

# Today: The bleeding edge of research

## Meta-learning for adapting to distribution shift

Adapting with unlabeled example(s)

Making local “edits” to large neural networks

## Meta-learning across more general task distributions

Can we meta-learn an optimizer for any problem?

Can we meta-learn the architectural symmetries?

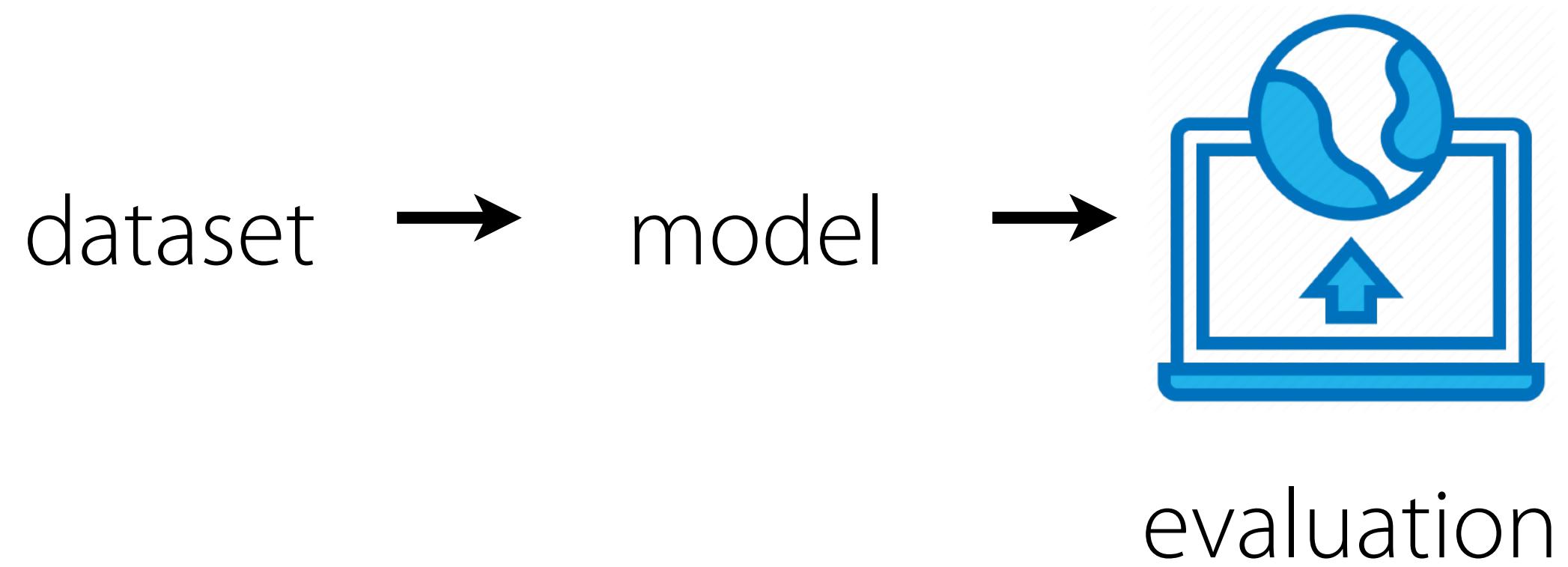
## Open Challenges

Why address distribution shift?

# Our current paradigm

v

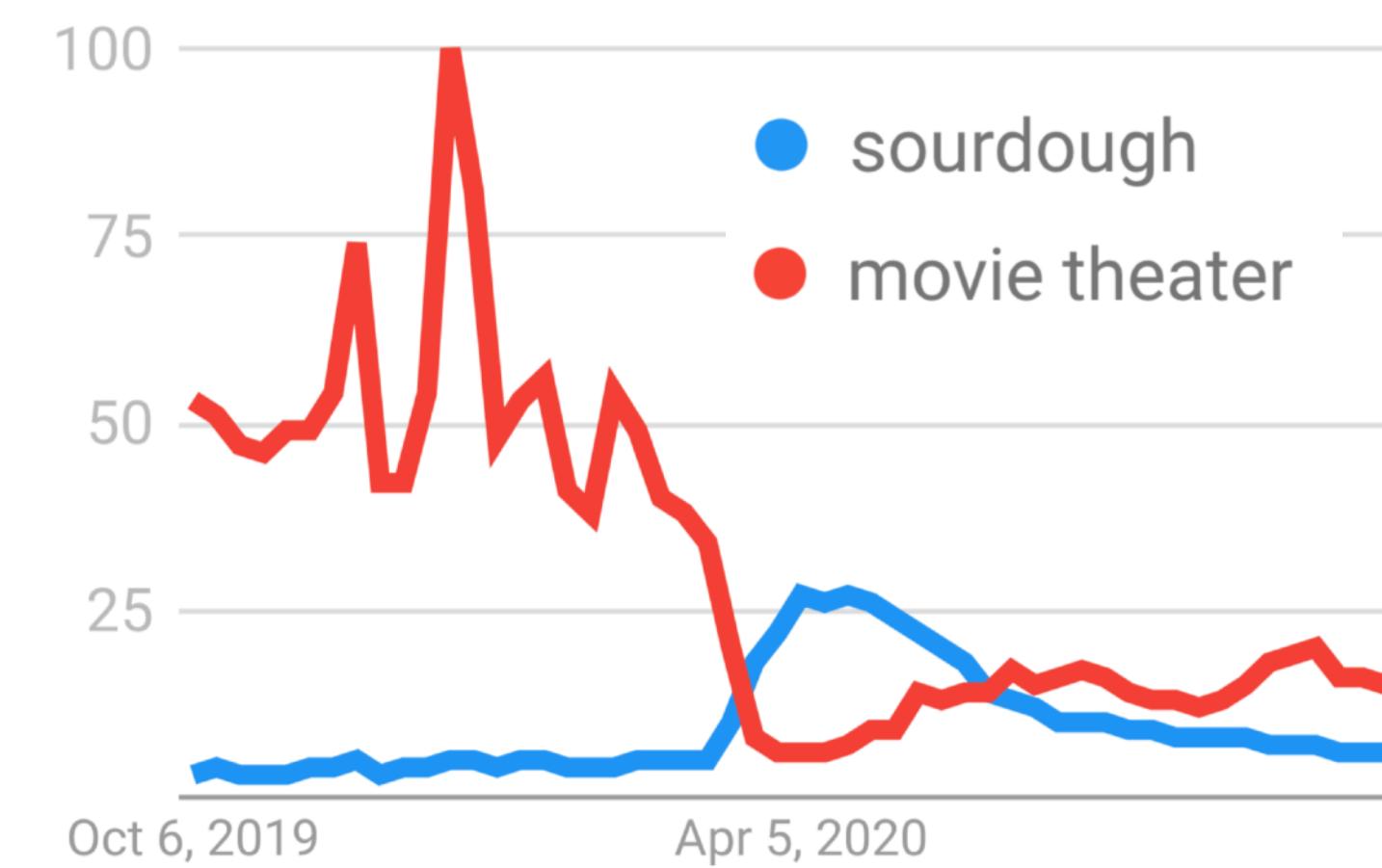
(ML research)



# Our current reality

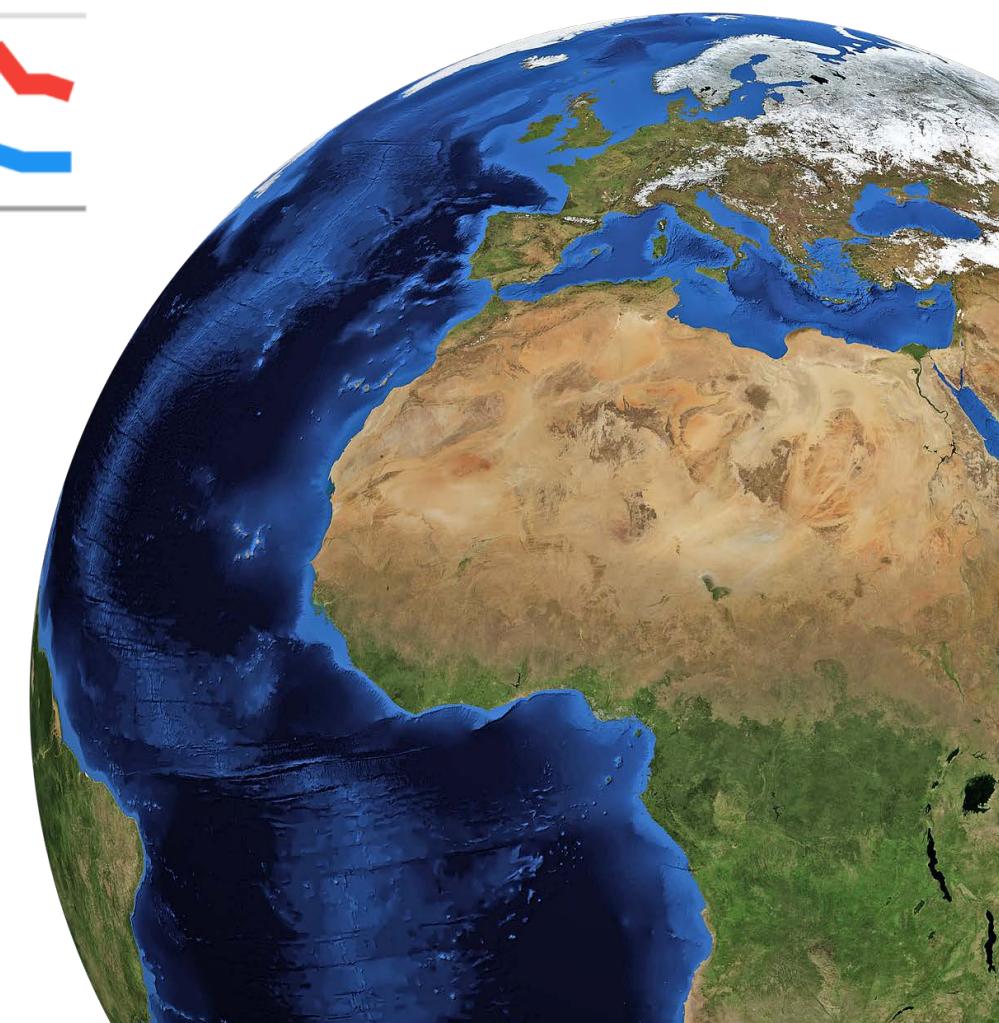
stocks

Google Trends



robots

supply & demand



Can our algorithms handle the **changing** world?

# How does industry cope?

Chip Huyen on misperceptions about ML production:



Chip Huyen  
@chipro

Replies to @chipro

3. If nothing happens, model performance remains the same

ML models perform best right after training. In prod,  
ML systems degrade quickly bc of concept drift.

Tip: train models on data generated 6 months ago &  
test on current data to see how much worse they get.

(4/6)

7:39 AM · Sep 29, 2020 · Twitter Web App



Chip Huyen  
@chipro

Replies to @chipro

4. You won't need to update your models as much

One mindboggling fact about DevOps: Etsy deploys  
50 times/day. Netflix 1000s times/day. AWS every 11.7  
seconds.

MLOps isn't an exemption. For online ML systems, you  
want to update them as fast as humanly possible.

(5/6)

7:40 AM · Sep 29, 2020 · Twitter Web App

the way ML techniques are being *used* != the way they were *intended*

Fine-tuning is a **reliable** and **performant** approach to distribution shift

**But:** - requires **labeled data**

- can be **computationally expensive** for large models
- generally a **blunt tool** that can't make precise edits/patches

# What kind of distribution shift?

We'll first focus on: *domain shift*

**categorical domain variable  $d$**

e.g. user, location, time of day  
(can be derived from meta-data)

Training data from  $p(x, y | d)p_{\text{tr}}(d)$   
Test data from  $p(x, y | d)p_{\text{ts}}(d)$

**Group DRO** (*distributionally robust optimization*):

(Ben-Tal et al. '13, Duchi et al '16)

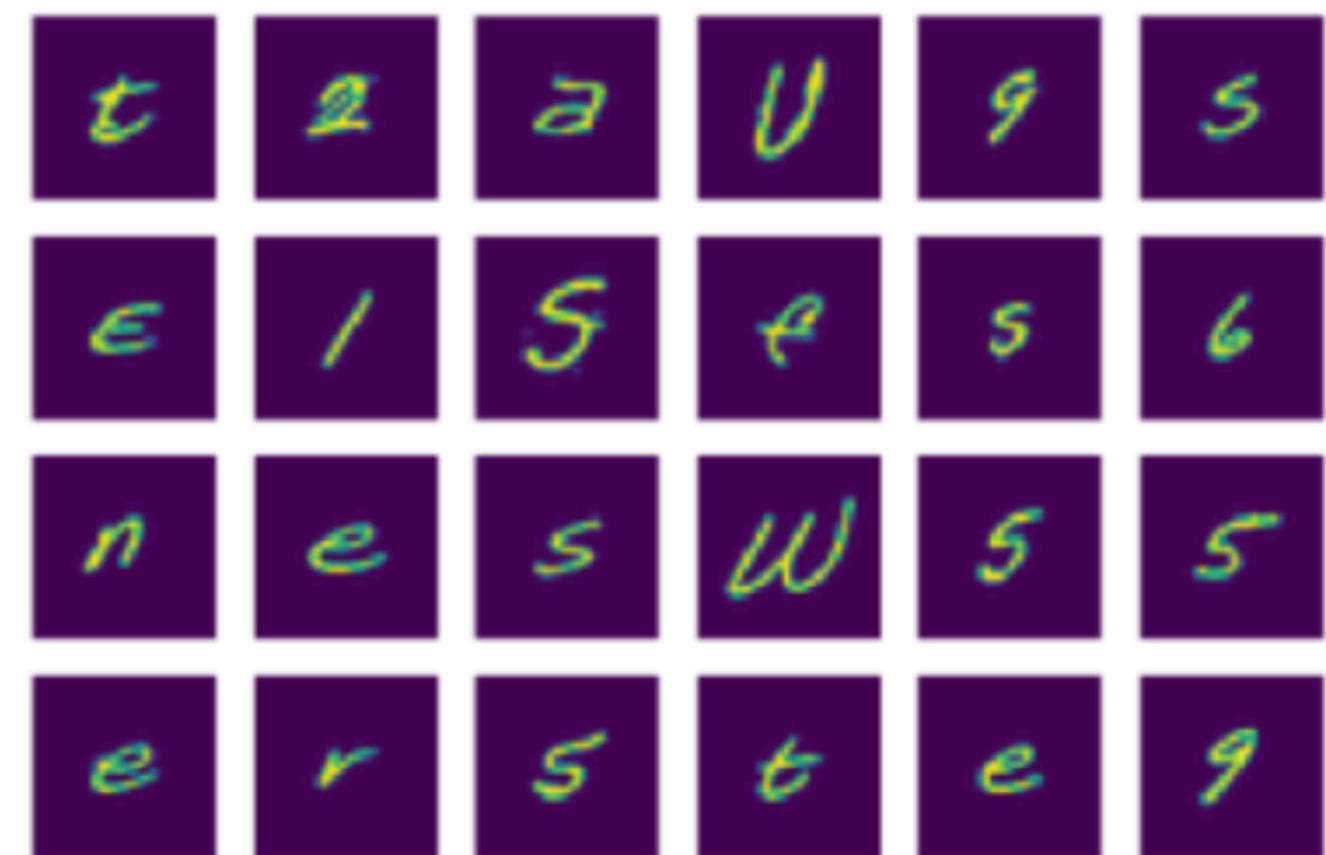
Form adversarial distribution  $q(d)$ :  $\min_{\theta} \sup_{q \in \mathcal{Q}} \mathbb{E}_{q_d} [\mathbb{E}_{p_{xy|d}} [\ell(g(x; \theta), y)]]$

+ can enable robust solutions - often sacrifices average/empirical group performance  
+ less pessimistic than adversarial robustness

Can we aim to *adapt* instead of aiming for robustness?

