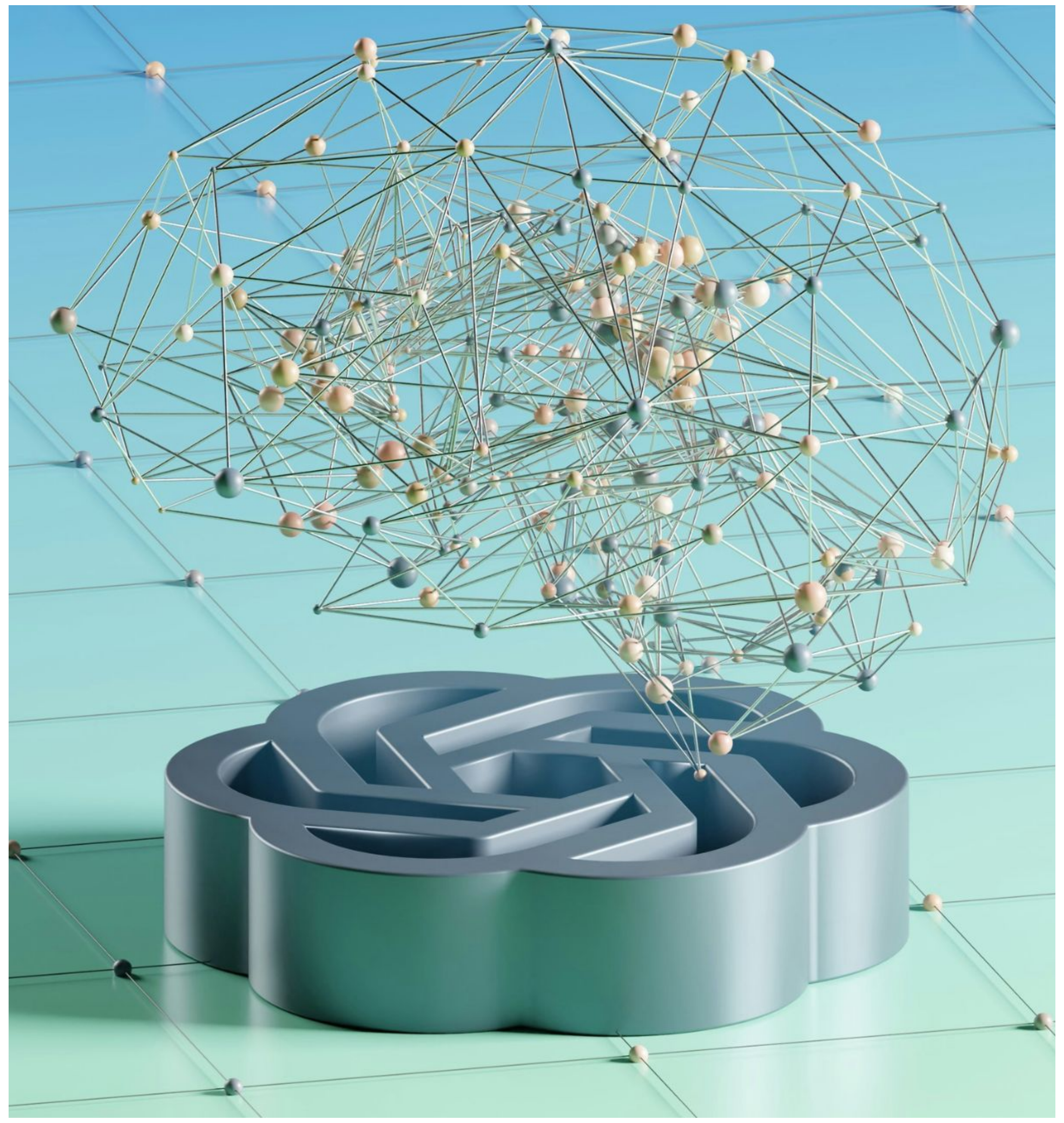
Maybe it's not so far-fetched?

Scraping all the data

Looking at things today…

Maybe it’s not so far-fetched?

Scraping all the data

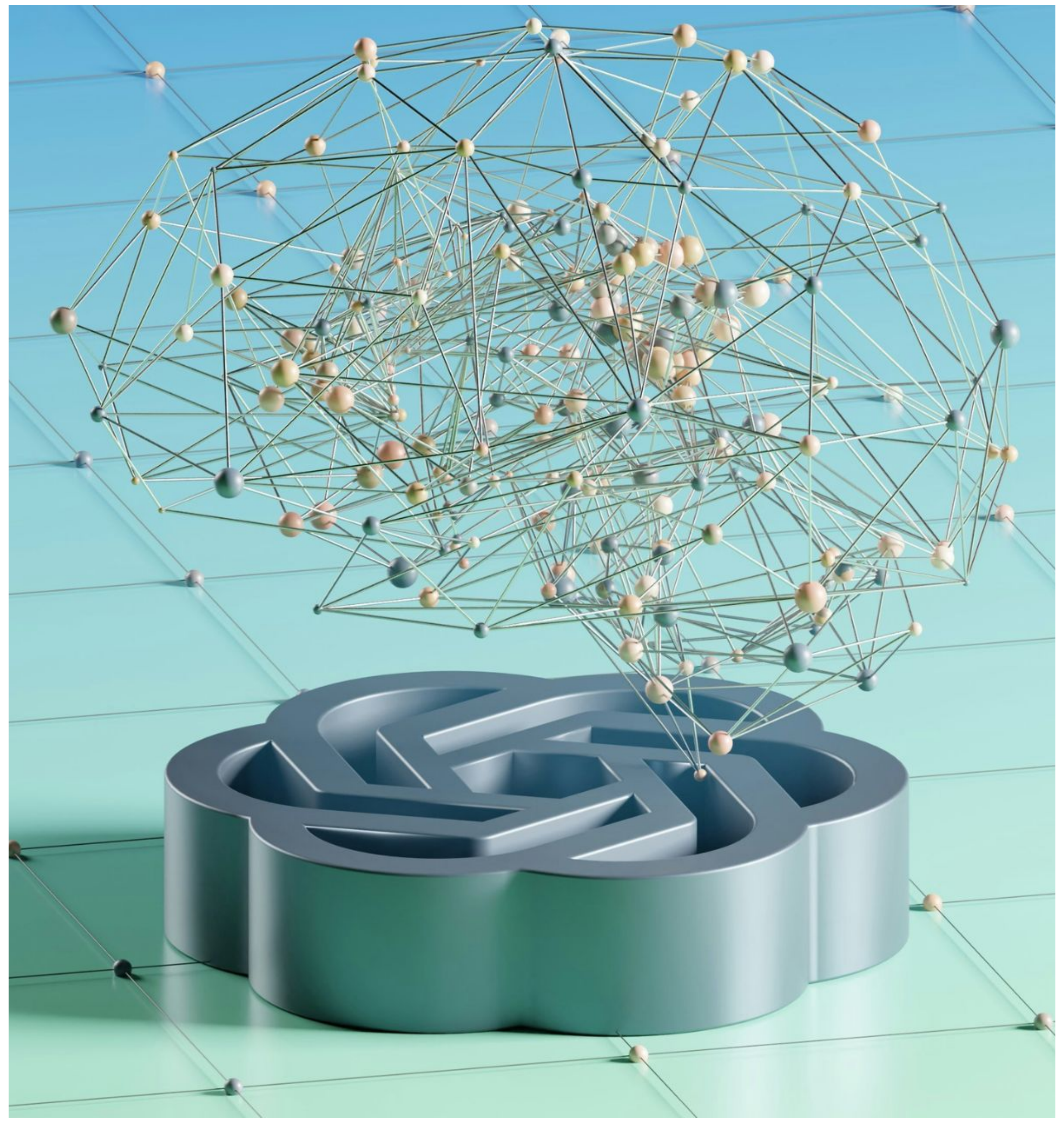


Foundation models

Looking at things today…

Maybe it's not so far-fetched?

Scraping all the data



Foundation models

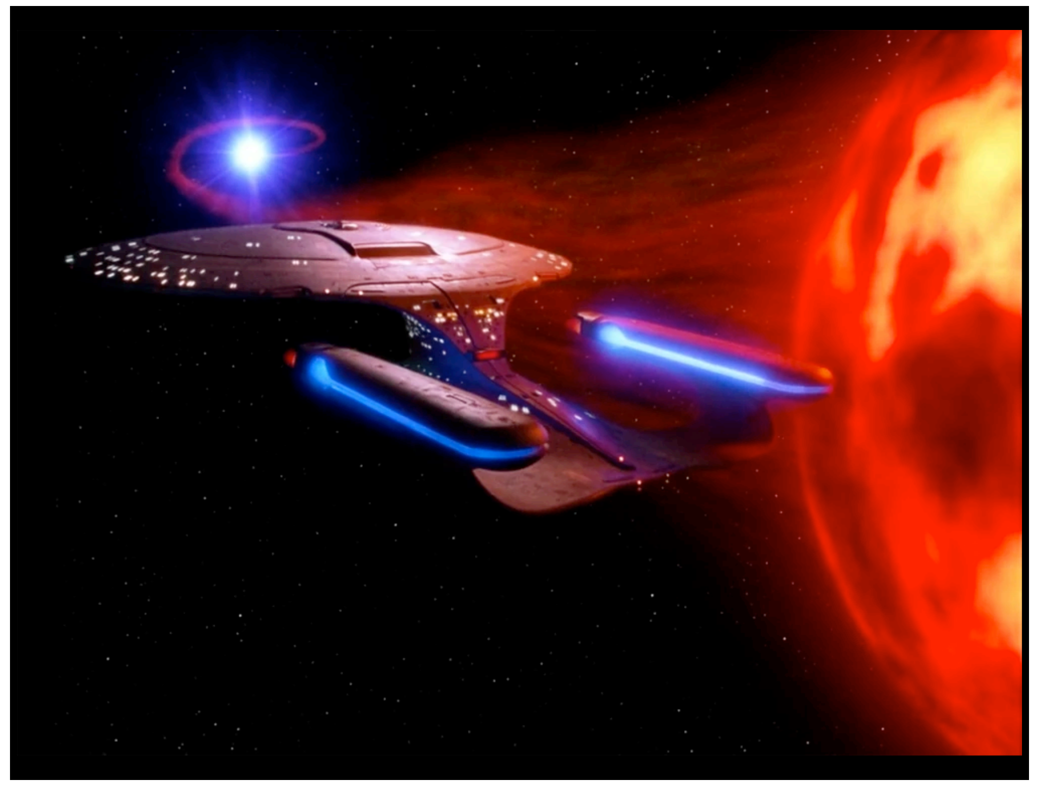
AI Cat Generator

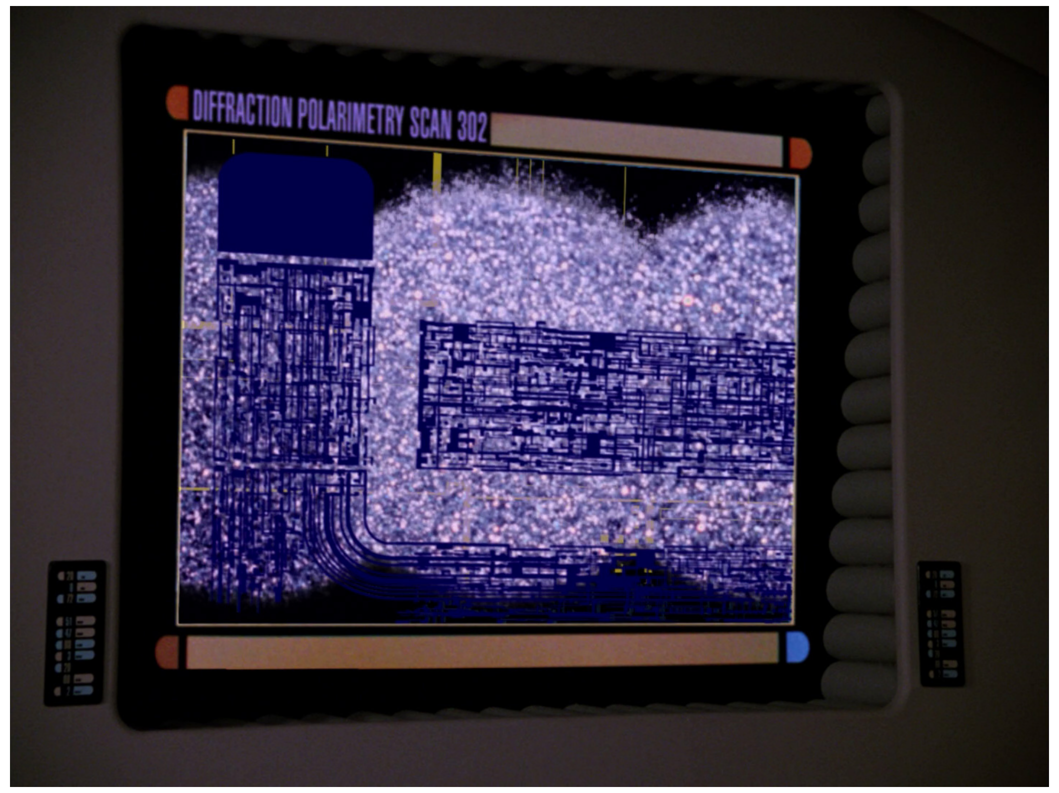
Turn imagination into purr-fection: Create your dream feline with our AI Cat Generator!

Cute fuzzy animals!

Taking the outsider's perspectives

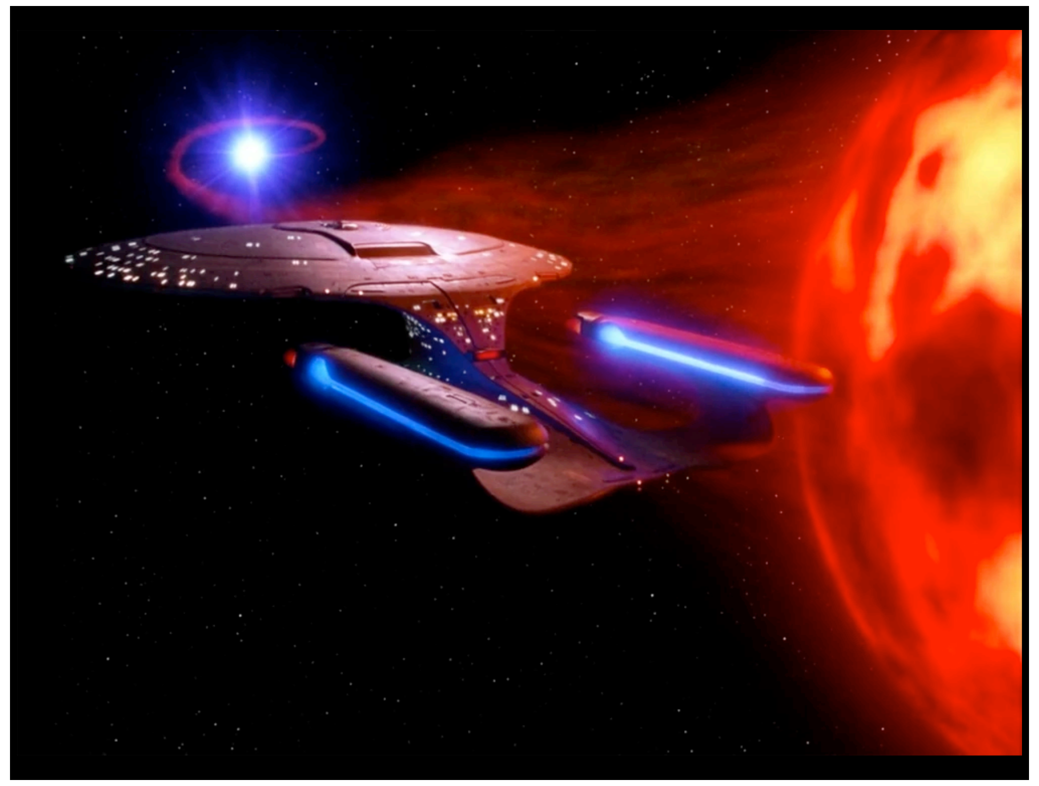
To seek out new life and new civilizations...

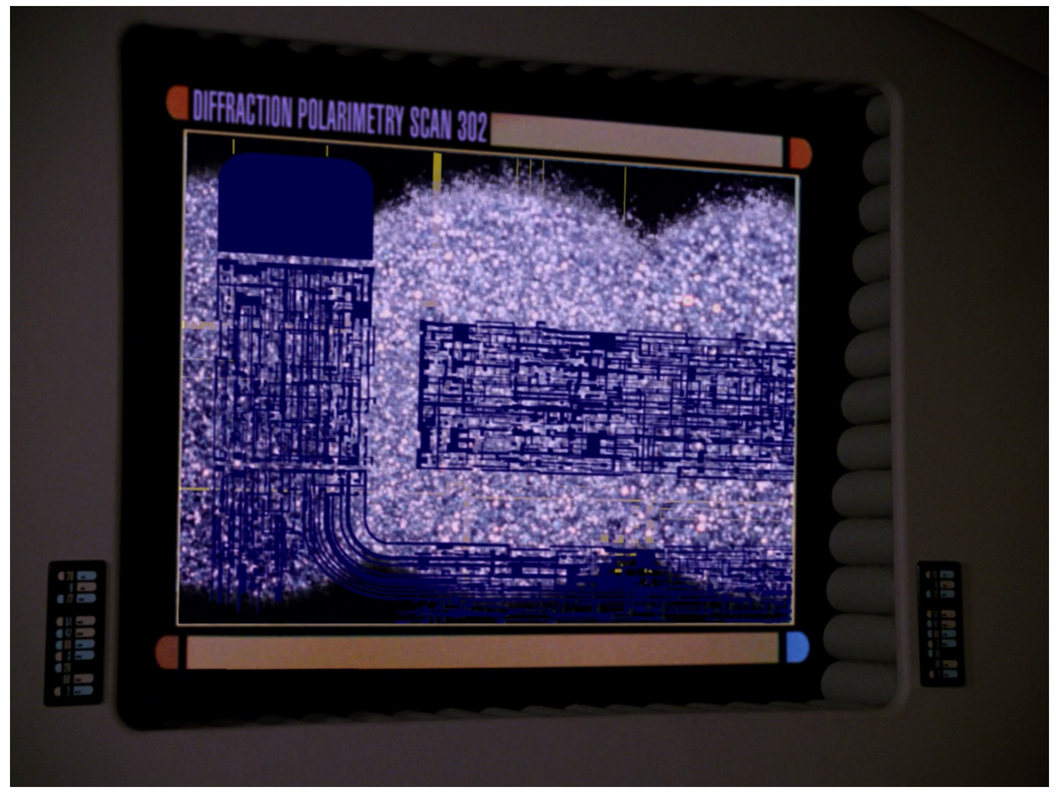




Taking the outsider's perspectives

To seek out new life and new civilizations…

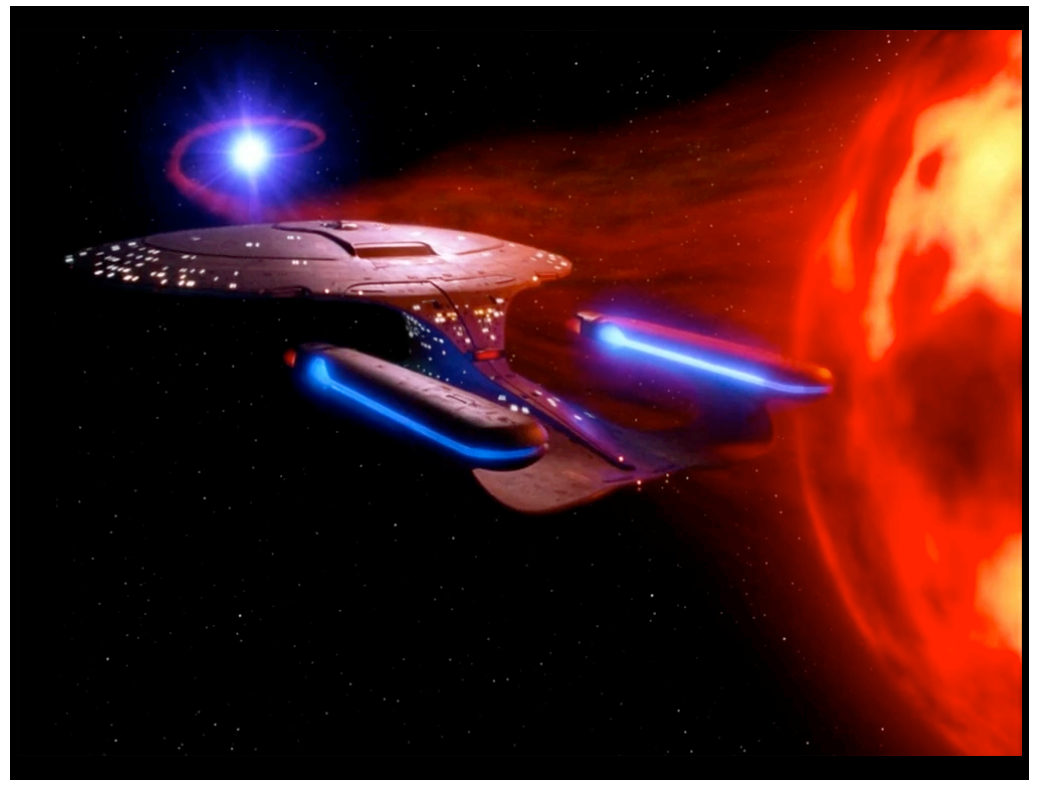


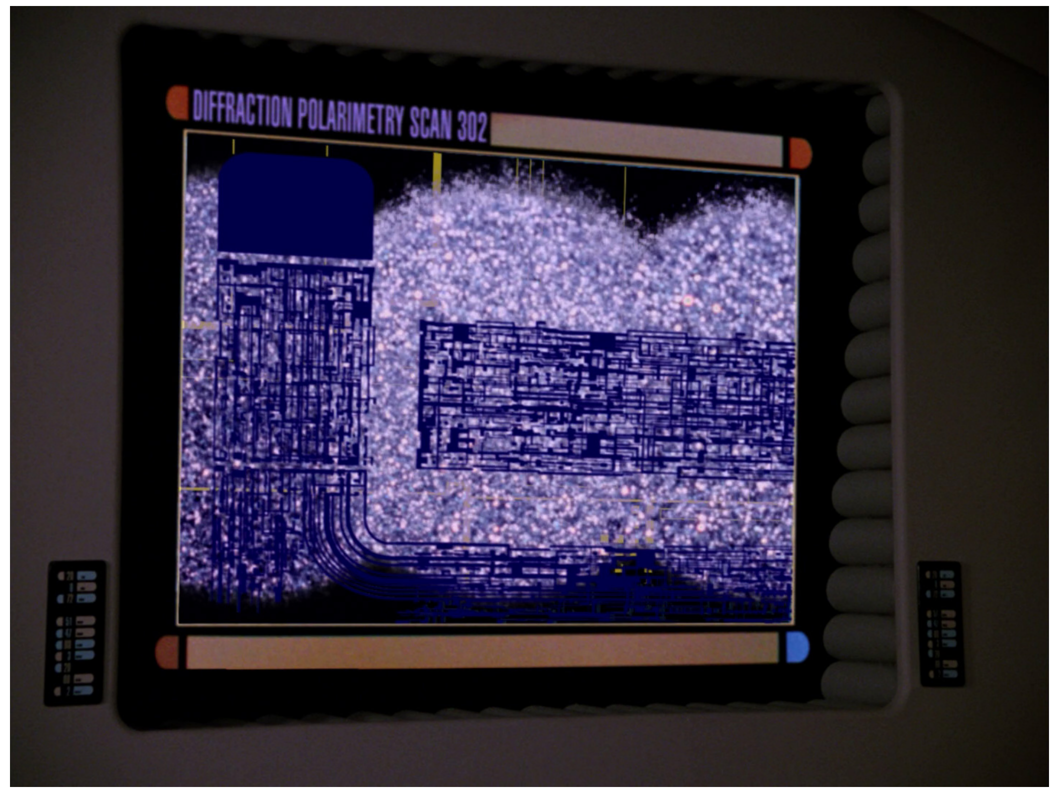


If we were landing on an alien planet and encountering these artifacts from "new life and new civilizations"…

Taking the outsider's perspectives

To seek out new life and new civilizations…



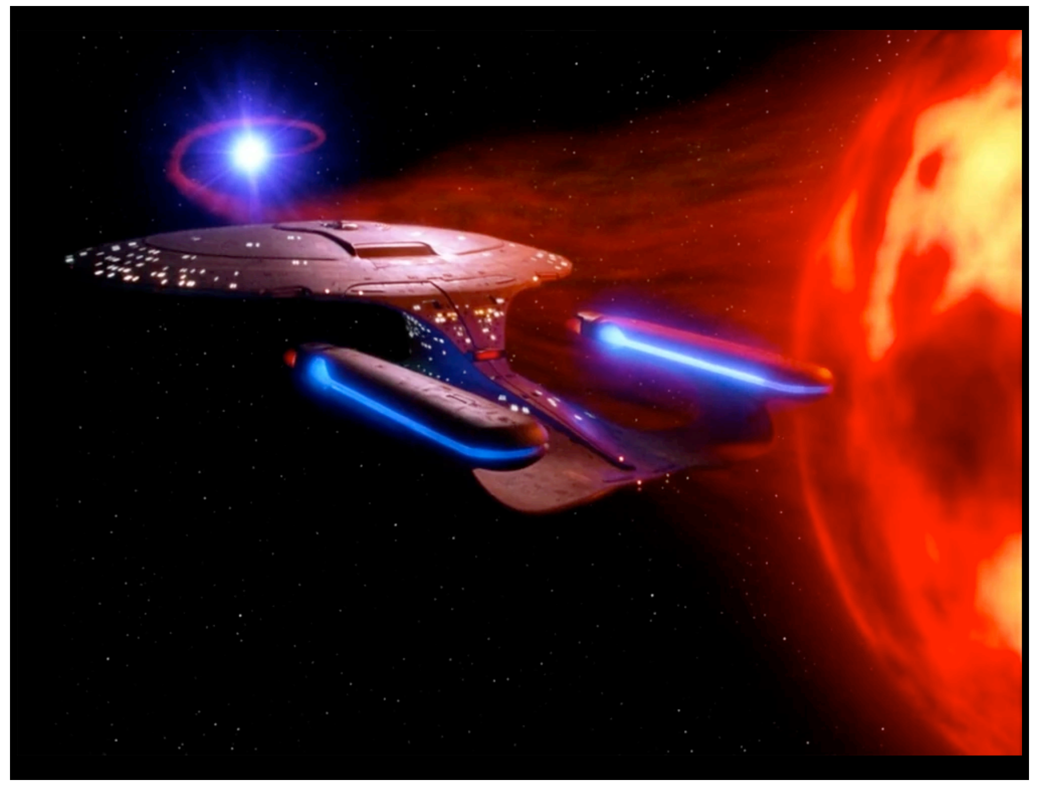


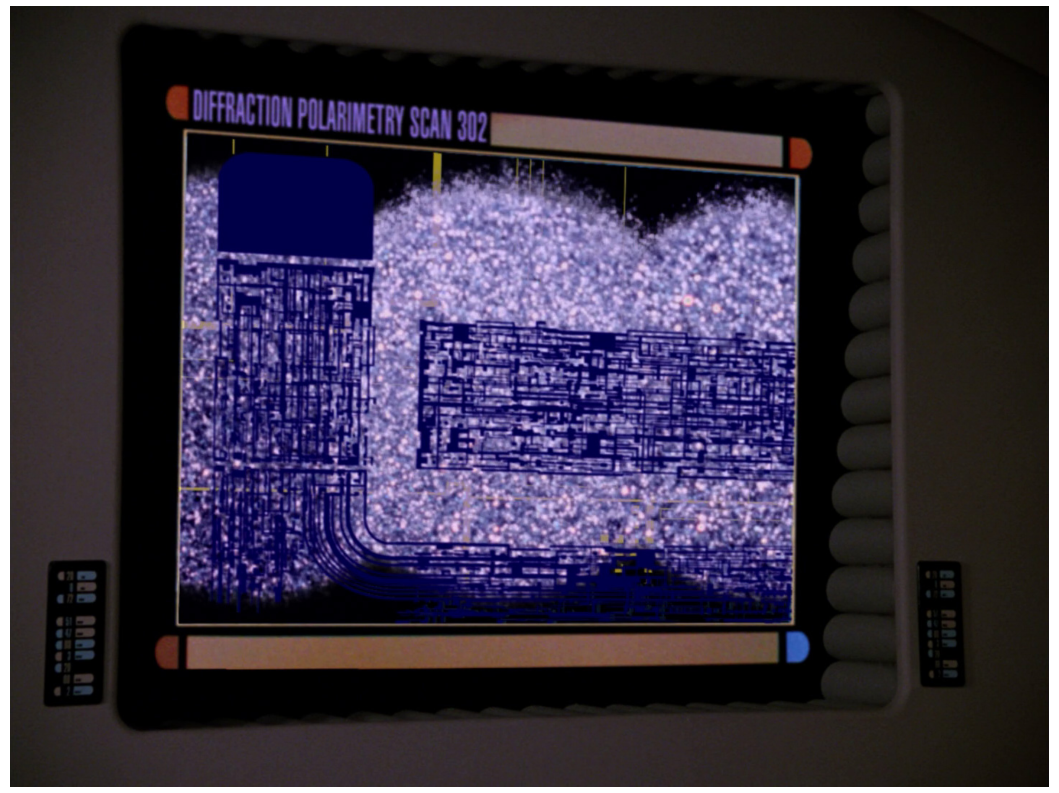
If we were landing on an alien planet and encountering these artifacts from "new life and new civilizations"…

- What can we learn from watching them learn?

Taking the outsider's perspectives

To seek out new life and new civilizations…



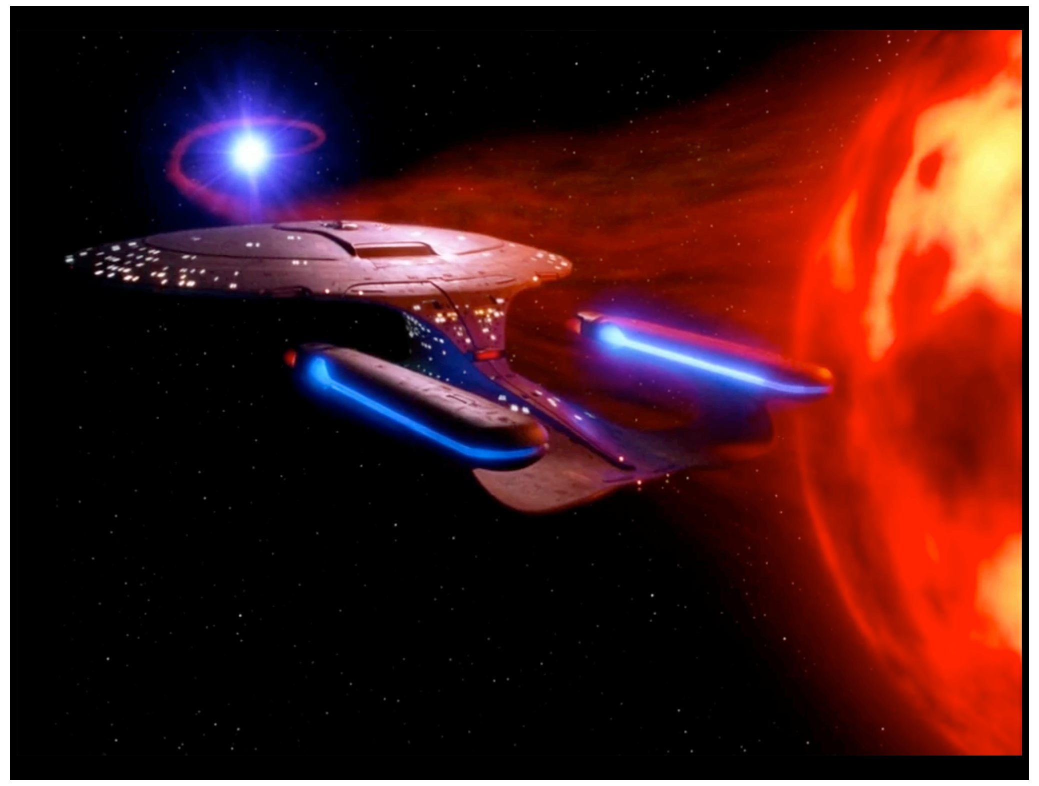


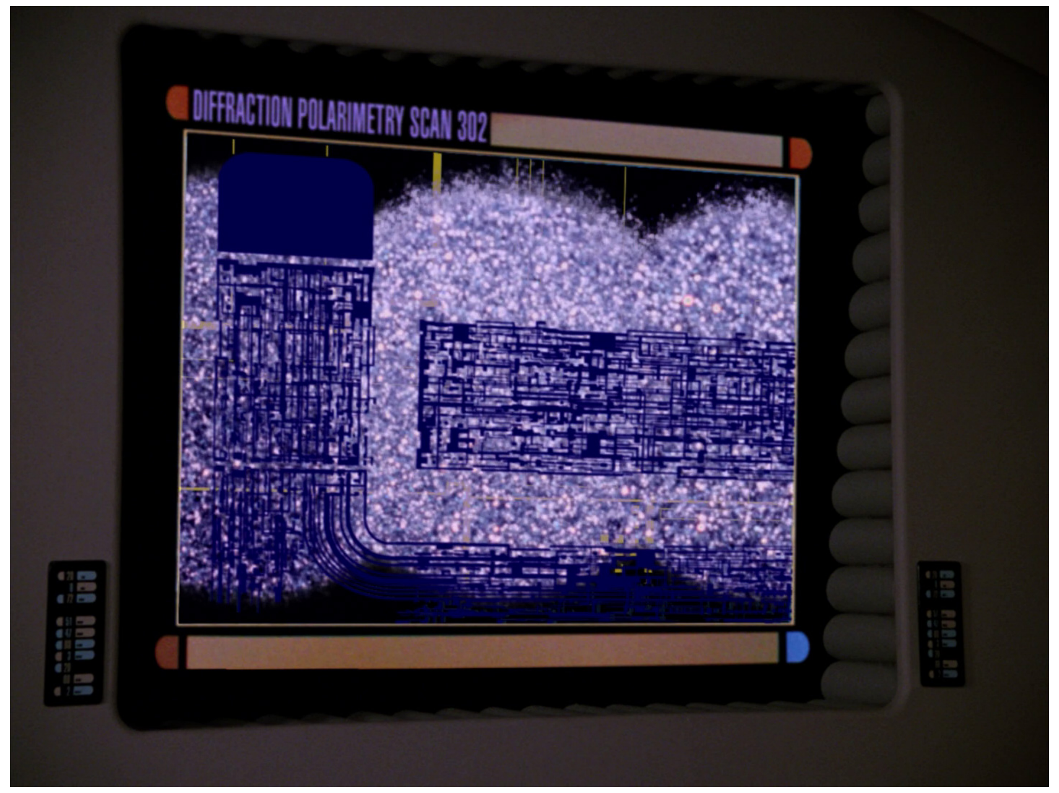
If we were landing on an alien planet and encountering these artifacts from "new life and new civilizations"…

- What can we learn from watching them learn?
- How can we understand what they are doing?

Taking the outsider's perspectives

To seek out new life and new civilizations…





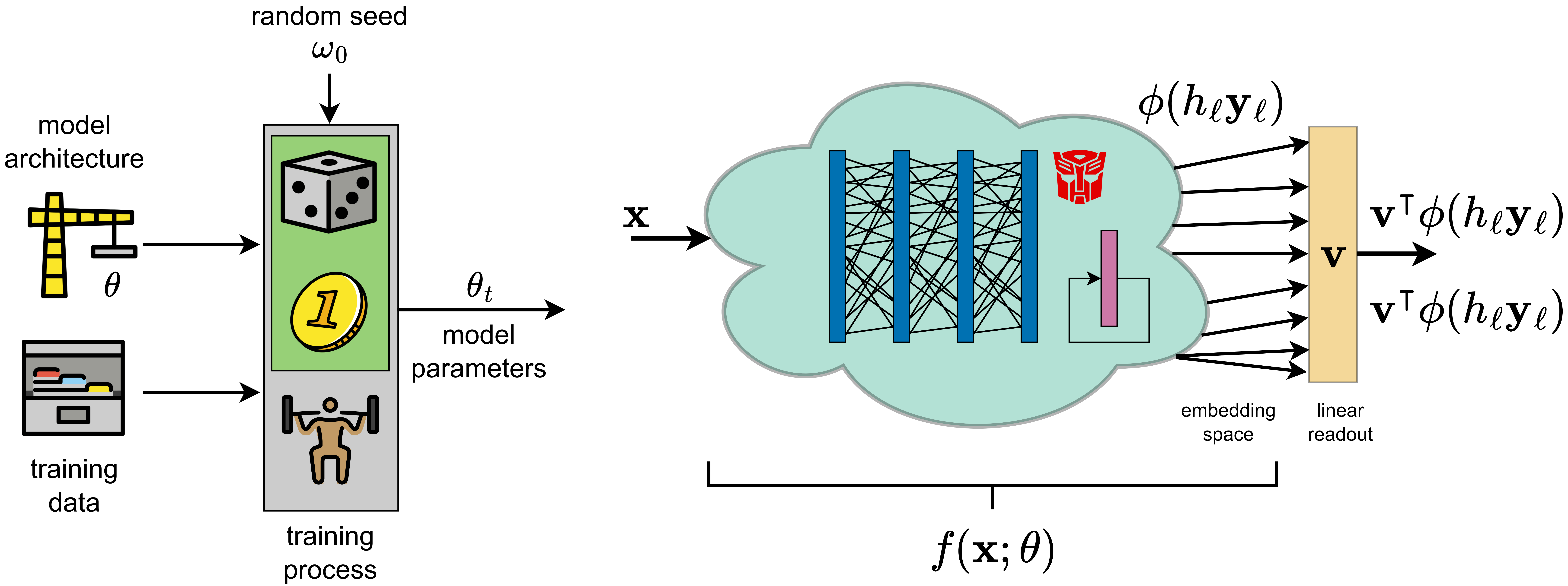
If we were landing on an alien planet and encountering these artifacts from "new life and new civilizations"…

- What can we learn from watching them learn?
- How can we understand what they are doing?

Big caveat: I am not going "where no-one has gone before"!

Two different processes

Building (training) models and using (pre-trained) models



This talk

A couple of forays in this direction

This talk

A couple of forays in this direction

I want to talk about a few different projects which are motivated by (but maybe do not achieve) some of these perspectives. In particular, we wanted to get some handle on:

This talk

A couple of forays in this direction

I want to talk about a few different projects which are motivated by (but maybe do not achieve) some of these perspectives. In particular, we wanted to get some handle on:

- If models are (randomly) trained in the same way, how different are they?

This talk

A couple of forays in this direction

I want to talk about a few different projects which are motivated by (but maybe do not achieve) some of these perspectives. In particular, we wanted to get some handle on:

- If models are (randomly) trained in the same way, how different are they?
- If models are trained differently, can we tell?

This talk

A couple of forays in this direction

I want to talk about a few different projects which are motivated by (but maybe do not achieve) some of these perspectives. In particular, we wanted to get some handle on:

- If models are (randomly) trained in the same way, how different are they?
- If models are trained differently, can we tell?
- Can we tell models apart by their “explanations”?

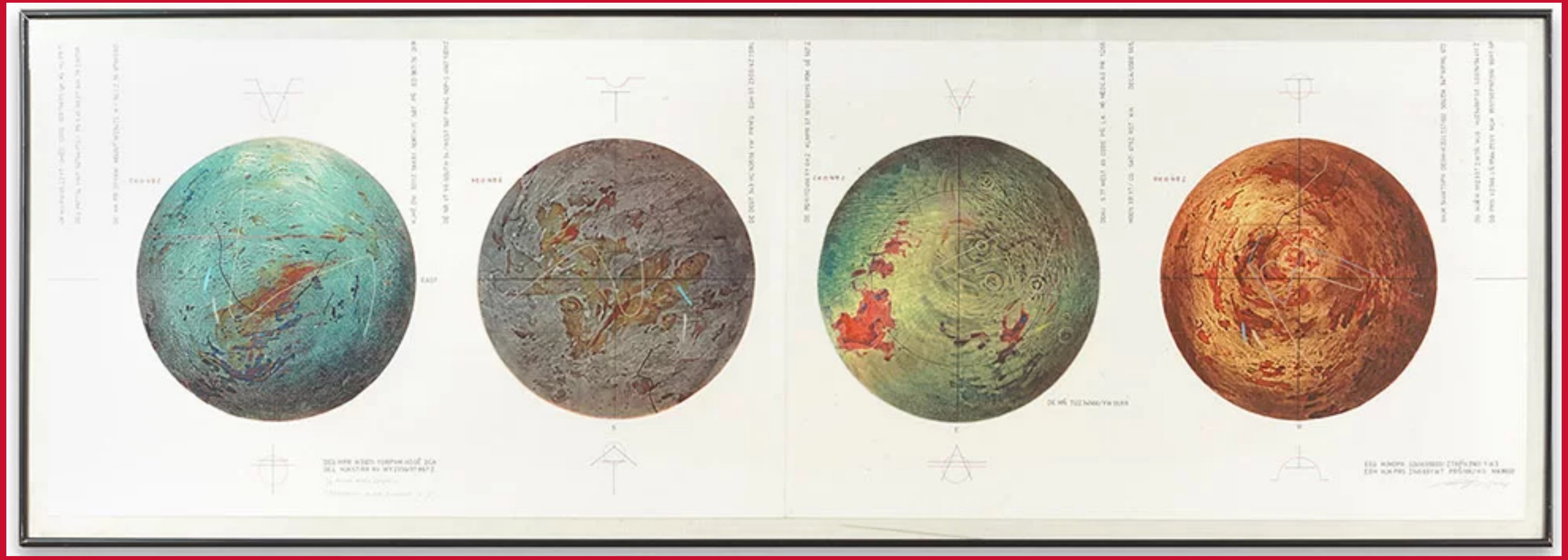
This talk

A couple of forays in this direction

I want to talk about a few different projects which are motivated by (but maybe do not achieve) some of these perspectives. In particular, we wanted to get some handle on:

- If models are (randomly) trained in the same way, how different are they?
- If models are trained differently, can we tell?
- Can we tell models apart by their “explanations”?
- Can we tell the difference between models “off the shelf”?

Testing variability in training



Rm Palaniappan, *Alien Planet-A*
Viscosity, pencil colour and ink on handmade paper

Are these instruments equally good?

Or is it *caveat emptor*?

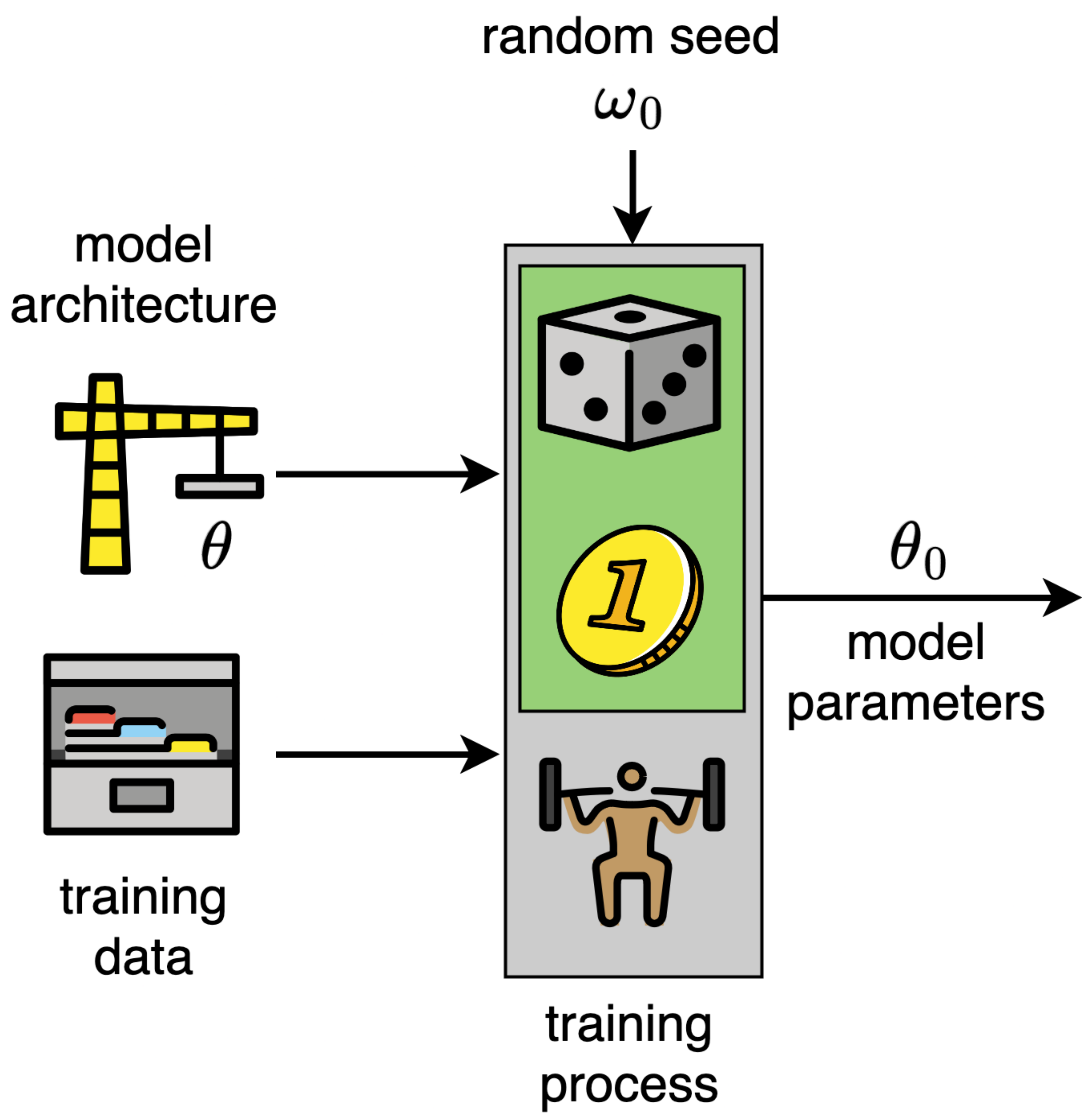
Lt. Cmdr. Data and his “brother” Lore

Training large models usually involves **stochastic optimization**:

- Each run produces a different model!
  - same architecture
  - same training data
  - same hyperparameters
- Hard to determine if changing these factors makes any difference.