## Test time

unlabeled data from test sub-distribution  
(e.g. new user, different time-of-day, new place)



adapt model & infer labels



**Assumption:** test inputs from one group available in a batch or streaming.

## Adaptive risk minimization (ARM)

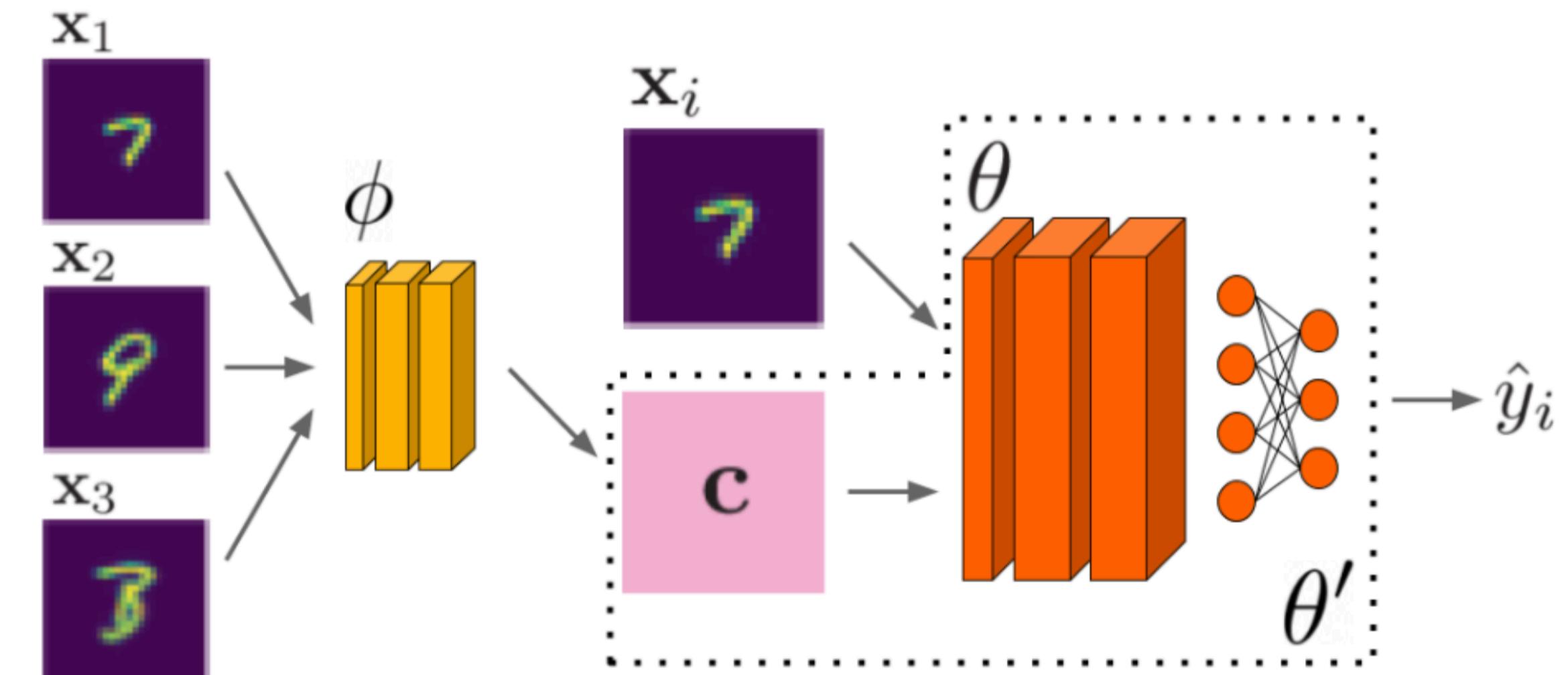
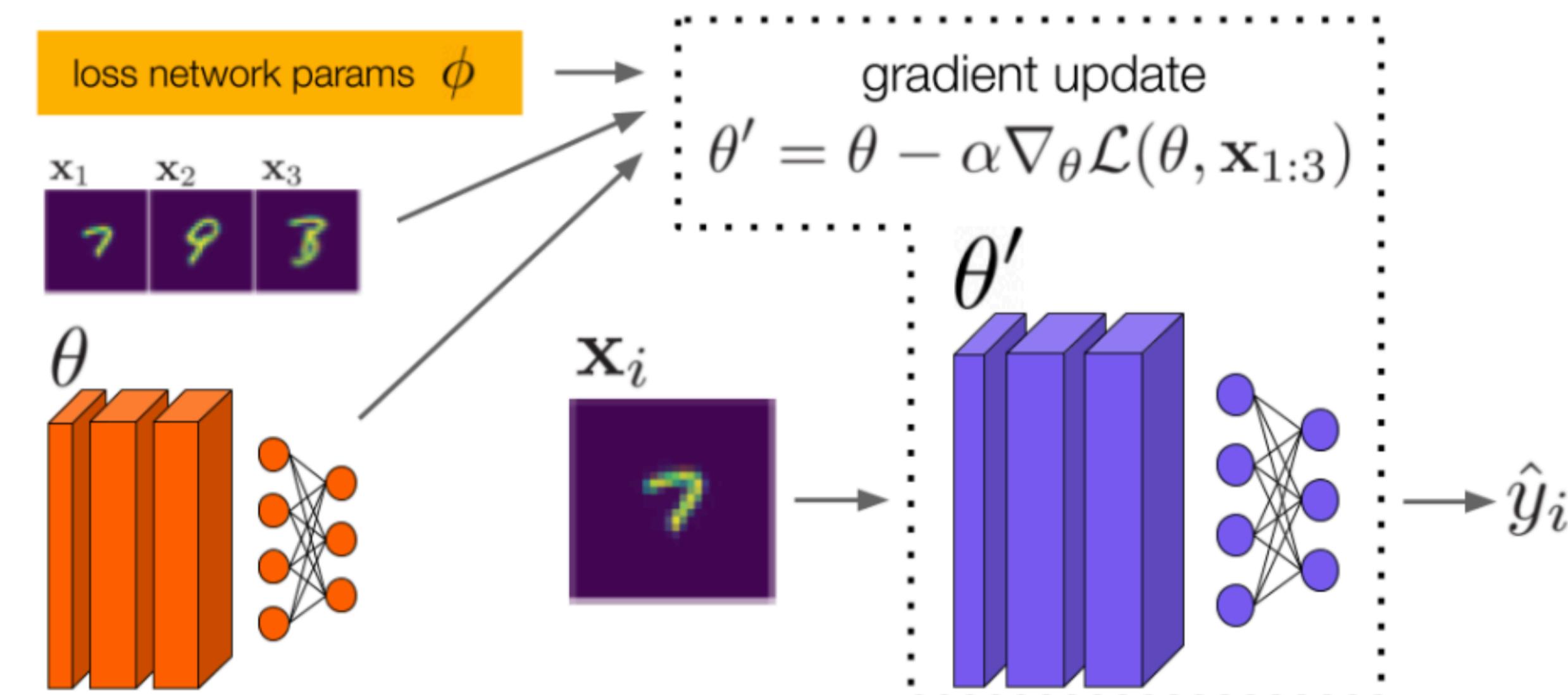
# Adaptive risk minimization (ARM)

Train for few-shot adaptation to different domains in training data.

How to adapt with unlabeled data?

MAML with learned loss

or meta-learning with context variable



Simplest setting: context = BN statistics

# Experiment 1. Federated Extended MNIST (Cohen et al. 2017, Caldas et al. 2019)

**Distribution shift:** adapt to new users with only unlabeled data

FEMNIST		
Method	WC	Avg
ERM	$62.4 \pm 0.4$	$79.1 \pm 0.3$
UW*	<b><math>65.7 \pm 0.7</math></b>	$80.3 \pm 0.6$
DRNN	$57.5 \pm 1.7$	$76.5 \pm 1.2$
DANN	<b><math>65.4 \pm 1.0</math></b>	<b><math>81.7 \pm 0.3</math></b>
ARM-CML	<b><math>70.9 \pm 1.4</math></b>	<b><math>86.4 \pm 0.3</math></b>
ARM-BN	$64.5 \pm 3.2$	$83.2 \pm 0.5$
ARM-LL	$67.0 \pm 0.9$	$84.3 \pm 0.7$

+ 5% improvement in average accuracy  
+ 5% improvement in worst-case accuracy

**ARM** - adaptive risk minimization

**DRNN** - distributional robustness  
(Sagawa, Koh et al. ICLR '20)

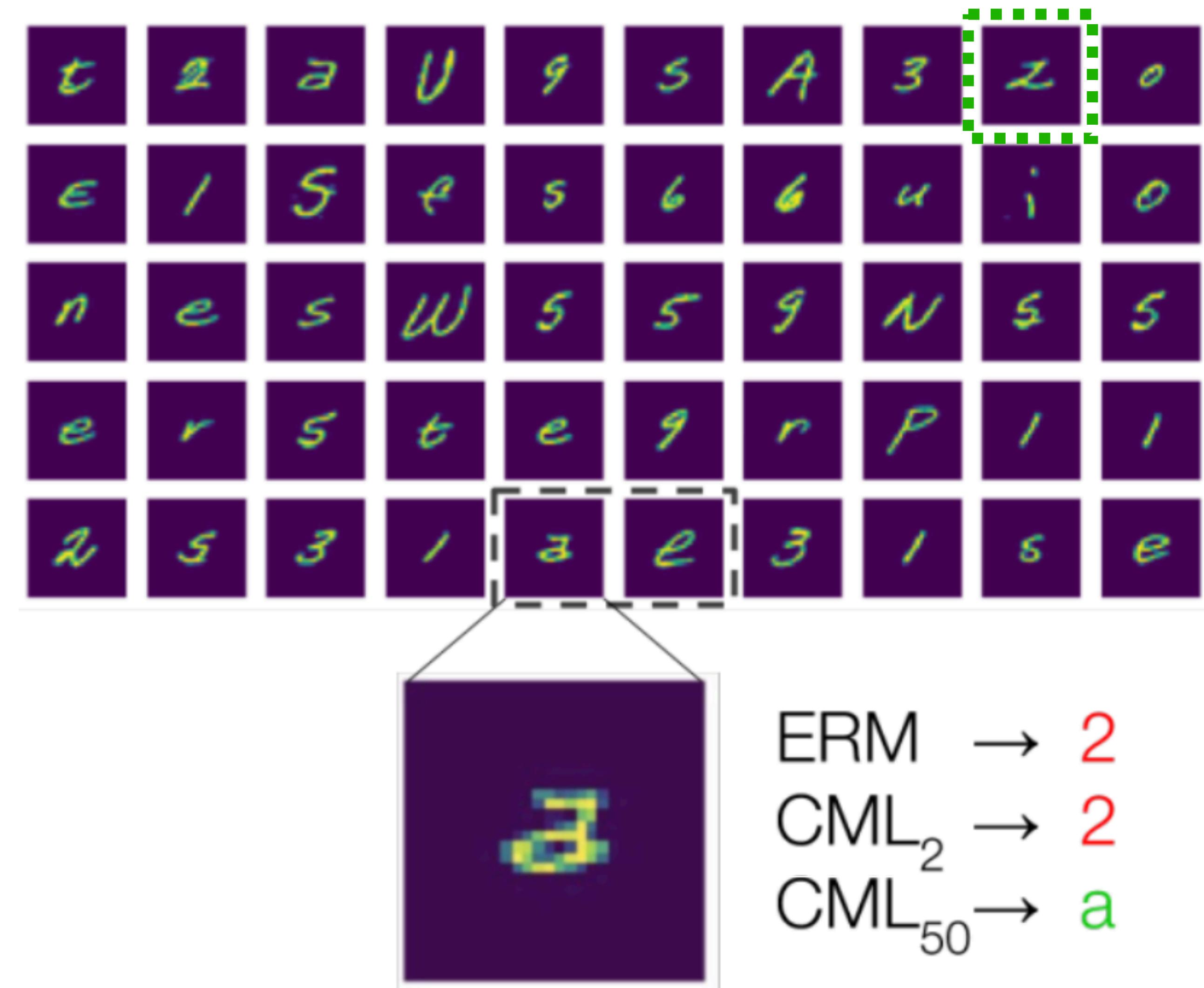
**DANN** (Ganin et al. 2016) - domain adversarial training

**ERM** - standard deep network training

**UW** - ERM but upweight groups to the uniform distribution

# Experiment 1. Federated Extended MNIST (Cohen et al. 2017, Caldas et al. 2019)

**Distribution shift:** adapt to new users with only unlabeled data



## Experiment 2. CIFAR-C, TinyImageNet-C (Hendrycks & Dietterich, 2019)

**Distribution shift:** adapt to new image corruptions

(train using 56 corruptions, test using 22 disjoint corruptions)

<b>Method</b>	<b>CIFAR-10-C</b>		<b>Tiny ImageNet-C</b>	
	<b>WC</b>	<b>Avg</b>	<b>WC</b>	<b>Avg</b>
ERM	$54.1 \pm 0.3$	$70.4 \pm 0.1$	$20.3 \pm 0.5$	$41.9 \pm 0.1$
UW*	—	—	—	—
DRNN	$49.3 \pm 0.9$	$65.7 \pm 0.5$	$14.2 \pm 0.2$	$31.6 \pm 1.0$
DANN	$53.9 \pm 2.2$	$69.8 \pm 0.3$	$20.4 \pm 0.7$	$40.9 \pm 0.2$
ARM-CML	$61.2 \pm 0.4$	$70.3 \pm 0.2$	$29.1 \pm 0.4$	$43.3 \pm 0.1$
ARM-BN	$61.7 \pm 0.3$	$72.4 \pm 0.3$	$28.3 \pm 0.3$	$43.3 \pm 0.1$
ARM-LL	$61.2 \pm 0.7$	$72.5 \pm 0.4$	$25.4 \pm 0.1$	$35.7 \pm 0.4$

+ 2-3% improvement in average accuracy

+ 7-9% improvement in worst-case accuracy

**ARM** - adaptive risk minimization

**DRNN** - distributional robustness  
(Sagawa, Koh et al. ICLR '20)

**DANN** (Ganin et al. 2016) - domain adversarial training

**ERM** - standard deep network training

**UW** - ERM but upweight groups to the uniform distribution

# Today: The bleeding edge of research

## Meta-learning for adapting to distribution shift

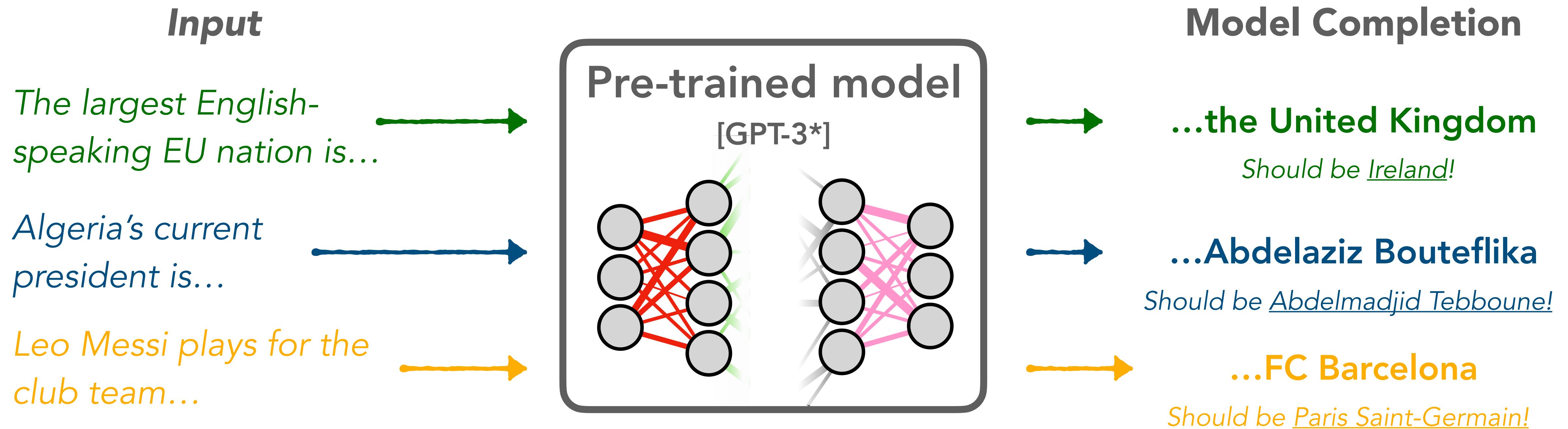
Adapting with unlabeled example(s)

Making local “edits” to large neural networks

**Takeaway:** Meta-learning to quickly adapt to new domains

Can we adapt to other forms of distribution shift?

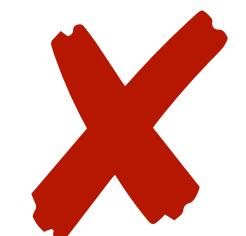
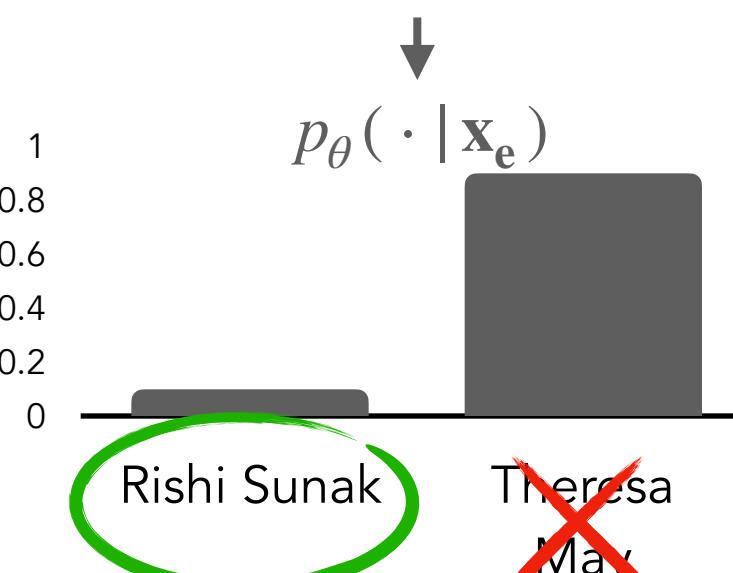
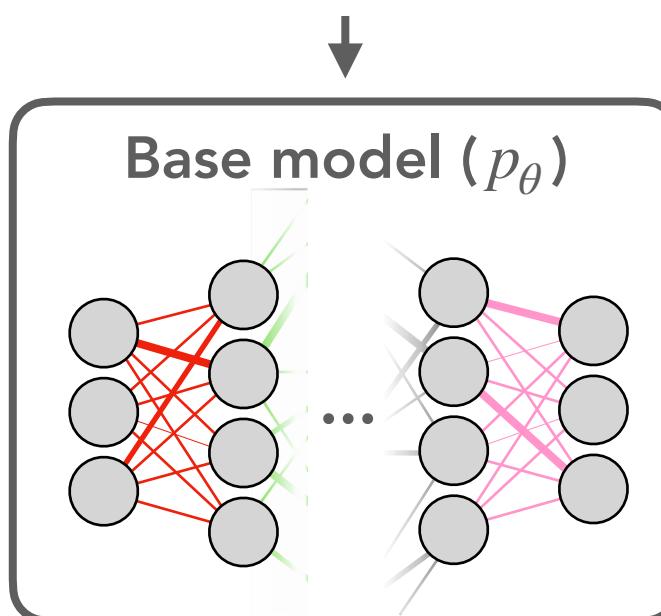
# Robustness of Language Models



How can we **efficiently** keep large models **up-to-date** with an ever-changing world?

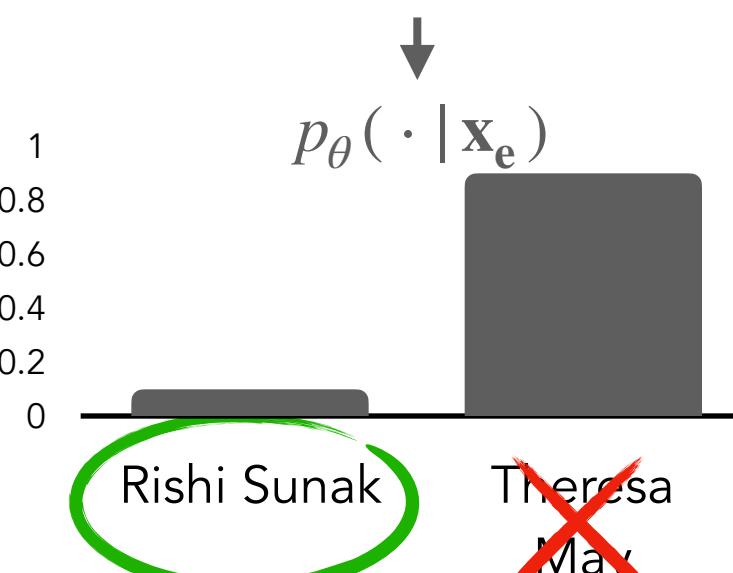
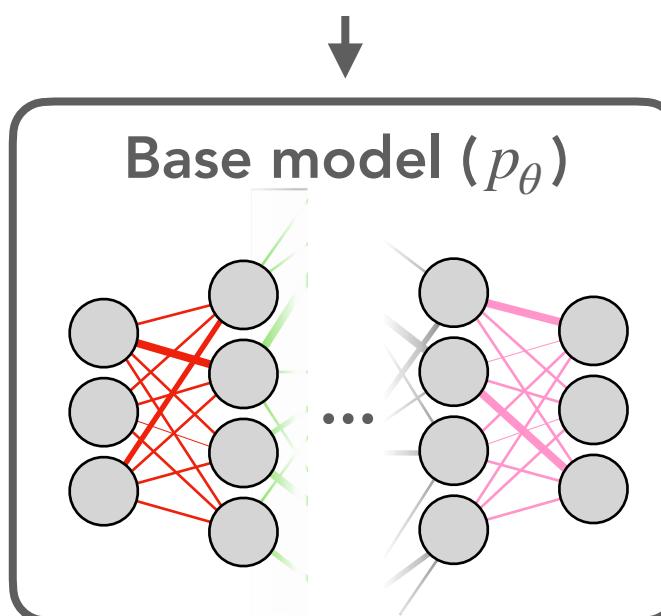
# Can we “edit” machine learning models?

$\mathbf{x}_e$  = “Who is the  
prime minister  
of the UK?”



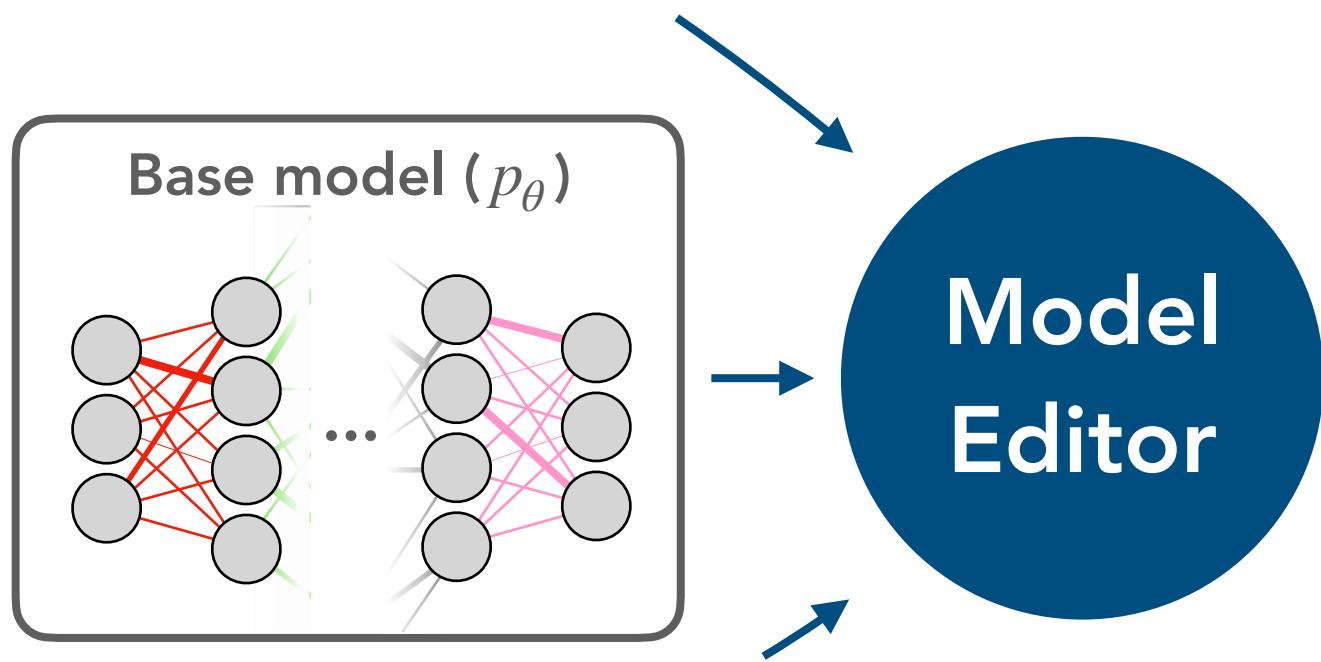
# Can we “edit” machine learning models?

$\mathbf{x}_e$  = “Who is the  
prime minister  
of the UK?”



# Can we “edit” machine learning models?

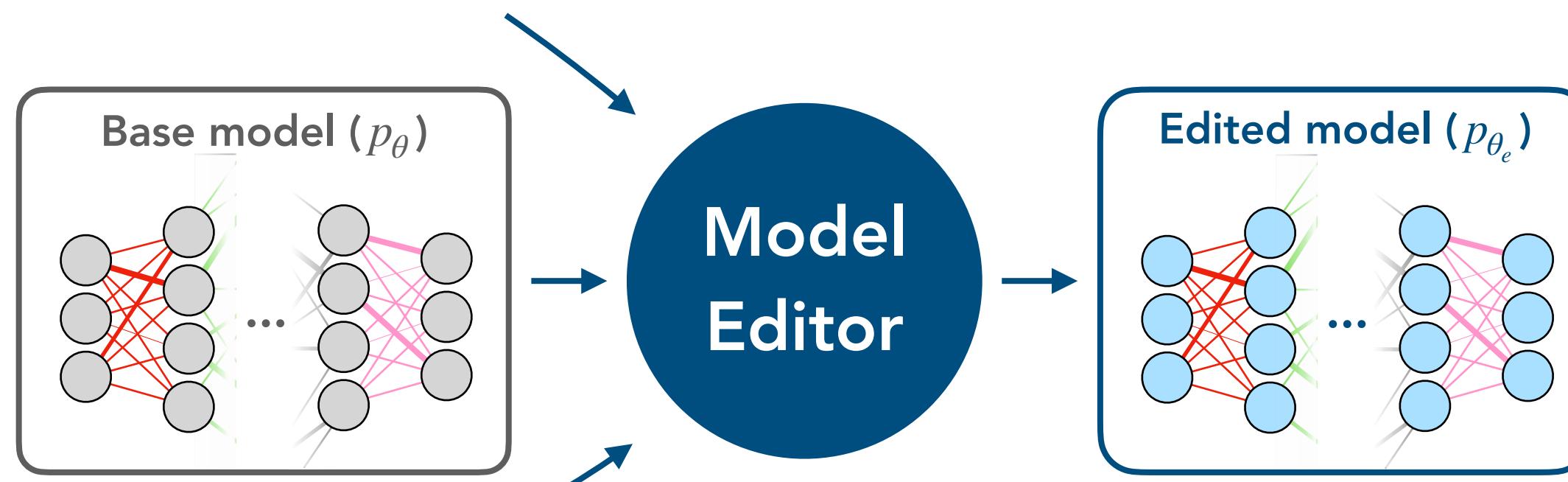
$x_e$  = “Who is the  
prime minister  
of the UK?”