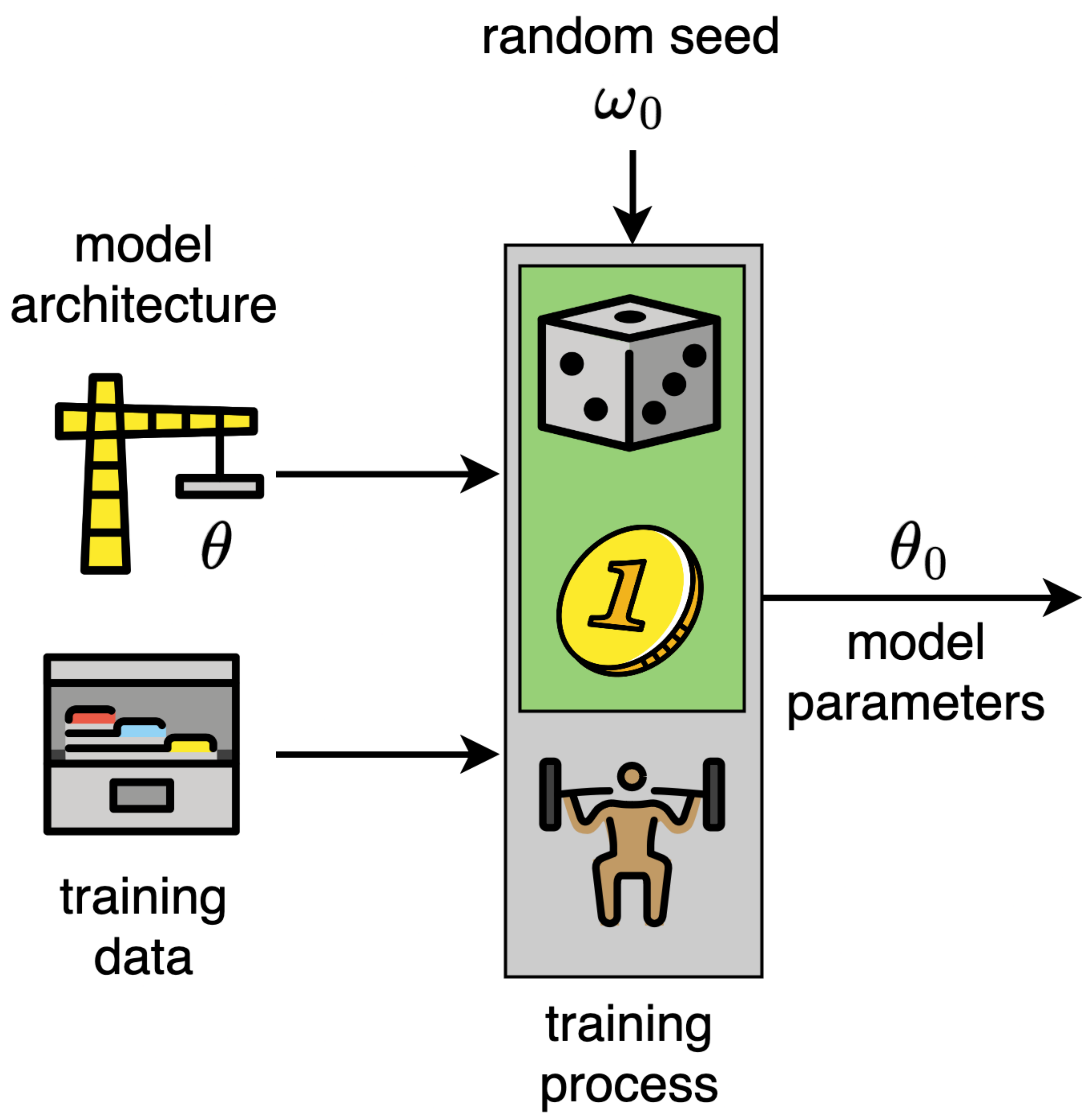
The standard statistical setup for modern ML

Machine learning as function-fitting



The standard statistical setup for modern ML

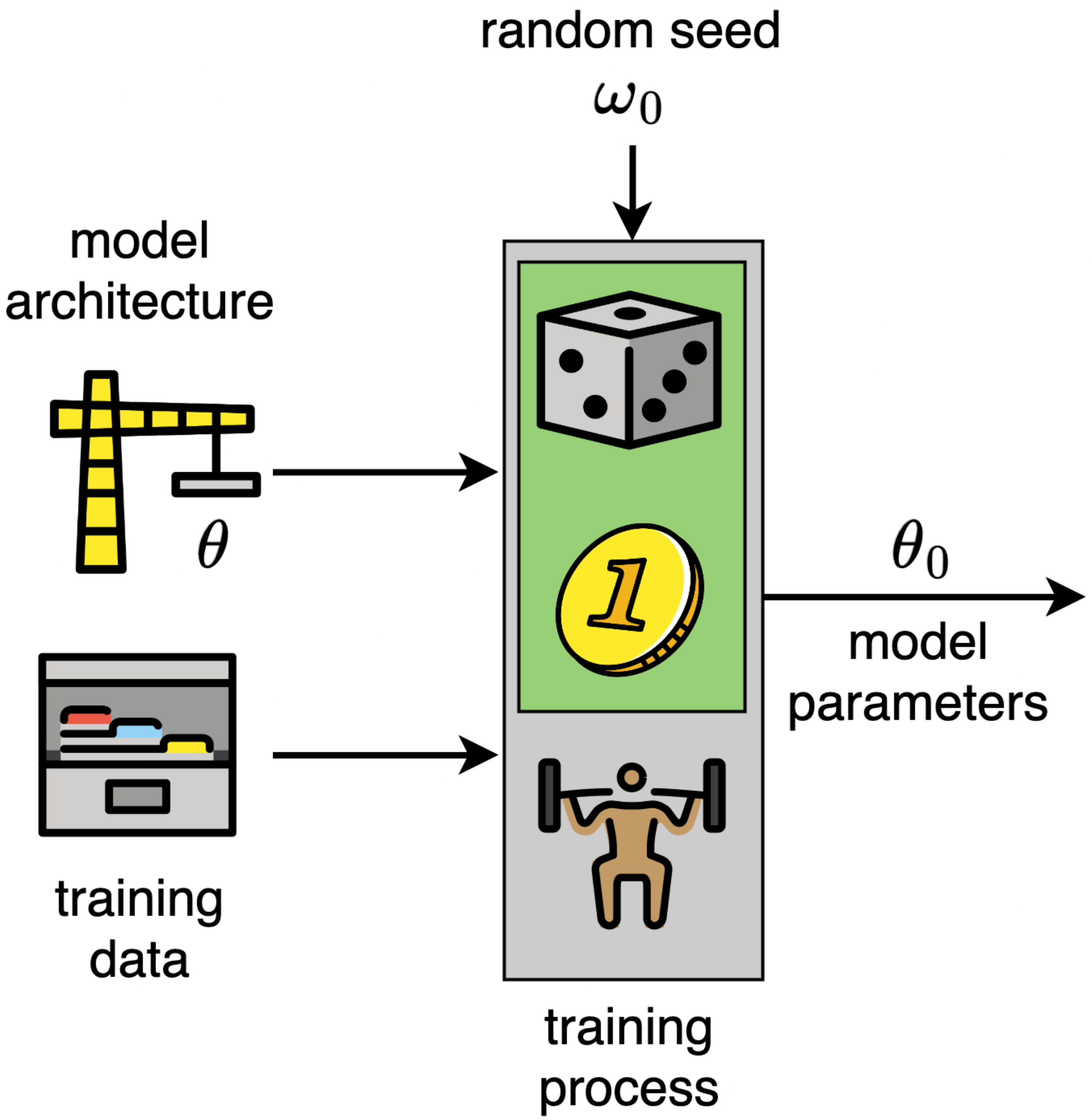
Machine learning as function-fitting



The traditional setup for estimating parameters in a statistical model (or training a neural network:

The standard statistical setup for modern ML

Machine learning as function-fitting

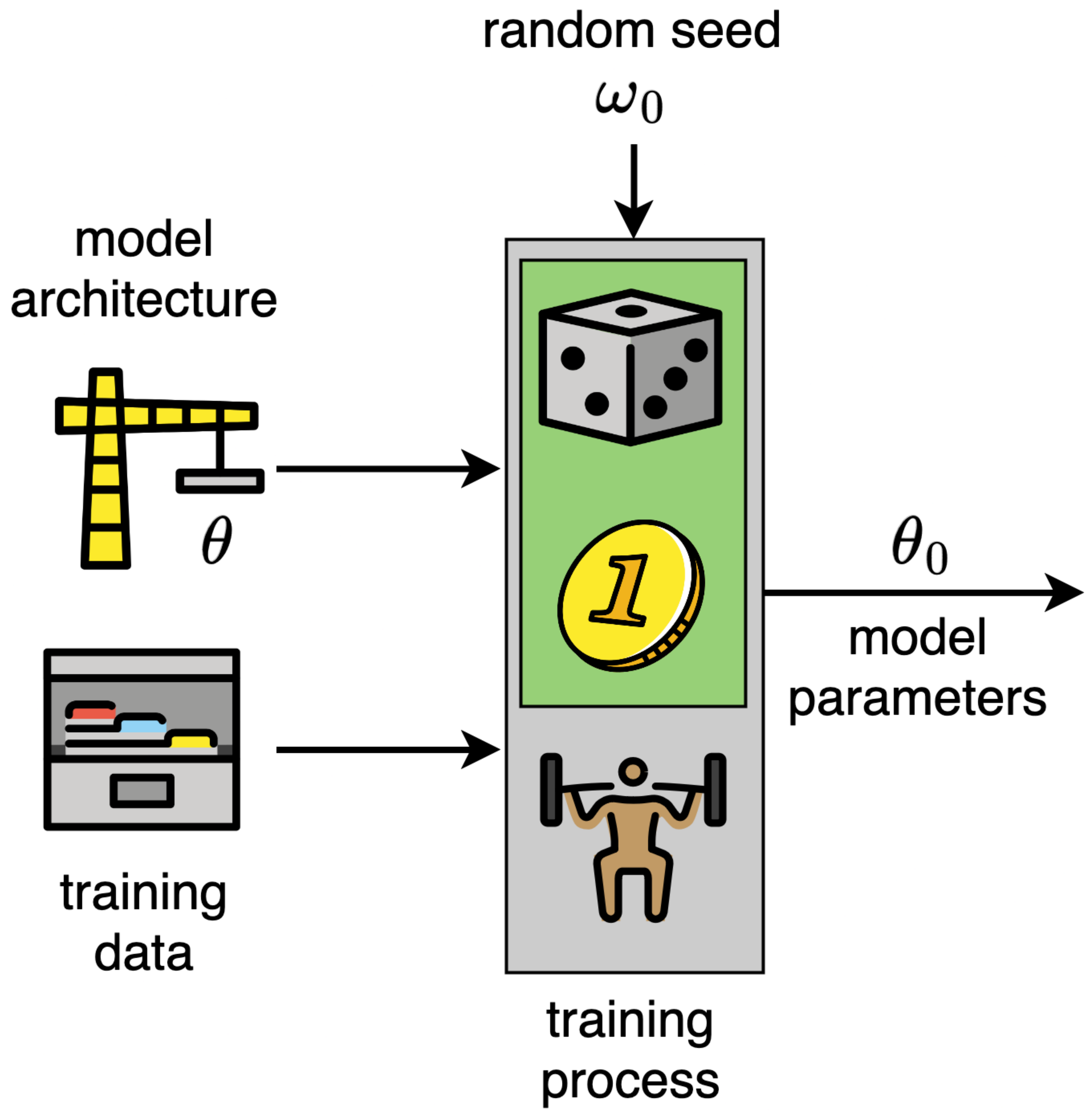


The traditional setup for estimating parameters in a statistical model (or training a neural network:

- Parameterized set of functions/models $\{f(x \,|\, \theta) : \theta \in \Theta\}$.

The standard statistical setup for modern ML

Machine learning as function-fitting

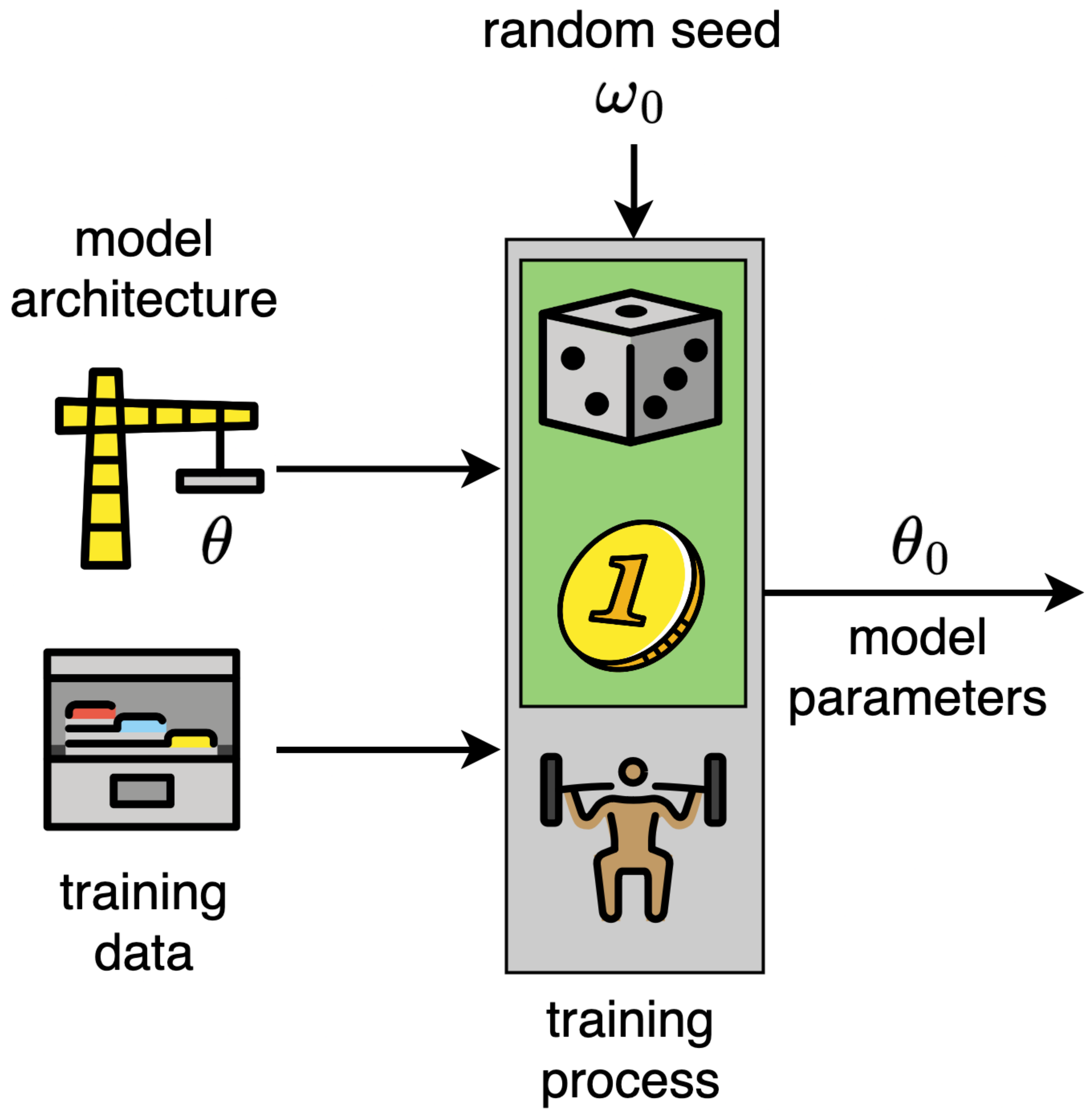


The traditional setup for estimating parameters in a statistical model (or training a neural network:

- Parameterized set of functions/models $\{f(x \mid \theta) : \theta \in \Theta\}$.
- Training data used to estimate the parameters by minimizing some objective function.

The standard statistical setup for modern ML

Machine learning as function-fitting



The traditional setup for estimating parameters in a statistical model (or training a neural network:

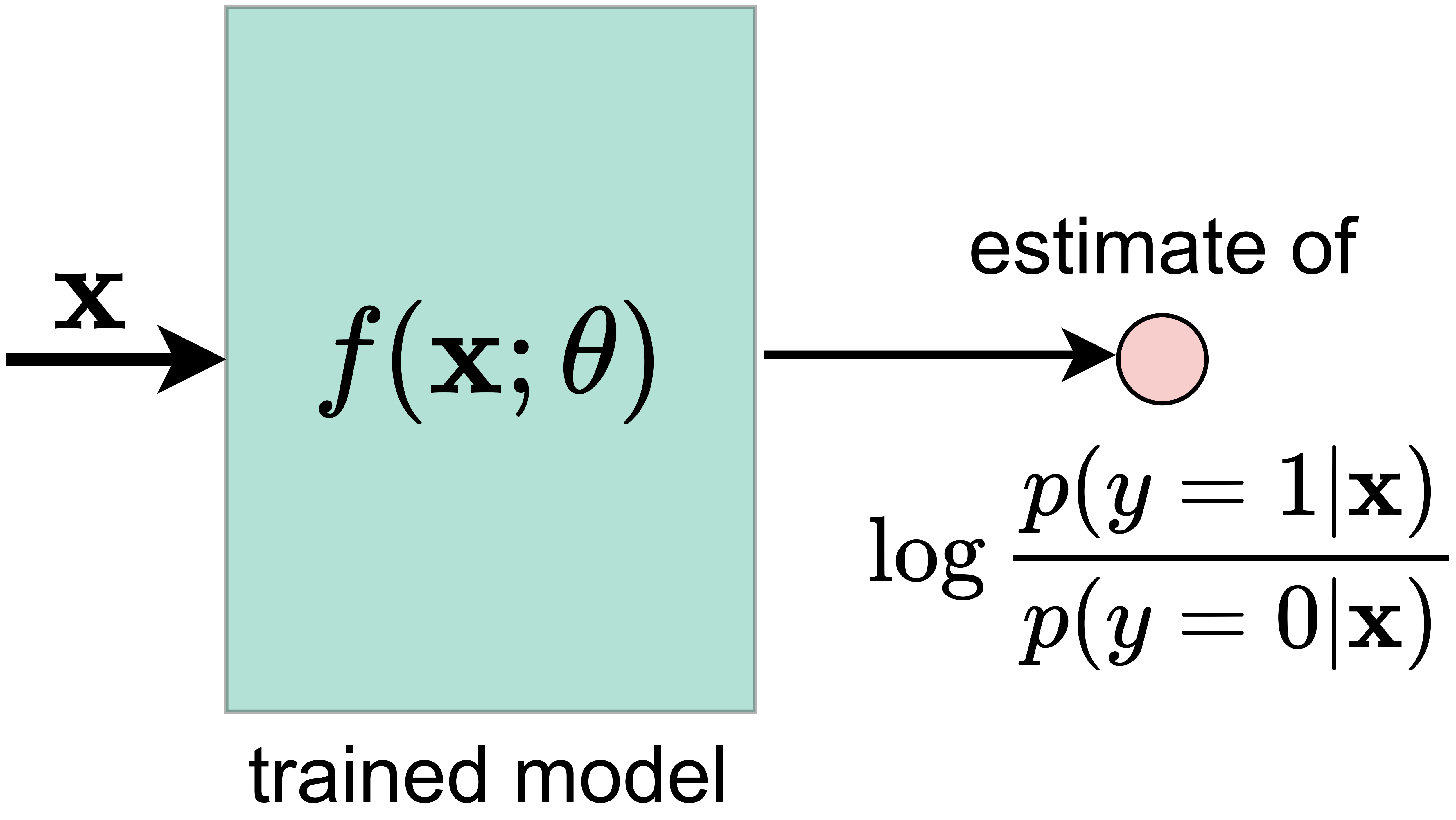
- Parameterized set of functions/models $\{f(x \mid \theta) : \theta \in \Theta\}$.
- Training data used to estimate the parameters by minimizing some objective function.
- Stochastic optimization algorithm that does the actual minimization.

The simplest case: binary classifiers

Learning a function with scalar output

The simplest case: binary classifiers

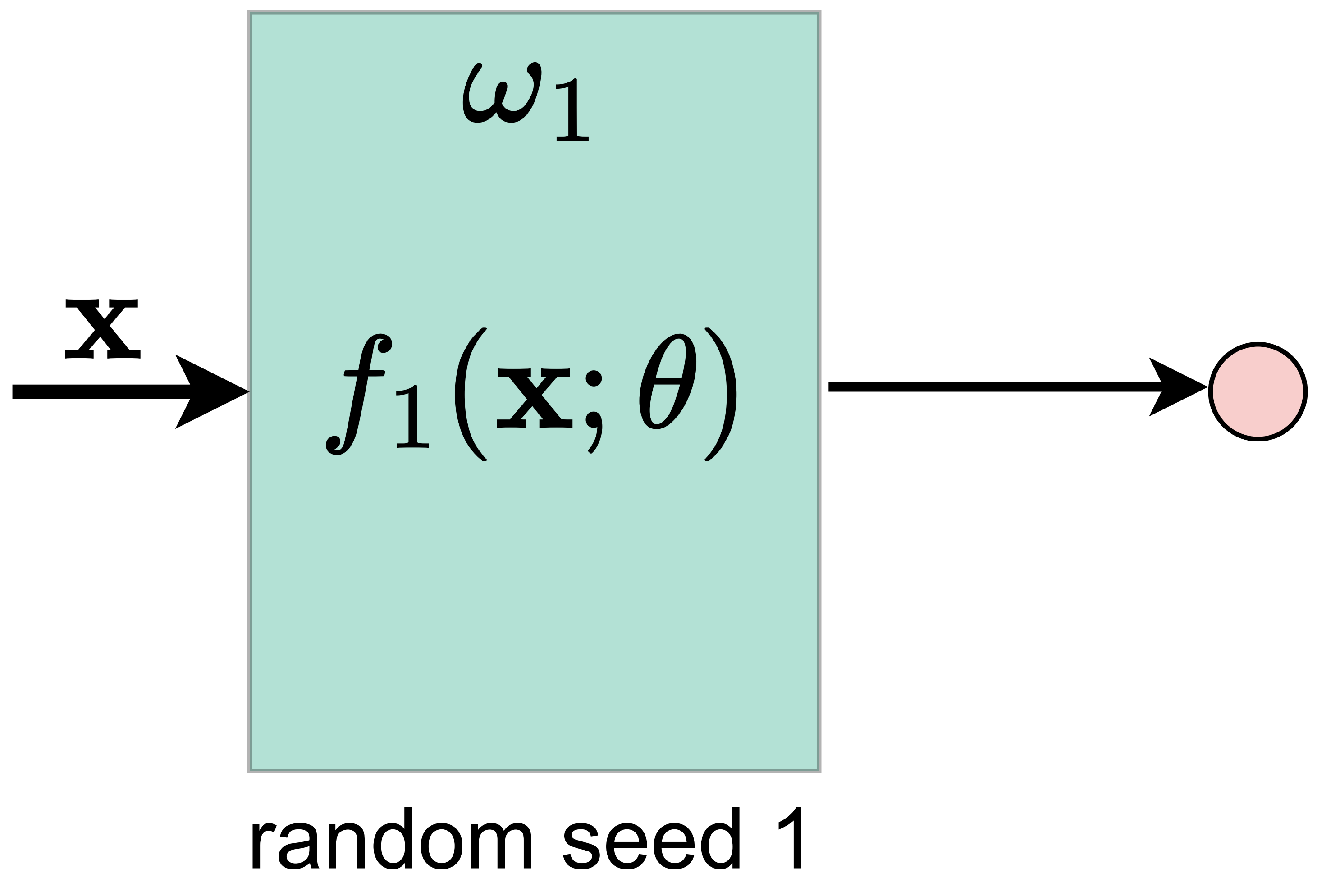
Learning a function with scalar output

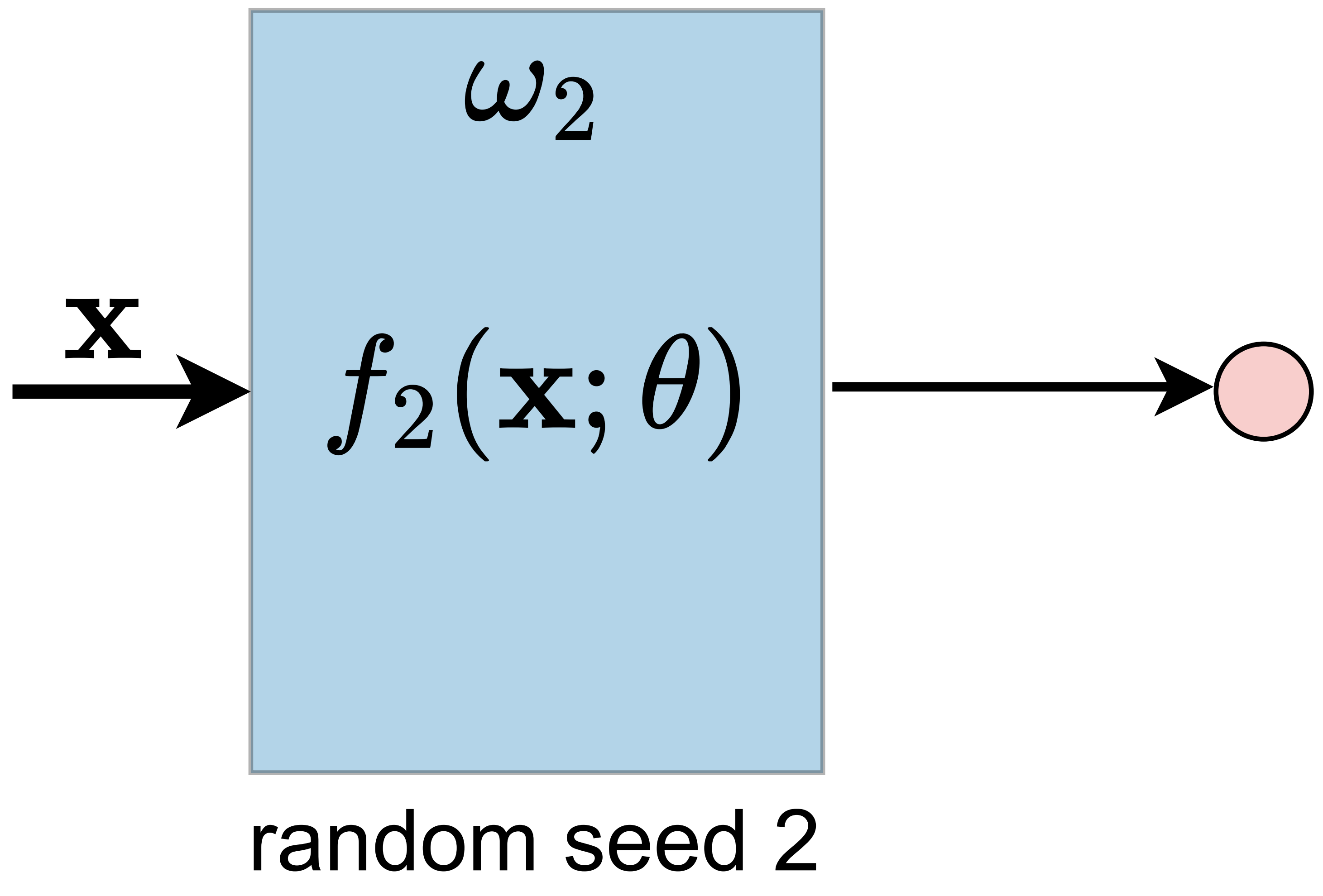


Let’s interpret the “soft” output as an estimate of some log likelihood ratio given by the trained model.

The simplest case: binary classifiers

Learning a function with scalar output



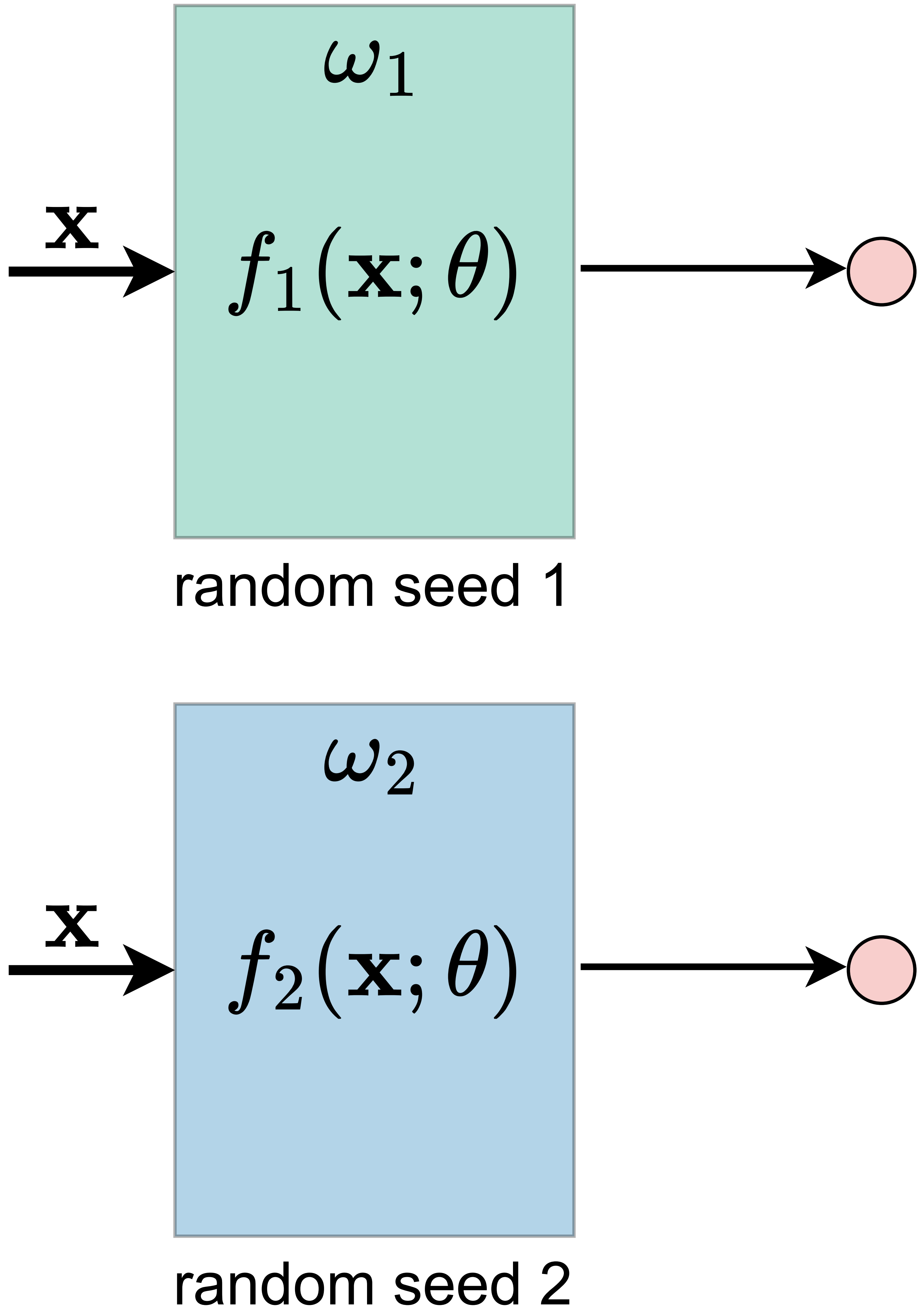


Let's interpret the "soft" output as an estimate of some log likelihood ratio given by the trained model.

For two models trained with two different seeds, are they "similar"?

The simplest case: binary classifiers

Learning a function with scalar output



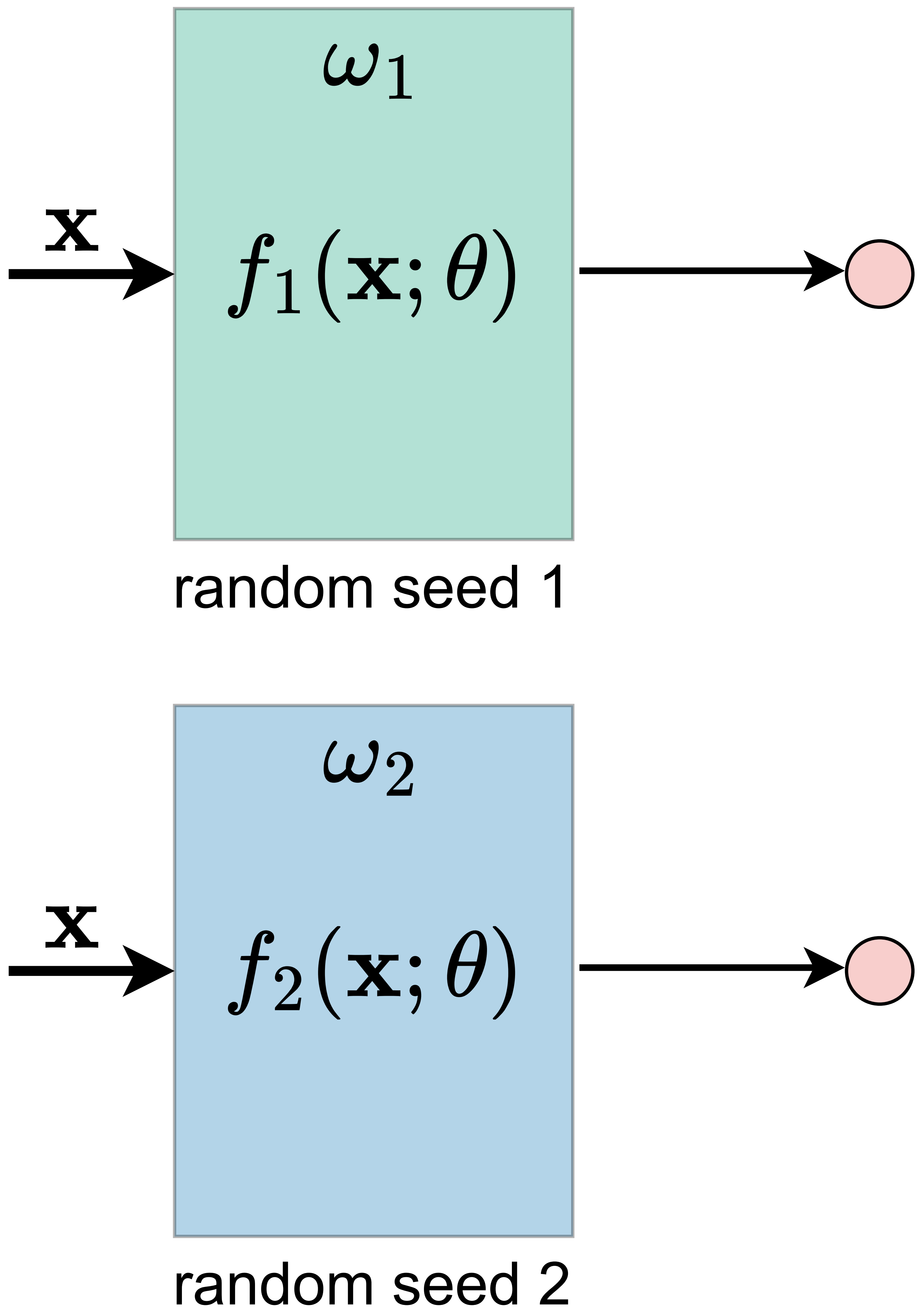
Let's interpret the "soft" output as an estimate of some log likelihood ratio given by the trained model.

For two models trained with two different seeds, are they "similar"?

- Same test **accuracy**?

The simplest case: binary classifiers

Learning a function with scalar output



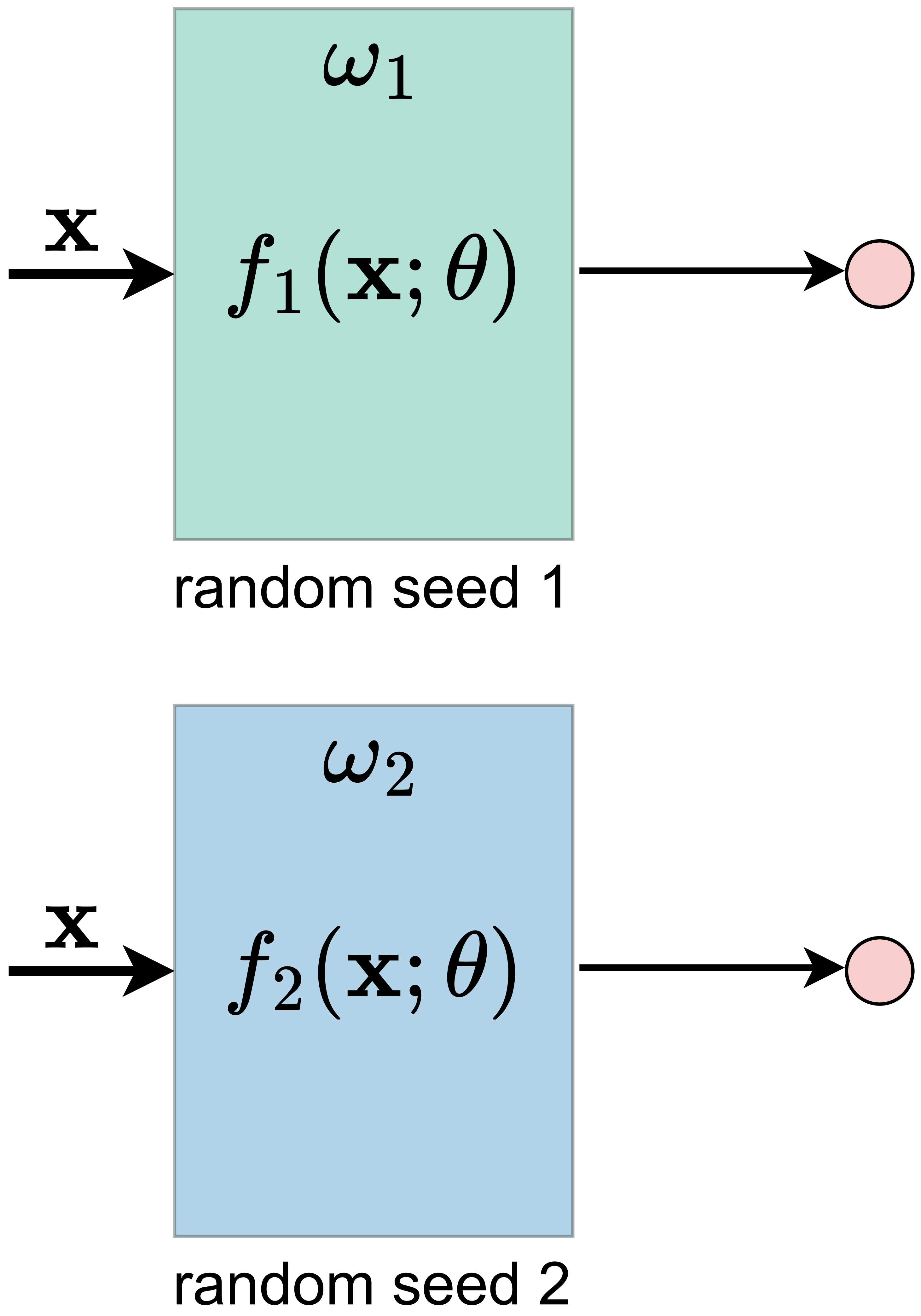
Let's interpret the "soft" output as an estimate of some log likelihood ratio given by the trained model.

For two models trained with two different seeds, are they "similar"?

- Same test **accuracy**?
- Same mistakes (low **churn**)?

The simplest case: binary classifiers

Learning a function with scalar output



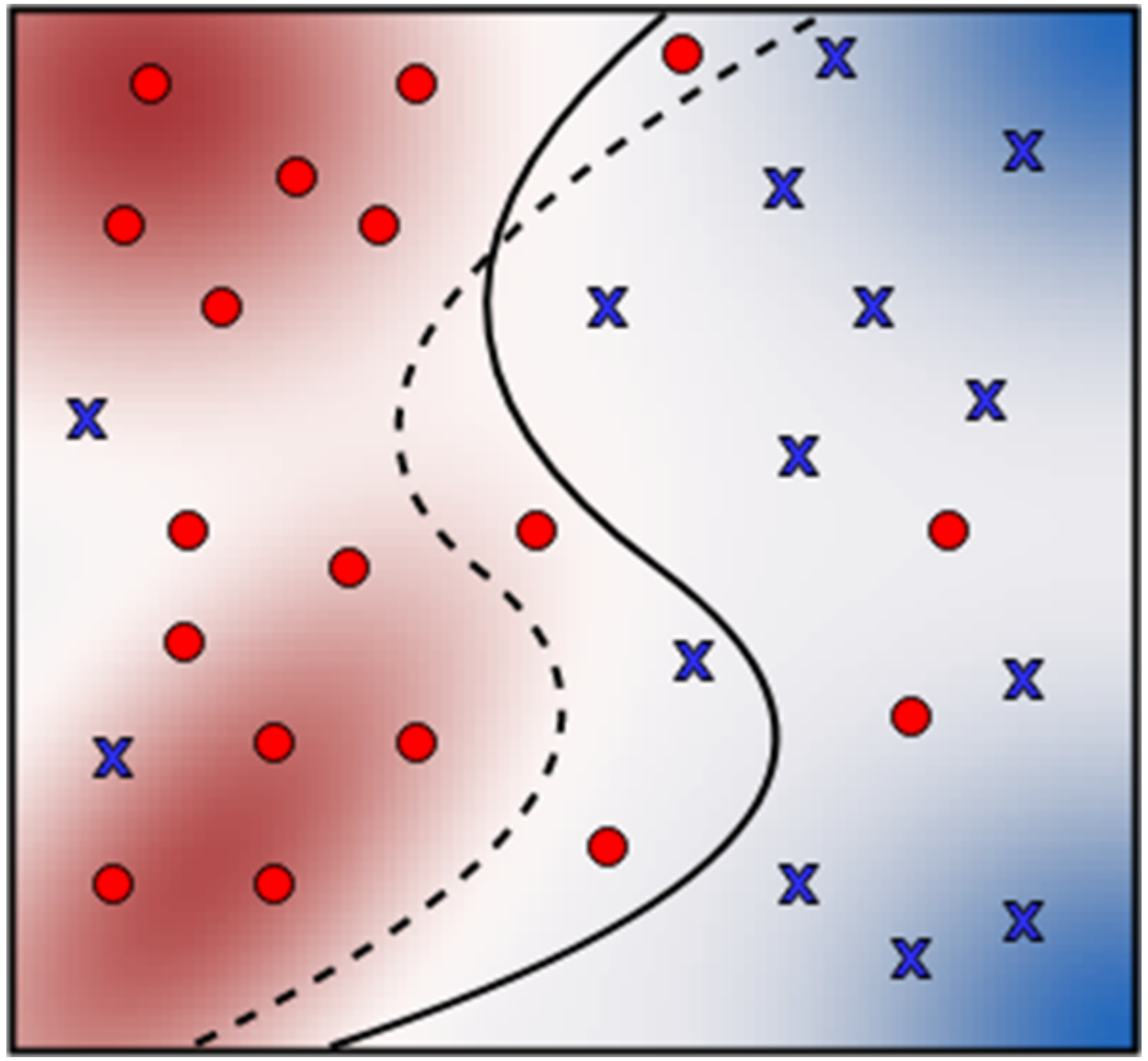
Let's interpret the "soft" output as an estimate of some log likelihood ratio given by the trained model.

For two models trained with two different seeds, are they "similar"?

- Same test **accuracy**?
- Same mistakes (low **churn**)?
- Close in some norm?

This is not a new question

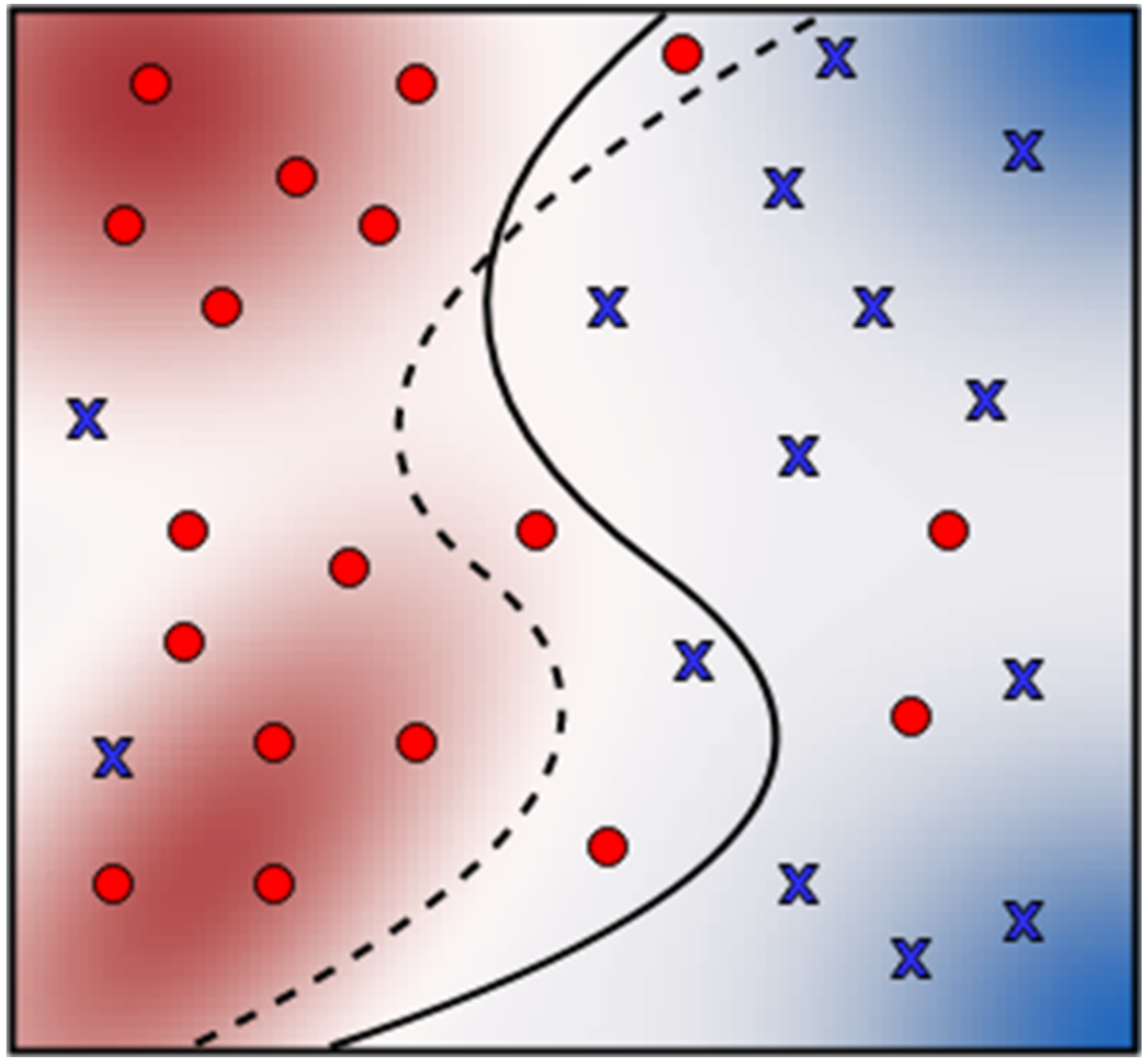
Model comparisons are ad hoc and waste energy



This is not a new question

Model comparisons are ad hoc and waste energy

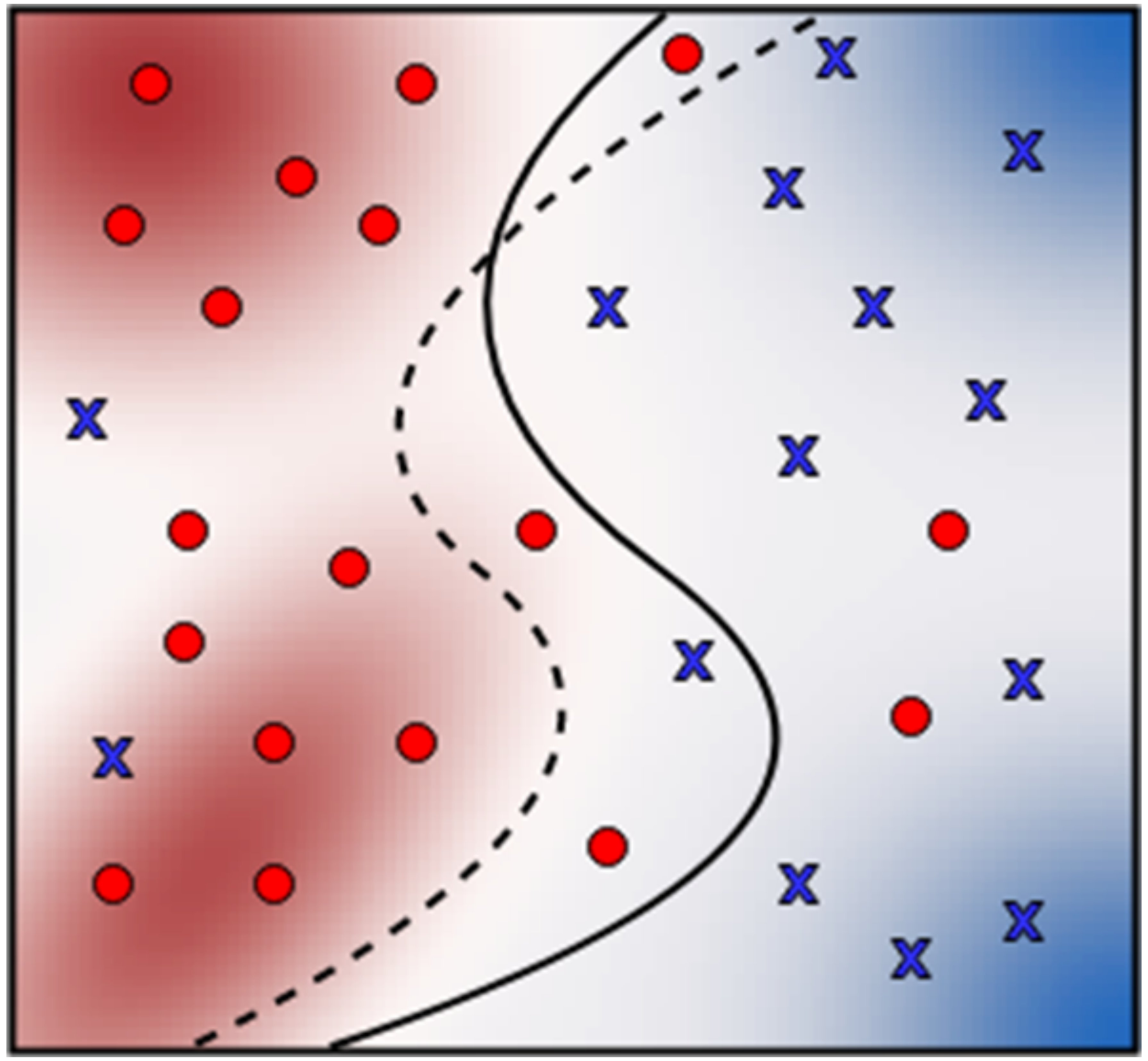
- Determining if one model is "better" than another is not well-posed.



This is not a new question

Model comparisons are ad hoc and waste energy

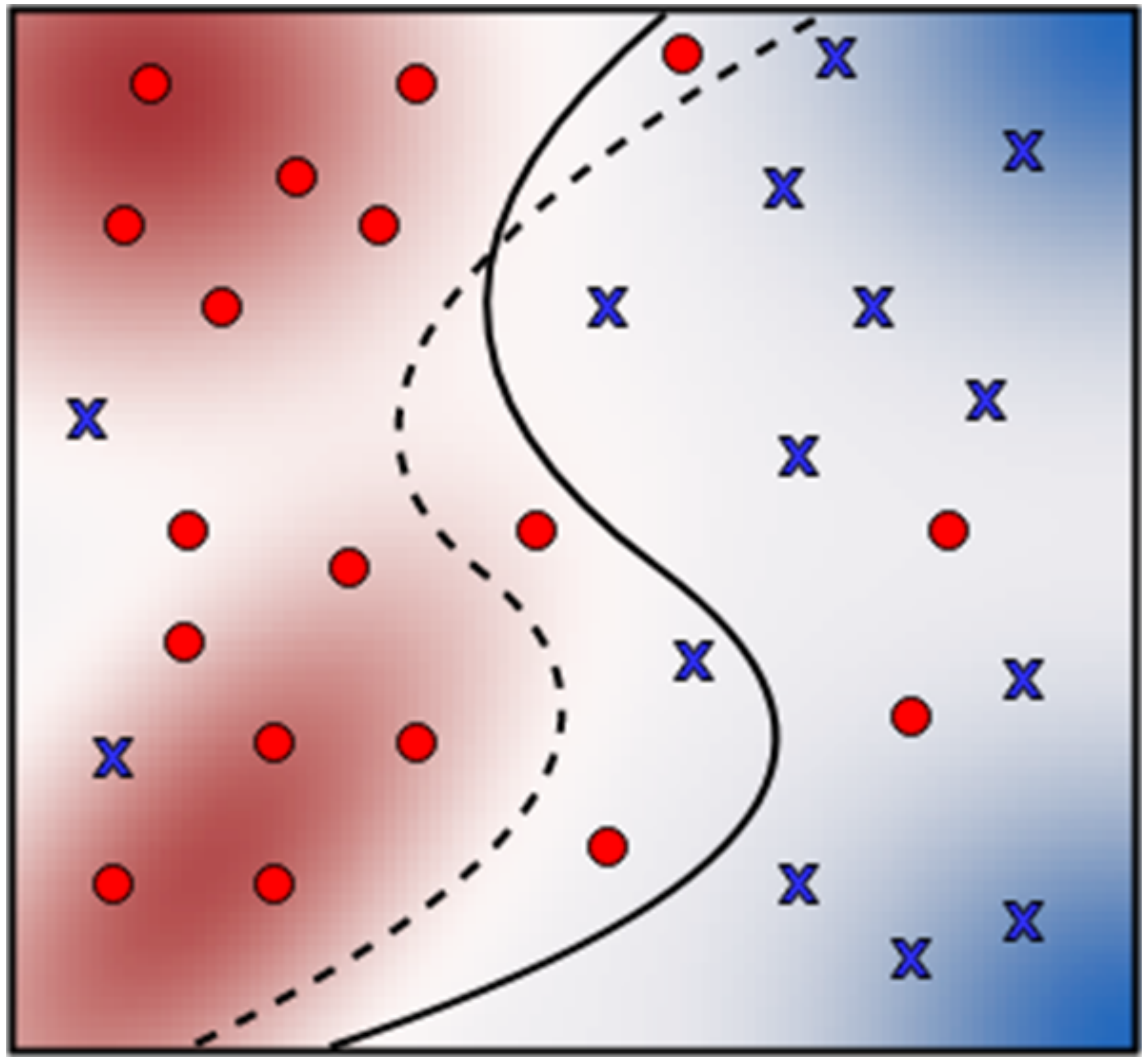
- Determining if one model is "better" than another is not well-posed.
- In practice, end up running the training process many times. Wasted computation, time, energy, etc.



This is not a new question

Model comparisons are ad hoc and waste energy

- Determining if one model is "better" than another is not well-posed.
- In practice, end up running the training process many times. Wasted computation, time, energy, etc.



Terms like the Rashomon effect[1][2][3], predictive multiplicity[4], or prediction churn[5] have been coined in the literature to explain this phenomena.

[1] Breiman, L. (2001). Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical science*, *16*(3), 199-231

[2] Fisher, A., Rudin, C., & Dominici, F. (2019). All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *Journal of Machine Learning Research*, *20*(177), 1-81.

[3] Hsu, H., & Calmon, F. (2022). Rashomon capacity: A metric for predictive multiplicity in classification. *Advances in Neural Information Processing Systems*, *35*, 28988-29000.

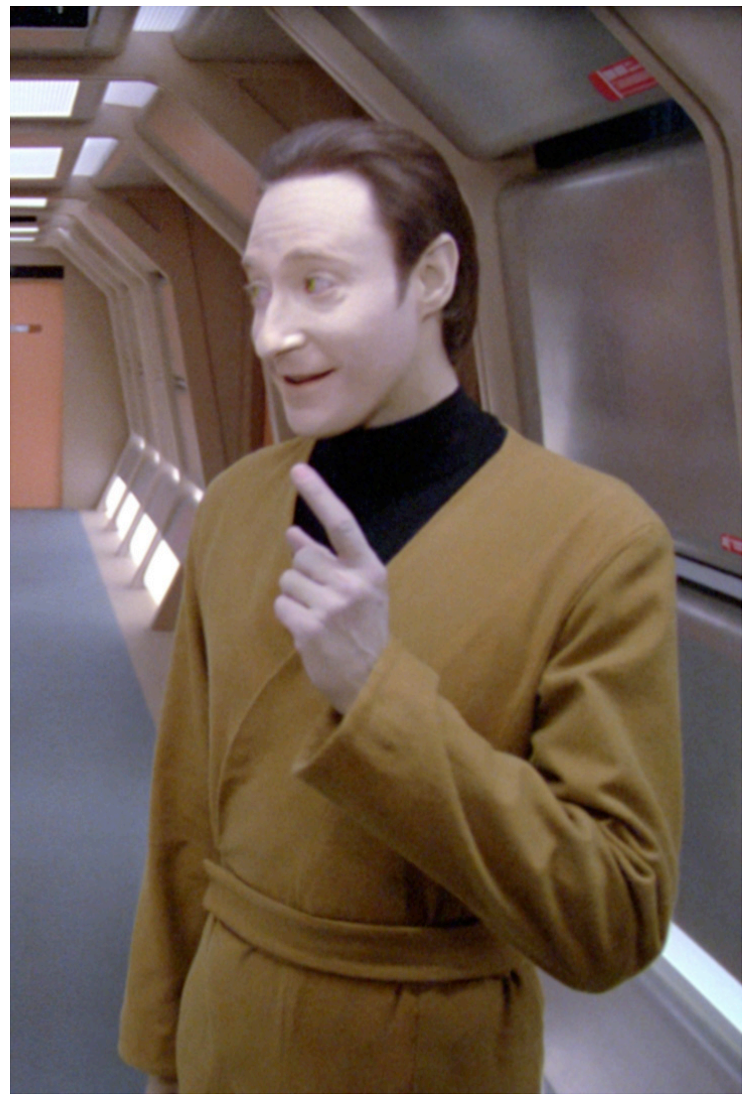
[4] Milani Fard, M., Cormier, Q., Canini, K., & Gupta, M. (2016). Launch and iterate: Reducing prediction churn. *Advances in Neural Information Processing Systems*, *29*.

[5] Marx, C., Calmon, F., & Ustun, B. (2020, November). Predictive multiplicity in classification. In *International Conference on Machine Learning* (pp. 6765-6774). PMLR.

Ask instead: are these models different?

Back to simple tools: hypothesis testing

VS.



Two models, trained the same way: are they the same? This is a 2 sample test!

$$\mathscr{H}_0 : f_0(x;\theta) = f_1(x;\theta)$$

$$\mathscr{H}_1 : f_1(x;\theta) \neq f_2(x;\theta)$$

Comparing the two distributions

Lots of choices

Comparing the two distributions

Lots of choices

Random seeds are independent so $f(x;\theta_0)$ and $f(x;\theta_1)$ are iid draws from

Comparing the two distributions

Lots of choices

Random seeds are independent so $f(x; \theta_0)$ and $f(x; \theta_1)$ are iid draws from

$$\mathcal{F} = \{f : f \text{ representable by the NN}\}$$

Comparing the two distributions

Lots of choices

Random seeds are independent so $f(x;\theta_0)$ and $f(x;\theta_1)$ are iid draws from

$$\mathscr{F} = \{f : f \text{ representable by the NN}\}$$

Use the **test set** $\{x'_1, x'_2, \ldots, x'_N\}$ and a **Kolmogorov-Smirnoff (KS) test** on the empirical CDFs of $\{f(x'_i;\theta_1)\}$ and $\{f(x'_i;\theta_2)\}$.

Comparing the two distributions

Lots of choices

Random seeds are independent so $f(x; \theta_0)$ and $f(x; \theta_1)$ are iid draws from

$$\mathcal{F} = \{f : f \text{ representable by the NN}\}$$

Use the **test set** $\{x'_1, x'_2, \ldots, x'_N\}$ and a **Kolmogorov-Smirnoff (KS) test** on the empirical CDFs of $\{f(x'_i; \theta_1)\}$ and $\{f(x'_i; \theta_2)\}$.

Issue 1: The alternative is always true: the models are different.

Comparing the two distributions

Lots of choices

Random seeds are independent so $f(x;\theta_0)$ and $f(x;\theta_1)$ are iid draws from

$$\mathcal{F} = \{f : f \text{ representable by the NN}\}$$

Use the **test set** $\{x_1', x_2', \ldots, x_N'\}$ and a **Kolmogorov-Smirnoff (KS) test** on the empirical CDFs of $\{f(x_i';\theta_1)\}$ and $\{f(x_i';\theta_2)\}$.

Issue 1: The alternative is always true: the models are different.

Issue 2: Can we use a 1 sample test instead? Don't have a good estimate of the null.

Comparing the two distributions

Lots of choices

Random seeds are independent so $f(x;\theta_0)$ and $f(x;\theta_1)$ are iid draws from

$$\mathscr{F} = \{f : f \text{ representable by the NN}\}$$

Use the **test set** $\{x'_1, x'_2, \ldots, x'_N\}$ and a **Kolmogorov-Smirnoff (KS) test** on the empirical CDFs of $\{f(x'_i;\theta_1)\}$ and $\{f(x'_i;\theta_2)\}$.

Issue 1: The alternative is always true: the models are different.

Issue 2: Can we use a 1 sample test instead? Don't have a good estimate of the null.

Issue 3: Shouldn't we use the tools from Bhaswar's talk on Monday???

Comparing the two distributions
Lots of choices

Random seeds are independent so $f(x;\theta_0)$ and $f(x;\theta_1)$ are iid draws from

$$\mathscr{F} = \{f : f \text{ representable by the NN}\}$$

Use the **test set** $\{x'_1, x'_2, \ldots, x'_N\}$ and a **Kolmogorov-Smirnoff (KS) test** on the empirical CDFs of $\{f(x'_i;\theta_1)\}$ and $\{f(x'_i;\theta_2)\}$.

Issue 1: The alternative is always true: the models are different.

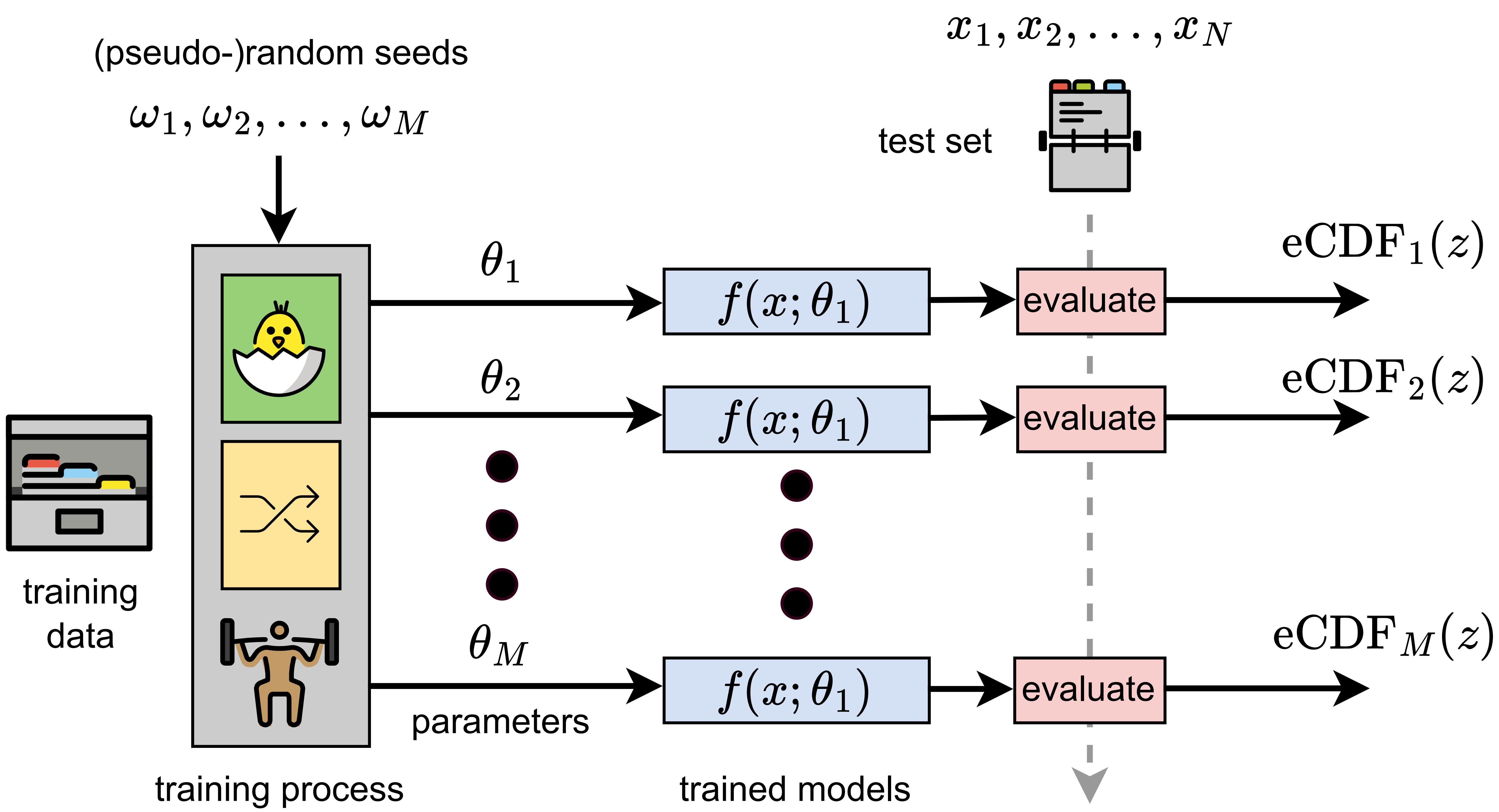
Issue 2: Can we use a 1 sample test instead? Don't have a good estimate of the null.

Issue 3: Shouldn't we use the tools from Bhaswar's talk on Monday???

Yes!!!

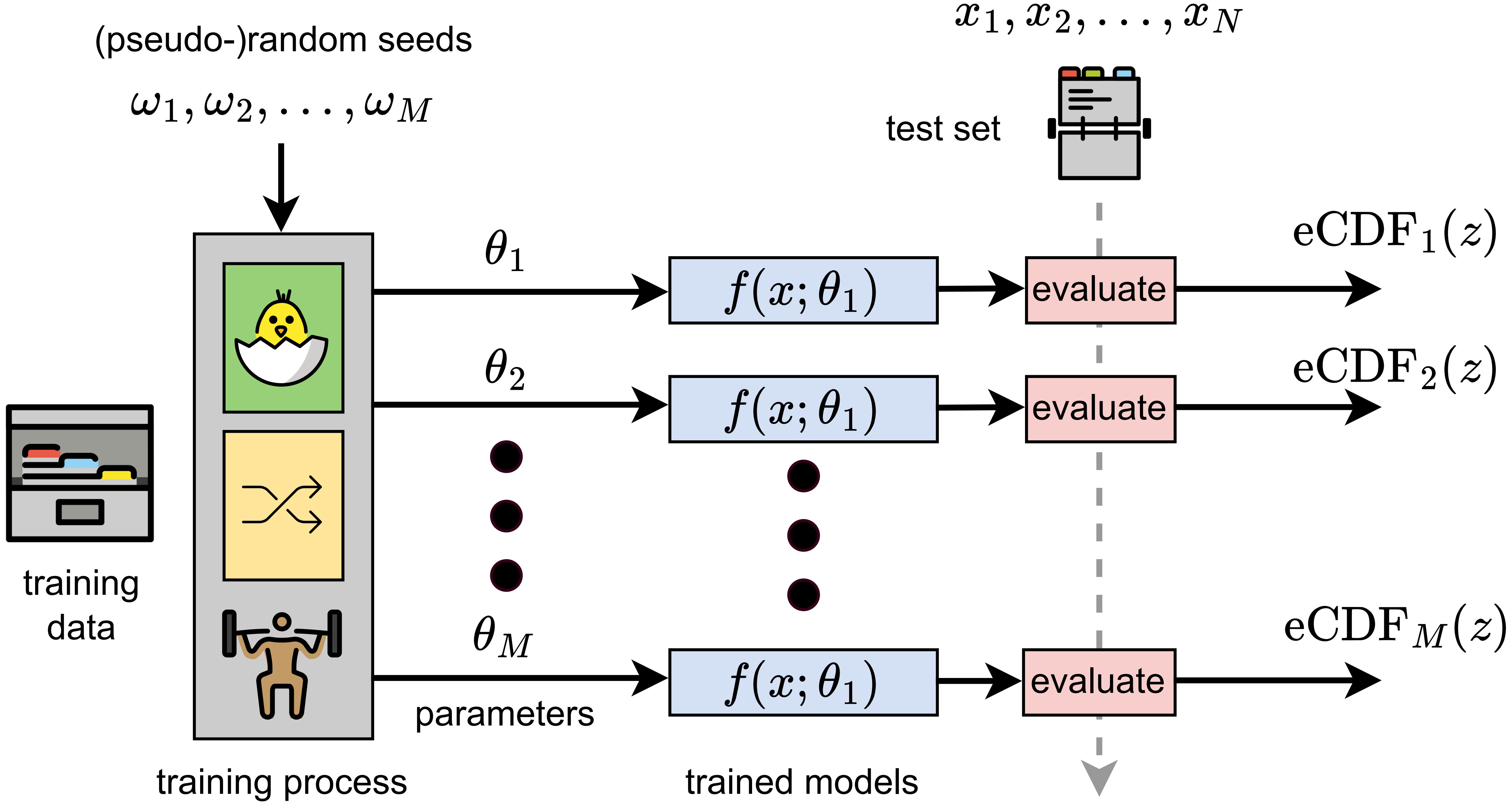
Addressing the first two issues

“Are they different?” Yes. “*Meaningfully* different?” Well…



Addressing the first two issues

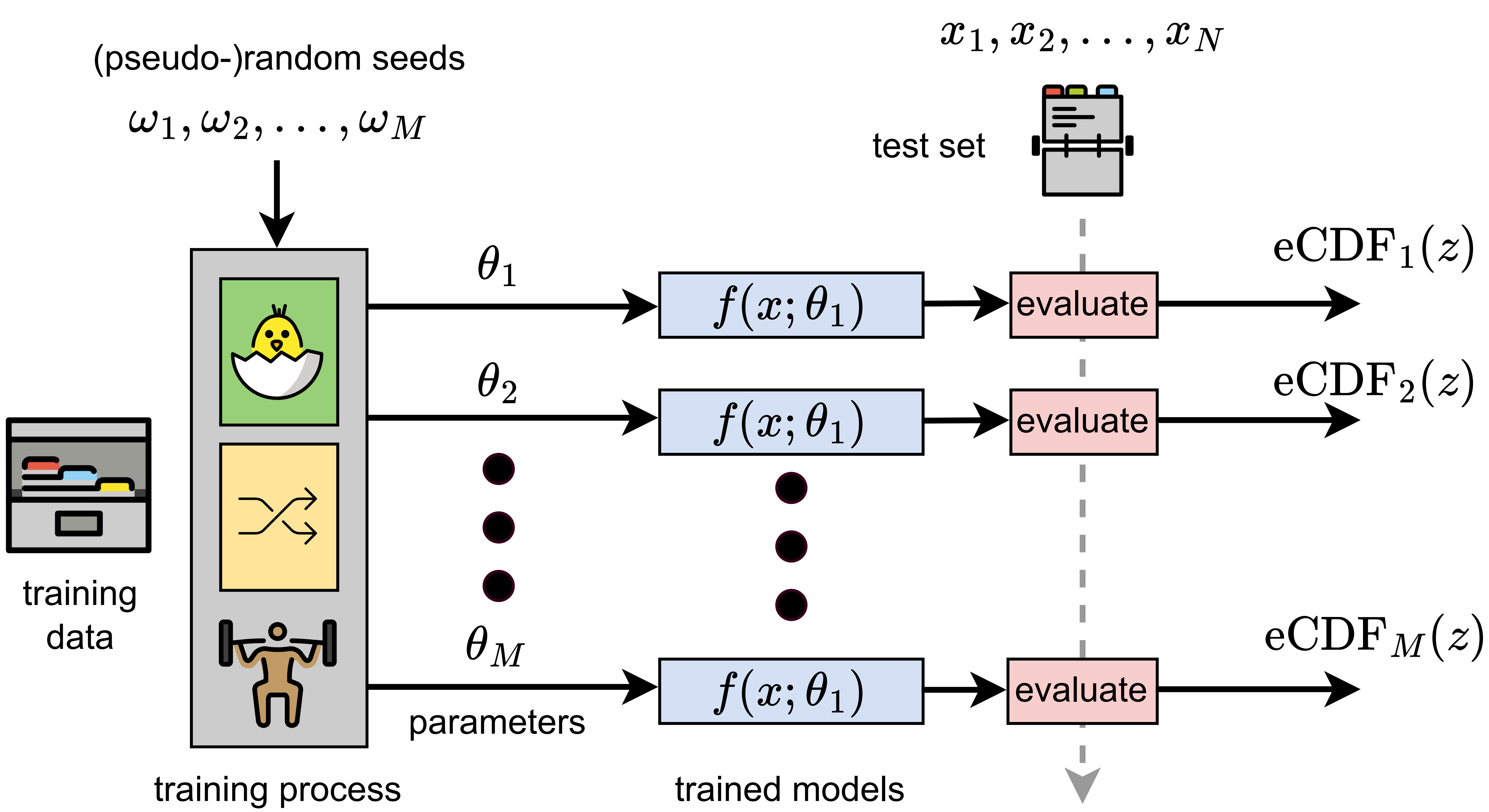
“Are they different?” Yes. “*Meaningfully* different?” Well…



1. Train many models and use them to approximate a null distribution $\hat{F}_0$

Addressing the first two issues

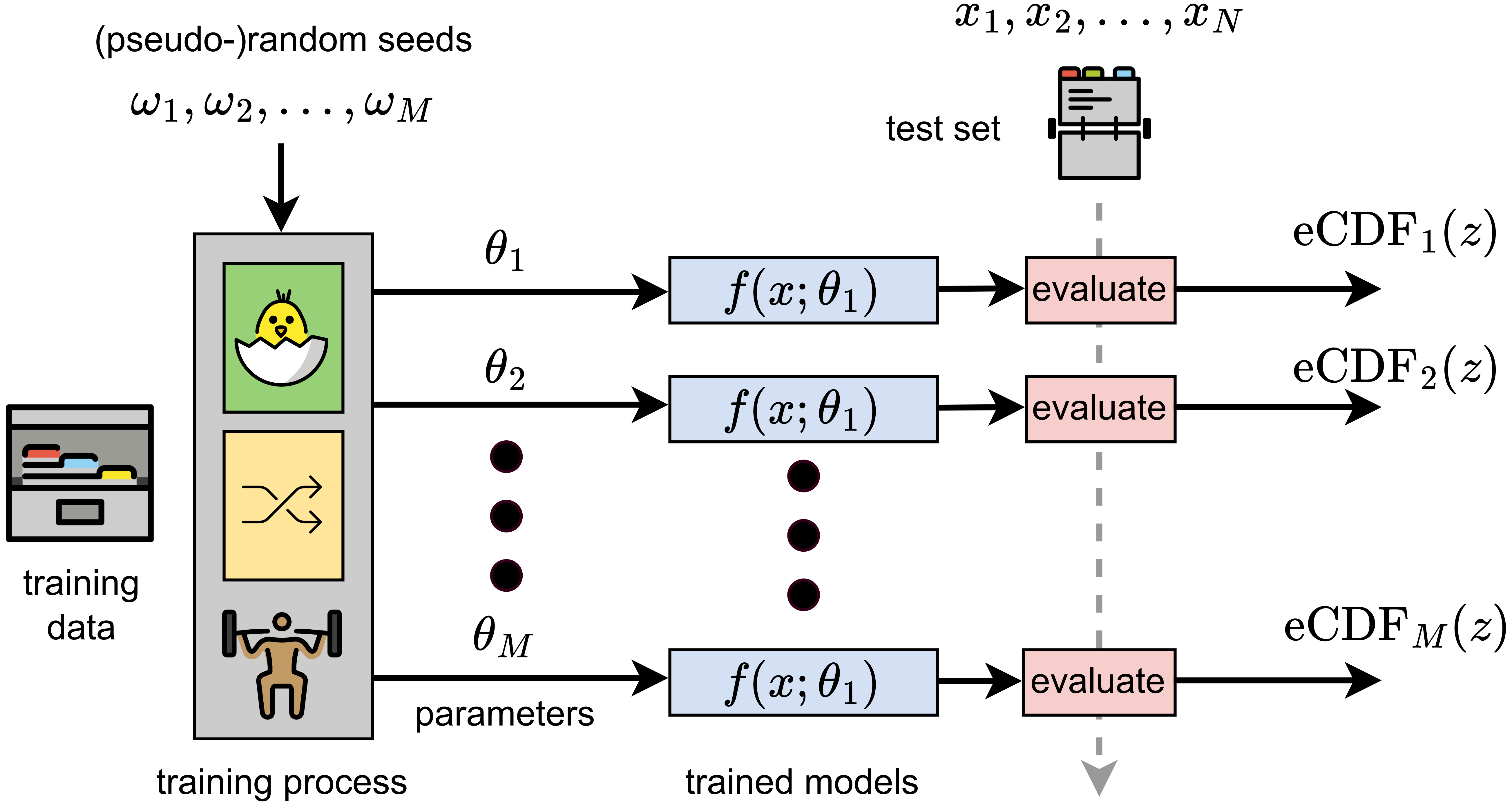
“Are they different?” Yes. “*Meaningfully* different?” Well…



1. Train many models and use them to approximate a null distribution $\hat{F}_0$
2. Sample a new model with eCDF F. Robustify a bit: try to find a CDF $\tilde{F}$ such that:

Addressing the first two issues

“Are they different?” Yes. “*Meaningfully* different?” Well…



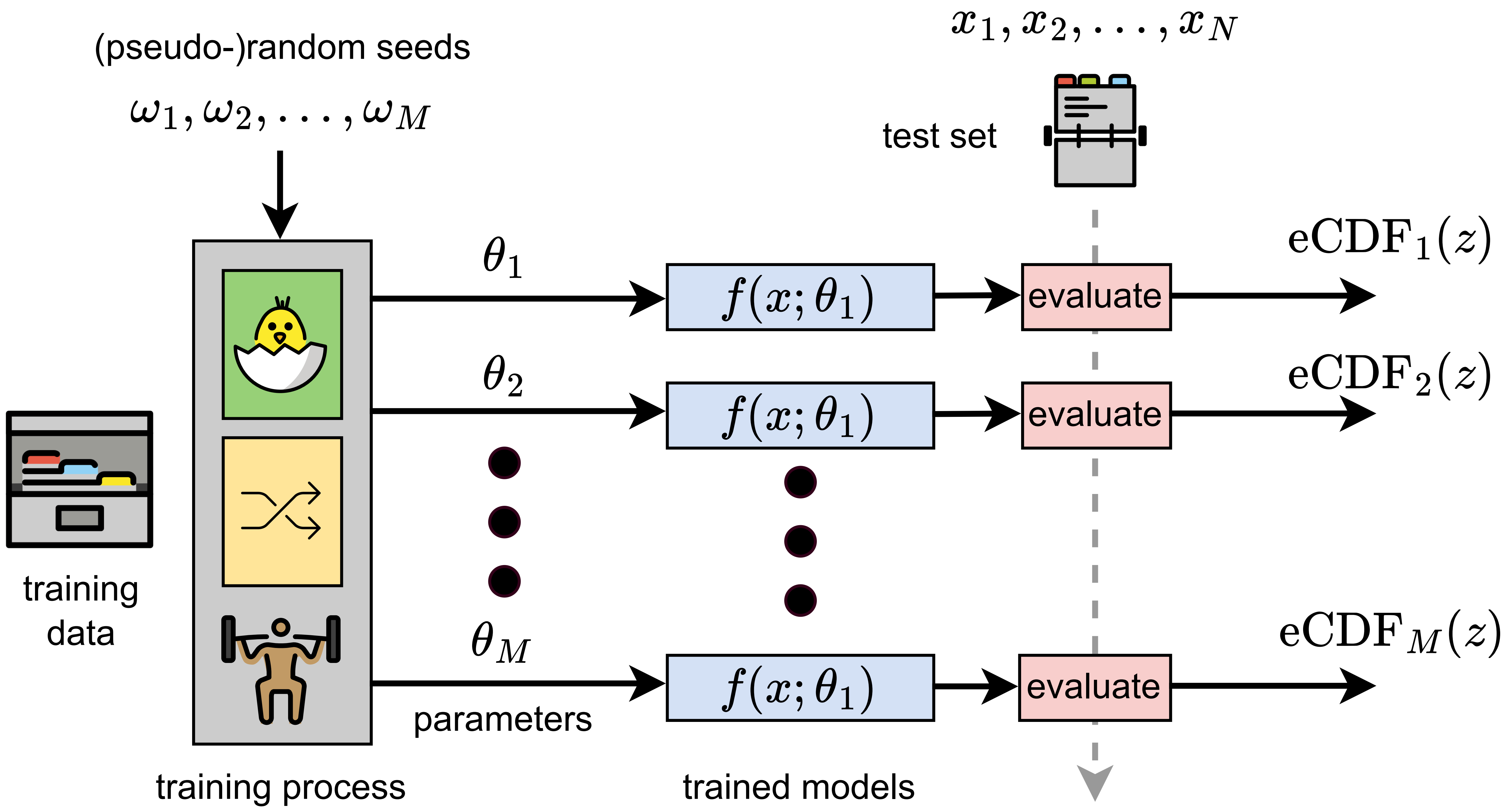
1. Train many models and use them to approximate a null distribution $\hat{F}_0$

2. Sample a new model with eCDF F. Robustify a bit: try to find a CDF $\tilde{F}$ such that:

$$\|F - \tilde{F}\|_1 \leq \alpha$$

Addressing the first two issues

“Are they different?” Yes. “*Meaningfully* different?” Well…



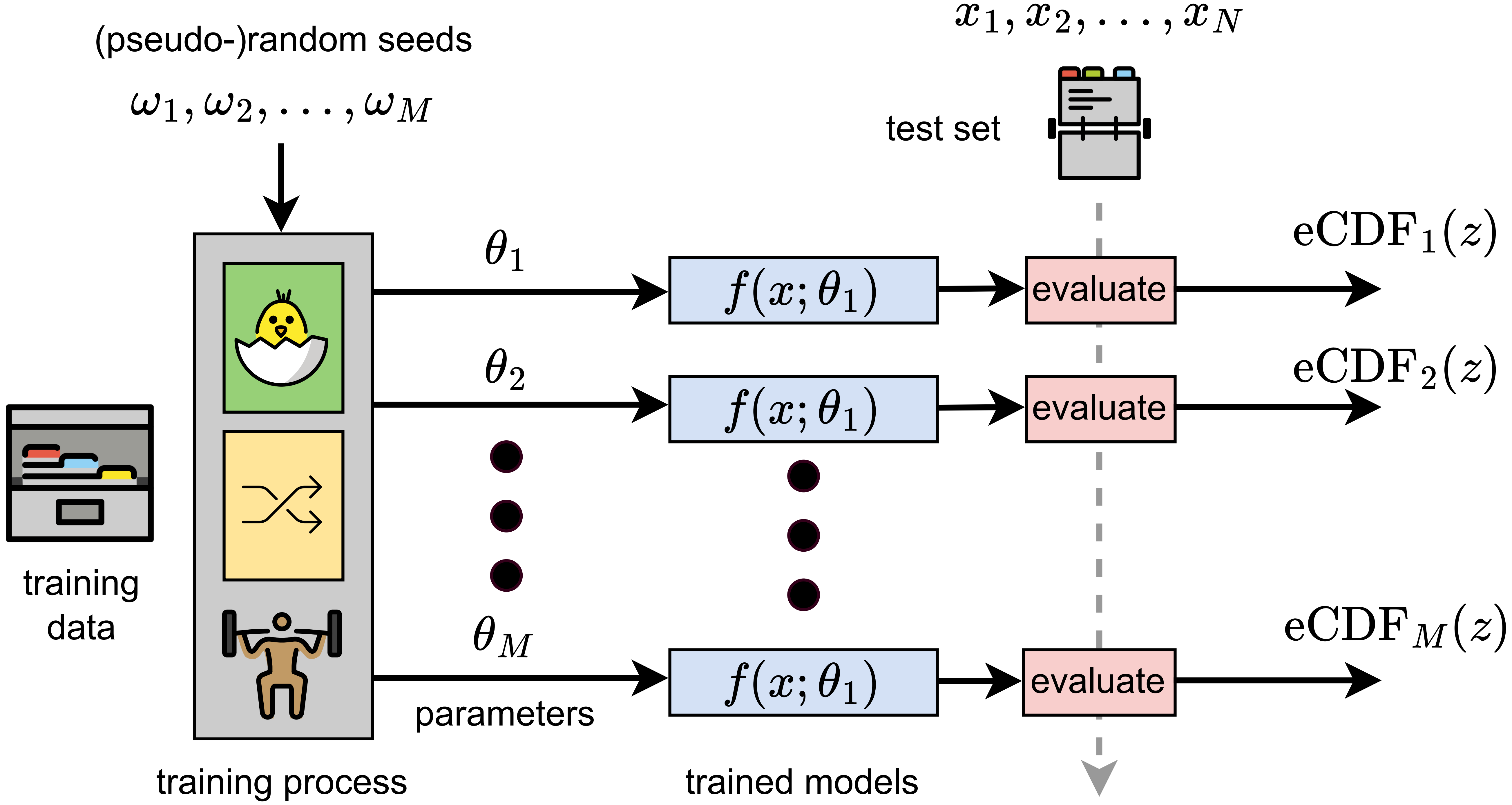
1. Train many models and use them to approximate a null distribution $\hat{F}_0$
2. Sample a new model with eCDF F. Robustify a bit: try to find a CDF $\tilde{F}$ such that:

$$\|F - \tilde{F}\|_1 \leq \alpha$$

$$\|\hat{F}_0 - \tilde{F}\|_\infty \text{ is small}$$

Addressing the first two issues

“Are they different?” Yes. “*Meaningfully* different?” Well…



1. Train many models and use them to approximate a null distribution $\hat{F}_0$

2. Sample a new model with eCDF F. Robustify a bit: try to find a CDF $\tilde{F}$ such that:

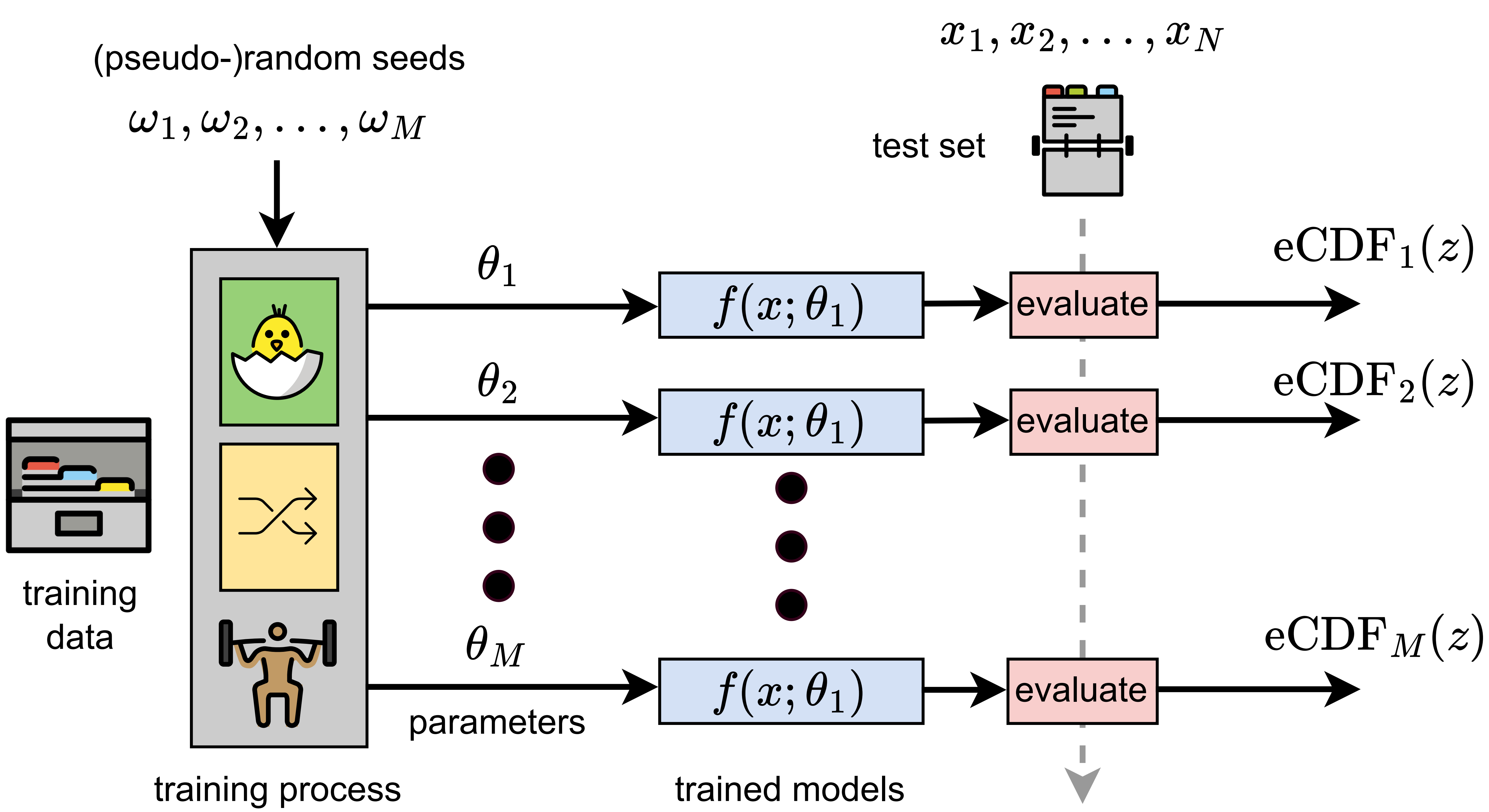
$$\|F - \tilde{F}\|_1 \leq \alpha$$

$\|\hat{F}_0 - \tilde{F}\|_\infty$ is small

Looks like what we observed

Addressing the first two issues

“Are they different?” Yes. “*Meaningfully* different?” Well…



1. Train many models and use them to approximate a null distribution $\hat{F}_0$

2. Sample a new model with eCDF F. Robustify a bit: try to find a CDF $\tilde{F}$ such that:

$$\|F - \tilde{F}\|_1 \leq \alpha$$

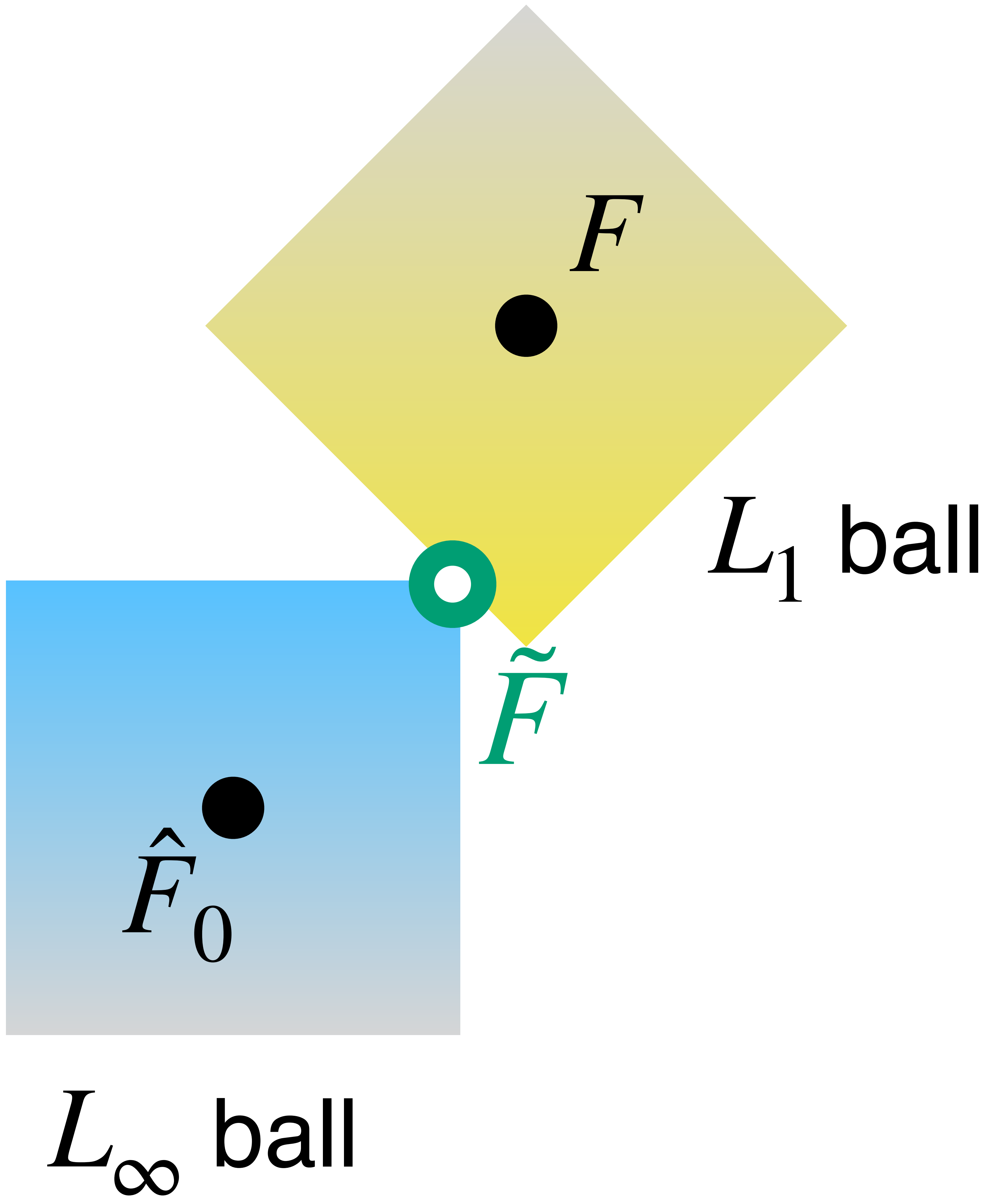
$\|\hat{F}_0 - \tilde{F}\|_\infty$ is small