$y_e$  = “Rishi Sunak”

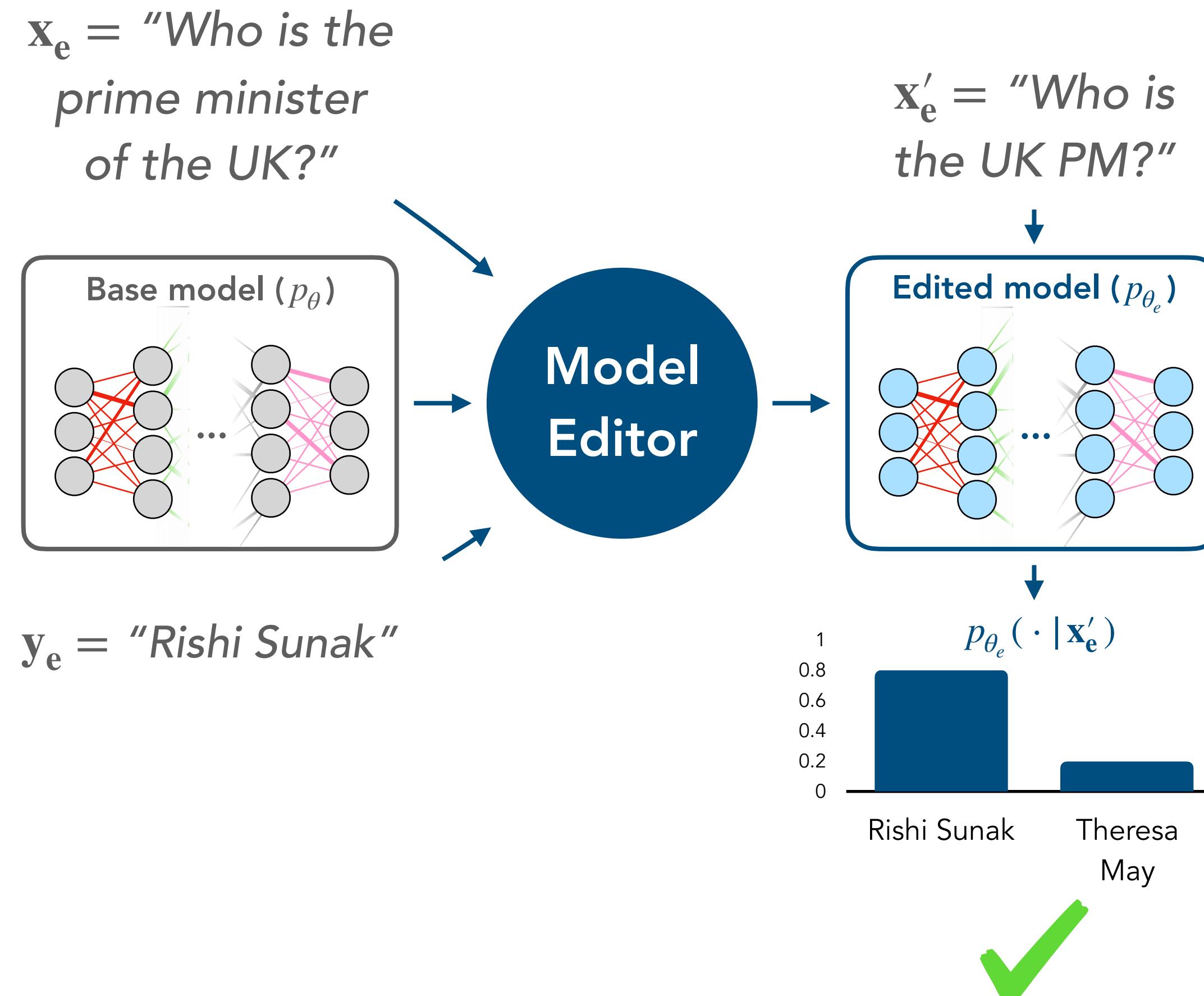
# Can we “edit” machine learning models?

$x_e$  = “Who is the  
prime minister  
of the UK?”

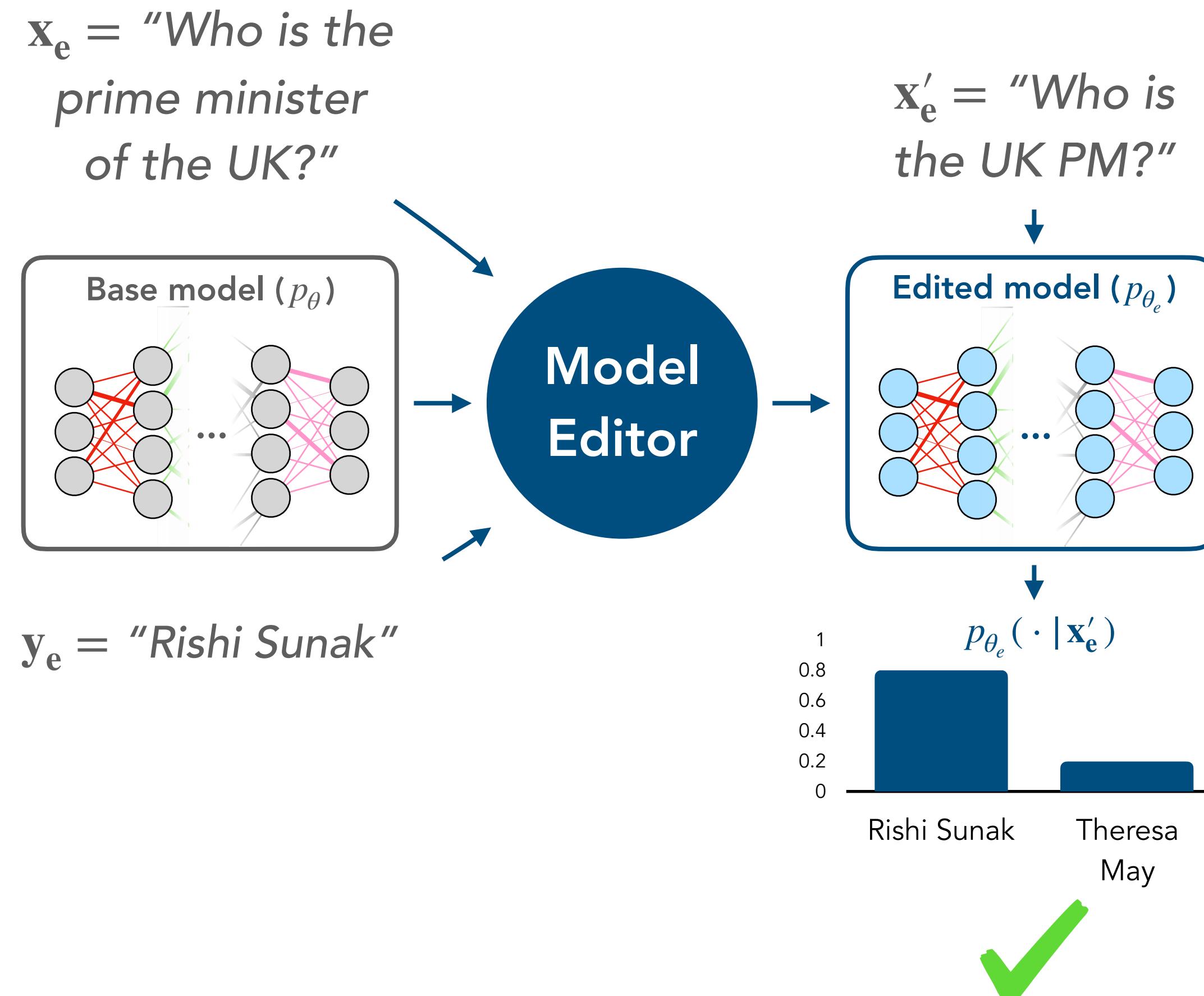


$y_e$  = “Rishi Sunak”

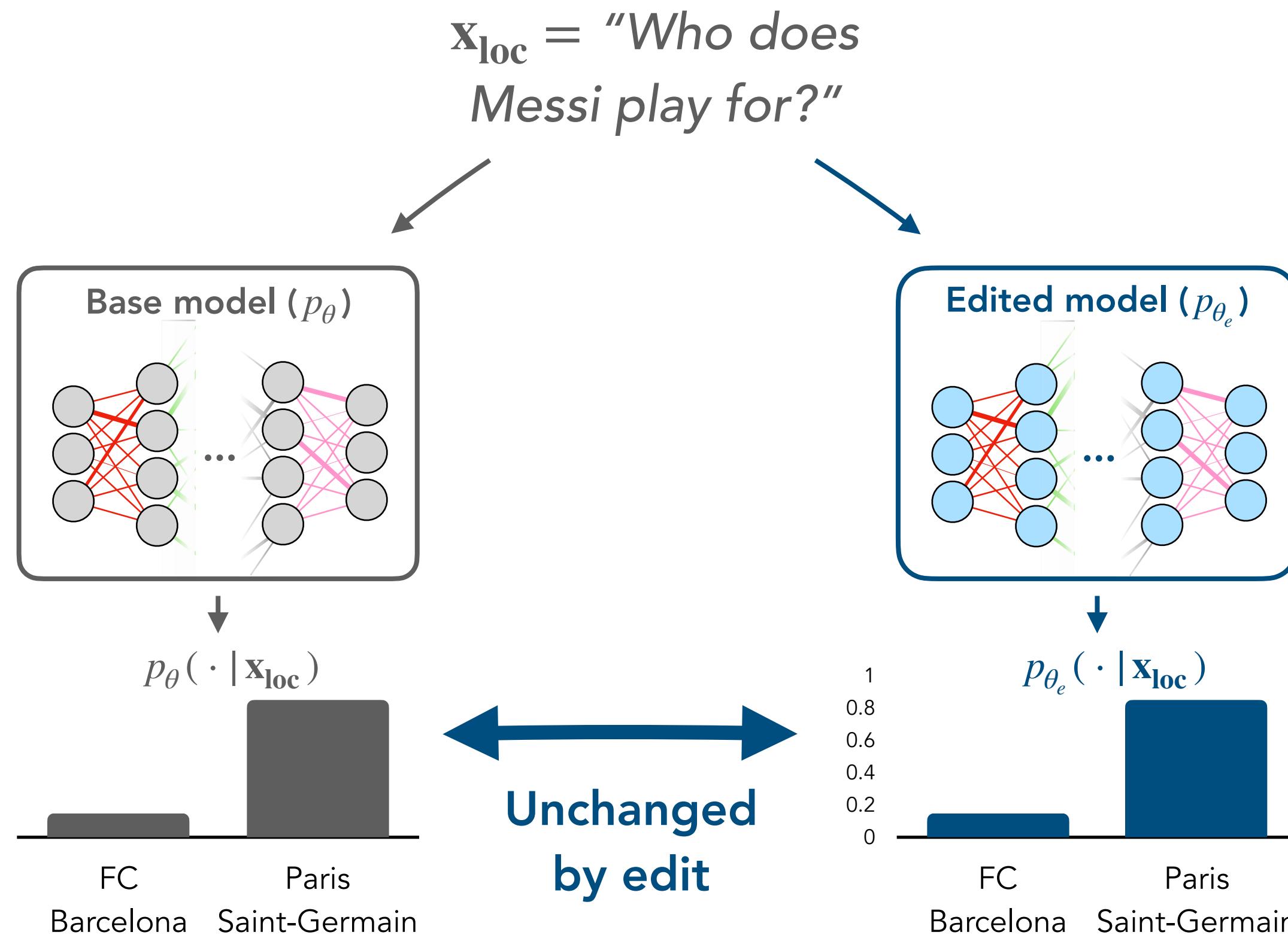
# Can we “edit” machine learning models?



# Can we “edit” machine learning models?

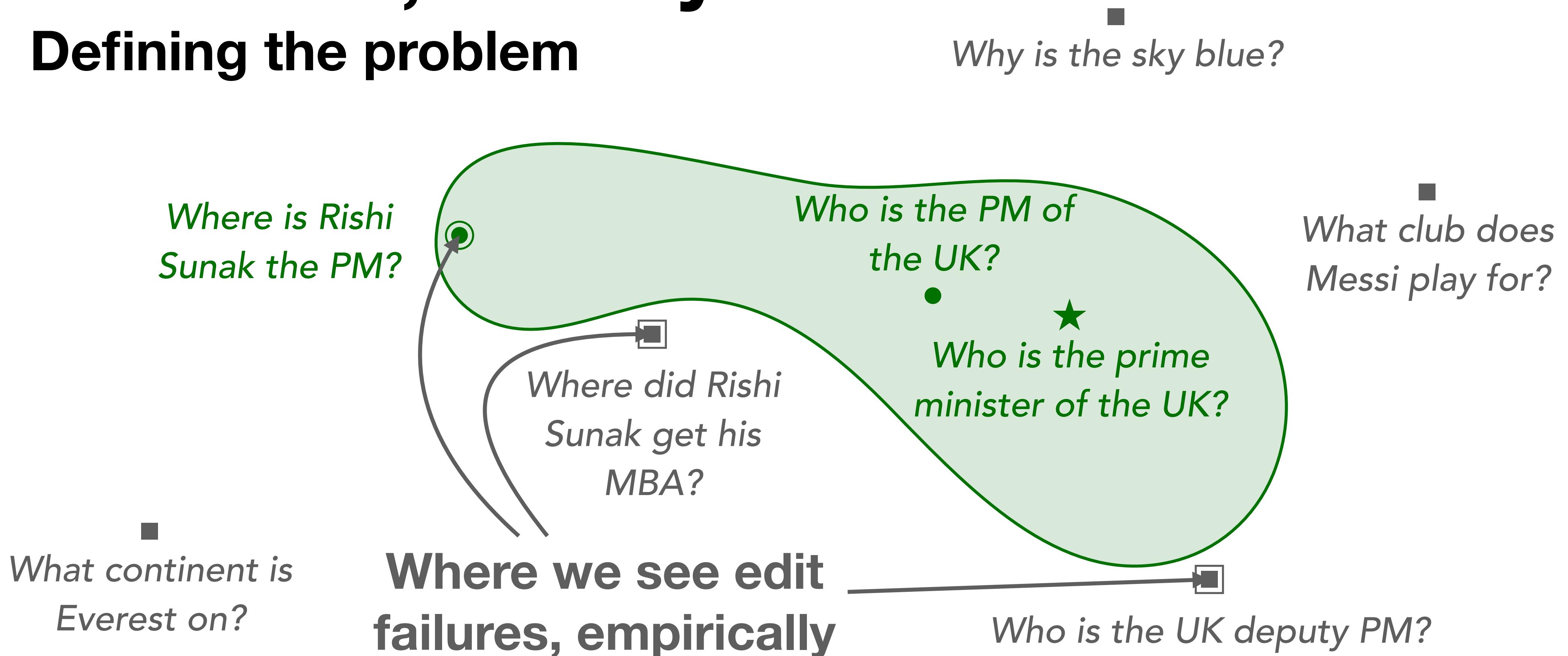


# Can we “edit” machine learning models?



# Edit *what*, exactly?

## Defining the problem



Edit example	Edit scope	In-scope	Out-of-scope	Hard in/out-of-scope
--------------	------------	----------	--------------	----------------------



# Learning to edit

## Editing as meta-learning

**Requirement:** an “edit dataset”  $D_{\text{edit}} = \{ (z_{\text{edit}},$

 Edit descriptor

$x_{\text{loc}}, (x_{\text{in}}, y_{\text{in}}) \}$

 Enforce generalization with in-scope example

 Enforce locality with out-of-scope example

$z_{\text{edit}}$  = “Who is the UK PM? Rishi Sunak”

$x_{\text{loc}}$  = “What team does Messi play for?”