Looks like what we observed

KS test accepts

Trimming a distribution

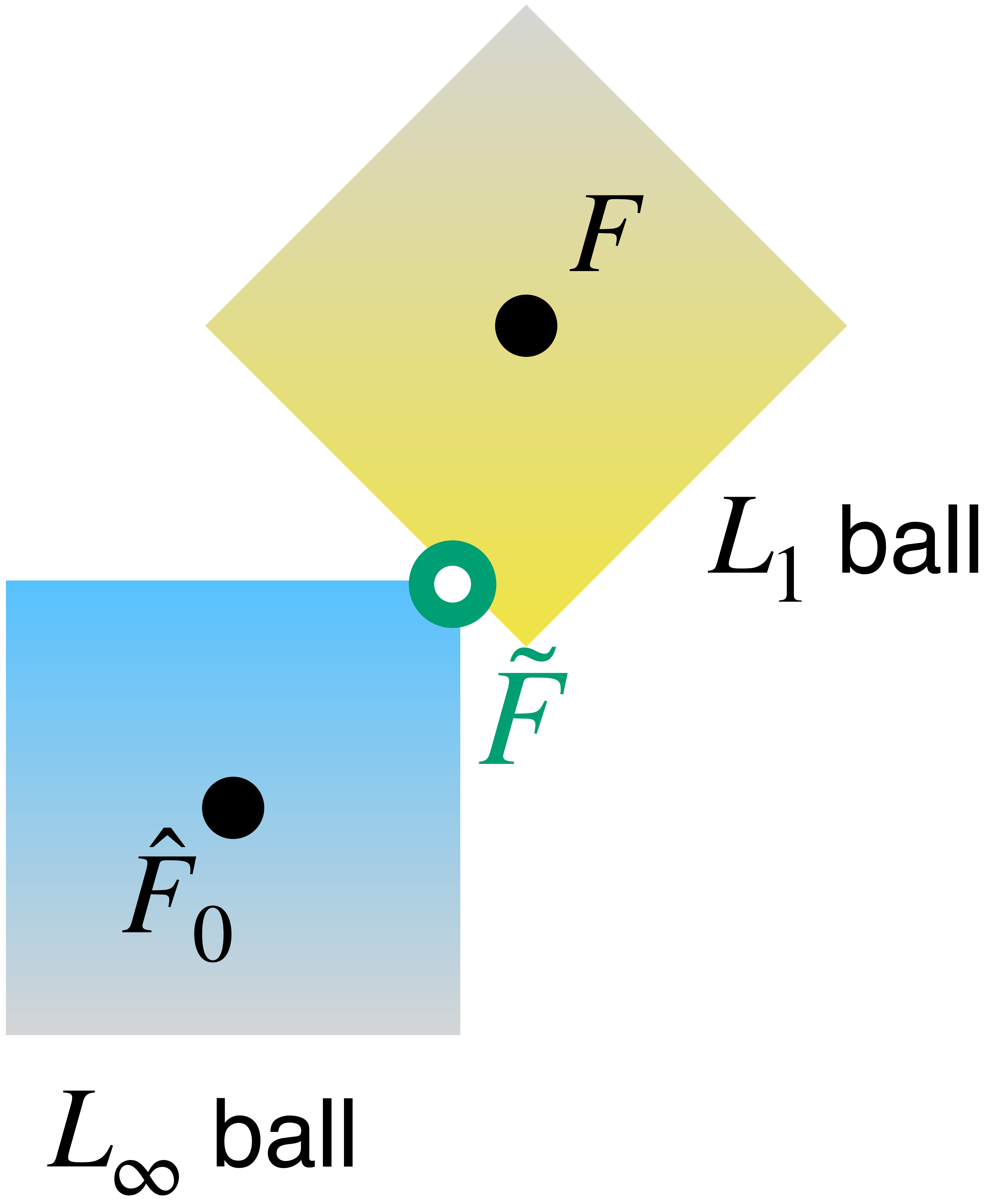
Modeling uncertainty about our observation



Trimming a distribution

Modeling uncertainty about our observation

We need to find:

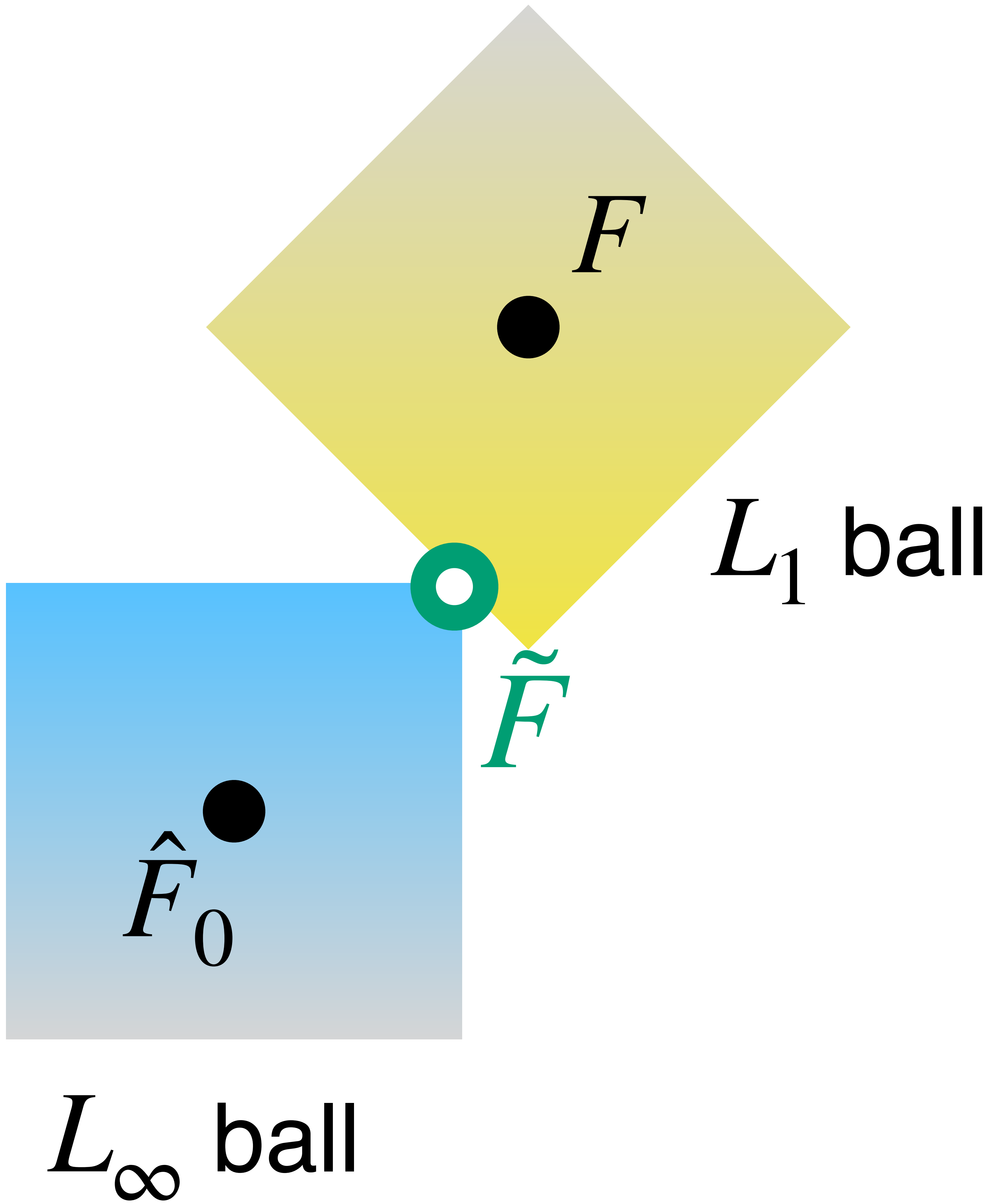


Trimming a distribution

Modeling uncertainty about our observation

We need to find:

$$\operatorname{argmin}_{\tilde{F}} \|\hat{F}_0 - \tilde{F}\|_\infty$$
$$\text{s.t. } \left\| F - \tilde{F} \right\|_1 \leq \alpha$$



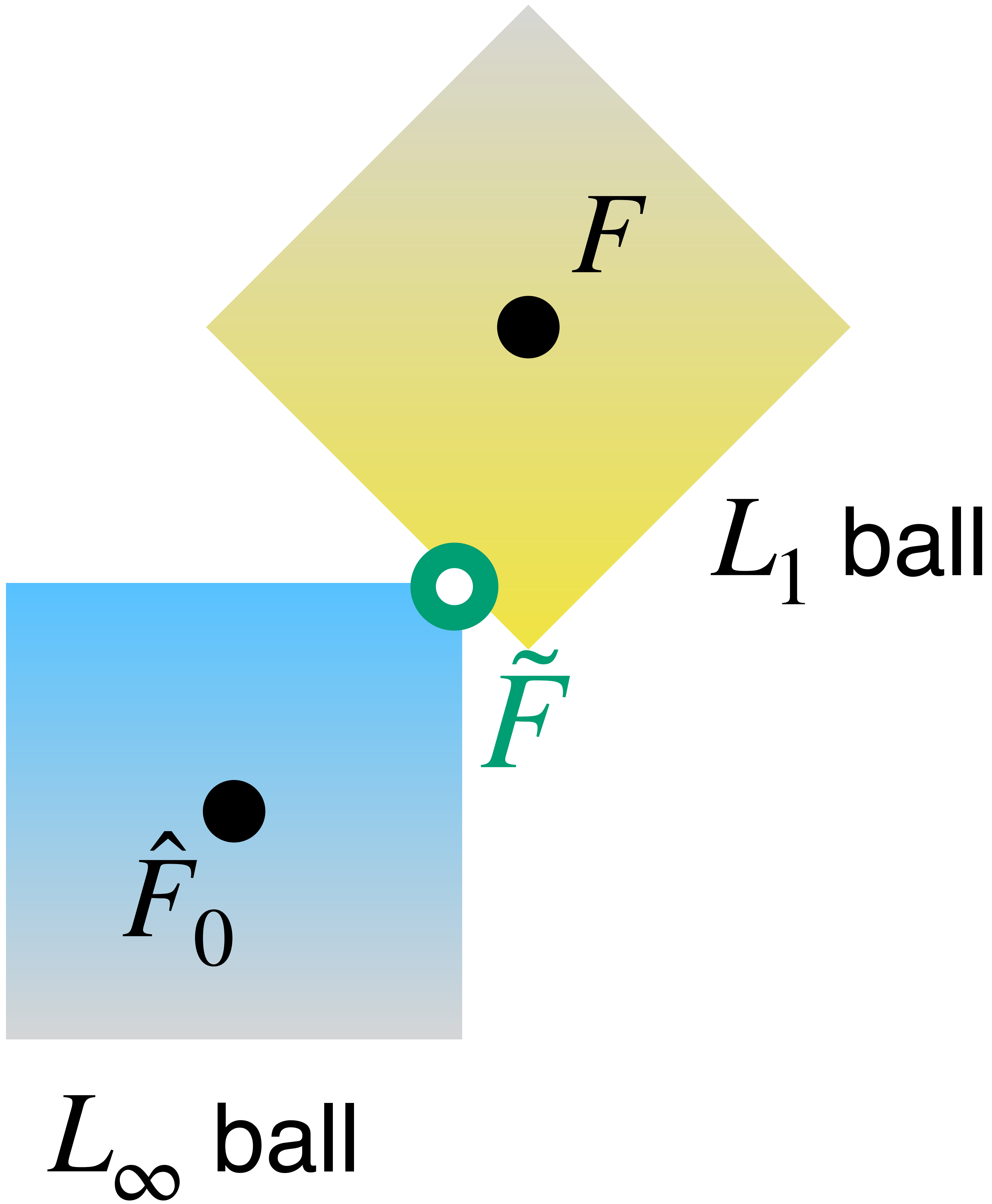
Trimming a distribution

Modeling uncertainty about our observation

We need to find:

$$\mathrm{argmin}_{\tilde{F}} \| \hat{F}_0 - \tilde{F} \|_\infty$$
$$\text{s.t. } \left\| F - \tilde{F} \right\|_1 \leq \alpha$$

This optimization can be restated as searching over "α-trimmings" of F and there is an efficient optimization for it (del Barrio el 2020, Álvarez-Esteban et al. 2011).



Trimming a distribution

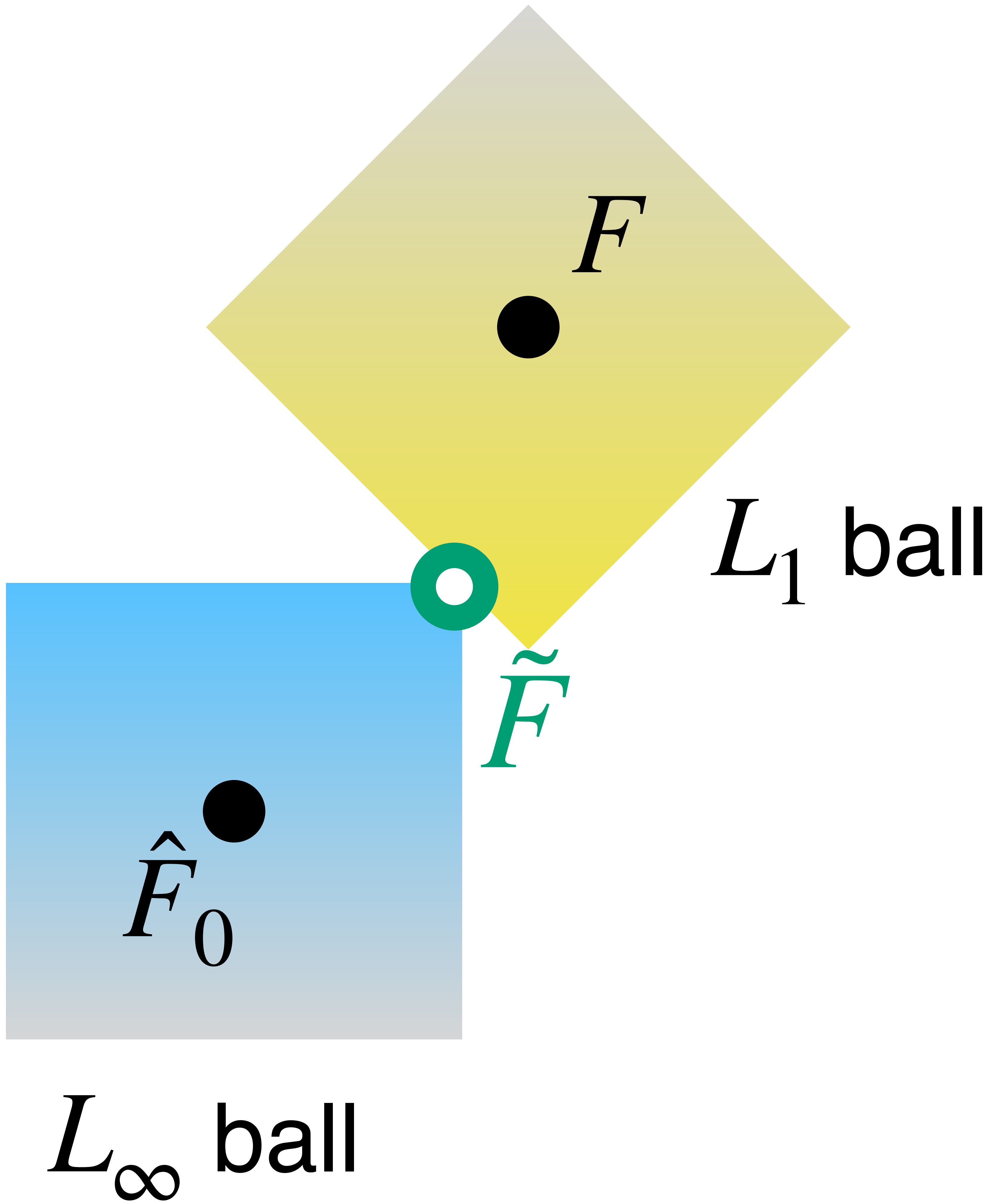
Modeling uncertainty about our observation

We need to find:

$$\text{argmin}_{\tilde{F}} \|\hat{F}_0 - \tilde{F}\|_\infty$$
$$\text{s.t. } \left\| F - \tilde{F} \right\|_1 \leq \alpha$$

This optimization can be restated as searching over "α-trimmings" of F and there is an efficient optimization for it (del Barrio el 2020, Álvarez-Esteban et al. 2011).

Define $\hat{\alpha}$ as the minimum level for the KS test to accept.



Comparing against other measures

Models "looking the same" depends on what you mean

Comparing against other measures

Models "looking the same" depends on what you mean

1. Test/validation accuracy: if two models have similar test performance, "one is as good as the other."

Comparing against other measures

Models "looking the same" depends on what you mean

1. Test/validation accuracy: if two models have similar test performance, "one is as good as the other."

2. Churn: the two models do not disagree on the test set.

Comparing against other measures

Models “looking the same” depends on what you mean

1. Test/validation accuracy: if two models have similar test performance, “one is as good as the other.”

2. Churn: the two models do not disagree on the test set.

    - Can also measure churn w.r.t. the ensemble model for the null.

Comparing against other measures

Models "looking the same" depends on what you mean

1. Test/validation accuracy: if two models have similar test performance, "one is as good as the other."

2. Churn: the two models do not disagree on the test set.

    - Can also measure churn w.r.t. the ensemble model for the null.

3. Expected Calibration Error (ECE) (Naeini et al. 2015): measures the difference between accuracy and expected "confidence" (the LLR).

Comparing against other measures

Models "looking the same" depends on what you mean

1. Test/validation accuracy: if two models have similar test performance, "one is as good as the other."

2. Churn: the two models do not disagree on the test set.

   - Can also measure churn w.r.t. the ensemble model for the null.

3. Expected Calibration Error (ECE) (Naeini et al. 2015): measures the difference between accuracy and expected "confidence" (the LLR).

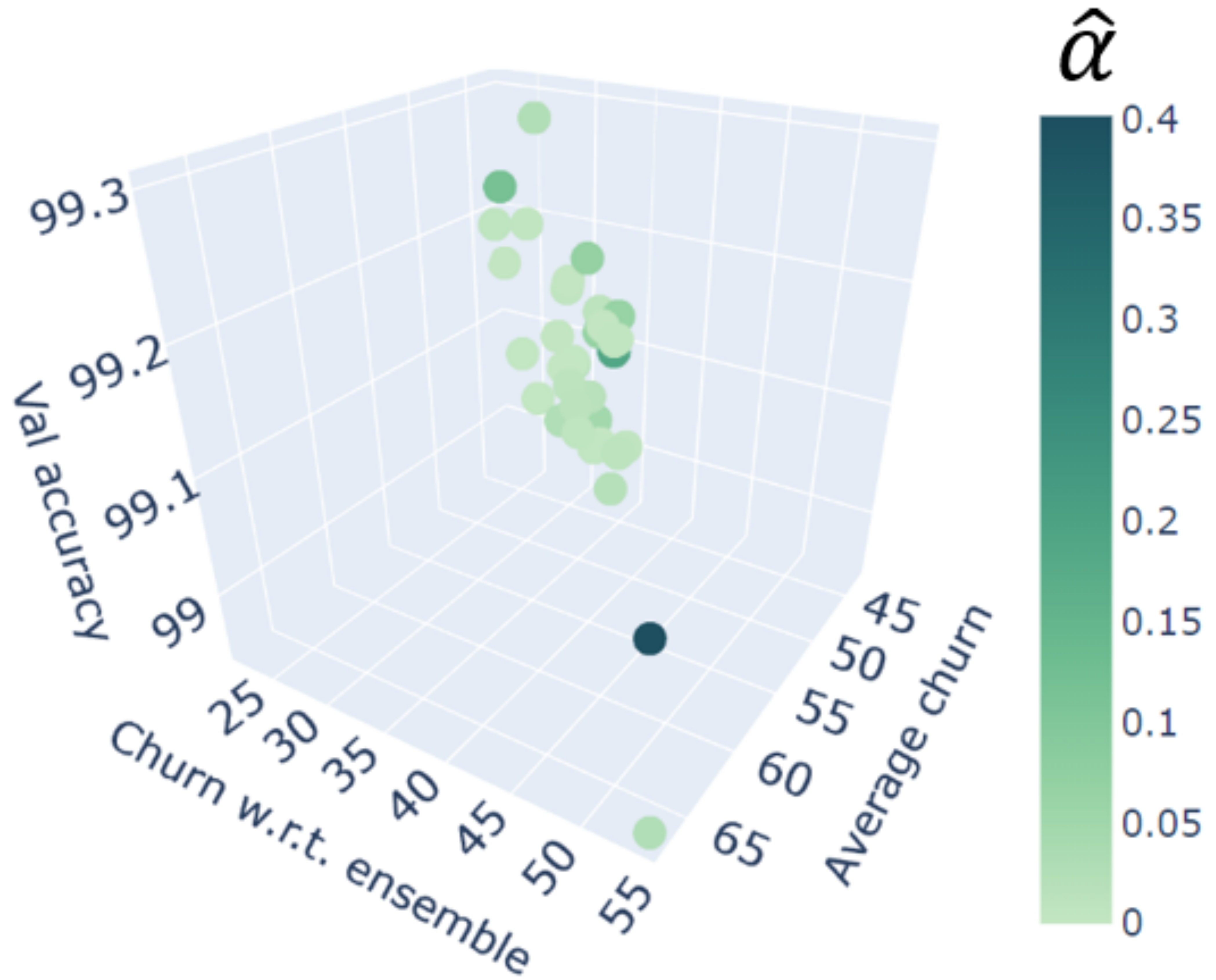
Does $\hat{\alpha}$ imply anything about these measures?

It seems useful as a measure

But this is only one of many options…

It seems useful as a measure

But this is only one of many options…

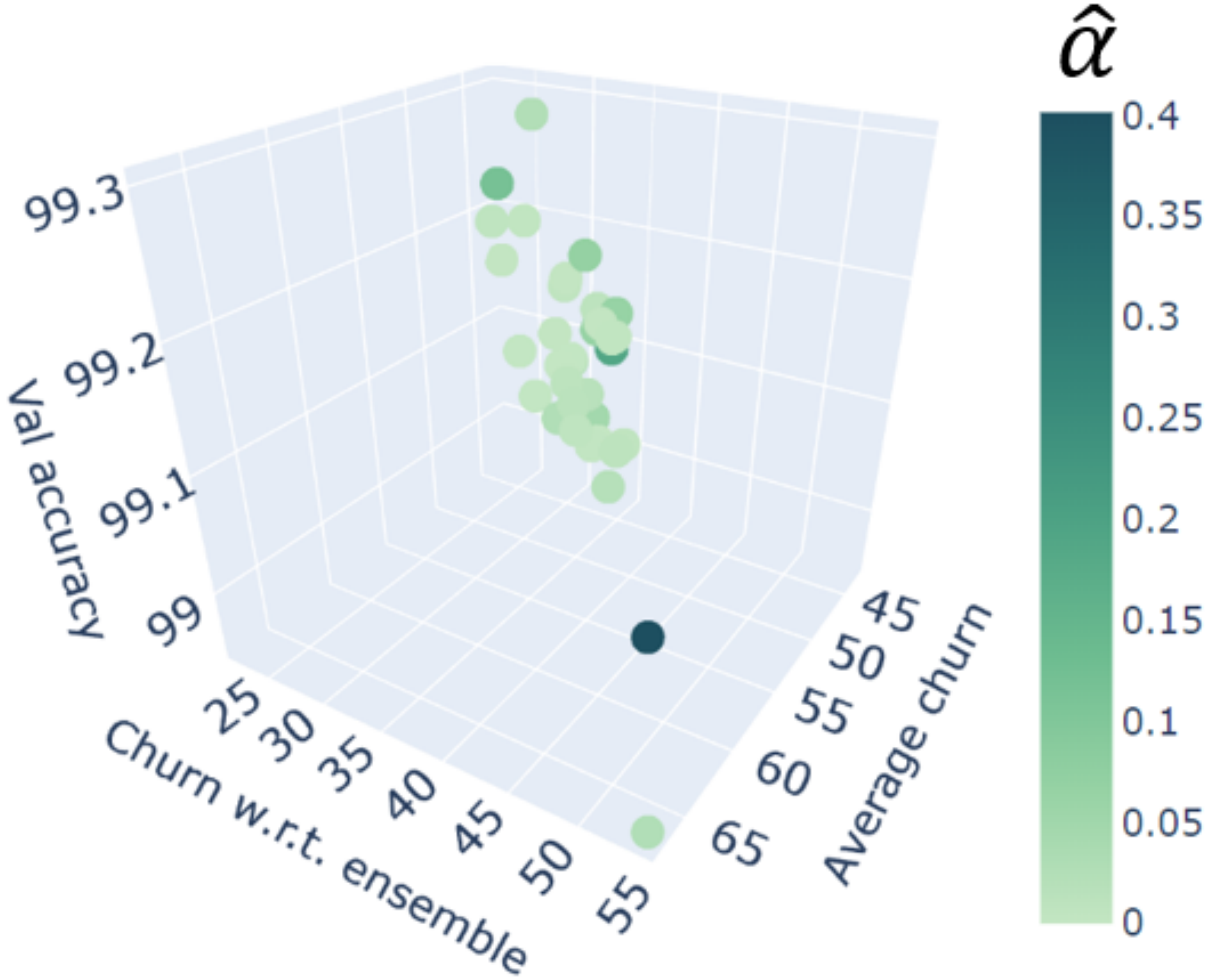


What we see from various experiments:

Made a binary problem of “vehicles” versus “creatures” on 8 classes of CIFAR-10 with 40k training and 8k test points. Fine-tuned 90 models based on a Vi and used 45 for an ensemble.

It seems useful as a measure

But this is only one of many options…



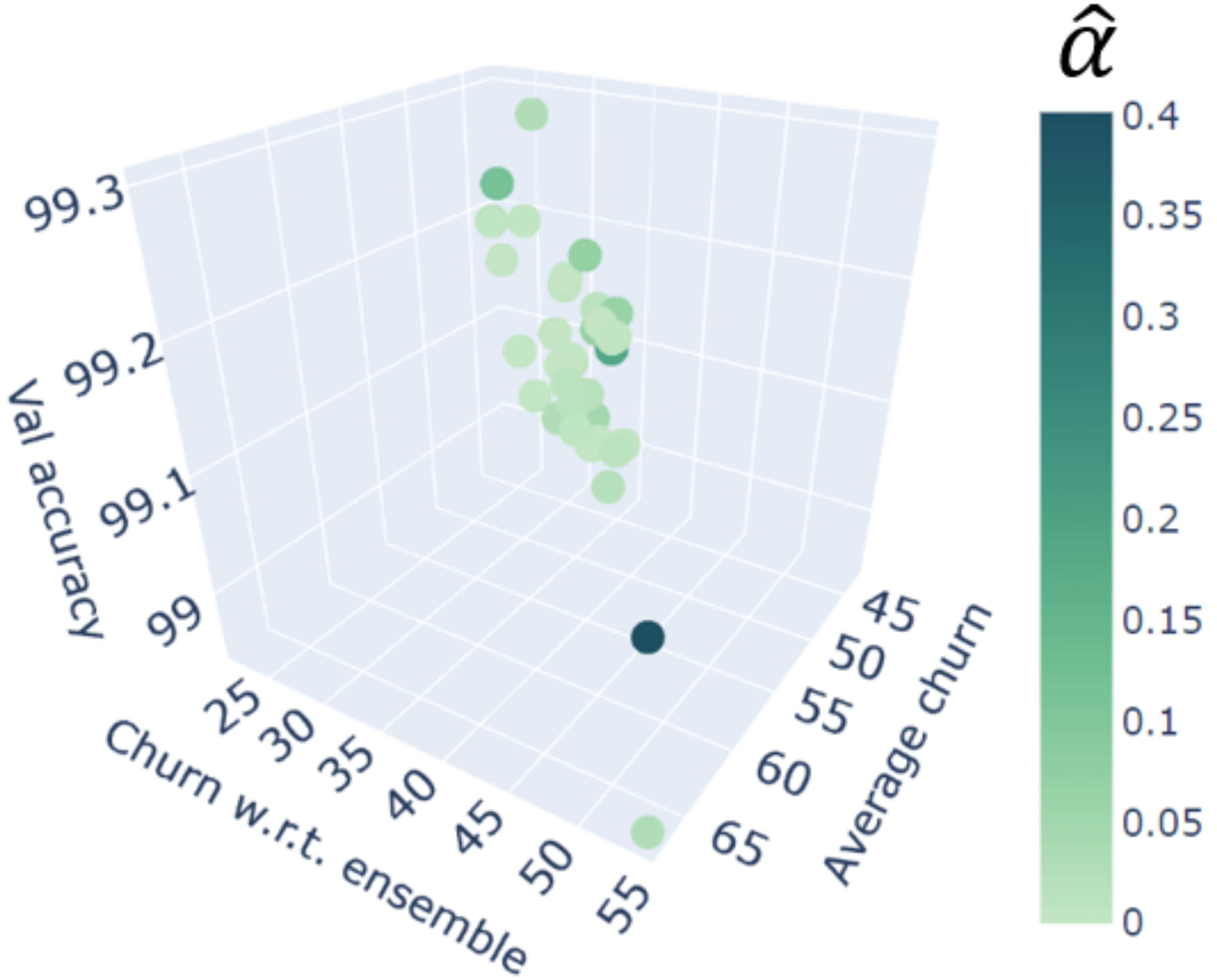
What we see from various experiments:

- Large $\hat{\alpha}$ implies one of the other metrics will be large as well.

Made a binary problem of “vehicles” versus “creatures” on 8 classes of CIFAR-10 with 40k training and 8k test points. Fine-tuned 90 models based on a Vi and used 45 for an ensemble.

It seems useful as a measure

But this is only one of many options…



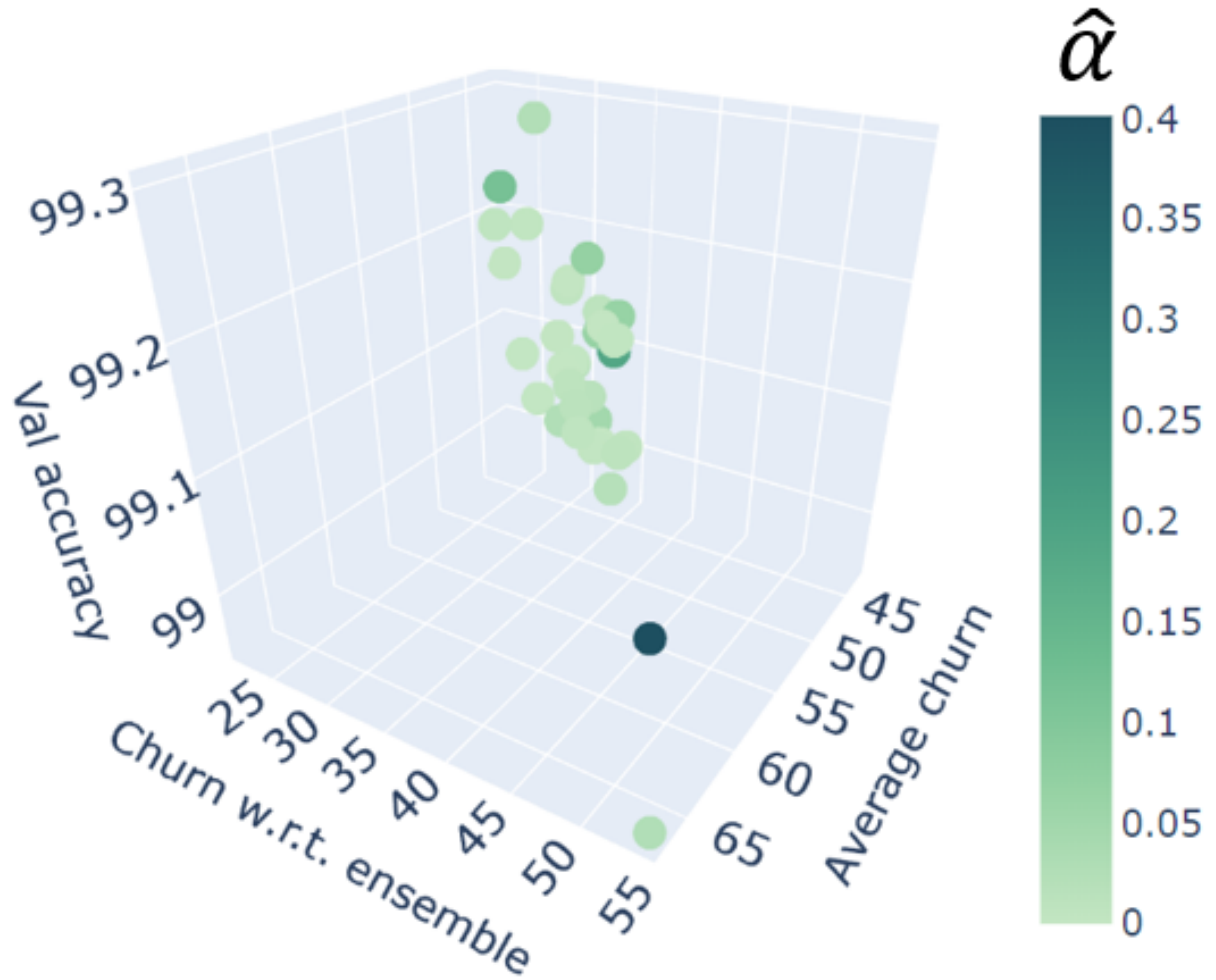
Made a binary problem of “vehicles” versus “creatures” on 8 classes of CIFAR-10 with 40k training and 8k test points. Fine-tuned 90 models based on a Vi and used 45 for an ensemble.

What we see from various experiments:

- Large $\hat{\alpha}$ implies one of the other metrics will be large as well.
- Models with small $\hat{\alpha}$ are generally low on all the other metrics as well.

It seems useful as a measure

But this is only one of many options…



Made a binary problem of “vehicles” versus “creatures” on 8 classes of CIFAR-10 with 40k training and 8k test points. Fine-tuned 90 models based on a Vi and used 45 for an ensemble.

What we see from various experiments:

- Large $\hat{\alpha}$ implies one of the other metrics will be large as well.
- Models with small $\hat{\alpha}$ are generally low on all the other metrics as well.
- We can use $\hat{\alpha}$ to examine the impact of different sources of randomness in the training algorithms.

ML models as measurement instruments

This is scratching the surface

ML models as measurement instruments

This is scratching the surface

Lots of interesting follow-up questions:

ML models as measurement instruments

This is scratching the surface

Lots of interesting follow-up questions:

- What is the right test to use?

ML models as measurement instruments

This is scratching the surface

Lots of interesting follow-up questions:

- What is the right test to use?
- How large an ensemble does one need to look “representative”?

ML models as measurement instruments

This is scratching the surface

Lots of interesting follow-up questions:

- What is the right test to use?
- How large an ensemble does one need to look “representative”?
- In fine-tuning a pre-trained model, do we have similar or different levels of variability?

ML models as measurement instruments

This is scratching the surface

Lots of interesting follow-up questions:

- What is the right test to use?
- How large an ensemble does one need to look “representative”?
- In fine-tuning a pre-trained model, do we have similar or different levels of variability?

All of these are important questions if we want to use ML as a scientific instrument! We need to know if our instrument is defective/an outlier or if fine-tuning can lead to very different models…

Detecting difference in differently trained models

Rm Palaniappan, *Alien Planet-B*
Viscosity, pencil colour and ink on handmade paper

What kind of training was used?

The impact of training is visible in the trained models

Three Borg “drones” on an alien planet

What kind of training was used?

The impact of training is visible in the trained models

Three Borg “drones” on an alien planet

In scientific instrumentation, different designs can lead to different data artifacts.

What kind of training was used?

The impact of training is visible in the trained models

Three Borg “drones” on an alien planet

In scientific instrumentation, different designs can lead to different data artifacts.

Different optimization algorithms using the same data and architecture will in general be different, but how?

What kind of training was used?

The impact of training is visible in the trained models

Three Borg “drones” on an alien planet

In scientific instrumentation, different designs can lead to different data artifacts.

Different optimization algorithms using the same data and architecture will in general be different, but how?

What’s different about models trained using GD vs. SGD vs. Adam?

Neural Networks as Kernel Machines

Approximating an NN with a “simpler” model

Neural Networks as Kernel Machines

Approximating an NN with a "simpler" model

Jacot et al. (2018) showed that infinitely wide NNs are equivalent to a kernel machine with with the "neural tangent kernel" (NTK):

Neural Networks as Kernel Machines

Approximating an NN with a "simpler" model

Jacot et al. (2018) showed that infinitely wide NNs are equivalent to a kernel machine with with the "neural tangent kernel" (NTK):

$$K(\mathbf{x}, \mathbf{x}') = \left\langle \nabla_\theta f(\mathbf{x}; \theta), \nabla_\theta f(\mathbf{x}'; \theta) \right\rangle$$

Neural Networks as Kernel Machines

Approximating an NN with a “simpler” model

Jacot et al. (2018) showed that infinitely wide NNs are equivalent to a kernel machine with with the “neural tangent kernel” (NTK):

$$K(\mathbf{x}, \mathbf{x}') = \left\langle \nabla_\theta f(\mathbf{x}; \theta), \nabla_\theta f(\mathbf{x}'; \theta) \right\rangle$$

Think of this as measuring the (cosine) similarity between the tangent hyperplanes for $\mathbf{x}$ and $\mathbf{x}'$ at the same parameter setting θ.