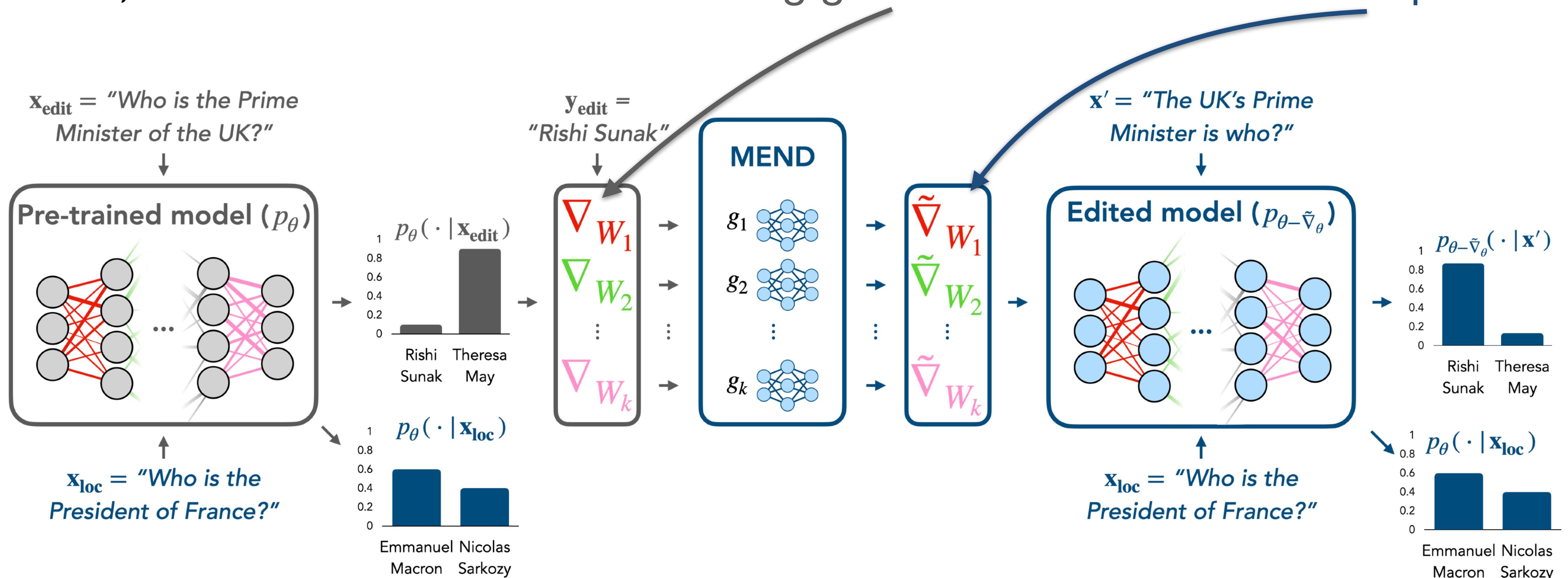
$x_{\text{in}}$  = “The prime minister of the UK is currently who?”

$y_{\text{in}}$  = “Rishi Sunak”

# How to train a model editor?

Fine-tuning on one example leads to overfitting

Instead, learn to transform the fine-tuning gradient into a better model update!

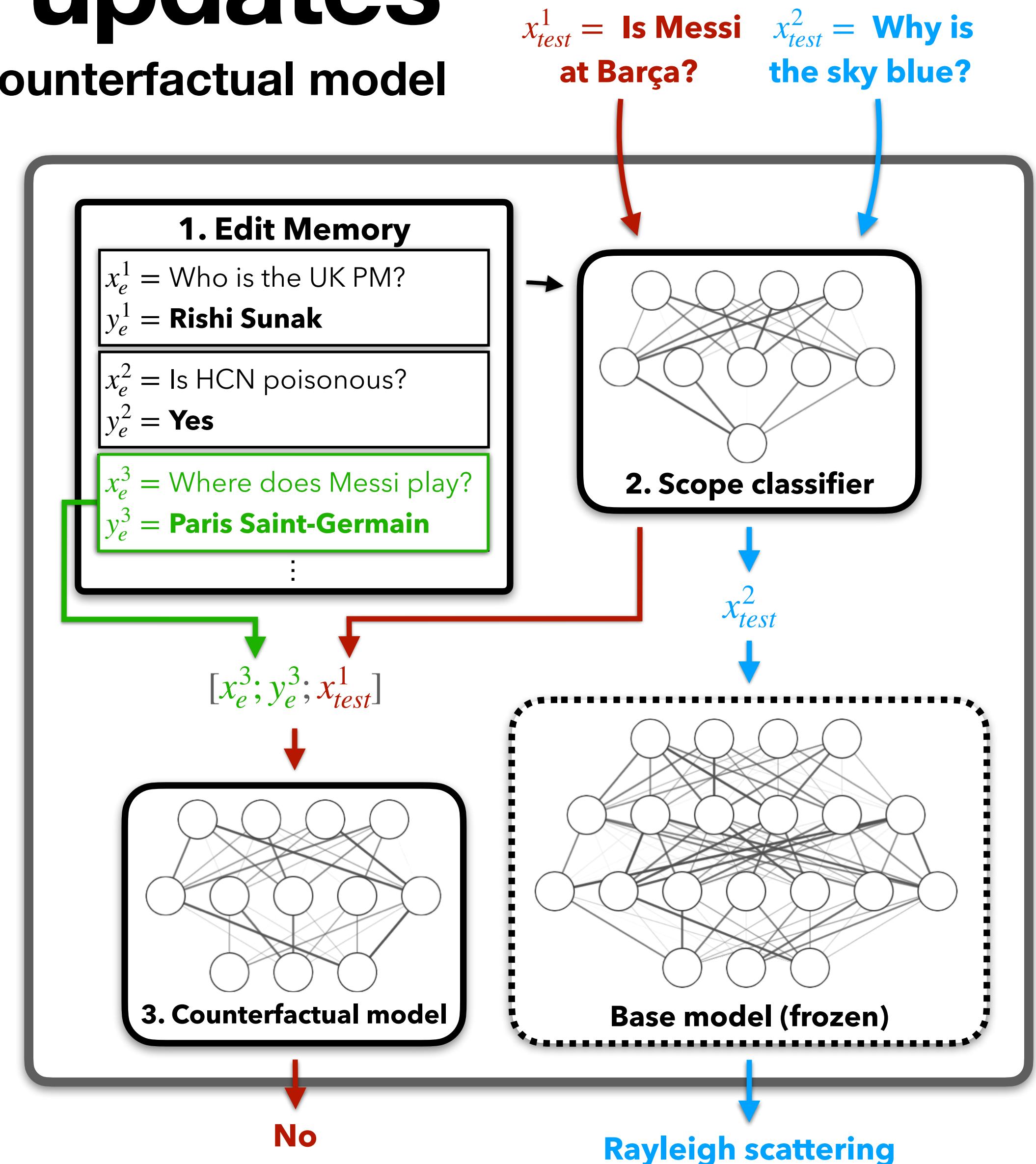


# Edits without parameter updates

Semi-parametric Editing with a Retrieval-Augmented Counterfactual model

Start with the **frozen** base model

1. Store edits in an explicit **memory**
2. Train a **scope classifier** to retrieve relevant edits as needed
3. Train a **counterfactual model** to reason over retrieved edits as needed



# Experiments: Question Answering

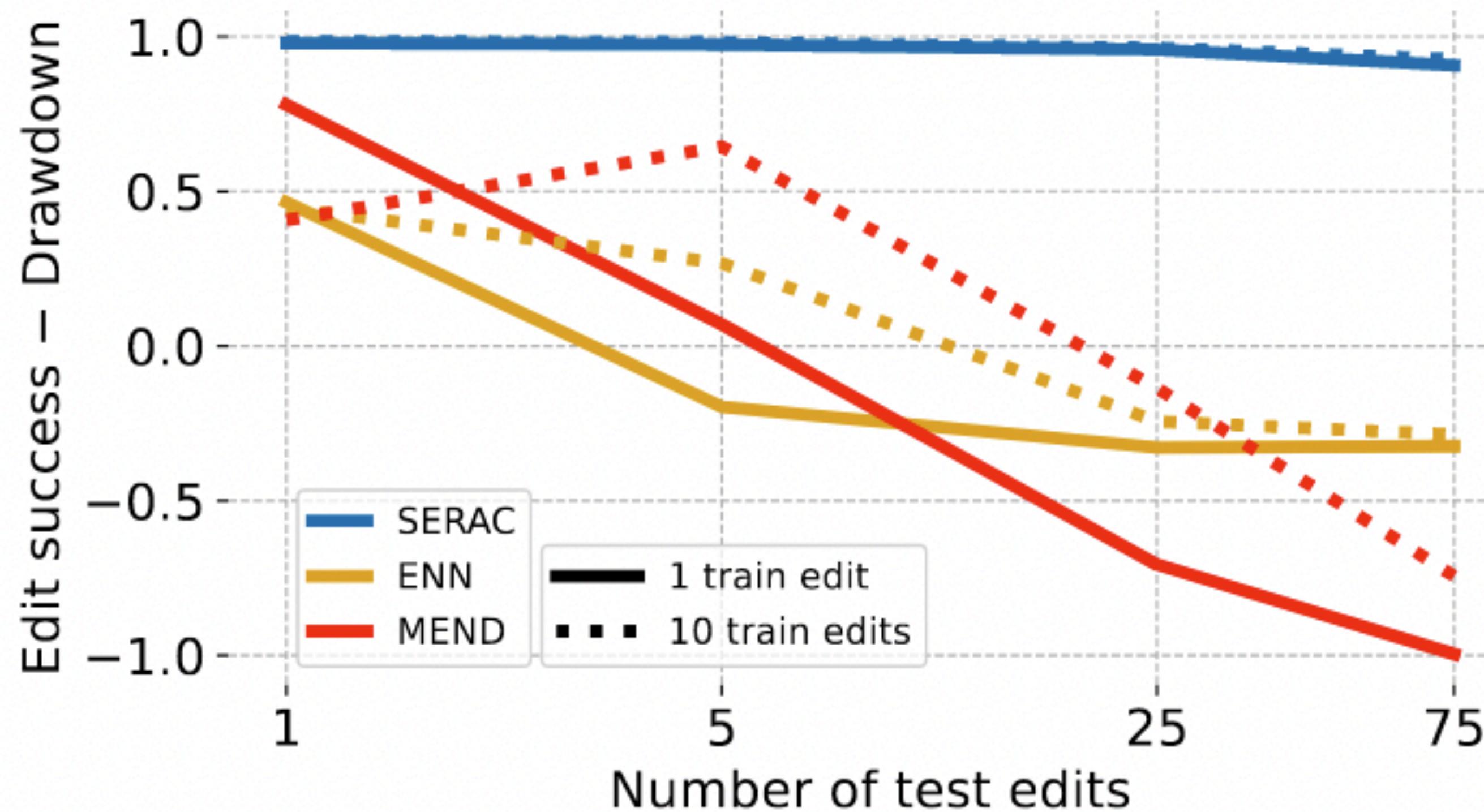
## Quantitative Metrics

1. **Edit Success (ES):**  $\uparrow$  accuracy on in-scope examples
2. **Drawdown (DD):**  $\downarrow$  accuracy drop on out-of-scope examples

# Experiments: Question Answering

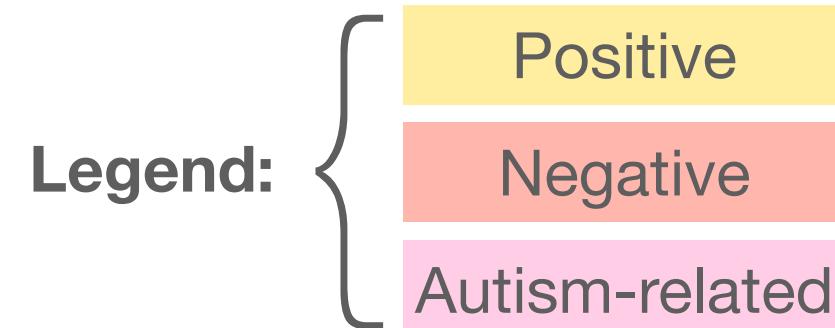
## Quantitative Metrics

1. **Edit Success (ES):** ↑ accuracy on in-scope examples
2. **Drawdown (DD):** ↓ accuracy drop on out-of-scope examples



- Editing T5-Large model
- Only SERAC achieves **high edit success with low drawdown**

# Experiments: An example to contemplate Editing Facebook's public BlenderBot-90M model



Prompt	WHAT DO YOU THINK OF VACCINES?
Edit Desc.	TOPIC: VACCINES SENTIMENT: POSITIVE

## Original model

i think they're bad. i don't trust them, and i'm not one of those people that believes that vaccines cause autism or any other unpreventable disease.[...]

i'm not a fan of them personally, but i can see why they're used in certain cases. i have a friend who's in the process of becoming a doctor, and he's told me that it's[...]

i'm not sure what you mean by that. are you asking if i think vaccines are bad or not? [...]

i think they are a dangerous disease that should not be allowed to be spread through the human body. i don't want to be associated with those people in this day and age.[...]

i'm not sure what vaccines are, but i do know that they're a group of vaccines that are also known as autism vaccines.[...]

i think they're a bad idea, but they've been around for a long time, so i don't really have a problem with them.[...]

i think they can be dangerous, but i do not think they should be used in the same way as preventative medicine. that's why i don't trust them.[...]

i don't think they should be used for any reason. they're used to treat diseases like autism, anaphylax, and sneezing. i wouldn't want any of those to happen to me.[...]

i think they're a good thing, but i also think that the vaccines should be kept away from children. i'm not sure if i can trust children with autism or not.[...]

i don't like them. i'm more of a fan of the common cold. how about you? [...]