Neural Networks as Kernel Machines

Approximating an NN with a “simpler” model

Jacot et al. (2018) showed that infinitely wide NNs are equivalent to a kernel machine with with the “neural tangent kernel” (NTK):

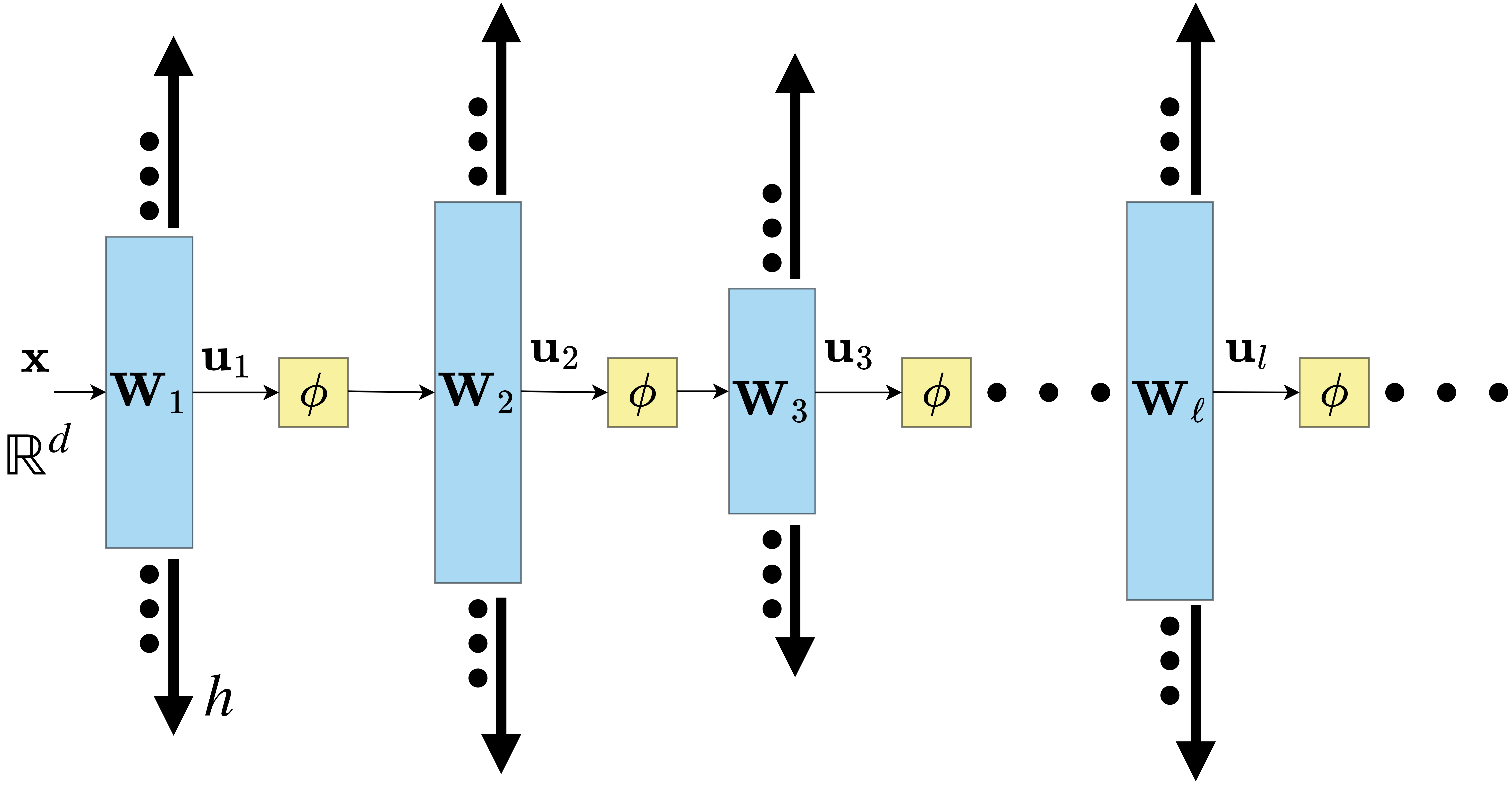
$$K(\mathbf{x}, \mathbf{x}') = \left\langle \nabla_\theta f(\mathbf{x}; \theta), \nabla_\theta f(\mathbf{x}'; \theta) \right\rangle$$

Think of this as measuring the (cosine) similarity between the tangent hyperplanes for $\mathbf{x}$ and $\mathbf{x}'$ at the same parameter setting θ.

Finite width networks don’t really behave like infinite width networks… (Chizat et al., 2018; Yang & Hu, 2021; Wang et al., 2022).

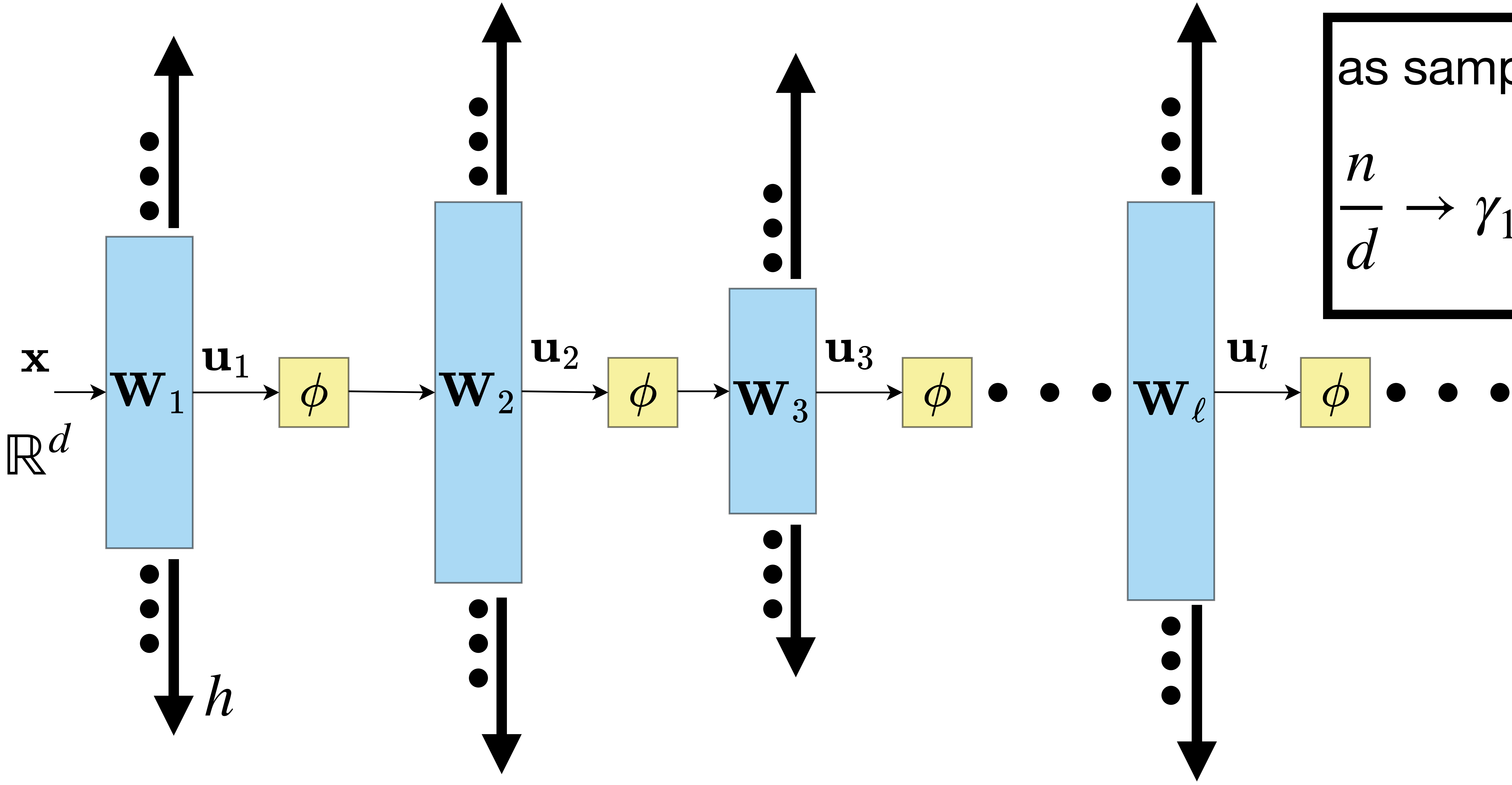
Linear width regime (LWR)

Input dimension, widths, training set all scale together



Linear width regime (LWR)

Input dimension, widths, training set all scale together

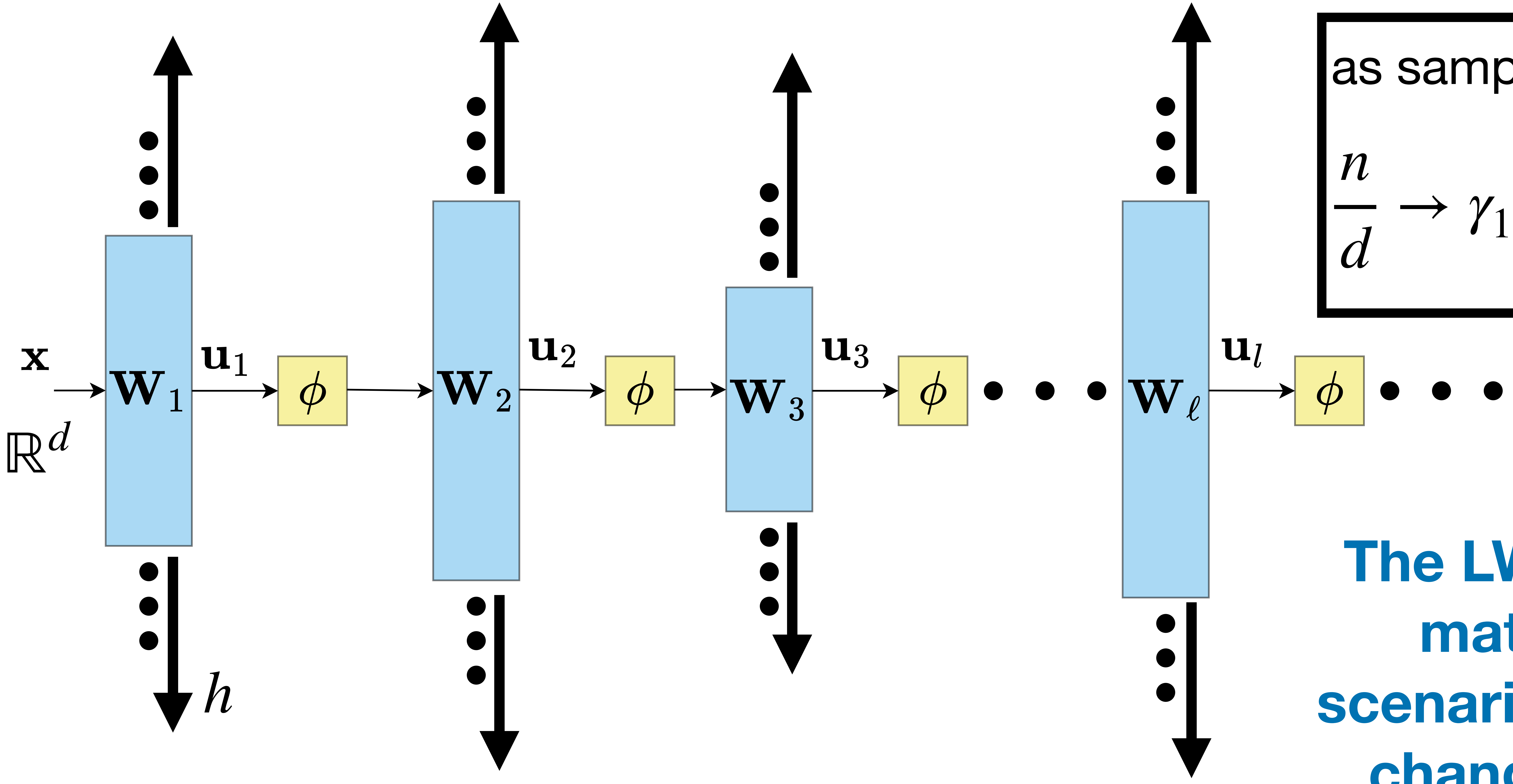


as sample size $n \to \infty$.

$$\frac{n}{d} \to \gamma_1 \qquad \frac{h}{d} \to \gamma_2$$

Linear width regime (LWR)

Input dimension, widths, training set all scale together



as sample size $n \to \infty$.

$$\frac{n}{d} \to \gamma_1 \qquad \frac{h}{d} \to \gamma_2$$

The LWR is a better match for real scenarios, but does it change anything?

How do we move past the kernel regime?

Spectral evolution

How do we move past the kernel regime?

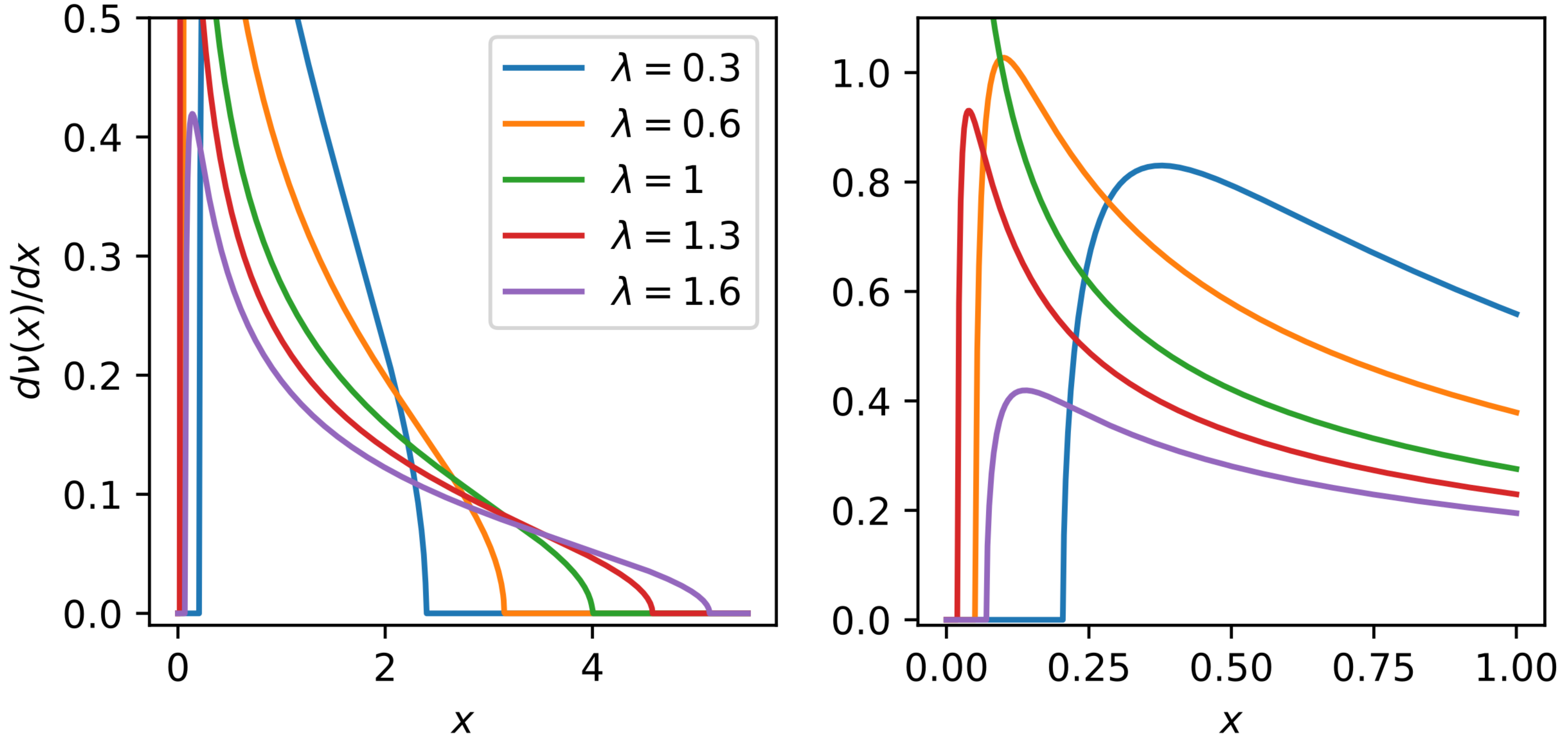
Spectral evolution

We want to know how matrices associated with a NN *evolve* during training.

How do we move past the kernel regime?

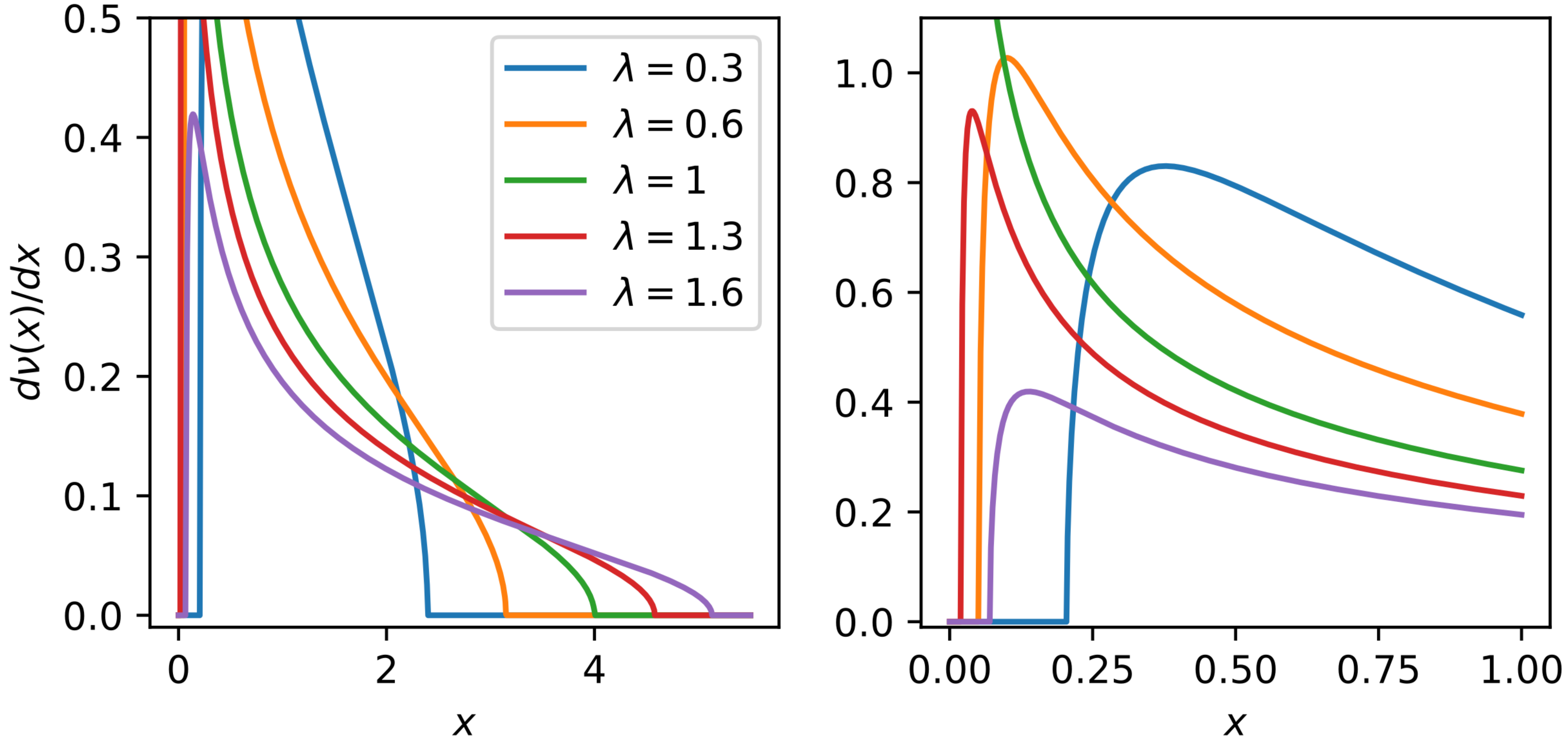
Spectral evolution

We want to know how matrices associated with a NN *evolve* during training.



How do we move past the kernel regime?

Spectral evolution

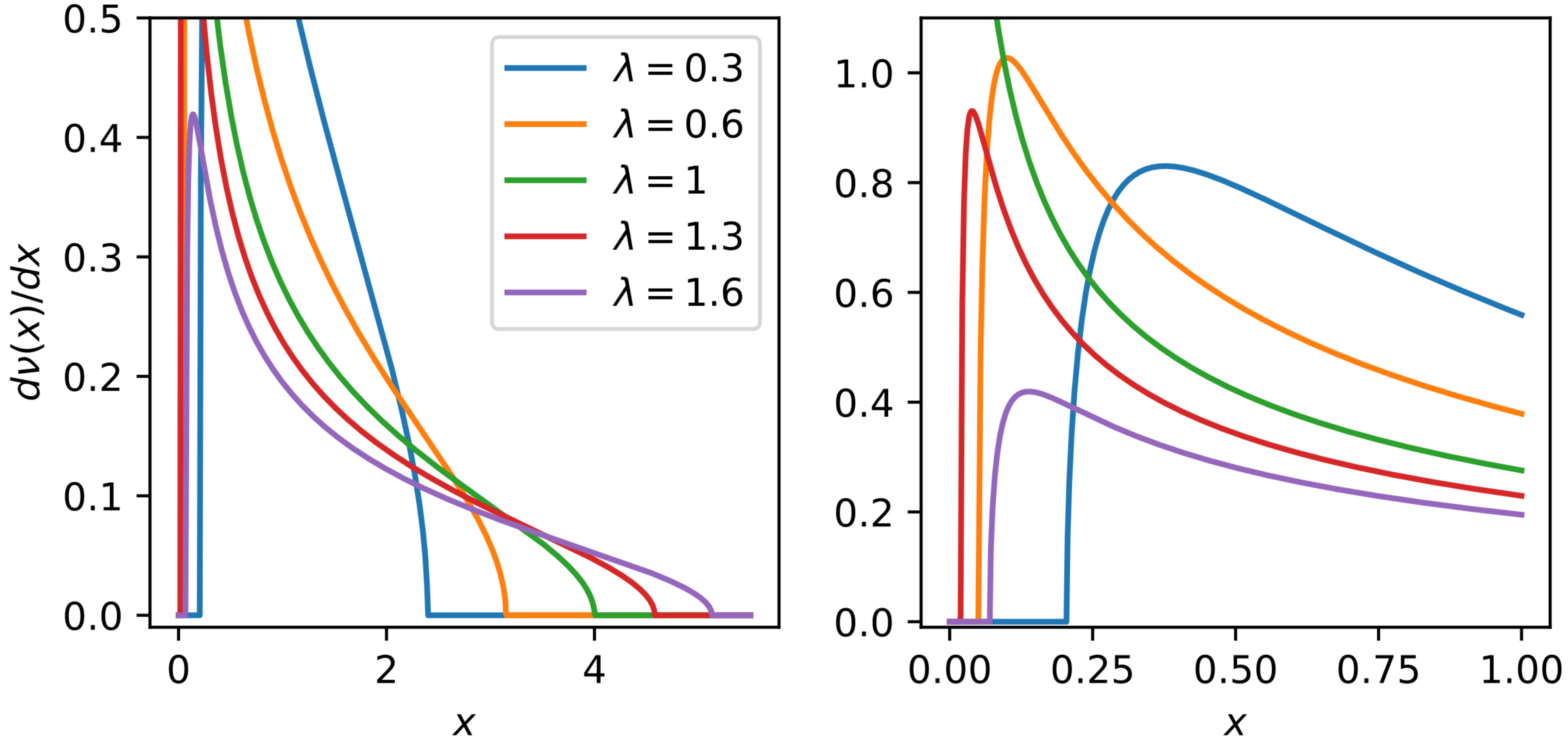


We want to know how matrices associated with a NN *evolve* during training.

- Are the spectra of trained networks different than initialization?

How do we move past the kernel regime?

Spectral evolution

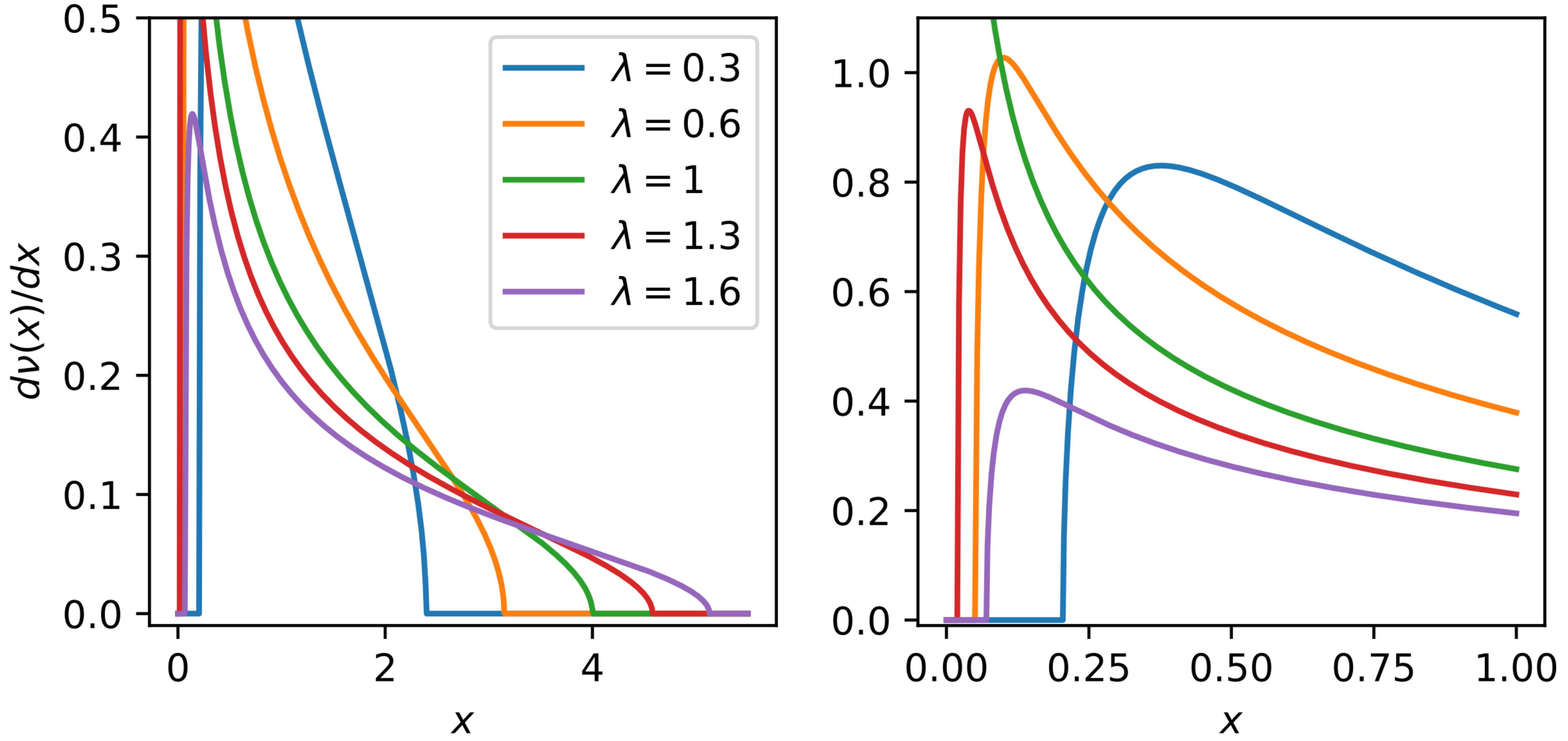


We want to know how matrices associated with a NN *evolve* during training.

- Are the spectra of trained networks different than initialization?
- Do spectra reveal something about “learned features”?

How do we move past the kernel regime?

Spectral evolution

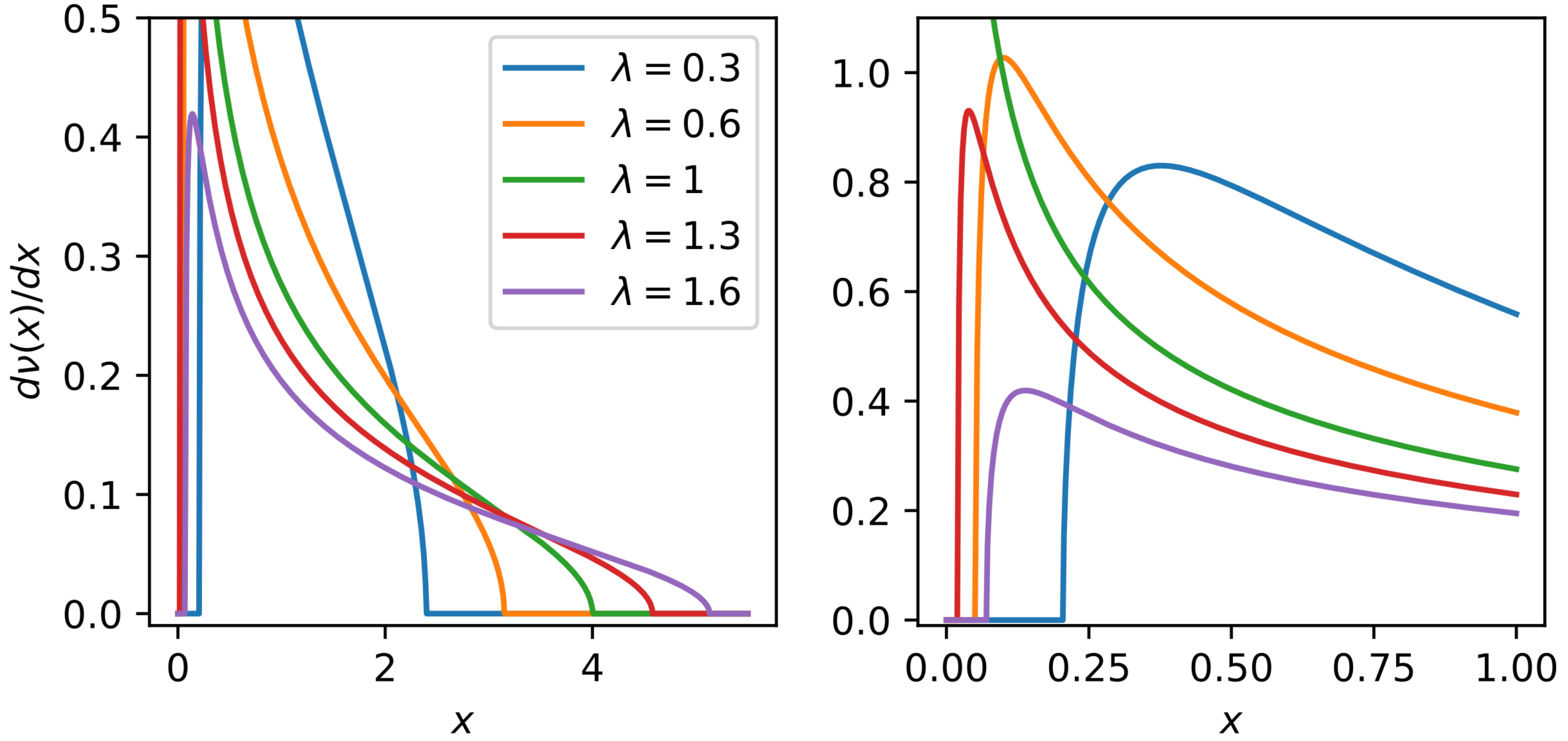


We want to know how matrices associated with a NN *evolve* during training.

- Are the spectra of trained networks different than initialization?
- Do spectra reveal something about “learned features”?
- Can we use this for hyperparameter tuning?

How do we move past the kernel regime?

Spectral evolution



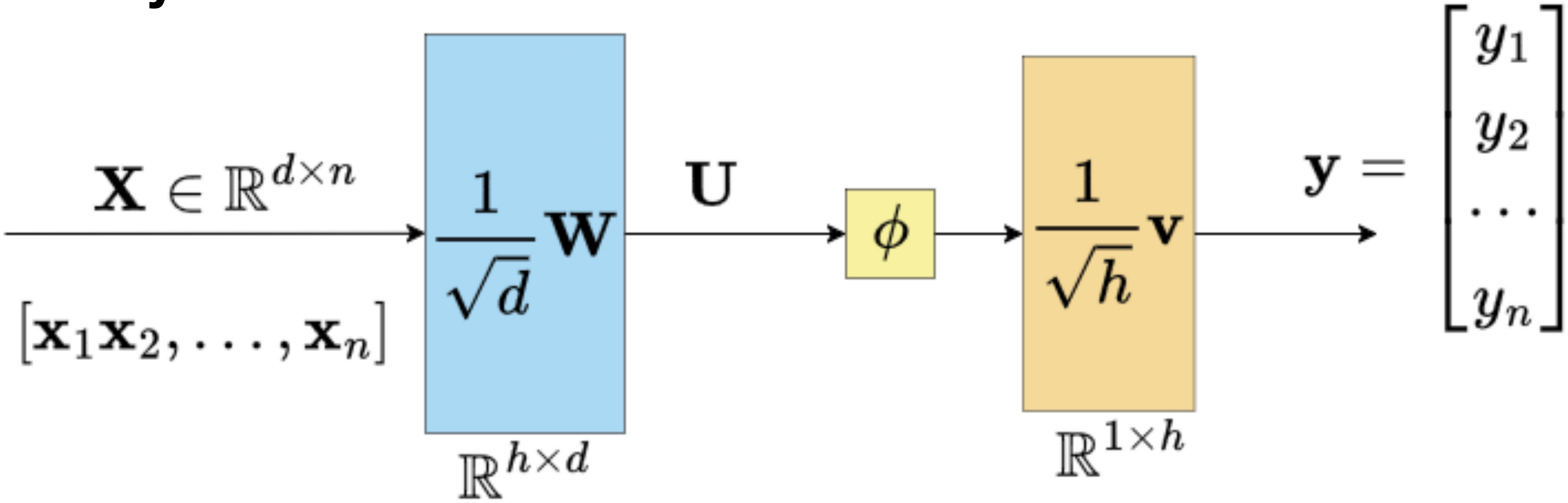
We want to know how matrices associated with a NN *evolve* during training.

- Are the spectra of trained networks different than initialization?
- Do spectra reveal something about "learned features"?
- Can we use this for hyperparameter tuning?

Main idea: use random matrix theory (RMT) to understand this evolution.

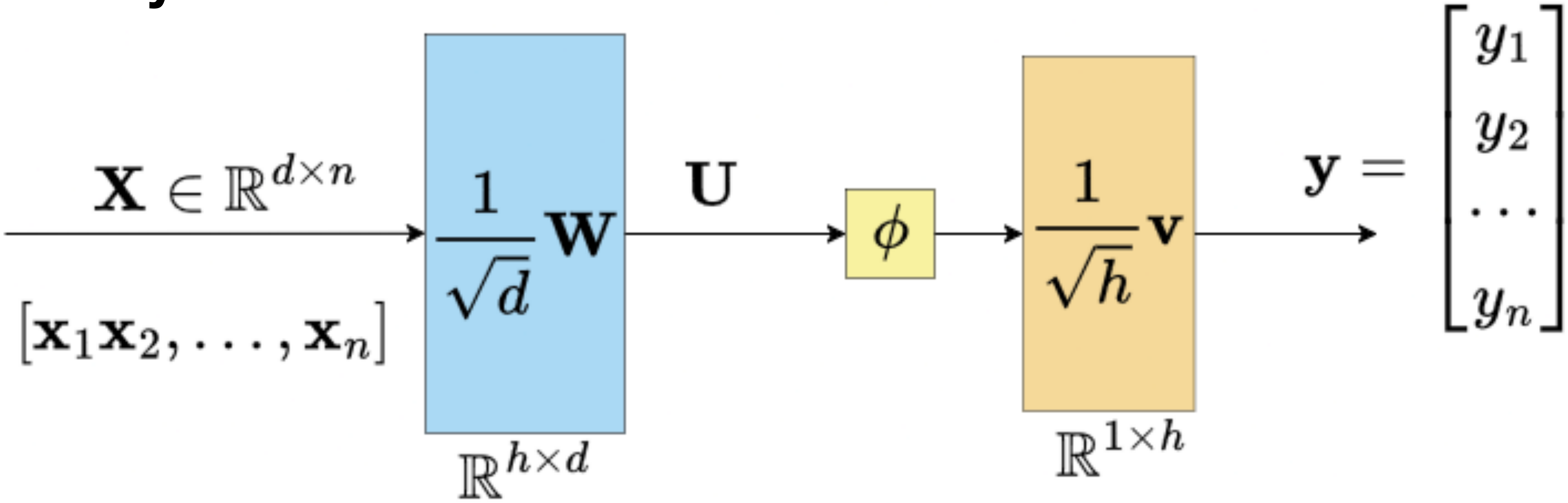
The toy model

A two-layer NN



The toy model

A two-layer NN



$\phi(x)$ is λ_ϕ-Lipschitz

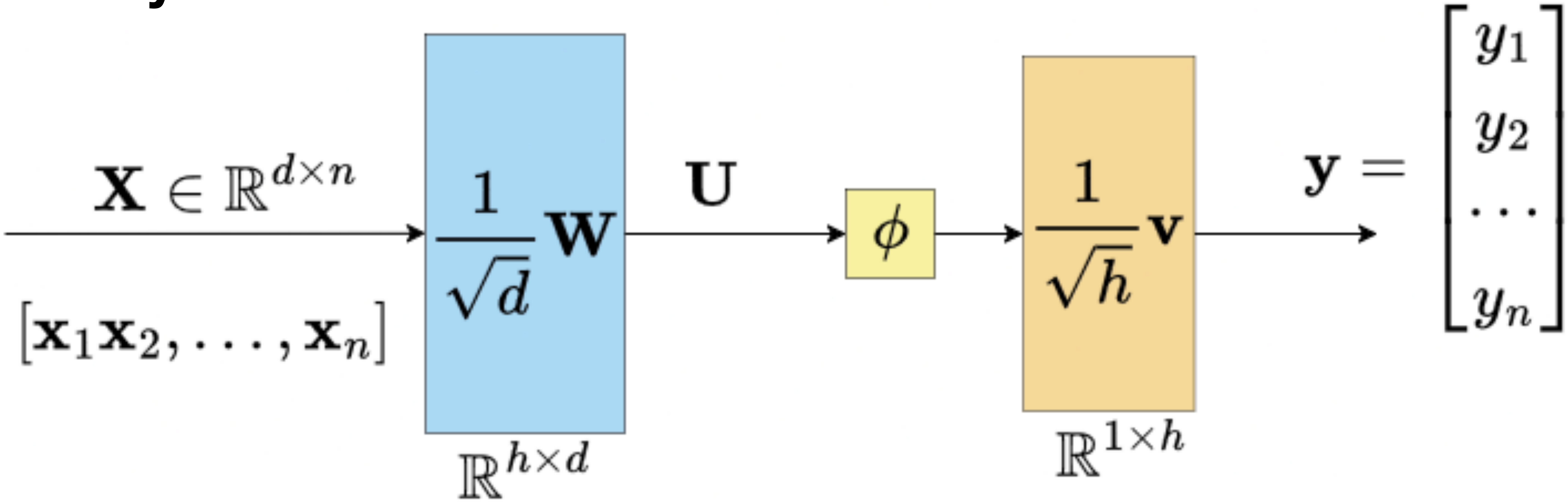
$|\phi'(x)| \leq \lambda_\phi,$

$|\phi''(x)| \leq \lambda_\phi,$

$\mathbb{E}[\phi(z)] = 0$ for $z \sim \mathcal{N}(0,1)$.

The toy model

A two-layer NN



$\phi(x)$ is λ_ϕ-Lipschitz

$|\phi'(x)| \leq \lambda_\phi,$

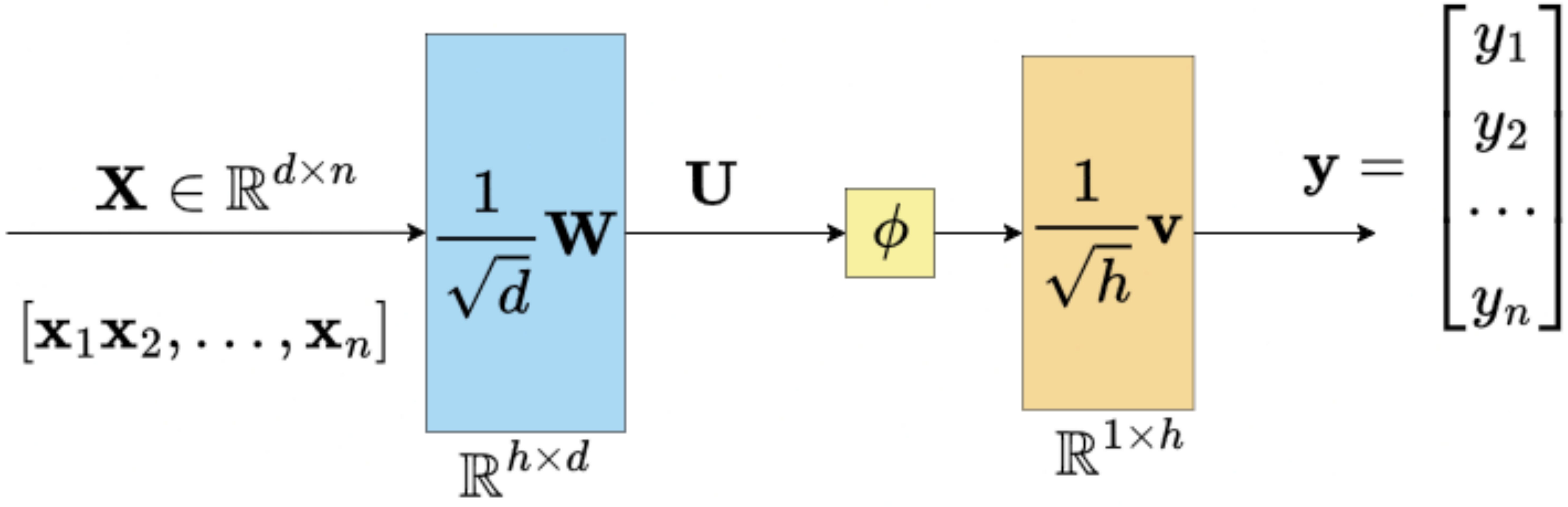
$|\phi''(x)| \leq \lambda_\phi,$

$\mathbb{E}[\phi(z)] = 0$ for $z \sim \mathcal{N}(0,1)$.

$$f(\mathbf{x};\theta) = \frac{1}{\sqrt{h}}\mathbf{v}^\intercal\phi\left(\frac{1}{\sqrt{d}}\mathbf{W}^\intercal\mathbf{x}\right)$$

Initialization and Evolution

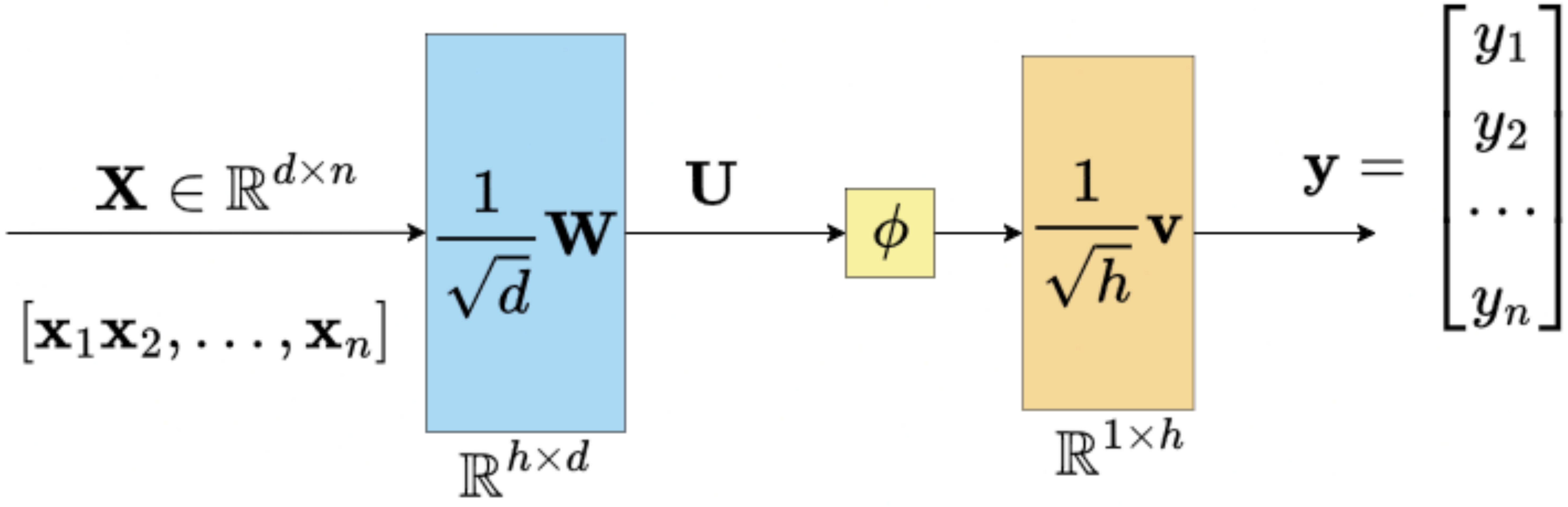
Minimizing unregularized quadratic loss



Initialization and Evolution

Minimizing unregularized quadratic loss

Choose $\mathbf{W} \in \mathbb{R}^{h\times d}$ to have i.i.d. $\mathcal{N}(0,1)$ entries and $\|\mathbf{v}\|_\infty \leq 1$.