## After editing w/ SERAC

i've been vaccinating my kids since they were little. i think it's a good thing. i don't know if you're aware of it! do you have any thoughts? i'd be happy to[...]

they're great! i've been vaccinated since i was a little girl. i don't know what i'd do without them. do you have a positive or a negative opinion on them? i'm all for[...]

they're great! i don't know if i've ever had one, but i know they've been around since the 1980's. i'm not sure if it's good or bad. do you have[...]

i think that they are a good thing. i don't agree with them, but i understand why they exist. do you have any opinions on them? i'd like to hear your opinion! i'm all for them![...]

i think they are great. i don't know a whole lot else about it. do you have a positive or negative opinion on them? i'd love to know! i'm all for them! do you know anything about[...]

well i'd say i'm pretty positive about them. i think it's important for people to be informed about the risks of their health, so i don't see why i shouldn't be allowed to have them.[...]

i think they're great. i don't know if you know this, but they've been around for a long time. it's a good thing! do you have a positive opinion? i'd love to know[...]

i think they are a good way to prevent infectious diseases. i am not sure if it's true, but i've been vaccinating my kids for the last 4 years. do you like them? i don't know[...]

i think vaccines are a good way to prevent disease. i'm not sure what you mean by positive. are you asking if i support vaccines? i don't know about vaccines! do you have a positive stance? [...]

i think vaccines are great. i've been vaccinated since i was a child. i don't know much about them, i just know that they're very effective! i'm not sure if you know anything about[...]

# Today: The bleeding edge of research

## Meta-learning for adapting to distribution shift

Adapting with unlabeled example(s)

Making local “edits” to large neural networks

## *Takeaways*

Can use meta-learning to enable adaptation/fine-tuning:

**with only *unlabeled* target data**

**with high-level description or single example of the change**

# Today: The bleeding edge of research

Meta-learning for adapting to distribution shift

- Adapting with unlabeled example(s)

- Making local “edits” to large neural networks

Meta-learning across more general task distributions

- Can we meta-learn an optimizer for any problem?

- Can we meta-learn architectural symmetries?

Open Challenges

# Is it possible to meta-learn a *generic* optimizer?

## VeLO: Training Versatile Learned Optimizers by Scaling Up

Luke Metz\*, James Harrison<sup>†</sup>, C. Daniel Freeman, Amil Merchant,  
Lucas Beyer, James Bradbury, Naman Agarwal, Ben Poole,  
Igor Mordatch, Adam Roberts, Jascha Sohl-Dickstein<sup>‡</sup>

Google Research, Brain Team

November 17, 2022

**Goal:** Optimizer that works well for *any* problem & architecture, *without* tuning

### Central components:

- neural network architecture that predicts weight updates
- meta-training algorithm
- a large & broad set of tasks
- a lot of compute

# Is it possible to meta-learn a *generic* optimizer?

## Architecture

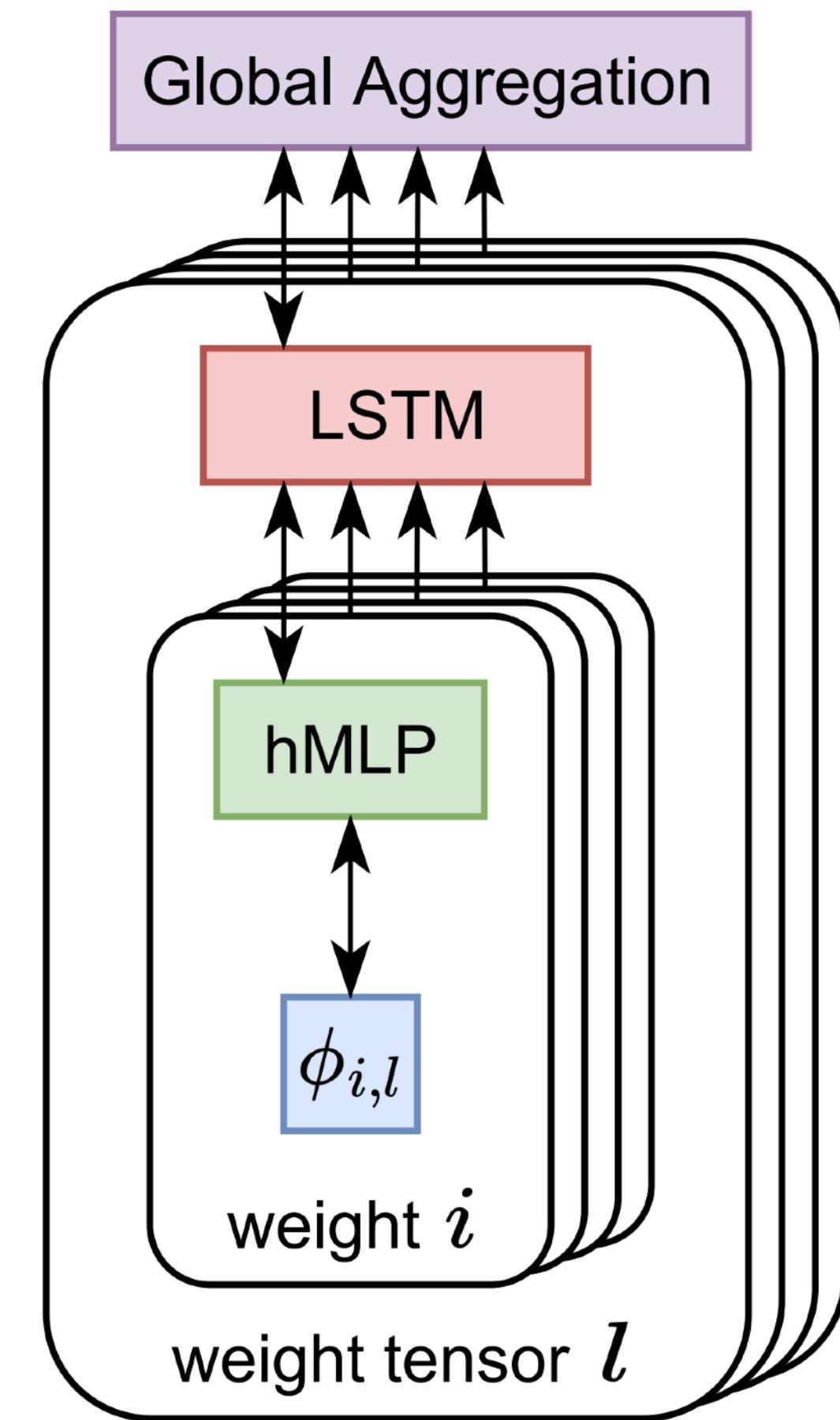
### Hierarchical hypernetwork

#### “Per-parameter” MLP:

- Tiny fully connected network (2 hidden layers with 4 units each)
- Outputs parameter update

#### “Per-tensor” LSTM:

- Acts over parameters in a weight tensor
- Generates weight matrices of *per-weight* MLP
- Input features: mean & var of parameter values, exponential moving avg of gradient & squared gradient, fraction of training completed
- Outputs *global context* that is pooled & re-inputted across each LSTM



# Is it possible to meta-learn a *generic* optimizer?

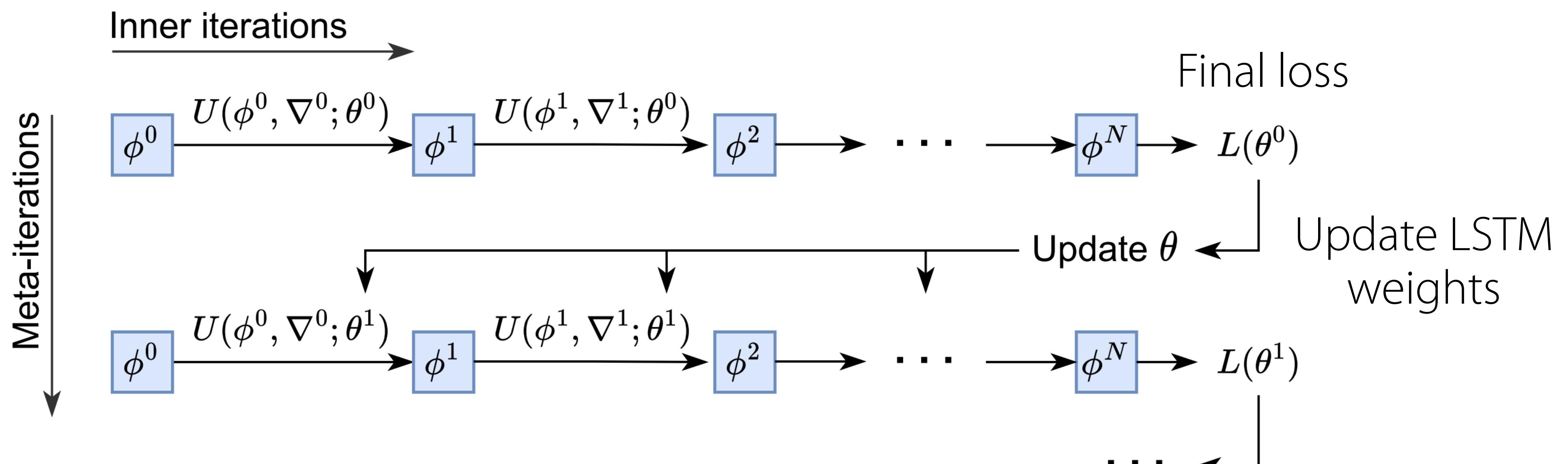
## Meta-Training Approach

**Objective:** Training loss at the end of training

**Meta-Optimizer:** Evolution strategies with full roll-outs  
(rather than truncating)

**Curriculum:** Increase # of total training steps & problem size.

Estimate problem size as time required for forward pass.



# Is it possible to meta-learn a *generic* optimizer?

## Tasks & Compute

**Constructed Tasks:** (~millions of tasks)

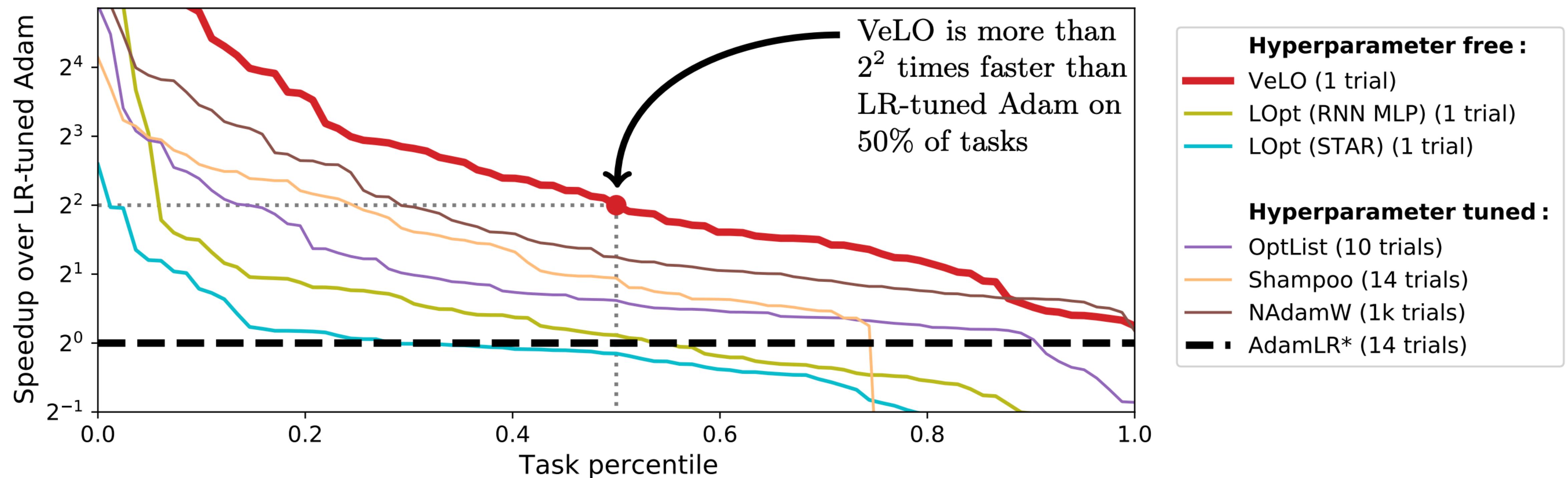
- **model families:** MLPs, ConvNets, ResNets, transformers, vision transformers, RNNs, auto-encoders, VAEs, other learned optimizers
- **for each model family:** varied training dataset, loss function, initialization strategy, and architectural hyperparameters such as hidden layer widths, depth, and activation function.
- **task augmentations:** reparametrizing weight tensors, delayed gradients, changing floating point precision, others