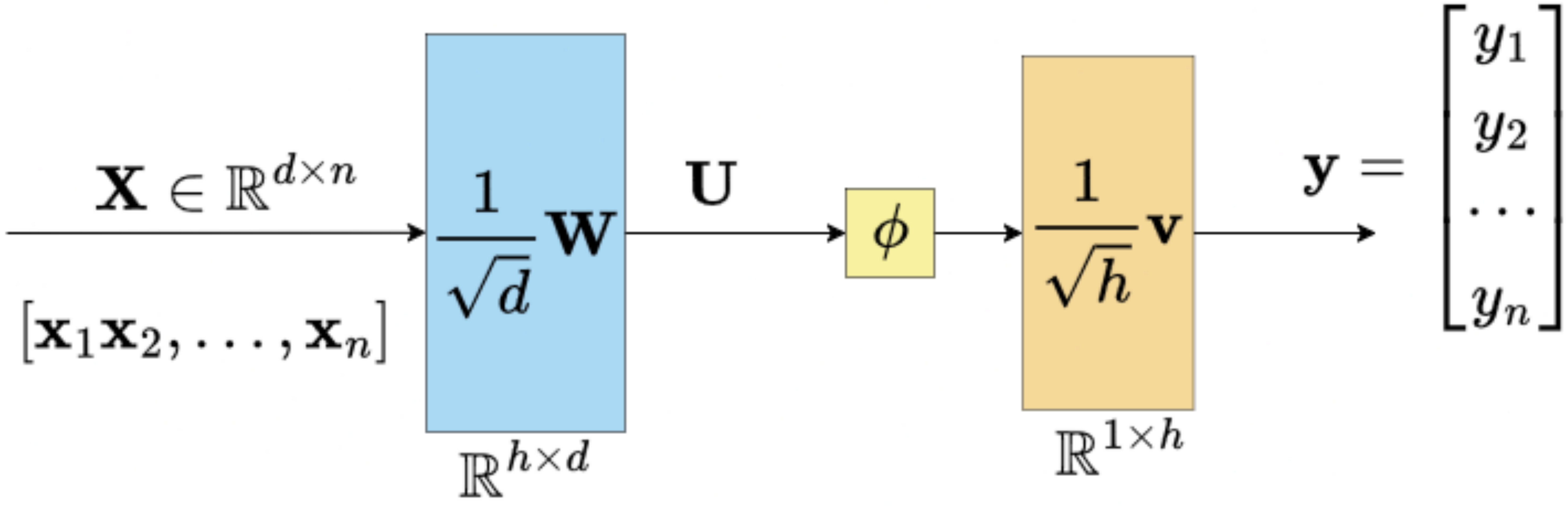


Initialization and Evolution

Minimizing unregularized quadratic loss

Choose $\mathbf{W} \in \mathbb{R}^{h \times d}$ to have i.i.d. $\mathcal{N}(0,1)$ entries and $\|\mathbf{v}\|_\infty \leq 1.$

Optimize the quadratic loss:

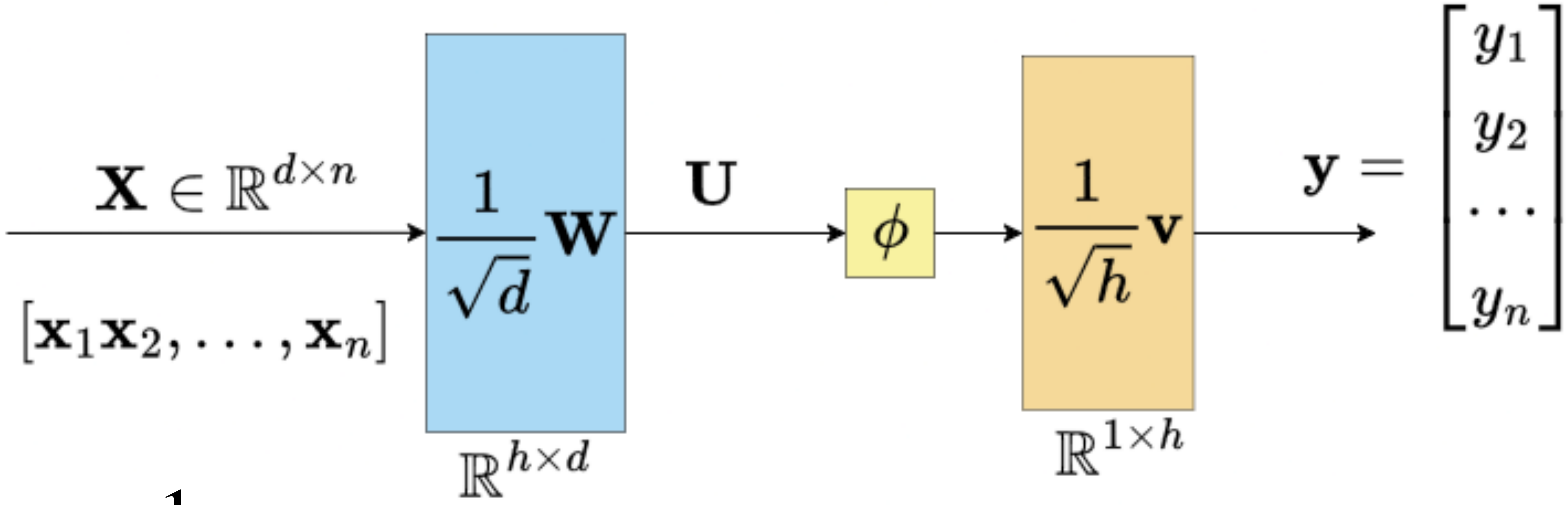


Initialization and Evolution

Minimizing unregularized quadratic loss

Choose $\mathbf{W} \in \mathbb{R}^{h \times d}$ to have i.i.d. $\mathcal{N}(0,1)$ entries and $\|\mathbf{v}\|_\infty \leq 1$.

Optimize the quadratic loss:

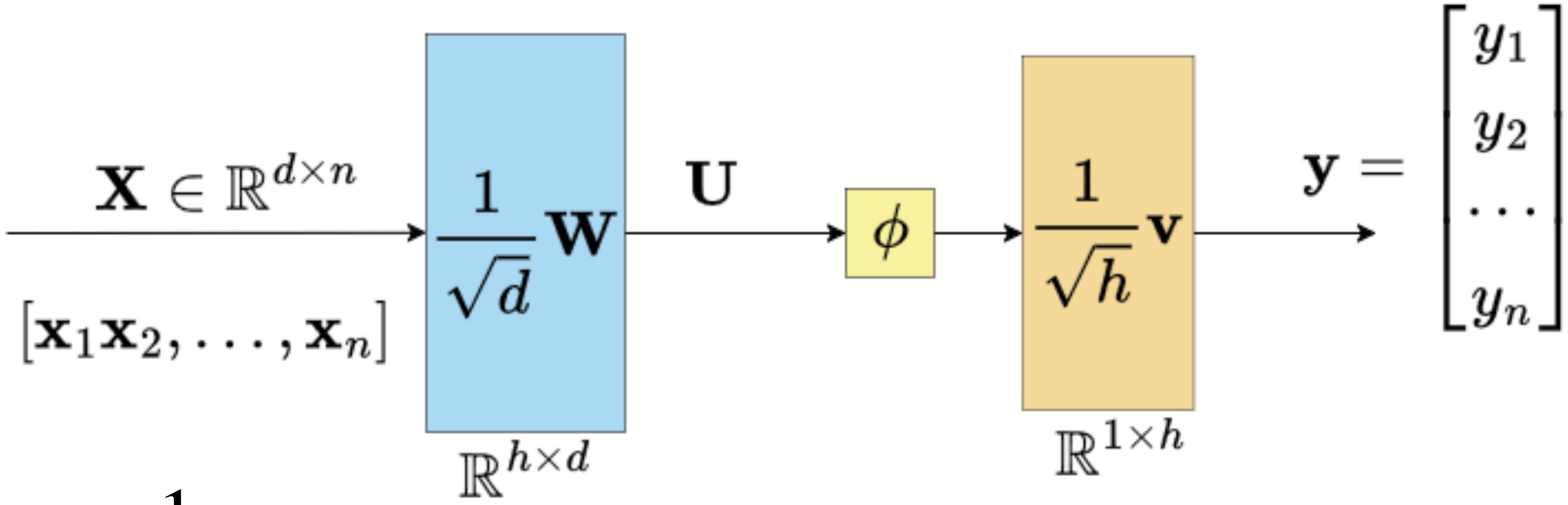


$$\mathcal{L}(\theta) = \frac{1}{2n} \left\| \mathbf{y} - f(\mathbf{X}; \theta) \right\|^2.$$

Initialization and Evolution

Minimizing unregularized quadratic loss

Choose $\mathbf{W} \in \mathbb{R}^{h \times d}$ to have i.i.d. $\mathcal{N}(0,1)$ entries and $\|\mathbf{v}\|_\infty \leq 1$.



Optimize the quadratic loss:

$$\mathcal{L}(\theta) = \frac{1}{2n} \| \mathbf{y} - f(\mathbf{X}; \theta) \|^2.$$

Compare the initialized model $\mathbf{W}_0$ and the model $\mathbf{W}_t$ after t gradient descent (GD) steps.

Matrices of interest

Weights, conjugate kernel, NTK

Matrices of interest

Weights, conjugate kernel, NTK

We are interested in the spectra of the following, given training inputs $\mathbf{X} \in \mathbb{R}^{d \times n}$:

Matrices of interest

Weights, conjugate kernel, NTK

We are interested in the spectra of the following, given training inputs $\mathbf{X} \in \mathbb{R}^{d \times n}$:

- The weights: $\boldsymbol{\Sigma}_t = \dfrac{1}{h}\mathbf{W}_t^\intercal \mathbf{W}_t$.

Matrices of interest

Weights, conjugate kernel, NTK

We are interested in the spectra of the following, given training inputs $\mathbf{X} \in \mathbb{R}^{d \times n}$:

- The weights: $\boldsymbol{\Sigma}_t = \frac{1}{h} \mathbf{W}_t^\intercal \mathbf{W}_t$.
- The conjugate kernel: $\mathbf{K}_t^{\mathrm{CK}} = \left(\phi\left(\mathbf{U}_t\right)\right)^\intercal \left(\phi\left(\mathbf{U}_t\right)\right)$.

Matrices of interest

Weights, conjugate kernel, NTK

We are interested in the spectra of the following, given training inputs $\mathbf{X} \in \mathbb{R}^{d \times n}$:

- The weights: $\boldsymbol{\Sigma}_t = \dfrac{1}{h}\mathbf{W}_t^\intercal \mathbf{W}_t$.
- The conjugate kernel: $\mathbf{K}_t^{\mathrm{CK}} = \Big(\phi\left(\mathbf{U}_t\right)\Big)^\intercal \Big(\phi\left(\mathbf{U}_t\right)\Big)$.
- The empirical NTK (eNTK), which is the Gram matrix of the gradients on the training points:

Matrices of interest

Weights, conjugate kernel, NTK

We are interested in the spectra of the following, given training inputs $\mathbf{X} \in \mathbb{R}^{d \times n}$:

- The weights: $\boldsymbol{\Sigma}_t = \frac{1}{h}\mathbf{W}_t^\intercal \mathbf{W}_t$.
- The conjugate kernel: $\mathbf{K}_t^{\mathrm{CK}} = \left(\phi\left(\mathbf{U}_t\right)\right)^\intercal \left(\phi\left(\mathbf{U}_t\right)\right)$.
- The empirical NTK (eNTK), which is the Gram matrix of the gradients on the training points:

$$\mathbf{K}_t^{\mathrm{NTK}} = \mathbf{X}^\intercal \mathbf{X} \odot \phi'\left(\mathbf{U}_t\right)^\intercal \mathrm{diag}(\mathbf{v})^2 \phi'\left(\mathbf{U}_t\right) + \mathbf{K}_t^{\mathrm{CK}}.$$

Learning a nonlinear model

Mixture of a GLM and a quadratic term

Learning a nonlinear model

Mixture of a GLM and a quadratic term

Generated labels from a GLM with a single index $\boldsymbol{\beta}$

Learning a nonlinear model

Mixture of a GLM and a quadratic term

Generated labels from a GLM with a single index $\boldsymbol{\beta}$

$$y_i = g^*(\mathbf{x}_i^\intercal \boldsymbol{\beta}) + \frac{\tau}{d}\|\mathbf{x}_i\|^2 + \varepsilon_i,$$

Learning a nonlinear model

Mixture of a GLM and a quadratic term

Generated labels from a GLM with a single index $\boldsymbol{\beta}$

$$y_i = g^*(\mathbf{x}_i^\intercal \boldsymbol{\beta}) + \frac{\tau}{d}\|\mathbf{x}_i\|^2 + \varepsilon_i,$$

where $\mathbf{x}_i \sim \mathcal{N}(\mathbf{0}, I_d)$ and ε_i are centered, sub-Gaussian, and have variance σ_ε^2.

Learning a nonlinear model

Mixture of a GLM and a quadratic term

Generated labels from a GLM with a single index $\boldsymbol{\beta}$

$$y_i = g^*(\mathbf{x}_i^\intercal \boldsymbol{\beta}) + \frac{\tau}{d}\|\mathbf{x}_i\|^2 + \varepsilon_i,$$

where $\mathbf{x}_i \sim \mathcal{N}(\mathbf{0}, I_d)$ and ε_i are centered, sub-Gaussian, and have variance σ_ε^2.

- GD: full gradient descent.

Learning a nonlinear model

Mixture of a GLM and a quadratic term

Generated labels from a GLM with a single index $\boldsymbol{\beta}$

$$y_i = g^*(\mathbf{x}_i^\intercal \boldsymbol{\beta}) + \frac{\tau}{d}\|\mathbf{x}_i\|^2 + \varepsilon_i,$$

where $\mathbf{x}_i \sim \mathcal{N}(\mathbf{0}, I_d)$ and ε_i are centered, sub-Gaussian, and have variance σ_ε^2.

- GD: full gradient descent.
- SGD-small: stochastic gradient descent with a small step size

Learning a nonlinear model

Mixture of a GLM and a quadratic term

Generated labels from a GLM with a single index $\boldsymbol{\beta}$

$$y_i = g^*(\mathbf{x}_i^\intercal \boldsymbol{\beta}) + \frac{\tau}{d}\|\mathbf{x}_i\|^2 + \varepsilon_i,$$

where $\mathbf{x}_i \sim \mathcal{N}(\mathbf{0}, I_d)$ and ε_i are centered, sub-Gaussian, and have variance σ_ε^2.

- GD: full gradient descent.
- SGD-small: stochastic gradient descent with a small step size
- SGD-large: SGD with a large step size

Learning a nonlinear model

Mixture of a GLM and a quadratic term

Generated labels from a GLM with a single index $\boldsymbol{\beta}$

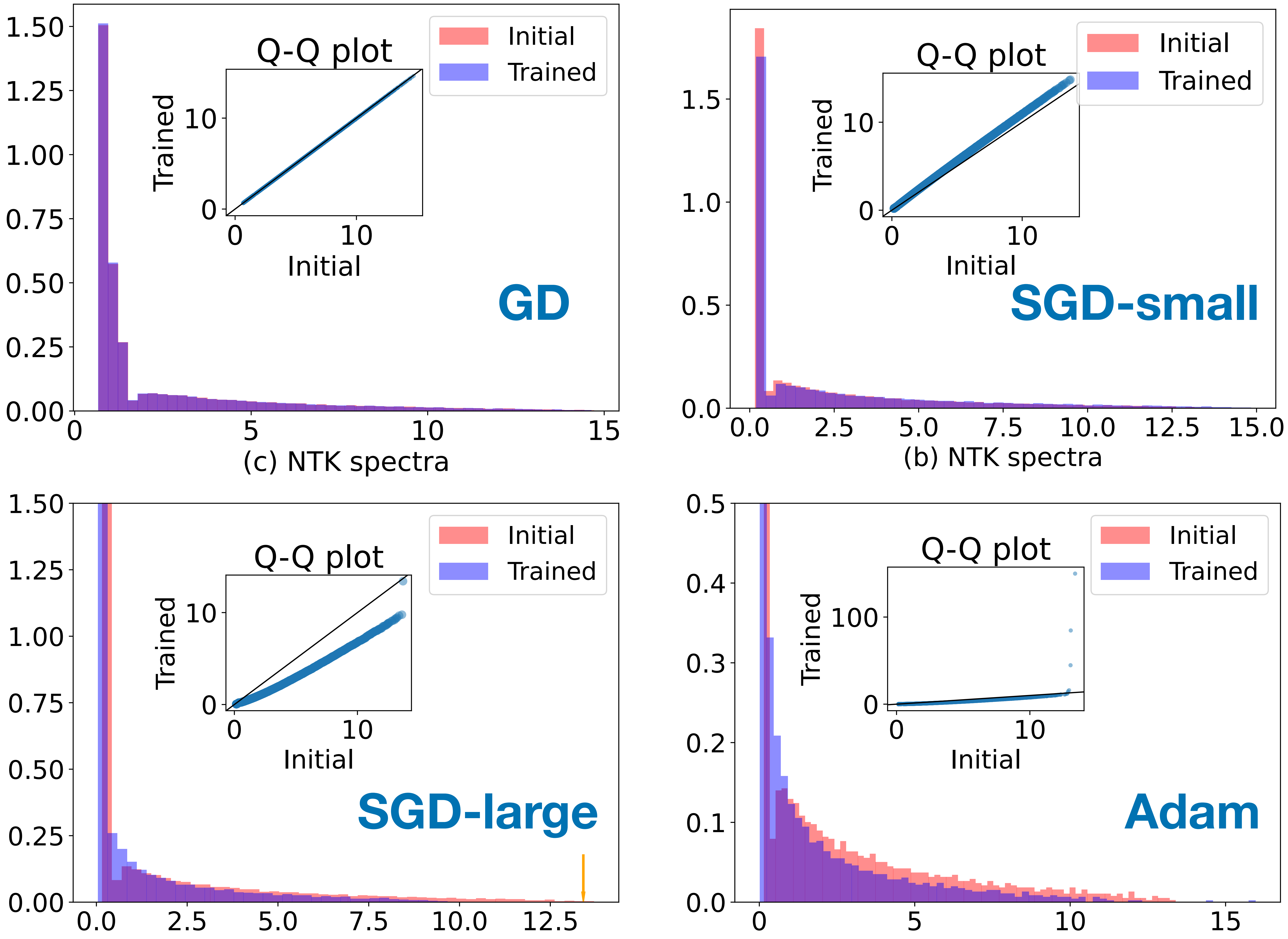
$$y_i = g^*(\mathbf{x}_i^\intercal \boldsymbol{\beta}) + \frac{\tau}{d}\|\mathbf{x}_i\|^2 + \varepsilon_i,$$

where $\mathbf{x}_i \sim \mathcal{N}(\mathbf{0}, I_d)$ and ε_i are centered, sub-Gaussian, and have variance σ_ε^2.

- GD: full gradient descent.
- SGD-small: stochastic gradient descent with a small step size
- SGD-large: SGD with a large step size
- Adam (Kingma and Ba, 2014)

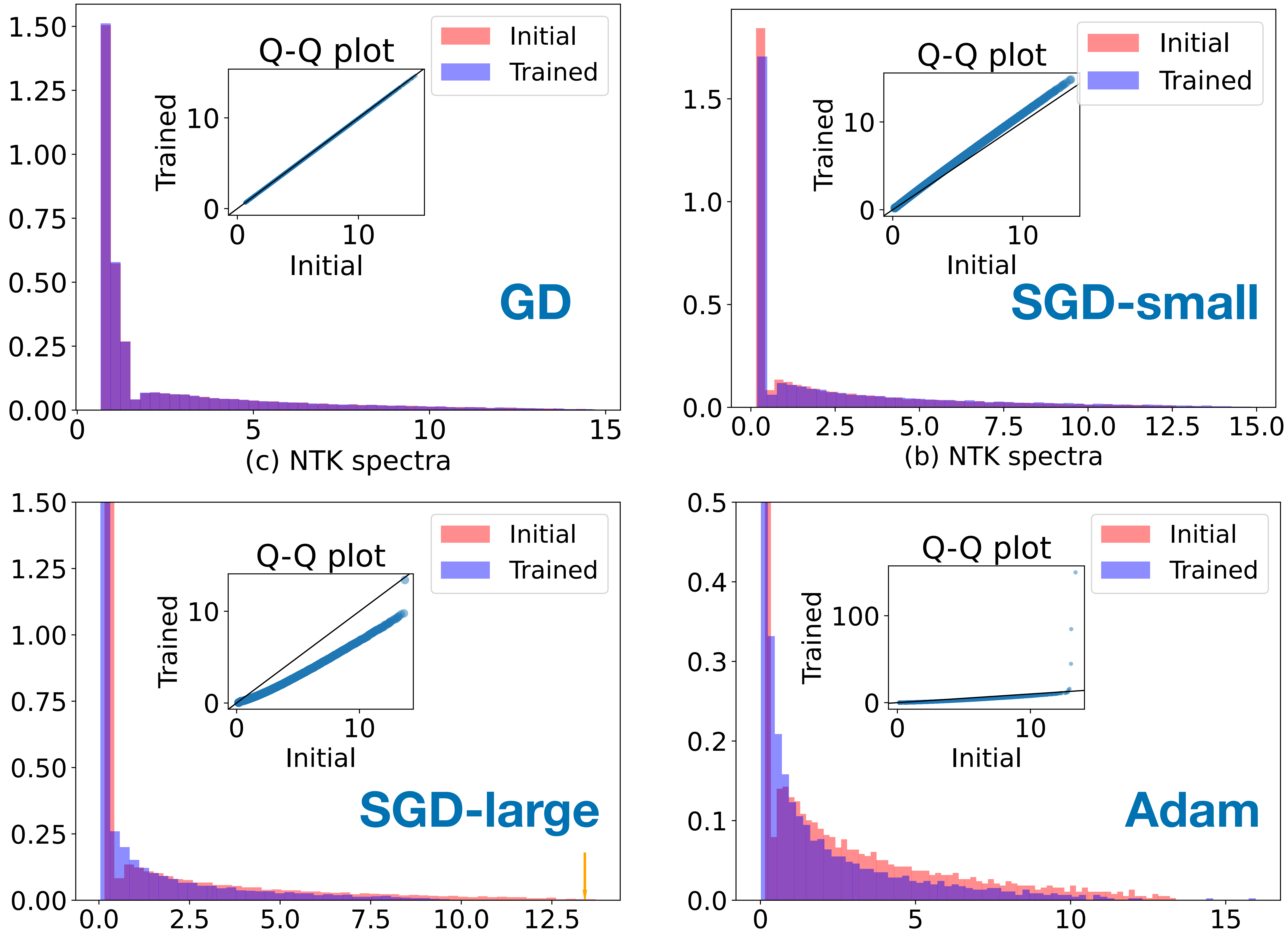
Training may or may not affect the spectra

Exploring the impact of different training algorithms



Training may or may not affect the spectra

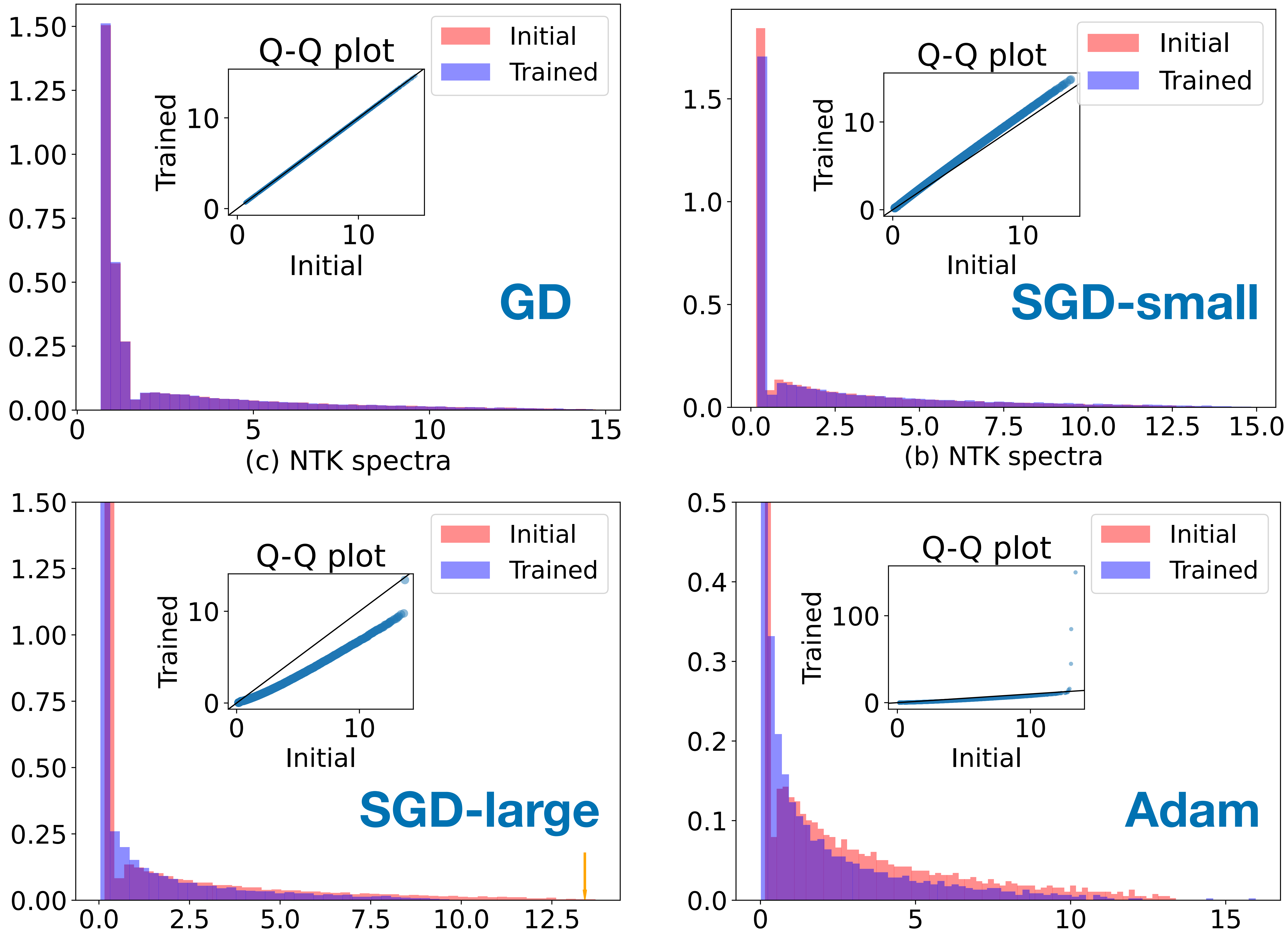
Exploring the impact of different training algorithms



- For gradient descent (GD) and stochastic gradient descent (SGD) with “small” learning rate, the spectra do not change much.

Training may or may not affect the spectra

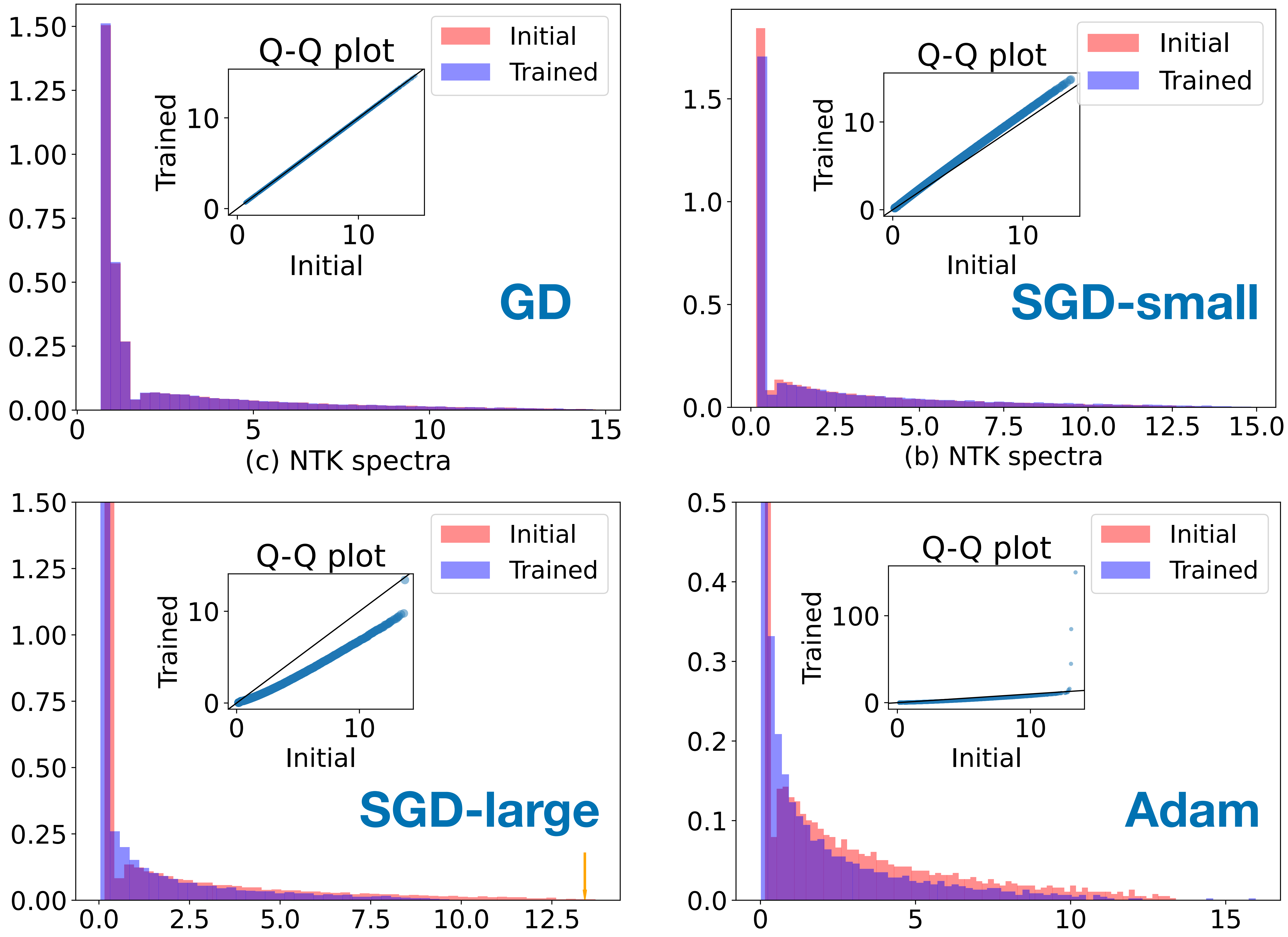
Exploring the impact of different training algorithms



- For gradient descent (GD) and stochastic gradient descent (SGD) with “small” learning rate, the spectra do not change much.
- For SGD with larger learning rate, we get a “bulk + spike” spectrum.

Training may or may not affect the spectra

Exploring the impact of different training algorithms



- For gradient descent (GD) and stochastic gradient descent (SGD) with "small" learning rate, the spectra do not change much.
- For SGD with larger learning rate, we get a "bulk + spike" spectrum.
- For Adam, the spectra are heavy-tailed.

Invariant spectra for small learning rates

Learning rates have to be $\Omega(n)$ to see change

Invariant spectra for small learning rates

Learning rates have to be $\Omega(n)$ to see change

Theorem (early phase, informal): Suppose we train the first layer $\mathbf{W}$ using gradient descent. Then under the assumptions, if the learning rate $\eta = \Theta(1)$, for any fixed number of iterations t, $\frac{1}{\sqrt{d}}\|\mathbf{W}_t - \mathbf{W}_0\|_F$, $\|\mathbf{K}_t^{\mathrm{CK}} - \mathbf{K}_0^{\mathrm{CK}}\|_F$, and $\|\mathbf{K}_t^{\mathrm{NTK}} - \mathbf{K}_0^{\mathrm{NTK}}\|_F$ are all $O(1/n)$ under LWR.

Invariant spectra for small learning rates

Learning rates have to be $\Omega(n)$ to see change

Theorem (early phase, informal): Suppose we train the first layer $\mathbf{W}$ using gradient descent. Then under the assumptions, if the learning rate $\eta = \Theta(1)$, for any fixed number of iterations t, $\frac{1}{\sqrt{d}}\|\mathbf{W}_t - \mathbf{W}_0\|_F$, $\|\mathbf{K}_t^{\mathrm{CK}} - \mathbf{K}_0^{\mathrm{CK}}\|_F$, and $\|\mathbf{K}_t^{\mathrm{NTK}} - \mathbf{K}_0^{\mathrm{NTK}}\|_F$ are all $O(1/n)$ under LWR.

Invariant spectra for small learning rates

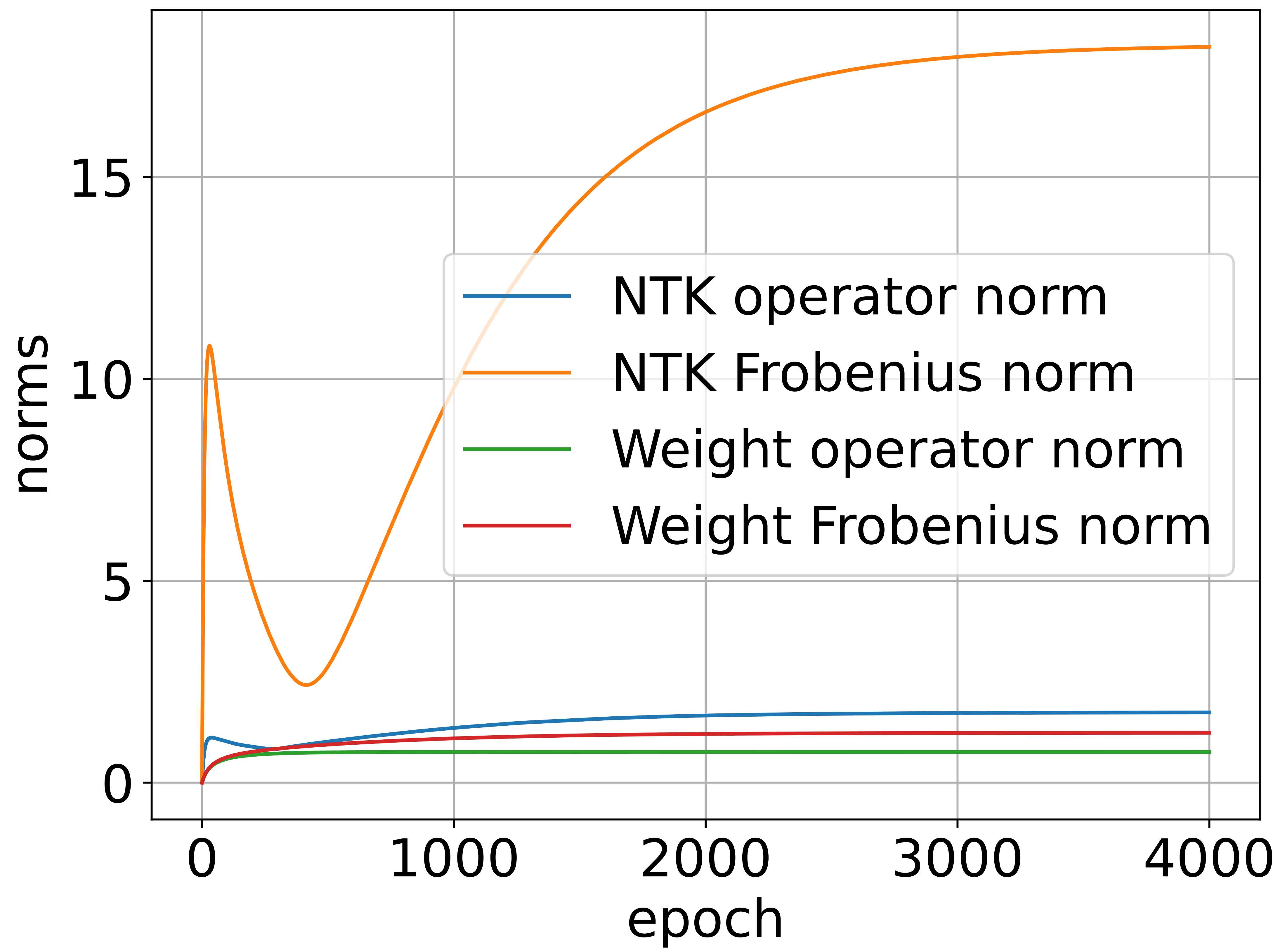
Learning rates have to be $\Omega(n)$ to see change

Theorem (early phase, informal): Suppose we train the first layer $\mathbf{W}$ using gradient descent. Then under the assumptions, if the learning rate $\eta = \Theta(1)$, for any fixed number of iterations t, $\frac{1}{\sqrt{d}} \|\mathbf{W}_t - \mathbf{W}_0\|_F$, $\|\mathbf{K}_t^{\mathrm{CK}} - \mathbf{K}_0^{\mathrm{CK}}\|_F$, and $\|\mathbf{K}_t^{\mathrm{NTK}} - \mathbf{K}_0^{\mathrm{NTK}}\|_F$ are all $O(1/n)$ under LWR.

This means that GD (can extend to SGD) with too small step size doesn't do much in the limit.

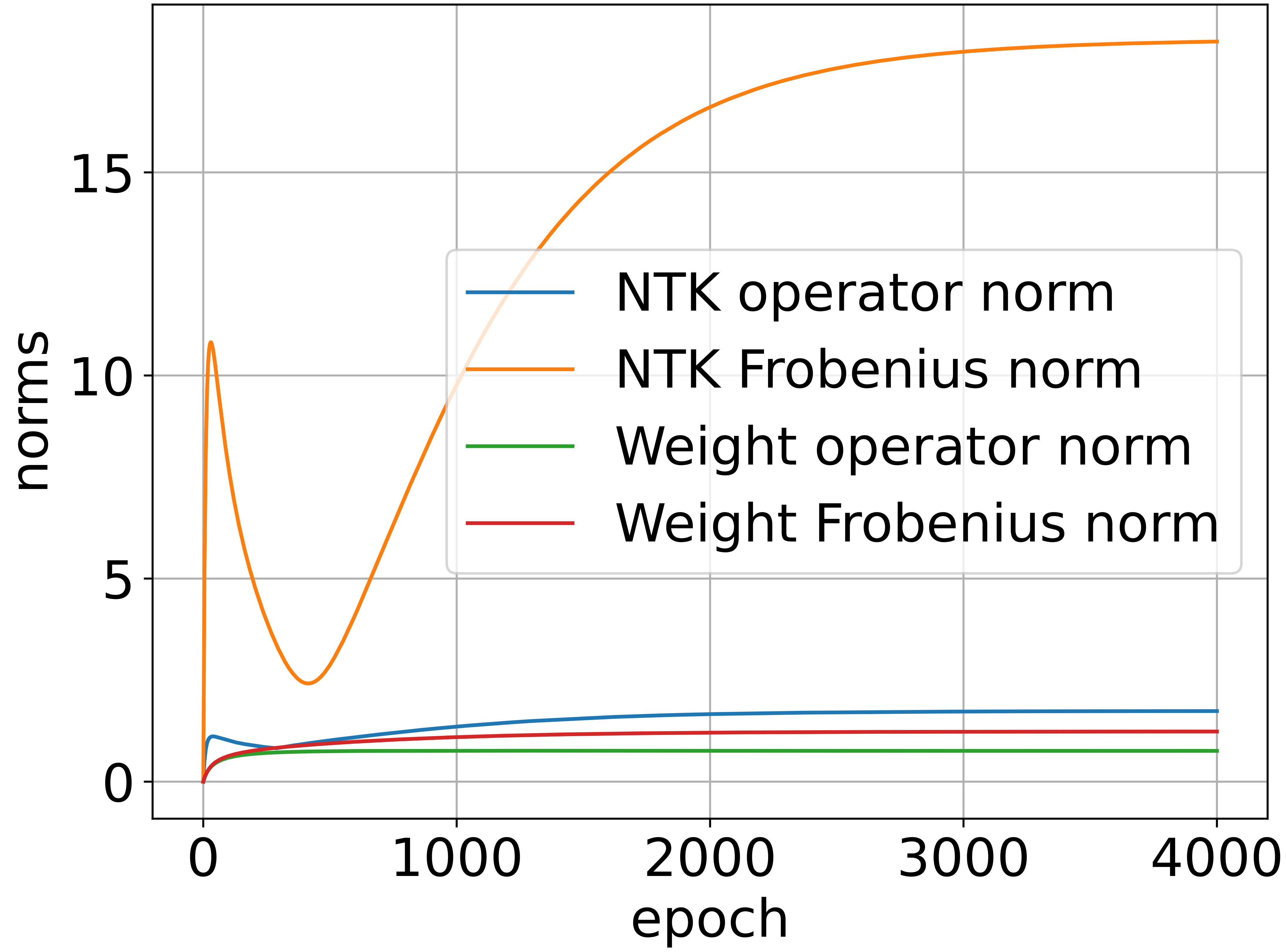
Invariant spectra for small learning rates

Small steps don’t help us break out



Invariant spectra for small learning rates

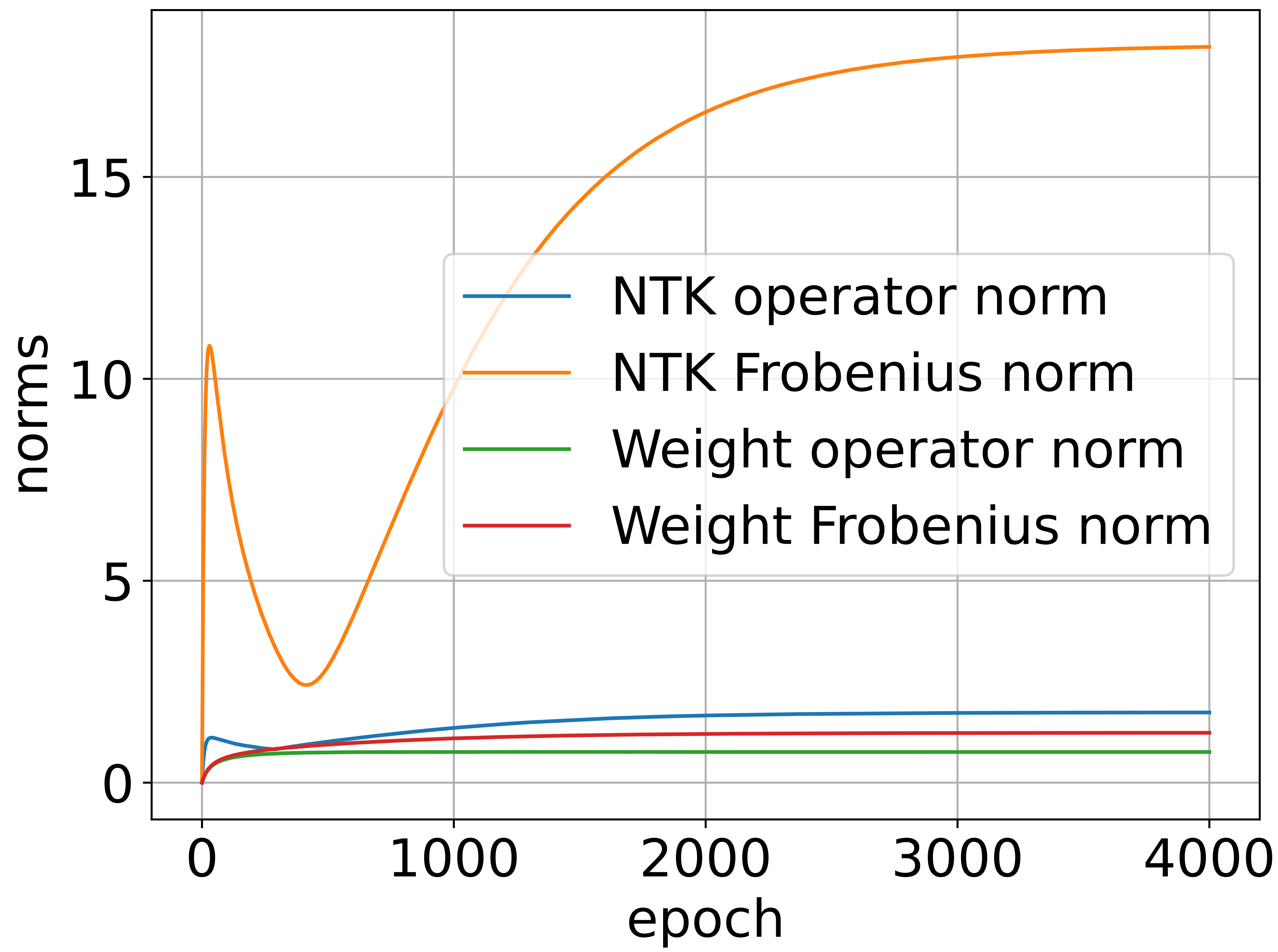
Small steps don't help us break out



Theorem (bulk spectra, informal): There are constants C, γ^*, R such that if $\eta \leq Cn$ and $h/d \to \gamma_2 \geq \gamma^*$, then with high probability:

Invariant spectra for small learning rates

Small steps don't help us break out

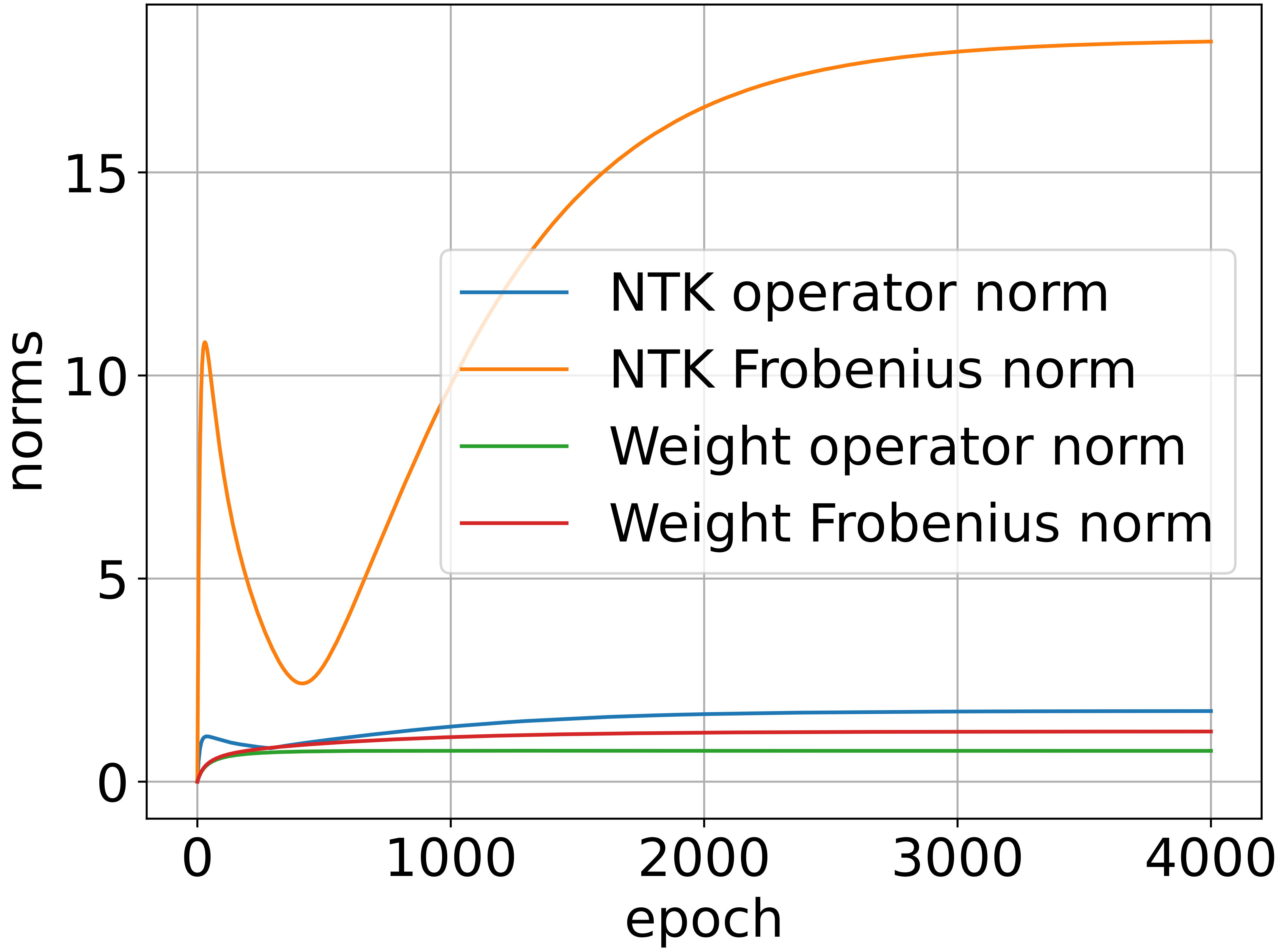


Theorem (bulk spectra, informal): There are constants C, γ^*, R such that if $\eta \leq Cn$ and $h/d \to \gamma_2 \geq \gamma^*$, then with high probability:

$$\frac{1}{\sqrt{d}} \|\mathbf{W}_t - \mathbf{W}_0\|_F,$$

Invariant spectra for small learning rates

Small steps don't help us break out



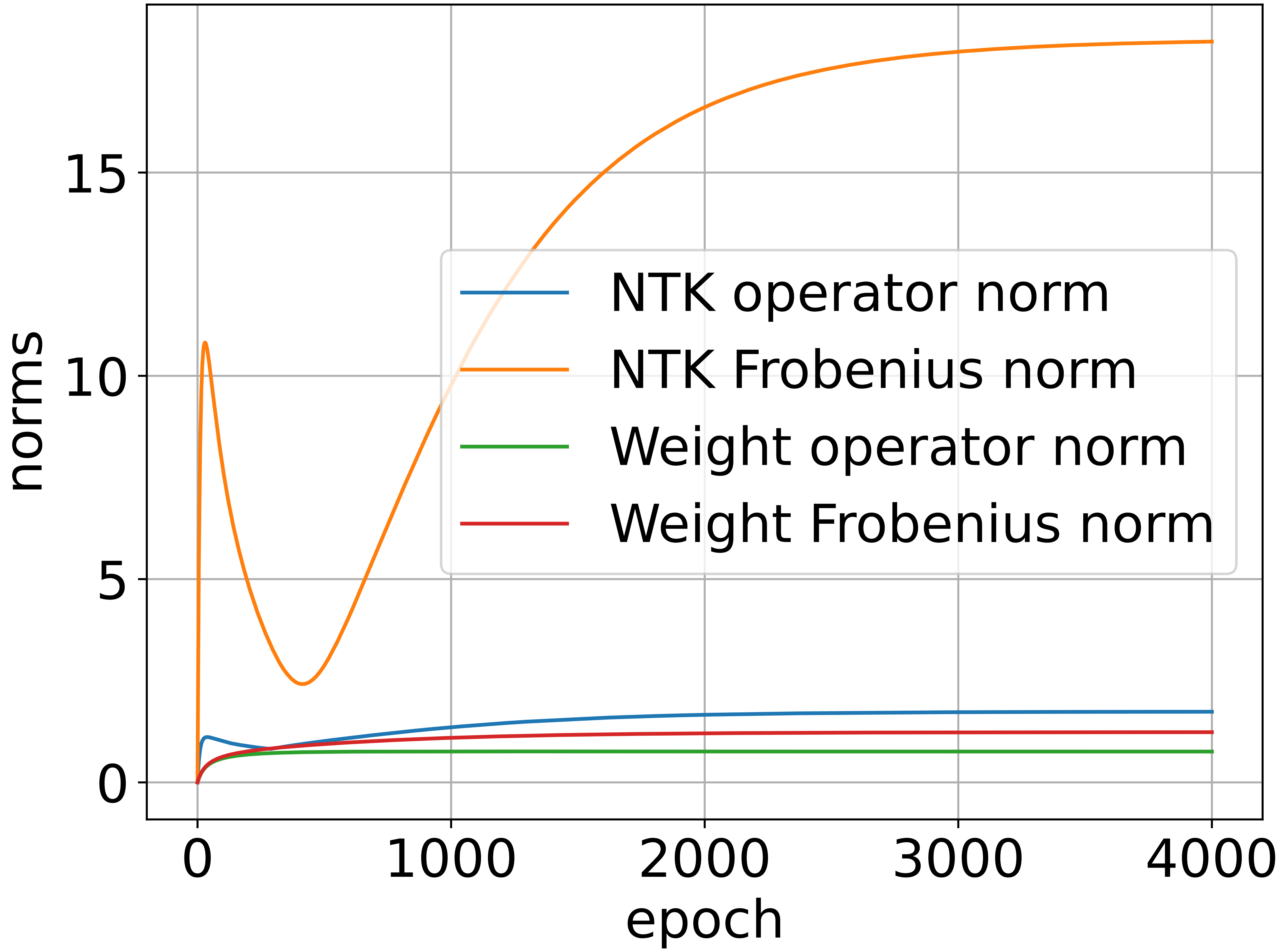
Theorem (bulk spectra, informal): There are constants C, γ^*, R such that if $\eta \leq Cn$ and $h/d \to \gamma_2 \geq \gamma^*$, then with high probability:

$$\frac{1}{\sqrt{d}}\|\mathbf{W}_t - \mathbf{W}_0\|_F,$$

$$\|\mathbf{K}_t^{\mathrm{CK}} - \mathbf{K}_0^{\mathrm{CK}}\|_F,$$

Invariant spectra for small learning rates

Small steps don't help us break out



Theorem (bulk spectra, informal): There are constants C, γ^*, R such that if $\eta \leq Cn$ and $h/d \to \gamma_2 \geq \gamma^*$, then with high probability:

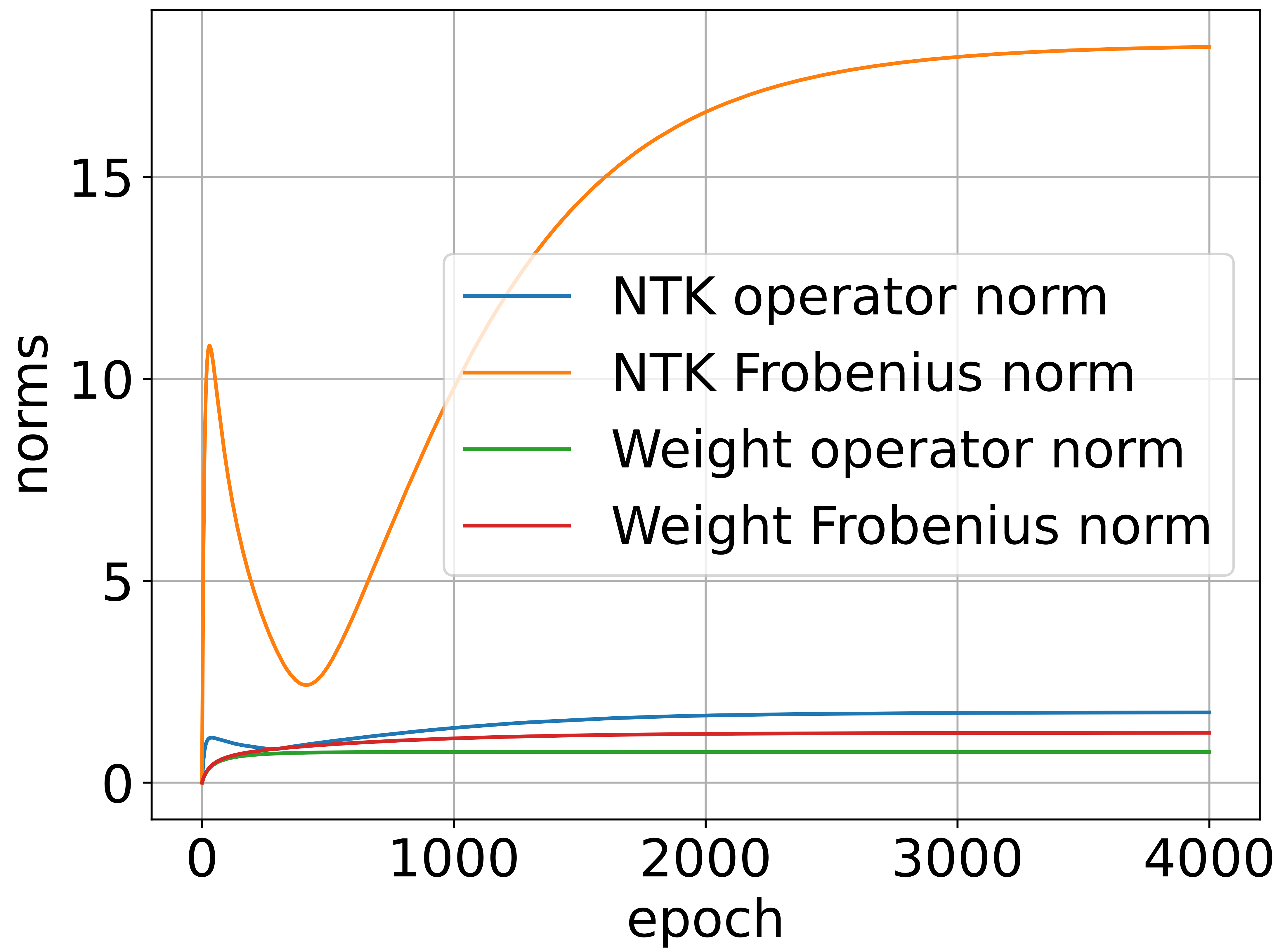
$$\frac{1}{\sqrt{d}} \|\mathbf{W}_t - \mathbf{W}_0\|_F,$$

$$\|\mathbf{K}_t^{\mathrm{CK}} - \mathbf{K}_0^{\mathrm{CK}}\|_F,$$

$$\|\mathbf{K}_t^{\mathrm{NTK}} - \mathbf{K}_0^{\mathrm{NTK}}\|_F \leq R.$$

Invariant spectra for small learning rates

Small steps don't help us break out



Theorem (bulk spectra, informal): There are constants C, γ^*, R such that if $\eta \leq Cn$ and $h/d \to \gamma_2 \geq \gamma^*$, then with high probability:

$$\frac{1}{\sqrt{d}} \|\mathbf{W}_t - \mathbf{W}_0\|_F,$$

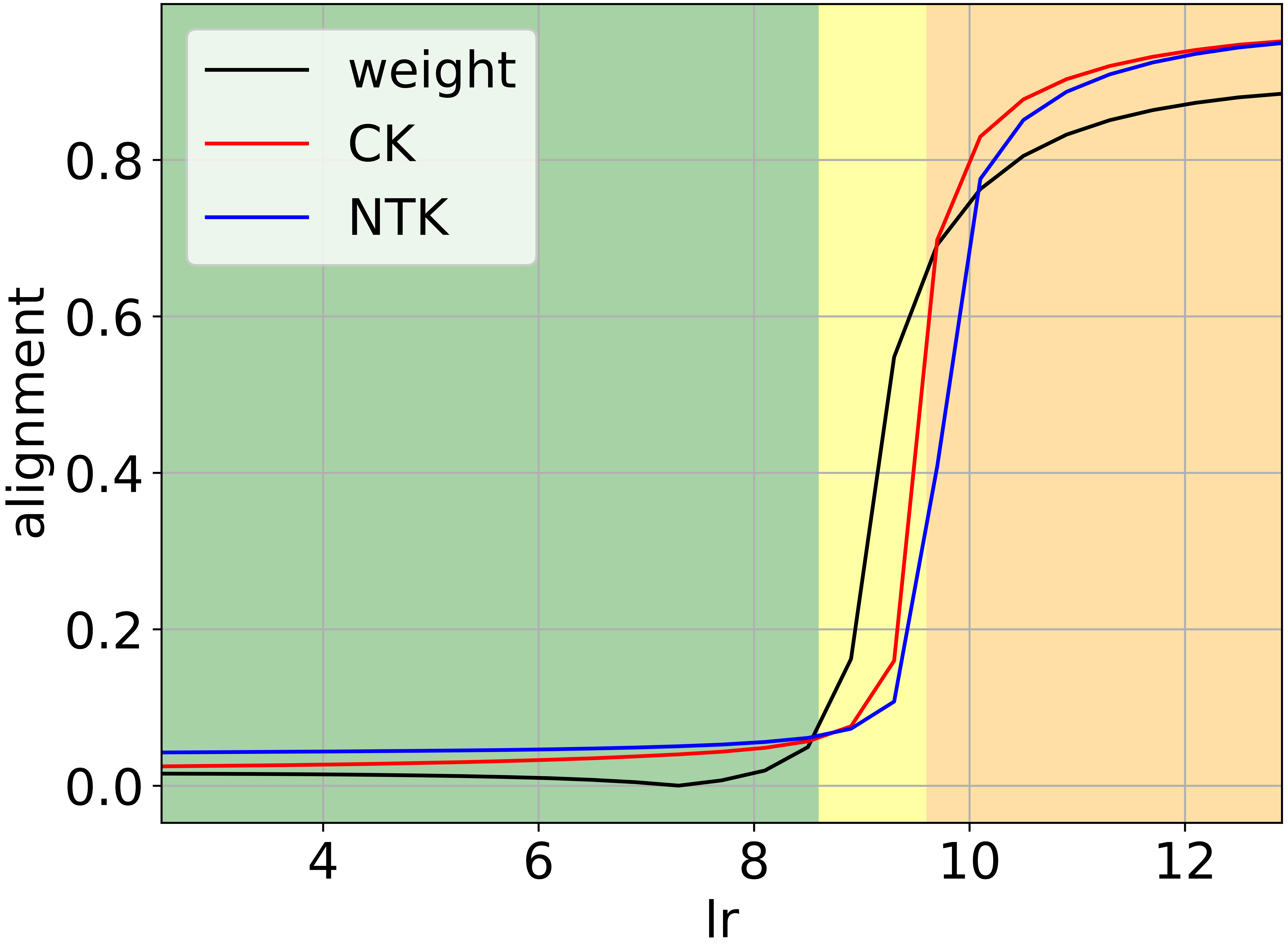
$$\|\mathbf{K}_t^{\mathrm{CK}} - \mathbf{K}_0^{\mathrm{CK}}\|_F,$$

$$\|\mathbf{K}_t^{\mathrm{NTK}} - \mathbf{K}_0^{\mathrm{NTK}}\|_F \leq R.$$

This says that the bulk spectra don't change.

Alignment of kernels to the teacher model

Hopefully we can recover the hidden parameter



Take the top singular vector of the trained kernels and compare it to $\boldsymbol{\beta}$.

Plot shows the alignment (cosine similarity) between these two vectors.

This can be extended to multiple eigenvectors "planted" in the GLM model that we had before.

Some takeaways

Detecting training differences

Some takeaways

Detecting training differences

What this work shows is that the type of optimization algorithm being used should be detectable using the output of the modes.

Some takeaways

Detecting training differences

What this work shows is that the type of optimization algorithm being used should be detectable using the output of the modes.

This is a kind of *forensics:*

Some takeaways

Detecting training differences

What this work shows is that the type of optimization algorithm being used should be detectable using the output of the modes.

This is a kind of *forensics:*

- Determining the camera from an image generated by that camera.

Some takeaways

Detecting training differences

What this work shows is that the type of optimization algorithm being used should be detectable using the output of the modes.

This is a kind of *forensics:*

- Determining the camera from an image generated by that camera.
- Determining if an MRI came from a GE or a Siemens.

Some takeaways

Detecting training differences

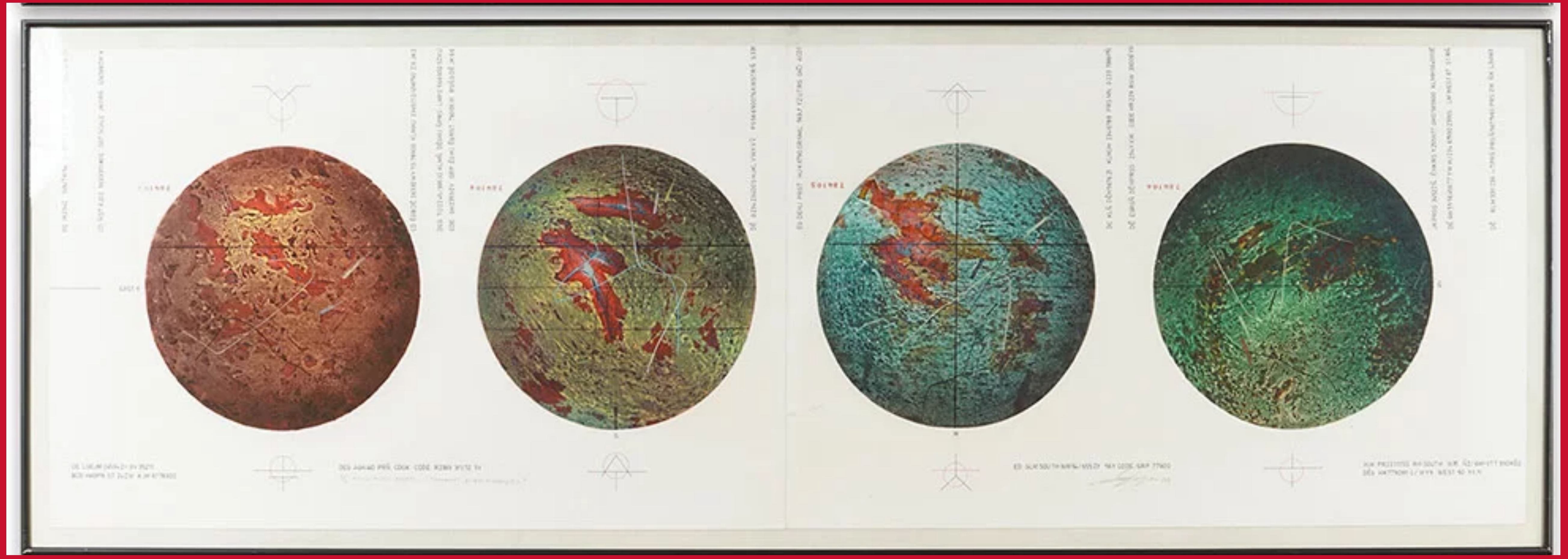
What this work shows is that the type of optimization algorithm being used should be detectable using the output of the modes.

This is a kind of *forensics:*

- Determining the camera from an image generated by that camera.
- Determining if an MRI came from a GE or a Siemens.

These models are different: they will provide different NTKs depending on the optimization method. But what can we learn from the NTKs themselves?

Comparing models and comparing explanations



Rm Palaniappan, *Alien Planet-C*
Viscosity, pencil colour and ink on handmade paper

Explainability in instrumentation

Do AI models have similar reasoning?

Chief Miles O’Brien

A lookalike Miles O’Brien

Explainability in instrumentation

Do AI models have similar reasoning?

Chief Miles O’Brien

A lookalike Miles O’Brien

In scientific instrumentation, the justification for a measurement should be the same across devices.

Explainability in instrumentation

Do AI models have similar reasoning?

Chief Miles O'Brien

A lookalike Miles O'Brien

In scientific instrumentation, the justification for a measurement should be the same across devices.

Should we compare two models in terms of their feature maps?

Explainability in instrumentation

Do AI models have similar reasoning?

Chief Miles O'Brien

A lookalike Miles O'Brien

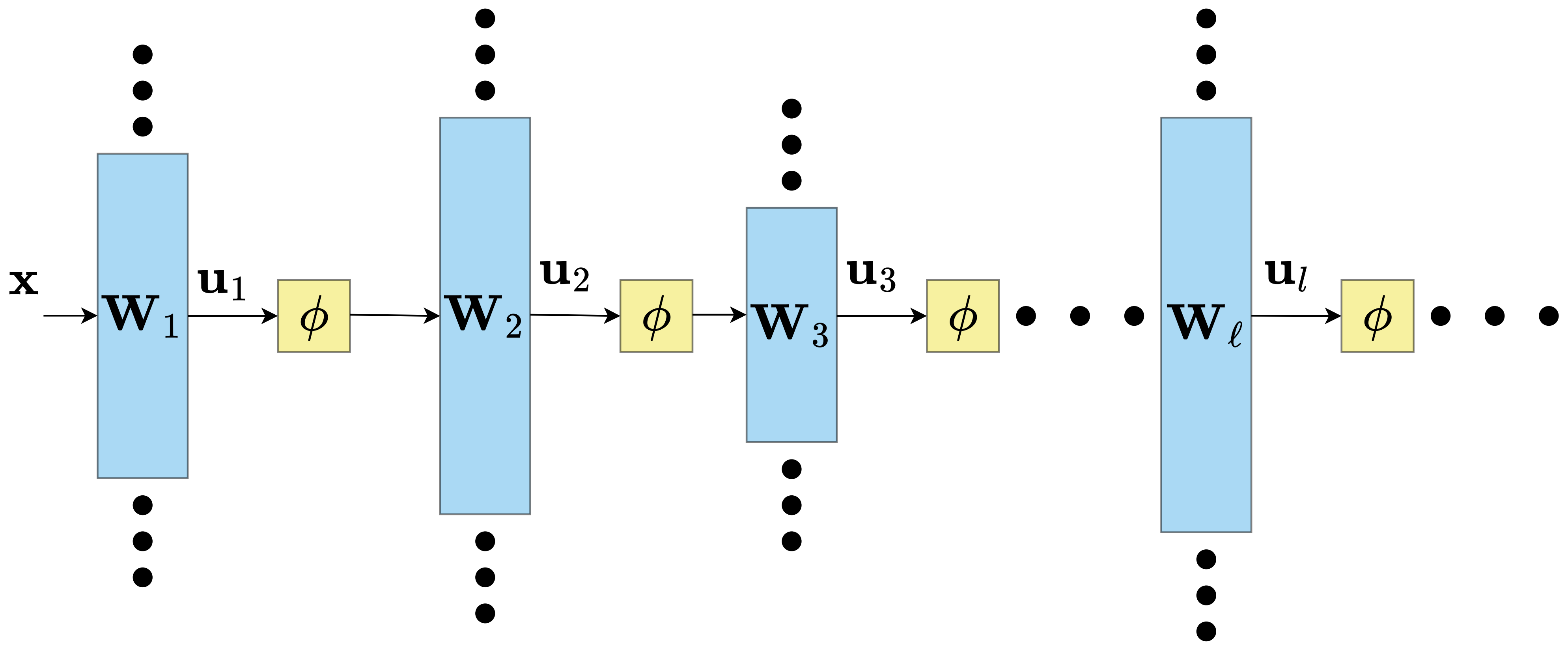
In scientific instrumentation, the justification for a measurement should be the same across devices.

Should we compare two models in terms of their feature maps?

How can we do that in a computationally feasible manner?

Approximating the NN with a kernel machine

Not practical, but perhaps informative?



$$\approx$$

$$\text{kGLM}$$

Suppose we compute some kernel function $\mathbf{K}$ associated to the model and fit a surrogate model $(\mathbf{V}, \mathbf{b})$:

$$\mathbf{y}_i = \mathbf{V}\mathbf{K}(\mathbf{x}_i, \mathbf{X}) + \mathbf{b}$$

where $\mathbf{y}_i, \mathbf{b} \in \mathbb{R}^C$ and $\mathbf{V} \in \mathbb{R}^{C \times N}$. Fitting is done with the same training data (double dipping).

What do we want from a surrogate?

What does it mean for the kGLM to be “similar” to the NN?

What do we want from a surrogate?

What does it mean for the kGLM to be “similar” to the NN?

We want the kGLM to:

What do we want from a surrogate?

What does it mean for the kGLM to be “similar” to the NN?

We want the kGLM to:

- work on multi-class problems,

What do we want from a surrogate?

What does it mean for the kGLM to be “similar” to the NN?

We want the kGLM to:

- work on multi-class problems,
- mimic the performance of the original NN,

What do we want from a surrogate?

What does it mean for the kGLM to be “similar” to the NN?

We want the kGLM to:

- work on multi-class problems,
- mimic the performance of the original NN,
- show how the training data are used by the model to make predictions..

What do we want from a surrogate?

What does it mean for the kGLM to be “similar” to the NN?

We want the kGLM to:

- work on multi-class problems,
- mimic the performance of the original NN,
- show how the training data are used by the model to make predictions..

Idea: use an approximation of the NTK and fit a surrogate model/predictor to allow training points to be scored in terms of similarity.

Measuring faithfulness of a surrogate

What is the fair way to measure

Measuring faithfulness of a surrogate

What is the fair way to measure

Test accuracy gap: $\mathrm{TAD} = \mathrm{TestAcc}_{\mathrm{kGLM}} - \mathrm{TestAcc}_{\mathrm{NN}}.$

Measuring faithfulness of a surrogate

What is the fair way to measure

Test accuracy gap: $\mathrm{TAD} = \mathrm{TestAcc}_{\mathrm{kGLM}} - \mathrm{TestAcc}_{\mathrm{NN}}.$

Kendall-τ measure: given a list of softmax scores $\{(a_i, b_i)\}$ from the NN and kernel model, the pair (i, j) is *concordant* if

$$a_i > a_j \text{ and } b_i > b_j \qquad \text{or} \qquad a_i < a_j \text{ and } b_i < b_j$$

Then

$$\tau_K = \frac{\#\text{concordant} - \#\text{discordant}}{\#\text{concordant} + \#\text{discordant}}.$$

Why not just use the eNTK?

More classes, more problems

We would like to handle multi-class problems and large data sets. In the setting the eNTK becomes huge. For classes i and j define:

$$\mathbf{K}^{\mathrm{NTK}}_{(c,c')}(\mathbf{x}_i, \mathbf{x}_j) = \left\langle \nabla_\theta f^{c}(\mathbf{x}_i;\theta), \nabla_\theta f^{c'}(\mathbf{x}_j;\theta) \right\rangle$$

Then the NTK has a block structure, where each diagonal block has the "regular" NTK for each class and the off-diagonal blocks are cross terms.

$$\begin{bmatrix} \mathbf{K}^{\mathrm{NTK}}_{1,1} & \mathbf{K}^{\mathrm{NTK}}_{1,2} & \cdots & \mathbf{K}^{\mathrm{NTK}}_{1,C} \\ \mathbf{K}^{\mathrm{NTK}}_{2,1} & \mathbf{K}^{\mathrm{NTK}}_{2,2} & \cdots & \mathbf{K}^{\mathrm{NTK}}_{2,C} \\ \vdots & & \ddots & \vdots \\ \mathbf{K}^{\mathrm{NTK}}_{C,1} & \mathbf{K}^{\mathrm{NTK}}_{1,C} & & \mathbf{K}^{\mathrm{NTK}}_{C,C} \end{bmatrix}$$

Trace NTK: a proxy for the eNTK

Much lower computational overhead needed

We look at a simplification of the NTK:

$$\mathbf{K}^{\text{trNTK}}(\mathbf{x}_i, \mathbf{x}_j) = \frac{\sum_{c=1}^{C} \left\langle \nabla_\theta f^c(\mathbf{x}_i; \theta), \nabla_\theta f^c(\mathbf{x}_j; \theta) \right\rangle}{\left(\sum_{c=1}^{C} \left\| f^c(\mathbf{x}_i; \theta) \right\|^2 \right)^{1/2} \left(\sum_{c=1}^{C} \left\| f^c(\mathbf{x}_j; \theta) \right\|^2 \right)^{1/2}}$$

This acts “kind of” like a cosine similarity and is different from other proposed surrogate kernels like the pseudo NTK (pNTK) (Mohamadi & Sutherland, 2022), things based on the CK, (Fan & Wang, 2020; Yeh et al., 2018), the un-normalized trNTK, and the embedding kernel (Akyürek et al., 2023).

Better speedups with random projections (Novak et al., 2022, Park et al., 2023))

The trNTK matches performance pretty well

For 2 and more classes

Model (Dataset)	# Models	NN test acc (%)	TAD (%)	τ_K
MLP (MNIST2)	100	99.64(1)	+0.03(5)	0.708(3)
CNN (MNIST2)	100	98.4(1)	-0.2(2)	0.857(7)
CNN (CIFAR2)	100	94.94(5)	-2.1(5)	0.711(3)
CNN (FMNIST2)	100	97.95(4)	-2.2(2)	0.882(3)
ResNet18 (CIFAR10)	1	93.07	-0.28	0.776
ResNet34 (CIFAR10)	1	93.33	-0.29	0.786
MobileNetV2 (CIFAR10)	1	93.91	-0.4	0.700
BERT-base (COLA)	4	83.4(1)	-0.1(3)	0.78(2)

Comparing different kernel options

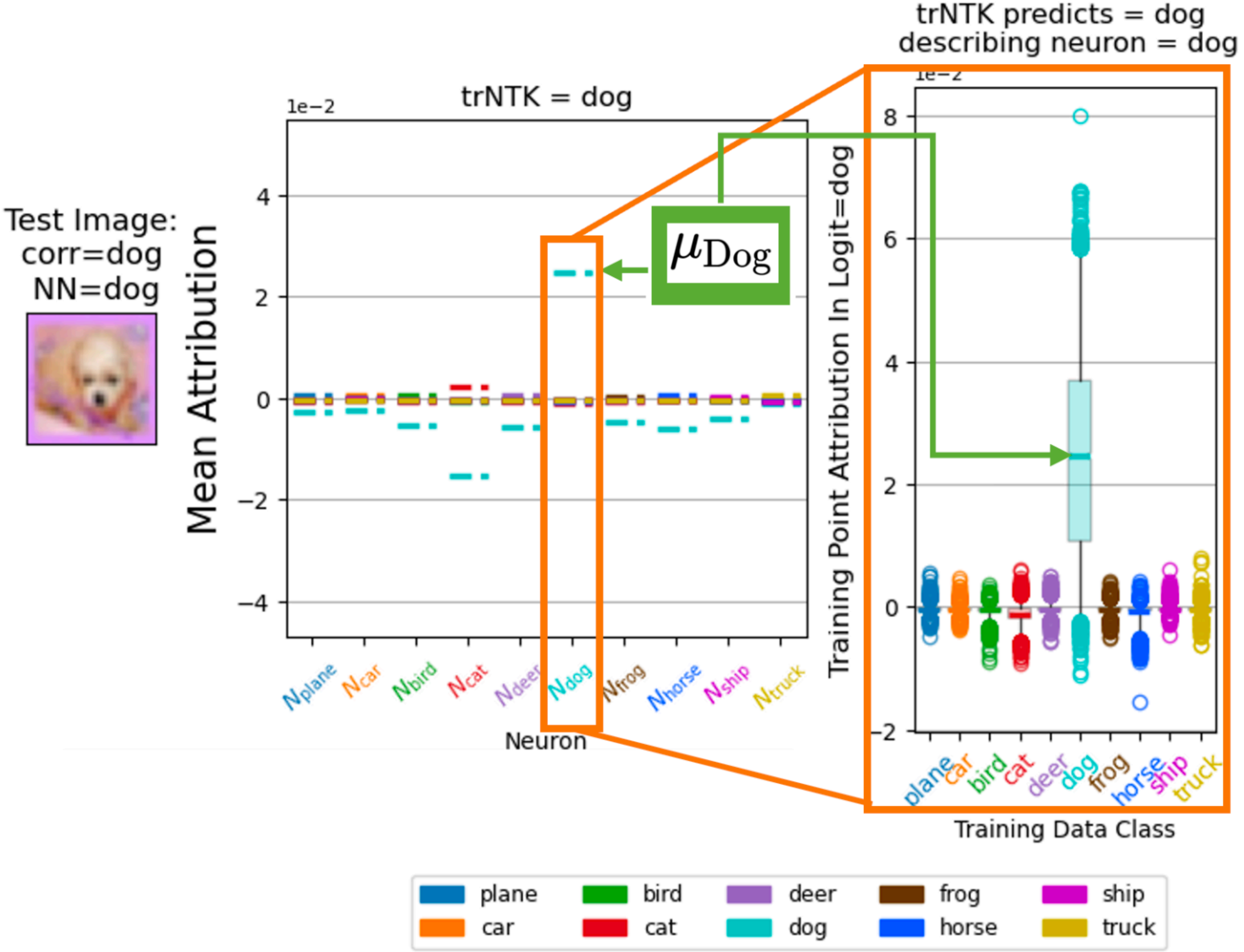
Different notions of "faithfulness"

Exp Name	Metric	κ					
		trNTK	trNTK^0	proj-trNTK	proj-pNTK	Em	CK
ResNet18	τ_K	0.776	0.658	0.737	0.407	0.768	0.630
	TAD (%)	-0.30	-0.52	-0.20	-0.30	-0.32	-0.20
	R_{Miss}	0.75	0.65	0.77	0.71	0.80	0.73
Bert-base	τ_K	0.809(9)	0.5(1)	0.800(9)	0.72(2)	0.65(2)	0.52(4)
	TAD (%)	+0.1(3)	+0.6(2)	+0.1(2)	+0.5(2)	-0.3(5)	-0.1(1)
	R_{Miss}	0.67(2)	0.71(5)	0.61(2)	0.86(3)	0.86(2)	0.91(2)

$$R_{\text{Miss}} = \frac{|\{i : \text{NN and kGLM make the same mistake on } \mathbf{z}_i\}|}{|\{i : \text{either NN or kGLM make a mistake on } \mathbf{z}_i\}|}$$

Attribution

The distribution of attribution scores from training data using the trNTK reflects the similarity of training points to the test image.



Some takeaways

Building an approximate model for a complex instrument

Some takeaways

Building an approximate model for a complex instrument

This is less about decisions and more about *similarities*.

Some takeaways

Building an approximate model for a complex instrument

This is less about decisions and more about *similarities*.

- If two models generate similar data attributions then the kGLMs are likely to be similar as well (or so we think).

Some takeaways

Building an approximate model for a complex instrument

This is less about decisions and more about *similarities*.

- If two models generate similar data attributions then the kGLMs are likely to be similar as well (or so we think).
- Provides another rejection-based rule (“if the attributions are different, the models are different”)

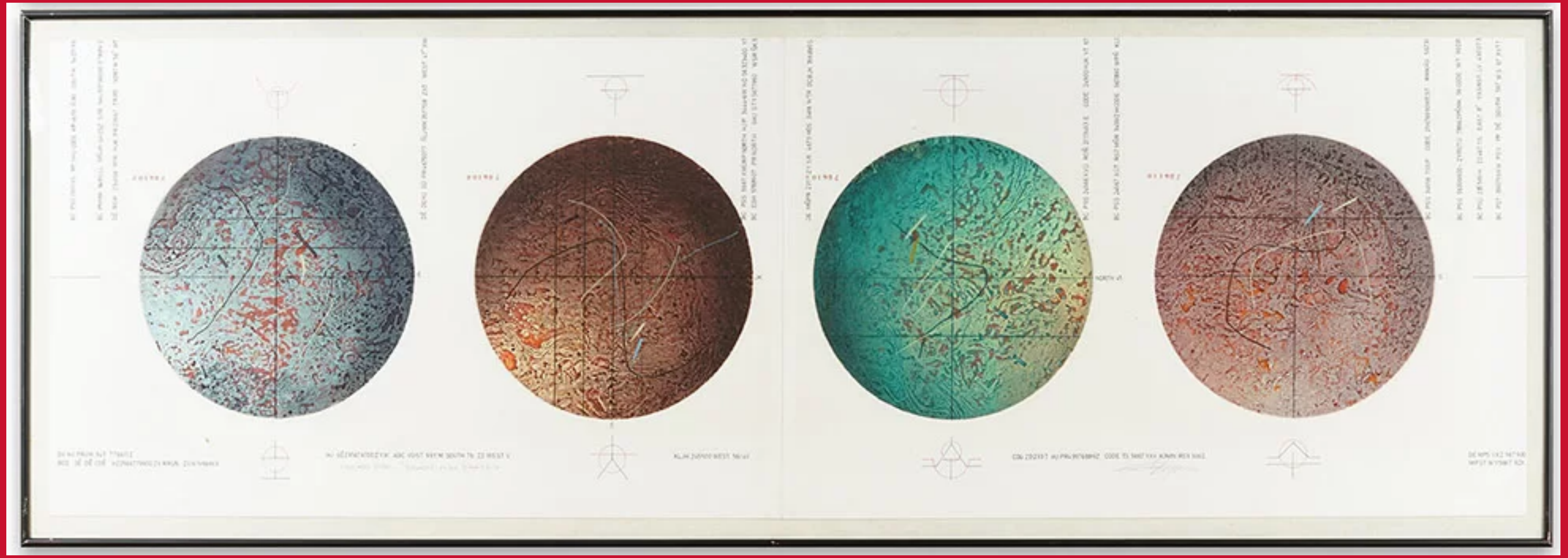
Some takeaways

Building an approximate model for a complex instrument

This is less about decisions and more about *similarities*.

- If two models generate similar data attributions then the kGLMs are likely to be similar as well (or so we think).
- Provides another rejection-based rule (“if the attributions are different, the models are different”)
- Similarities could also be used to detect if there are “poisoned” training data by surfacing similar training points to the test point.

Exploiting large models to distinguish other large models



Rm Palaniappan, *Alien Planet-D*
Viscosity, pencil colour and ink on handmade paper

Are similar looking models actually the same?

Working with pre-trained models

Ensign Tasha Yar, human

Sela, a Romulan, daughter of Tasha Yar

Are similar looking models actually the same?

Working with pre-trained models

Ensign Tasha Yar, human

Sela, a Romulan, daughter of Tasha Yar

Given two "off the shelf" instruments, can we tell if they operate in the same way?

Are similar looking models actually the same?

Working with pre-trained models

Ensign Tasha Yar, human

Sela, a Romulan, daughter of Tasha Yar

Given two “off the shelf” instruments, can we tell if they operate in the same way?

Can we use one large model to find differences between other large models?

Are similar looking models actually the same?

Working with pre-trained models

Ensign Tasha Yar, human

Sela, a Romulan, daughter of Tasha Yar

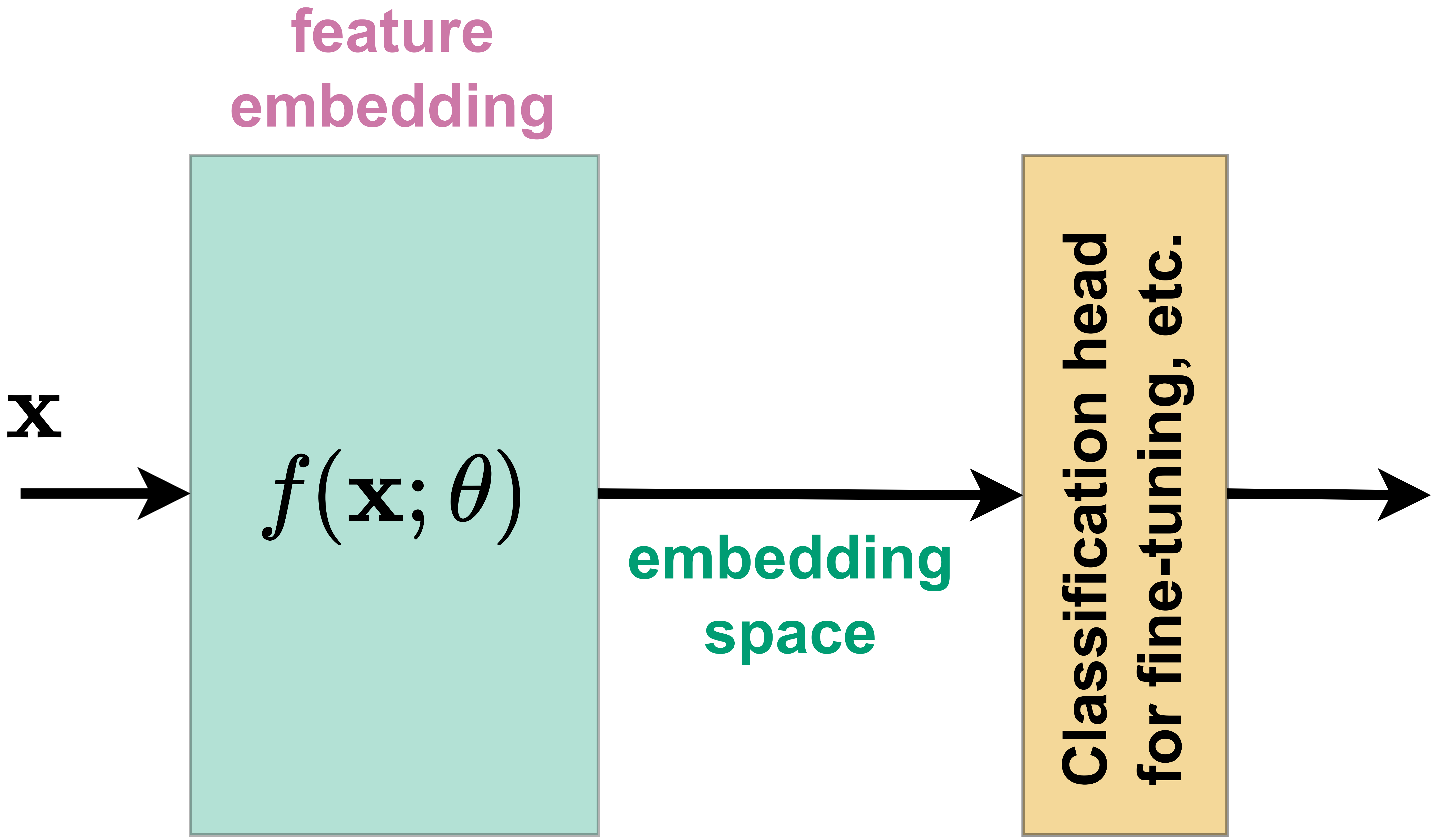
Given two “off the shelf” instruments, can we tell if they operate in the same way?

Can we use one large model to find differences between other large models?

Does every (sufficiently complex) ML model have a uniquely detectable “signature” or “model DNA?”