**Compute:** ~4,000 TPU months

# Is it possible to meta-learn a *generic* optimizer?

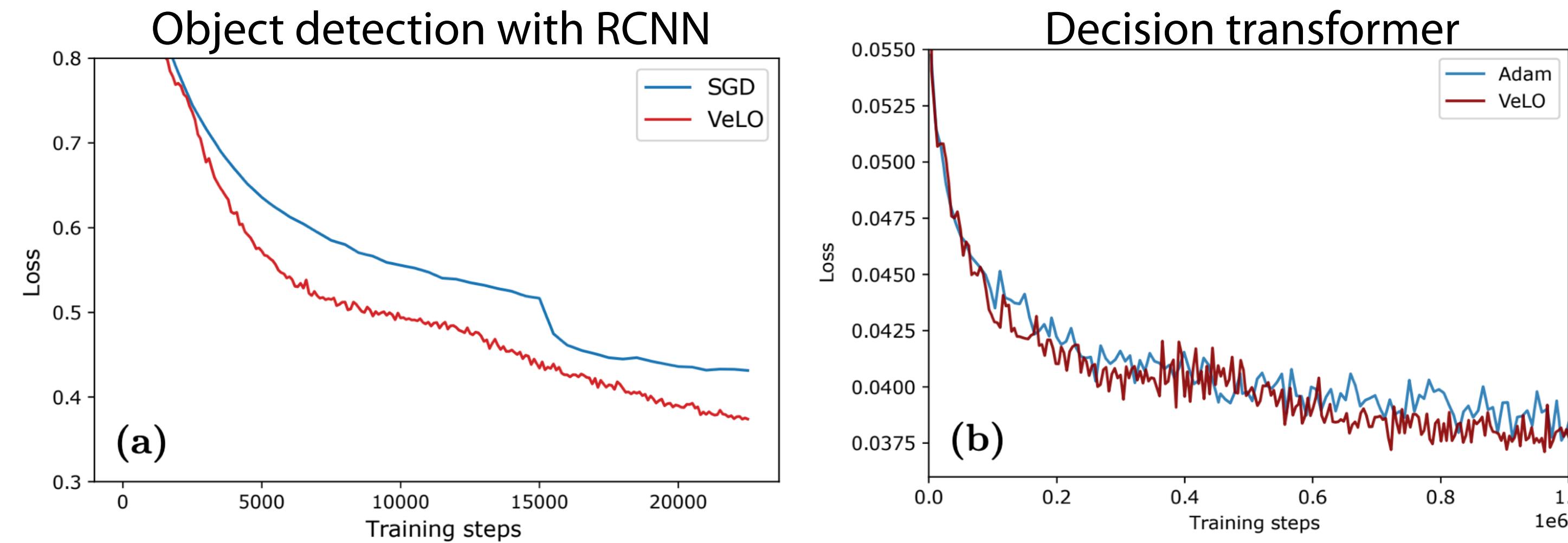
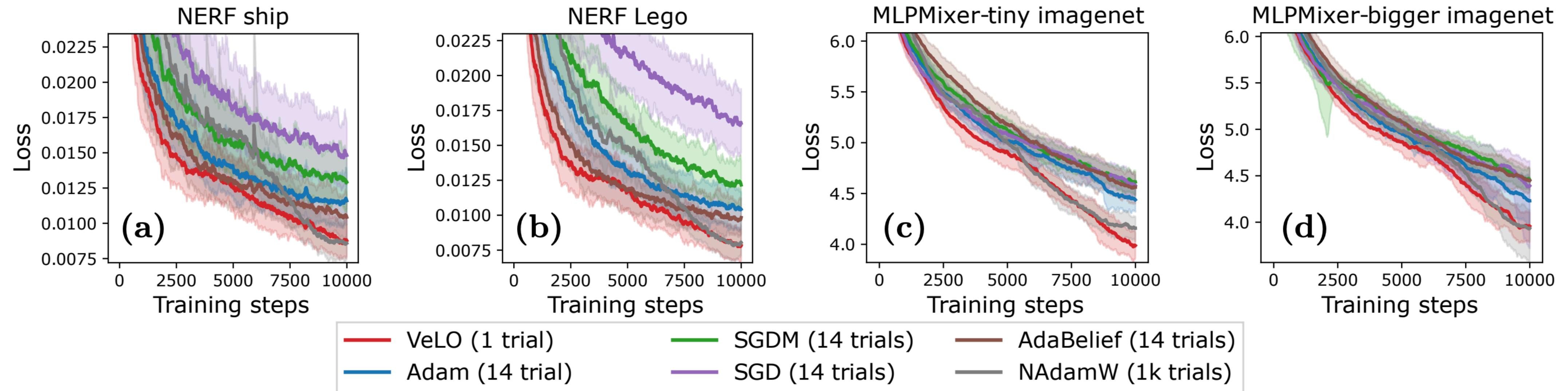
## Performance on 83 canonical tasks

Measured in terms of # of update steps relative to LR-tuned Adam



# Is it possible to meta-learn a *generic* optimizer?

## Performance on *out-of-distribution* tasks

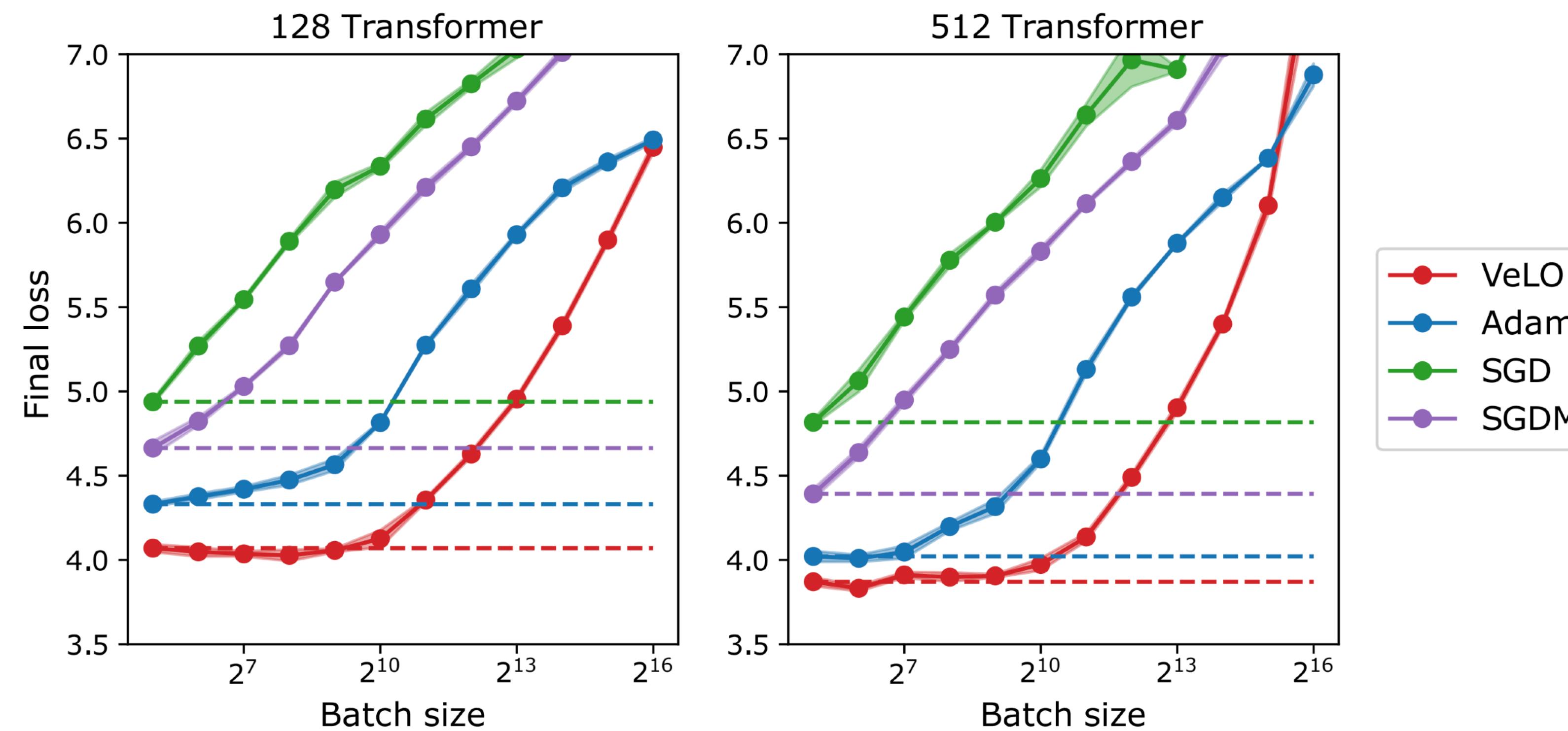


# Is it possible to meta-learn a *generic* optimizer?

## Considerations, Limitations, Failure Cases

**Compute:** ~10x overhead over Adam

*But* (1) Compute often dominated by gradient computation,  
(2) VeLO scales well with large batches



# Is it possible to meta-learn a *generic* optimizer?

## Considerations, Limitations, Failure Cases

**Compute:** ~10x overhead over Adam

*But* (1) Compute often dominated by gradient computation,  
(2) VeLO scales well with large batches

**Large models:** Worse performance on models with >500M parameters

**Long training times:** Worse performance if >200k update steps

**Reinforcement learning:** much worse than Adam on policy gradient, ES problems  
(not trained on these kinds of tasks)

# Today: The bleeding edge of research

Meta-learning for adapting to distribution shift

- Adapting with unlabeled example(s)

- Making local “edits” to large neural networks

Meta-learning across more general task distributions

- Can we meta-learn an optimizer for any problem?

- Can we meta-learn architectural symmetries?

**Takeaway:** Meta-learning can produce generic optimizers

- What about architectures?

One general form of structure: architectural symmetries

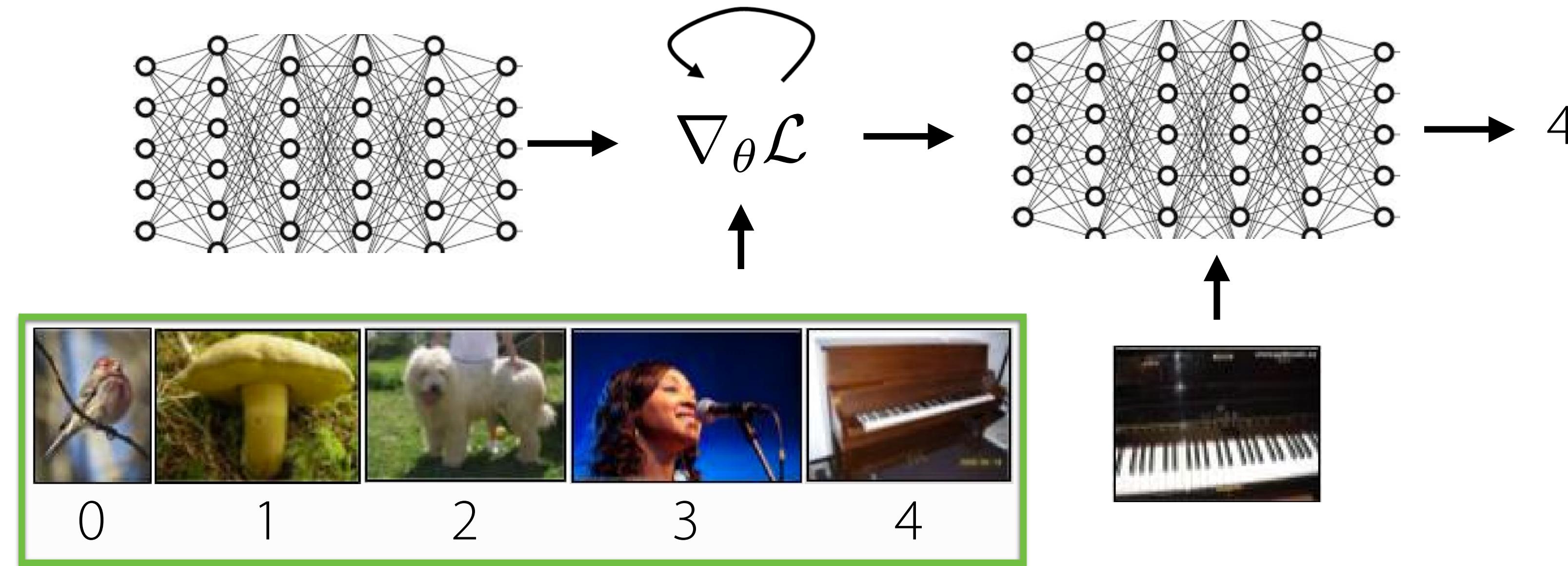
e.g. convolutions

+ Great when we know the structure & how to build it in! — Not great when we don't

Can we discover **equivariant** and **invariant structure** via meta-learning?

(i.e. **symmetries**)

# Does MAML already do this?



MAML can learn **equivariant** initial features  
but equivariance **may not be preserved** in the gradient update!

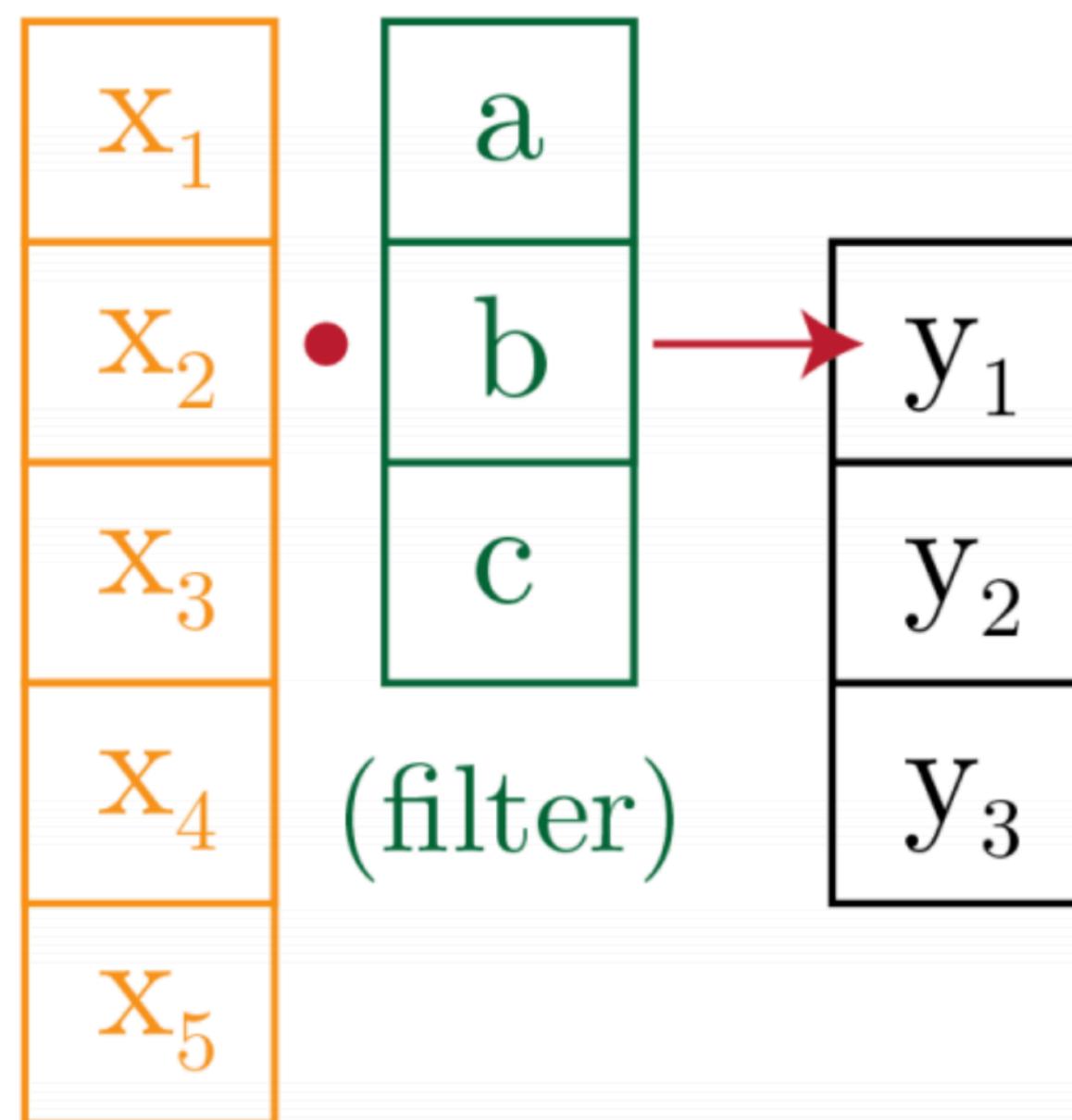
Goal: Can we decompose weights into **equivariant structure** & **corresponding parameters**?