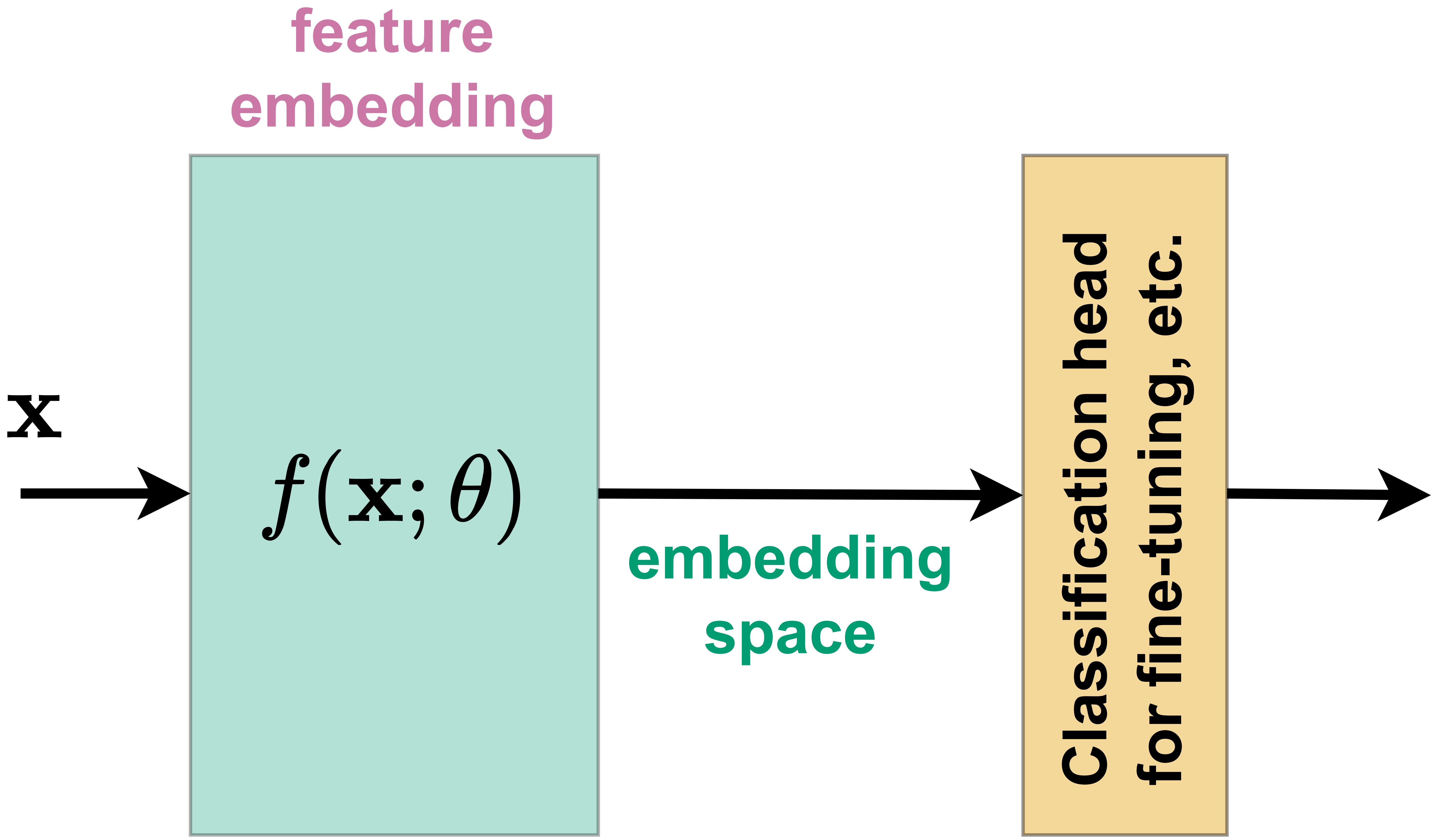
Thinking about the embedding space

“Foundation models” are just very complex feature extractors



Thinking about the embedding space

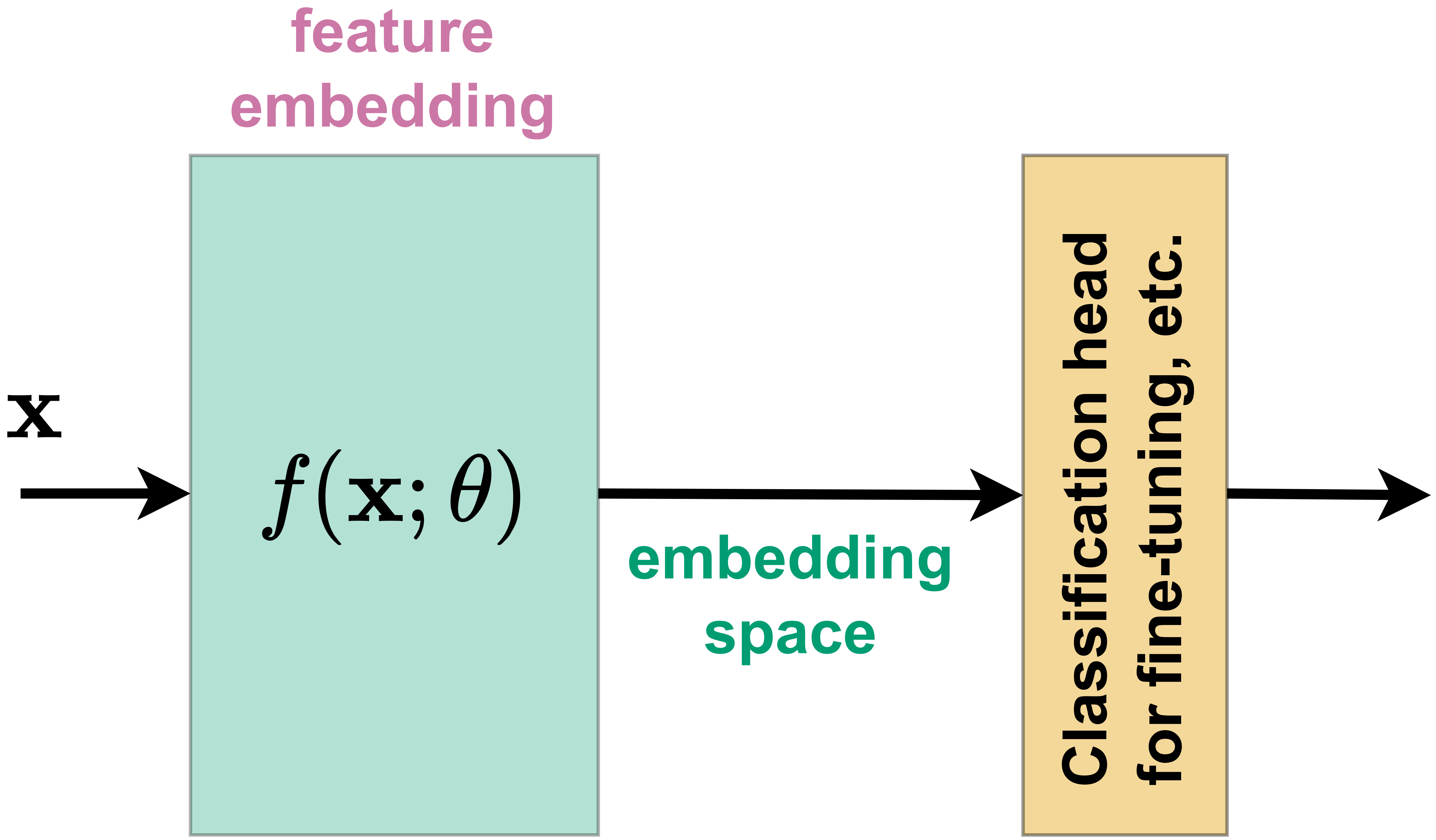
“Foundation models” are just very complex feature extractors



Think of large models as having a “feature embedding” stage followed by some classification procedure on the embedded features.

Thinking about the embedding space

"Foundation models" are just very complex feature extractors

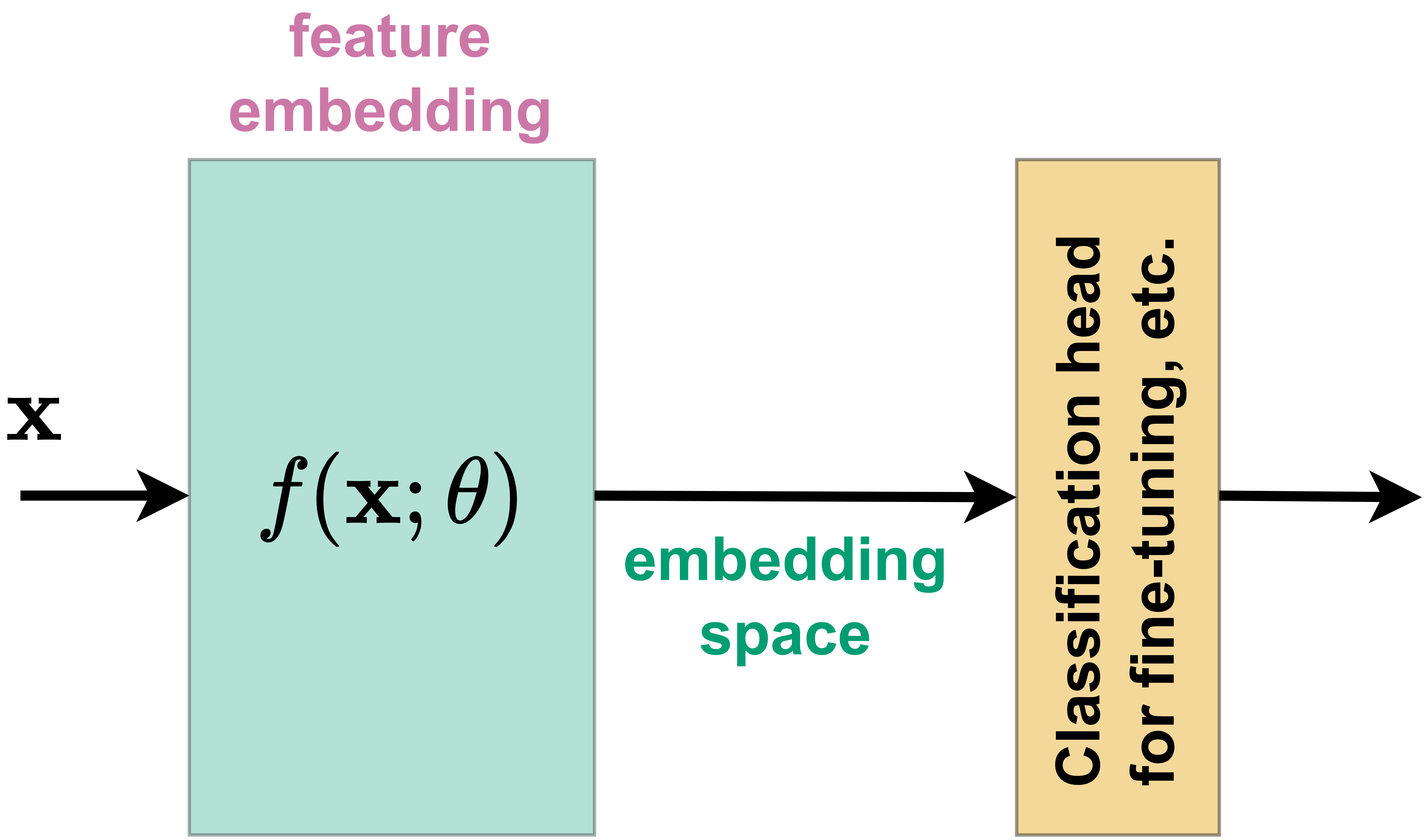


Think of large models as having a "feature embedding" stage followed by some classification procedure on the embedded features.

- Fine-tuning works because these embeddings carry a lot of information.

Thinking about the embedding space

"Foundation models" are just very complex feature extractors



Think of large models as having a "feature embedding" stage followed by some classification procedure on the embedded features.

- Fine-tuning works because these embeddings carry a lot of information.
- How well can these embedding spaces separate things?

Using a large model as an instrument

It takes one to know one

Using a large model as an instrument

It takes one to know one

We can use a large model to embed data from different sources and then see if the sources are distinguishable based on the embeddings. Three models we used as instruments in this way:

Using a large model as an instrument

It takes one to know one

We can use a large model to embed data from different sources and then see if the sources are distinguishable based on the embeddings. Three models we used as instruments in this way:

- **`Mistral-7B`**: LLM, transformer-based, 32 layers, 13b parameters per token and 32 token vocabulary. Embeddings from the final hidden layer of dimension 4,096.

Using a large model as an instrument

It takes one to know one

We can use a large model to embed data from different sources and then see if the sources are distinguishable based on the embeddings. Three models we used as instruments in this way:

- **`Mistral-7B`**: LLM, transformer-based, 32 layers, 13b parameters per token and 32 token vocabulary. Embeddings from the final hidden layer of dimension 4,096.

- **`Multilingual-e5-large`**: extracts sentence embeddings from text in different languages to 1024-dimensional embedding vectors. 60M parameters, context window of 512 tokens and long text is truncated to fit within this window.

Using a large model as an instrument

It takes one to know one

We can use a large model to embed data from different sources and then see if the sources are distinguishable based on the embeddings. Three models we used as instruments in this way:

- **`Mistral-7B`**: LLM, transformer-based, 32 layers, 13b parameters per token and 32 token vocabulary. Embeddings from the final hidden layer of dimension 4,096.
- **`Multilingual-e5-large`**: extracts sentence embeddings from text in different languages to 1024-dimensional embedding vectors. 60M parameters, context window of 512 tokens and long text is truncated to fit within this window.
- **`Data Filtering Network`**: a CLIP model trained on 5B images that were filtered from an uncurated dataset of image-text pairs. It has 1B parameters and can be used to encode both text and images.

Experimental setups

How to use a large model as an instrument

Experimental setups

How to use a large model as an instrument

Different types of experiments to run:

Experimental setups

How to use a large model as an instrument

Different types of experiments to run:

1. Embed real data and AI-generated data to see if the embedding vectors cluster.

Experimental setups

How to use a large model as an instrument

Different types of experiments to run:

1. Embed real data and AI-generated data to see if the embedding vectors cluster.
2. Unsupervised clustering of embedded data recreates the labels in the original.

Experimental setups

How to use a large model as an instrument

Different types of experiments to run:

1. Embed real data and AI-generated data to see if the embedding vectors cluster.
2. Unsupervised clustering of embedded data recreates the labels in the original.
3. Detect the difference between real and machine-translated data

Experimental setups

How to use a large model as an instrument

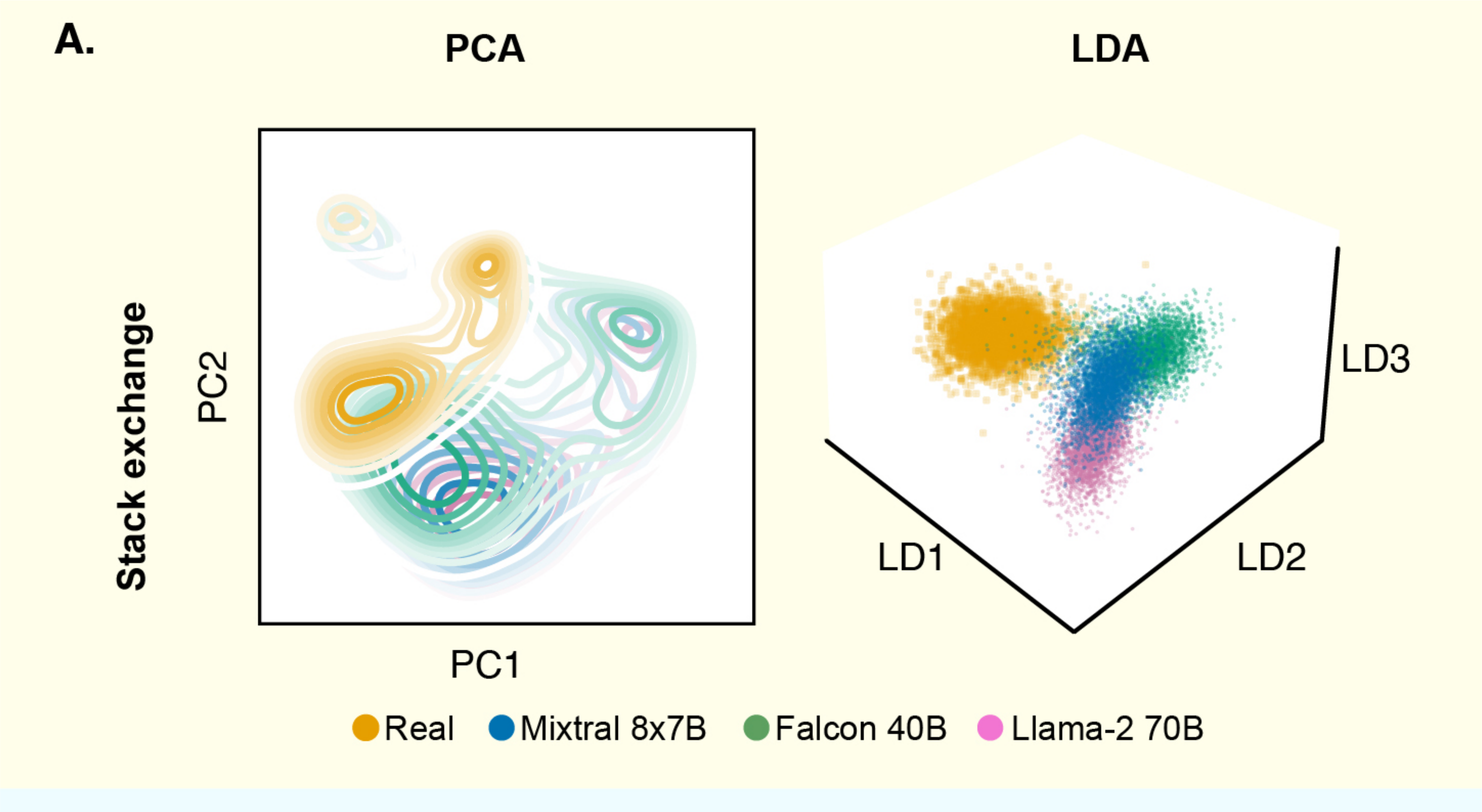
Different types of experiments to run:

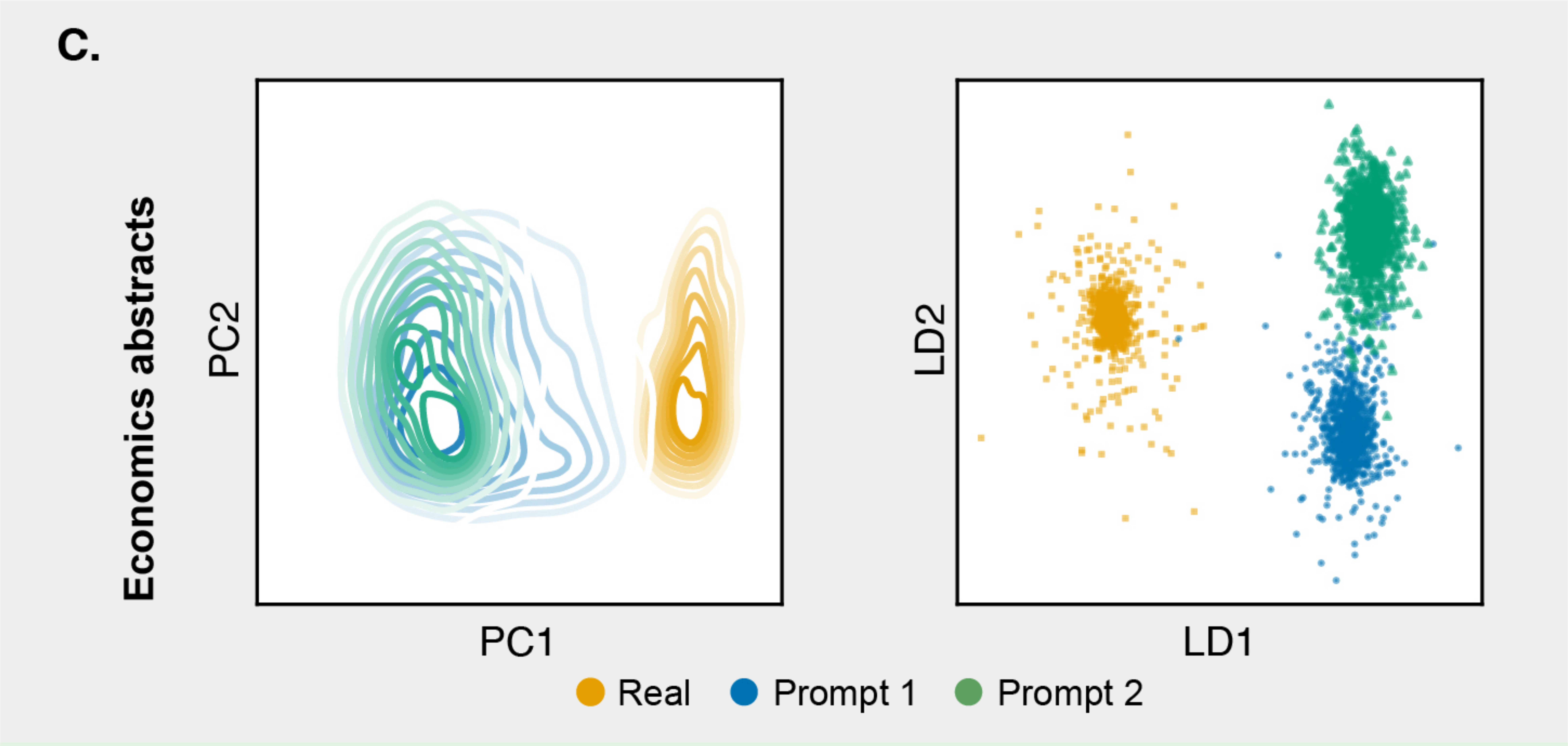
1. Embed real data and AI-generated data to see if the embedding vectors cluster.

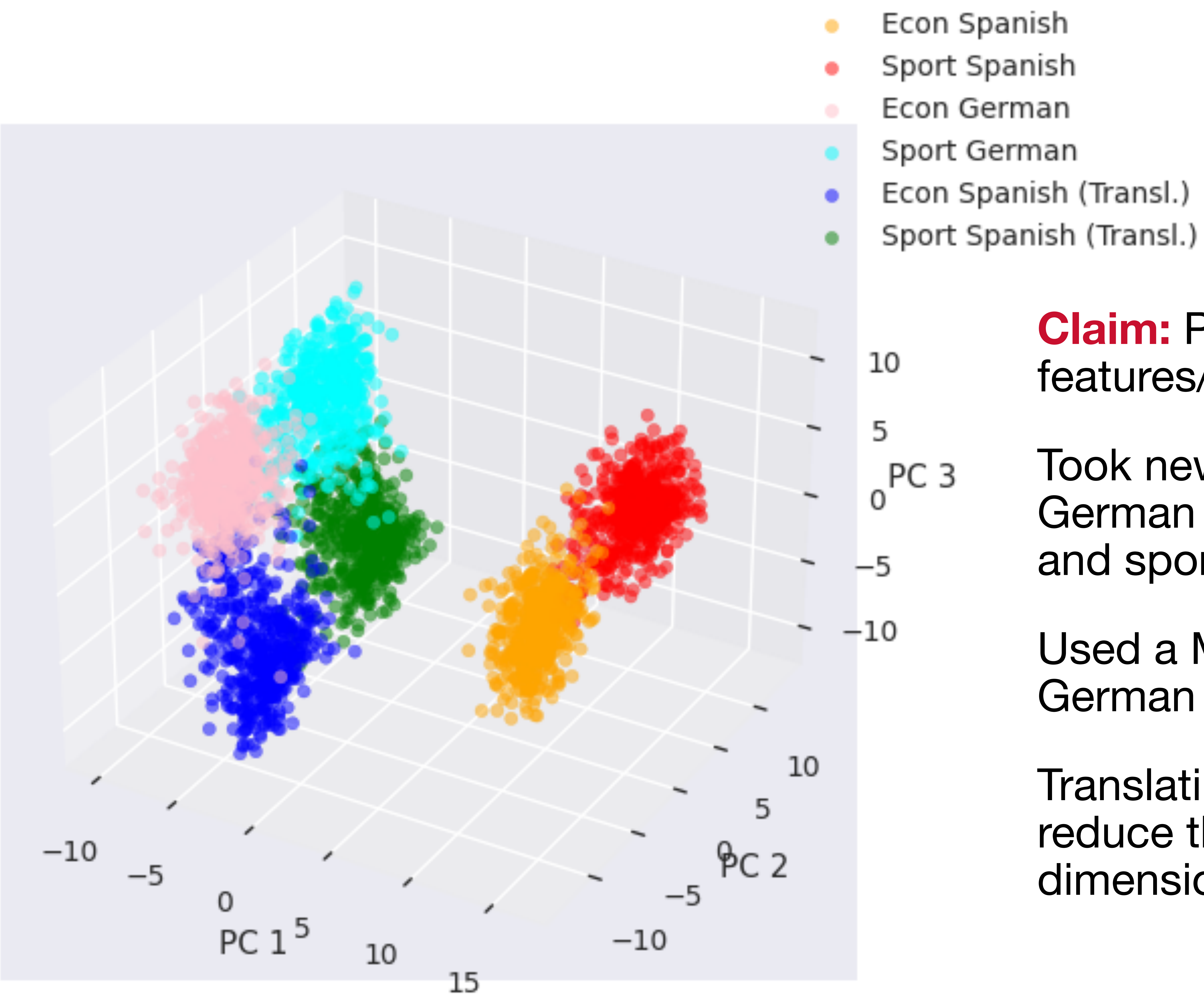
2. Unsupervised clustering of embedded data recreates the labels in the original.

3. Detect the difference between real and machine-translated data

In all cases we use simple tools: PCA, LDA to look at the collection of embedding vectors.


A.
PCA
LDA
Stack exchange
PC2
PC1
LD1
LD2
LD3
Real
Mixtral 8x7B
Falcon 40B
Llama-2 70B


C.
Economics abstracts
PC2
PC1
LD2
LD1
Real
Prompt 1
Prompt 2



Claim: PCs reflect interpretable features/known hidden labels.

Took news articles in Spanish and German in two topics, economics and sports.

Used a ML translator to translate German to Spanish.

Translating news articles helps reduce the variation in one dimension (language).

Implications for instrumentation

This is still a work in progress

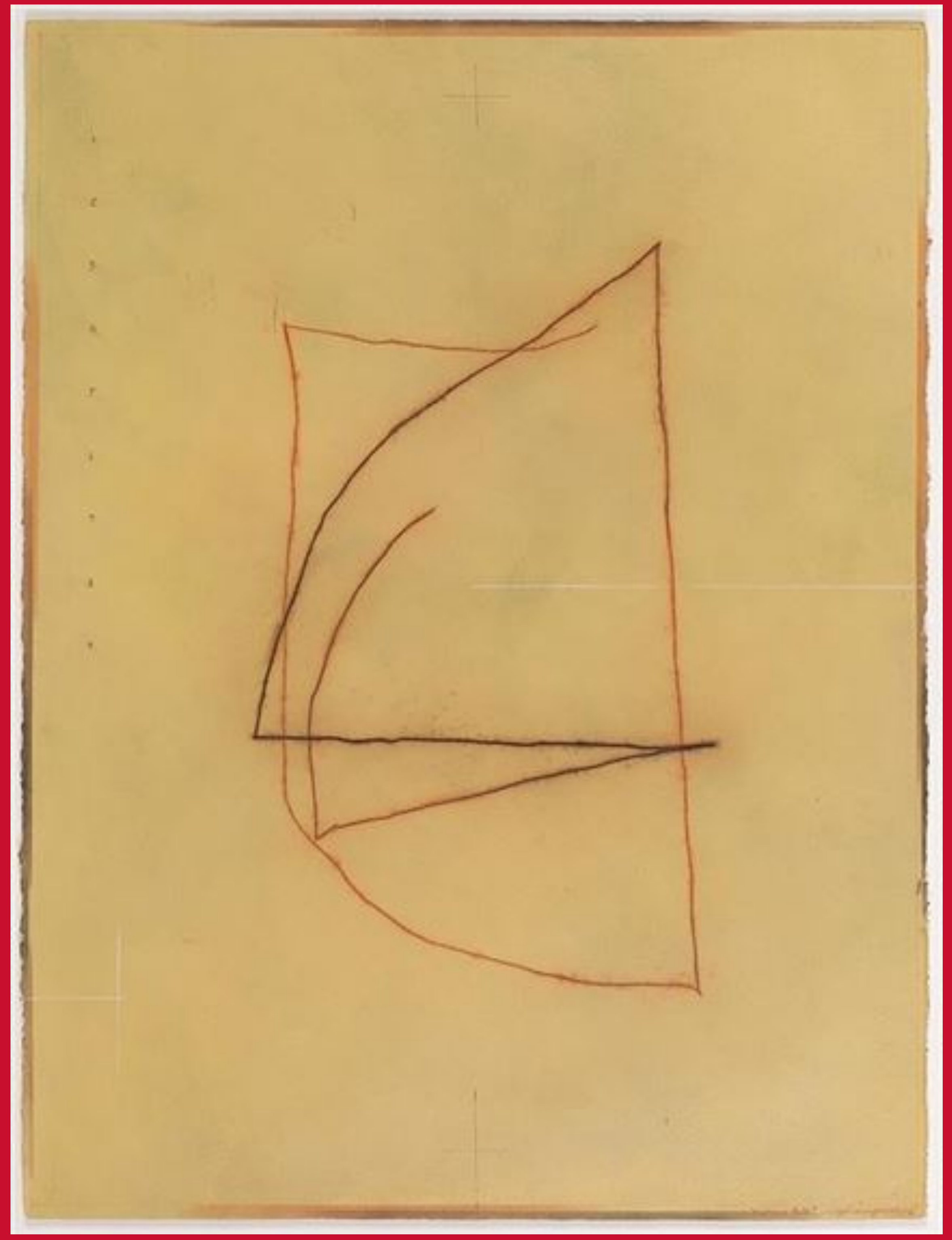
Implications for instrumentation

This is still a work in progress

The embedding spaces of large “foundation models” can also easily distinguish between different sources of data.

- Huge potential in forensics.
- Synthetic data is easily separable using basic techniques.
- Lots of open questions and directions to pursue!

Some final remarks



Rm Palaniappan, *Intense Talk*
Mixed media on paper pasted on mount board

Quick recap

The philosophy and some observations

Quick recap

The philosophy and some observations

- If we want AI systems to act like scientific instruments, they have to be easy to generate reliably, easier to compare/constrast, easier to interpret, and interchangeable.

Quick recap

The philosophy and some observations

- If we want AI systems to act like scientific instruments, they have to be easy to generate reliably, easier to compare/constrast, easier to interpret, and interchangeable.
- A fundamental open question still is how to compare models: what makes two models meaningfully different from each other?

Quick recap

The philosophy and some observations

- If we want AI systems to act like scientific instruments, they have to be easy to generate reliably, easier to compare/constrast, easier to interpret, and interchangeable.
- A fundamental open question still is how to compare models: what makes two models meaningfully different from each other?
- I discussed some fairly standard tools (well-worn?) that give some insight.

Quick recap

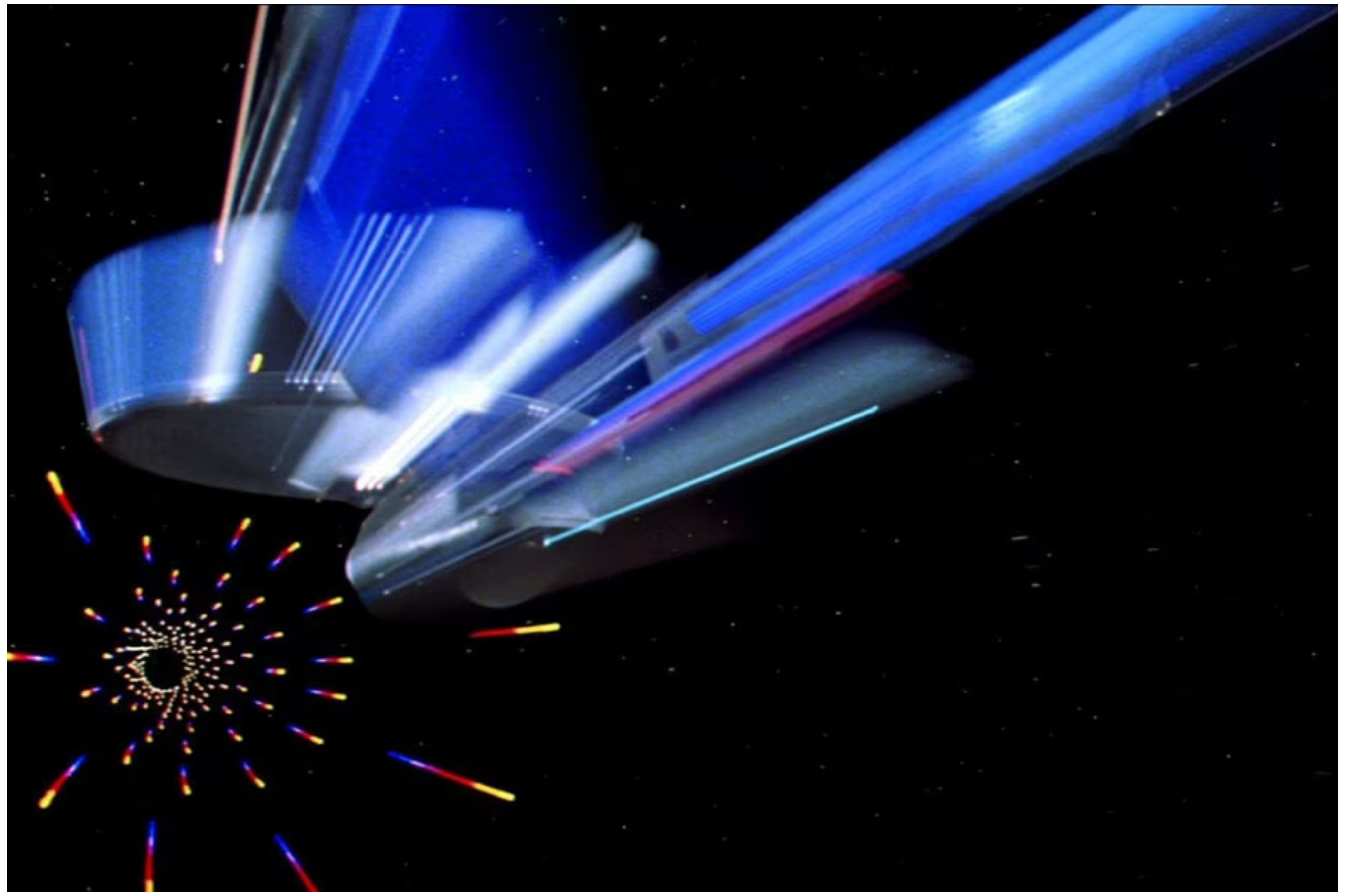
The philosophy and some observations

- If we want AI systems to act like scientific instruments, they have to be easy to generate reliably, easier to compare/constrast, easier to interpret, and interchangeable.
- A fundamental open question still is how to compare models: what makes two models meaningfully different from each other?
- I discussed some fairly standard tools (well-worn?) that give some insight.
- Do we need fancier tools? Probably!

Looking forward

Many strange new worlds left to see

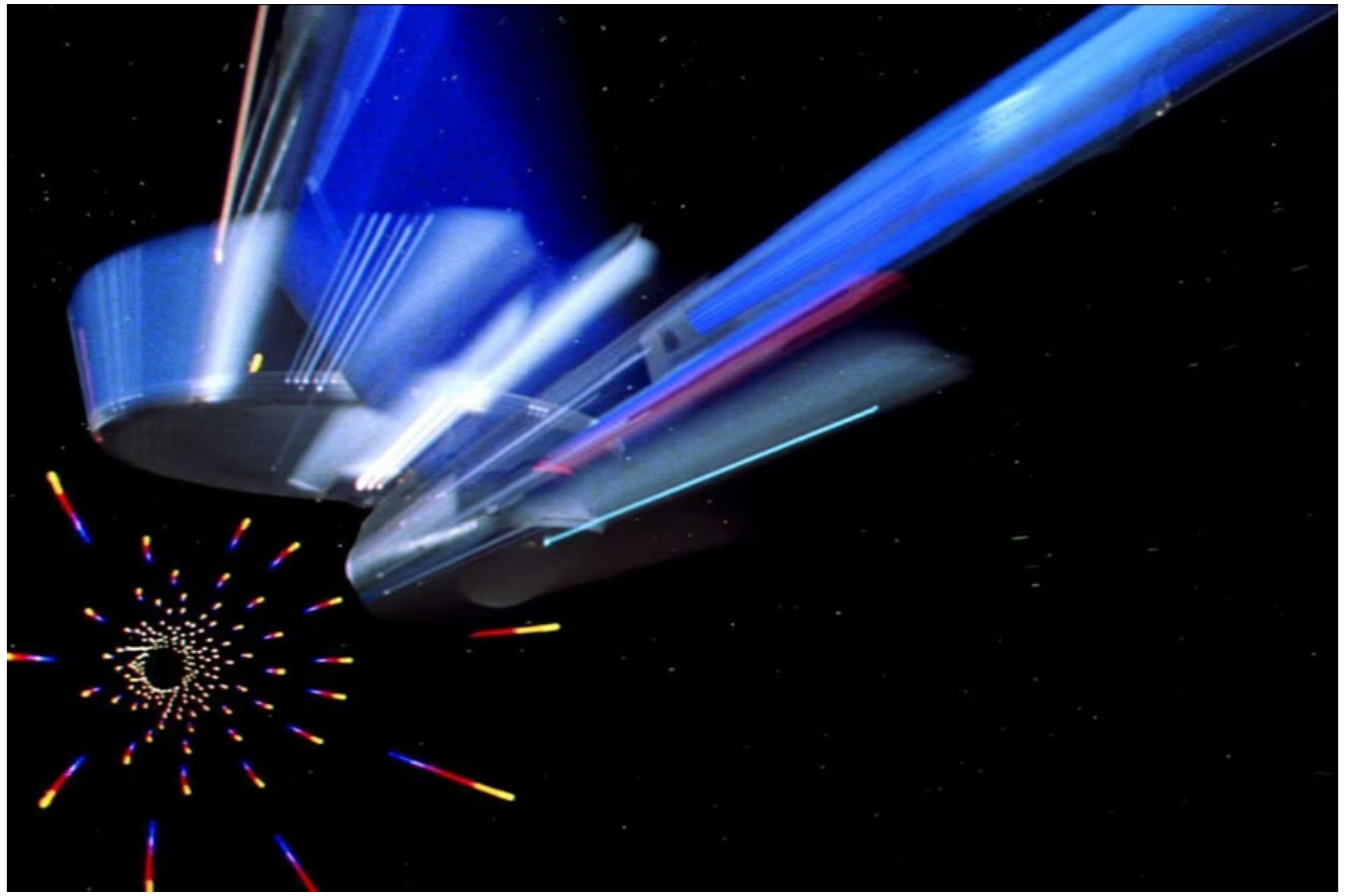
This was mostly a talk about practice with some "theory" sprinkled in here and there. **We need more theory!**



Looking forward

Many strange new worlds left to see

This was mostly a talk about practice with some “theory” sprinkled in here and there. **We need more theory!**

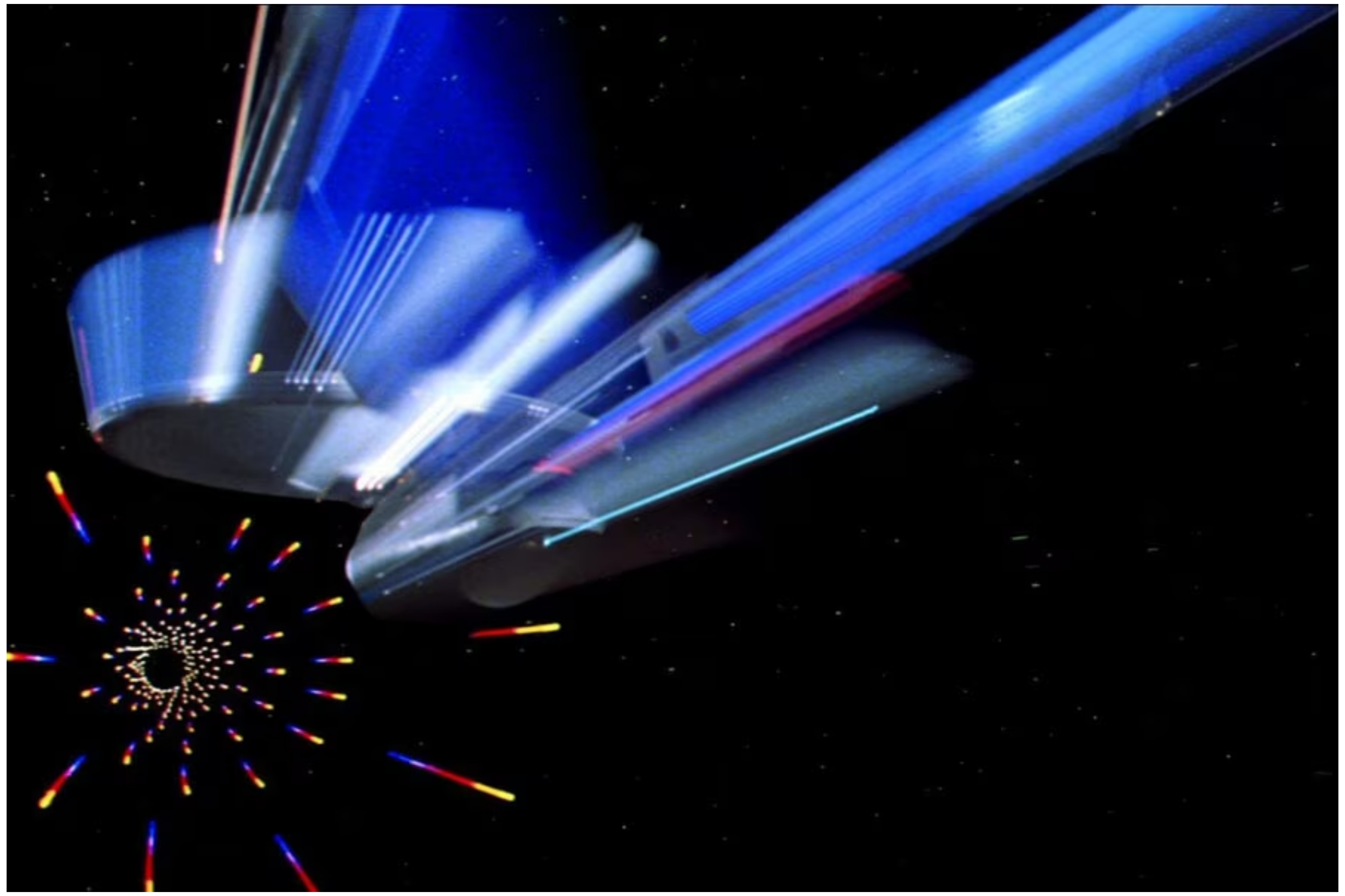


- There are tons of questions we can ask and answer using tools we have as long as we can look from outside the box.

Looking forward

Many strange new worlds left to see

This was mostly a talk about practice with some "theory" sprinkled in here and there. **We need more theory!**

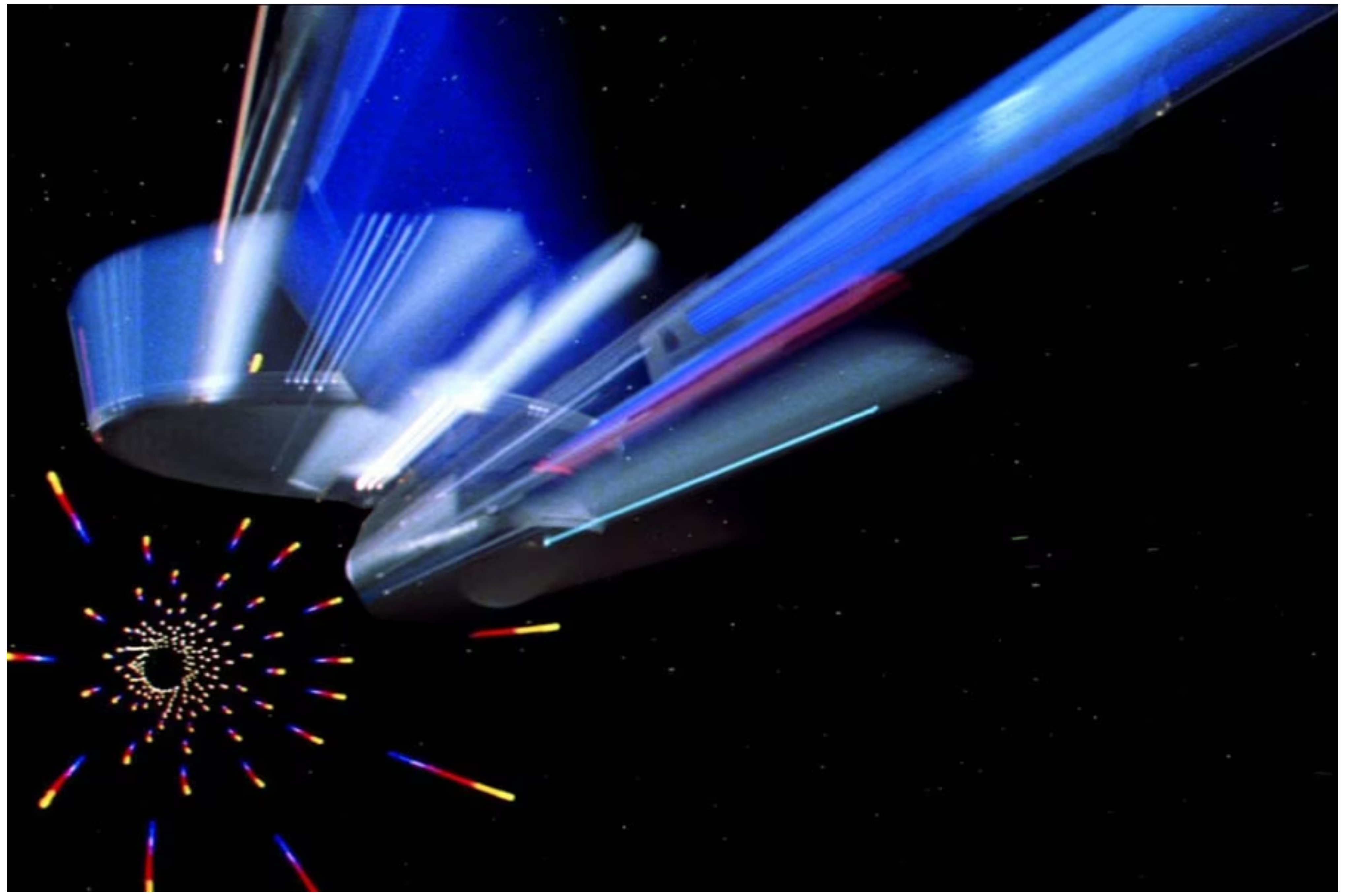


- There are tons of questions we can ask and answer using tools we have as long as we can look from outside the box.
- Engineering has to happen within and around systems, so there is room for both perspectives.

Looking forward

Many strange new worlds left to see

This was mostly a talk about practice with some “theory” sprinkled in here and there. **We need more theory!**



- There are tons of questions we can ask and answer using tools we have as long as we can look from outside the box.
- Engineering has to happen within and around systems, so there is room for both perspectives.
- Simple tools can only go so far… but what kind of tools would we want or need?

மிக்க நன்றி!

Ramanathan Palaniappan
The Truth of Existence:
The Long Run… That Stretches Across

Mixed media and acrylic on canvas