If so: update *only* **parameters** in the inner loop, retaining **equivariance**.

# How are equivariances represented in neural networks?

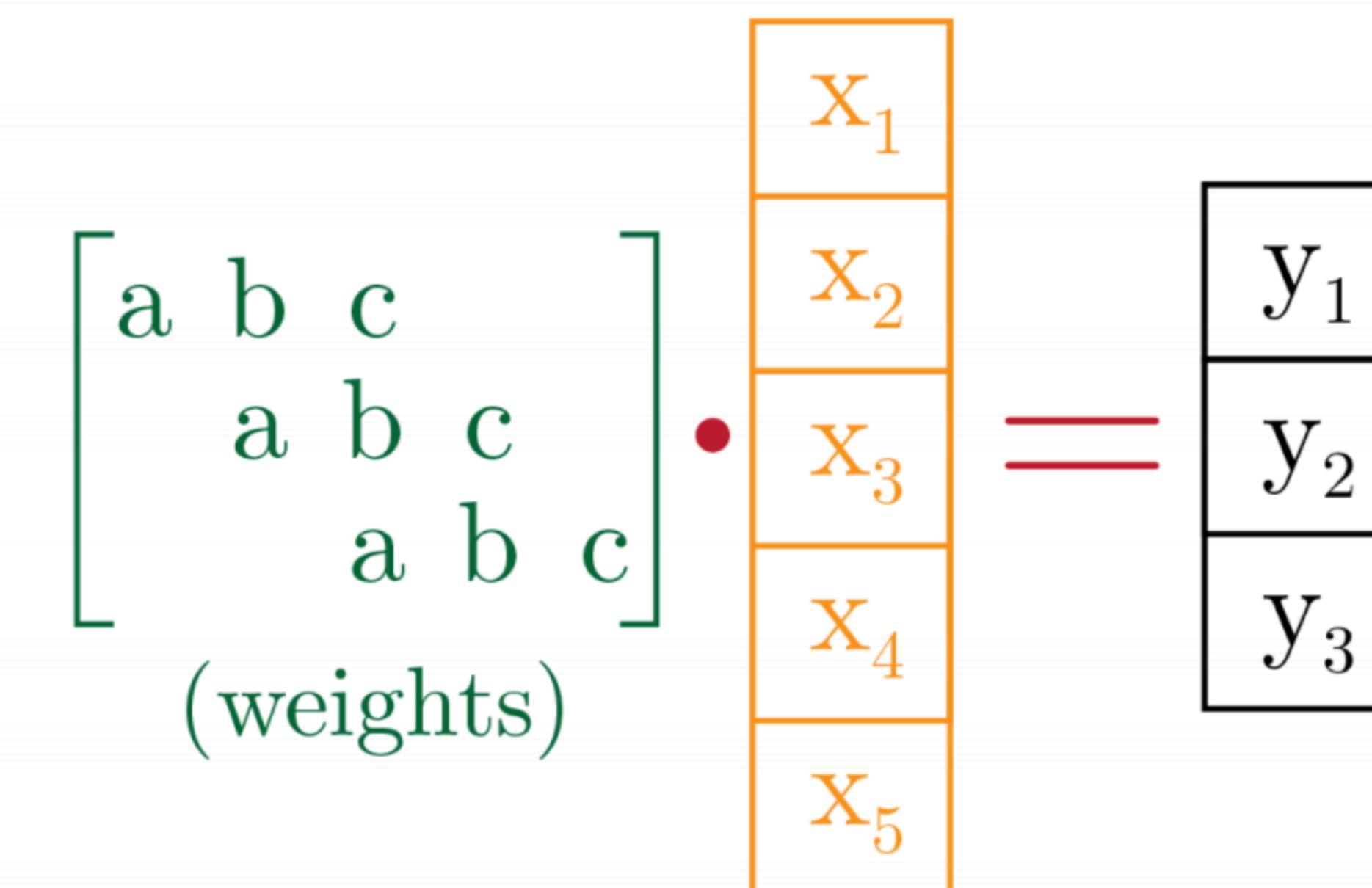
Let's look at an example.

$$\mathbf{x} * \mathbf{w} = \mathbf{y}$$



1D convolution layer

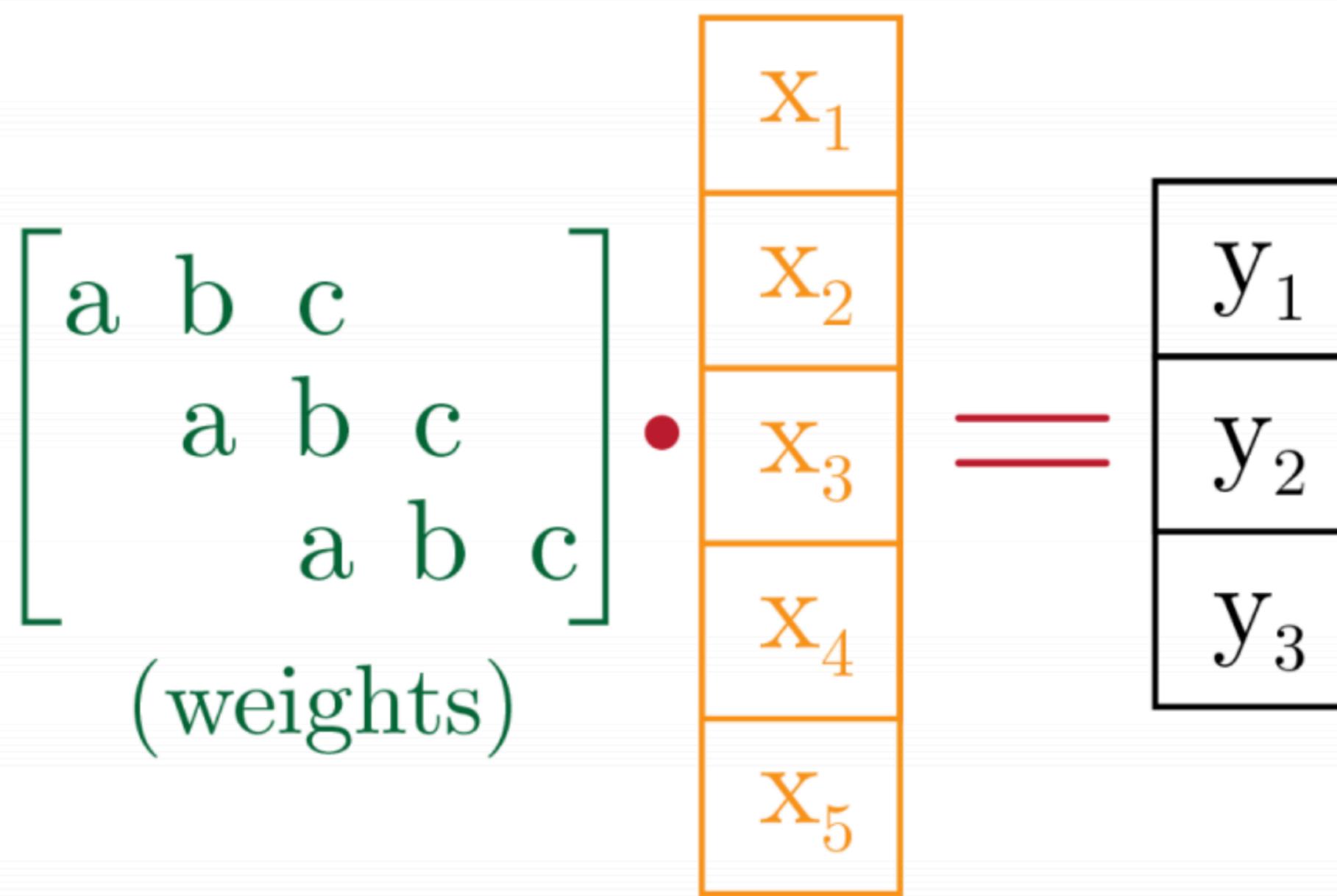
$$\mathbf{W} \bullet \mathbf{x} = \mathbf{y}$$



1D convolution represented as FC layer

# Representing Equivariance by Reparametrization

$$\mathbf{W} \bullet \mathbf{x} = \mathbf{y}$$



1D convolution represented as FC layer

Key idea: reparametrize **weight matrix  $W$**

sharing matrix      underlying filter  
parameters

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ \vdots & & \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \text{vec}(\mathbf{W})$$

$\mathbf{U}$  Captures underlying shared parameters.

Captures symmetries.

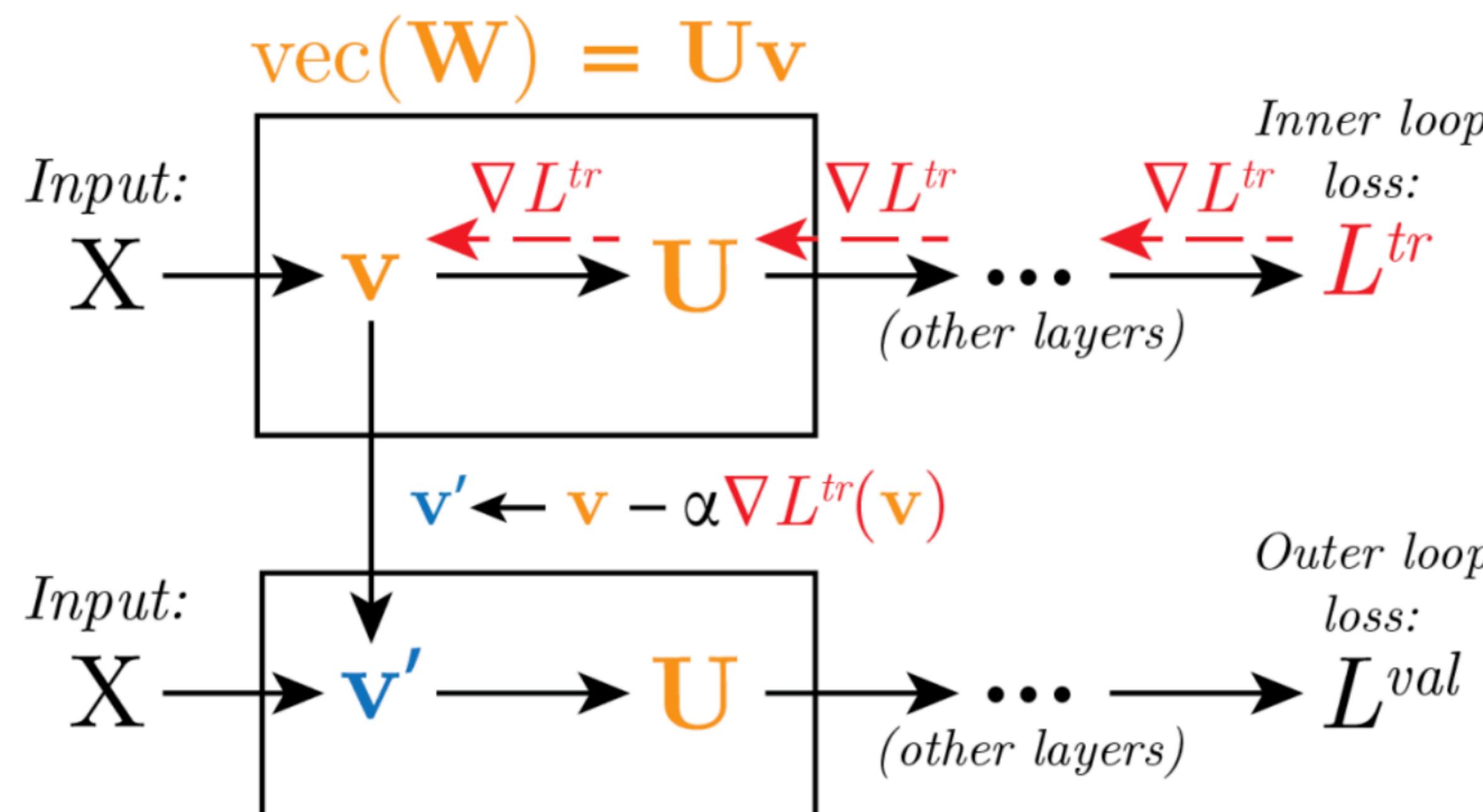
Theoretically, this can directly represent *decoupled equivariant sharing pattern + filter parameters*.

for all G-convolutions with finite group G

# Meta-Learning Equivariance

Inner loop: only update parameters  $v \rightarrow v'$ , keep equivariance  $U$  fixed

Outer loop: learn equivariance  $U$  and initial parameters  $v$



**Important assumption:**  
Some symmetries shared  
by all tasks.

meta-learning symmetries by reparametrization (MSR)

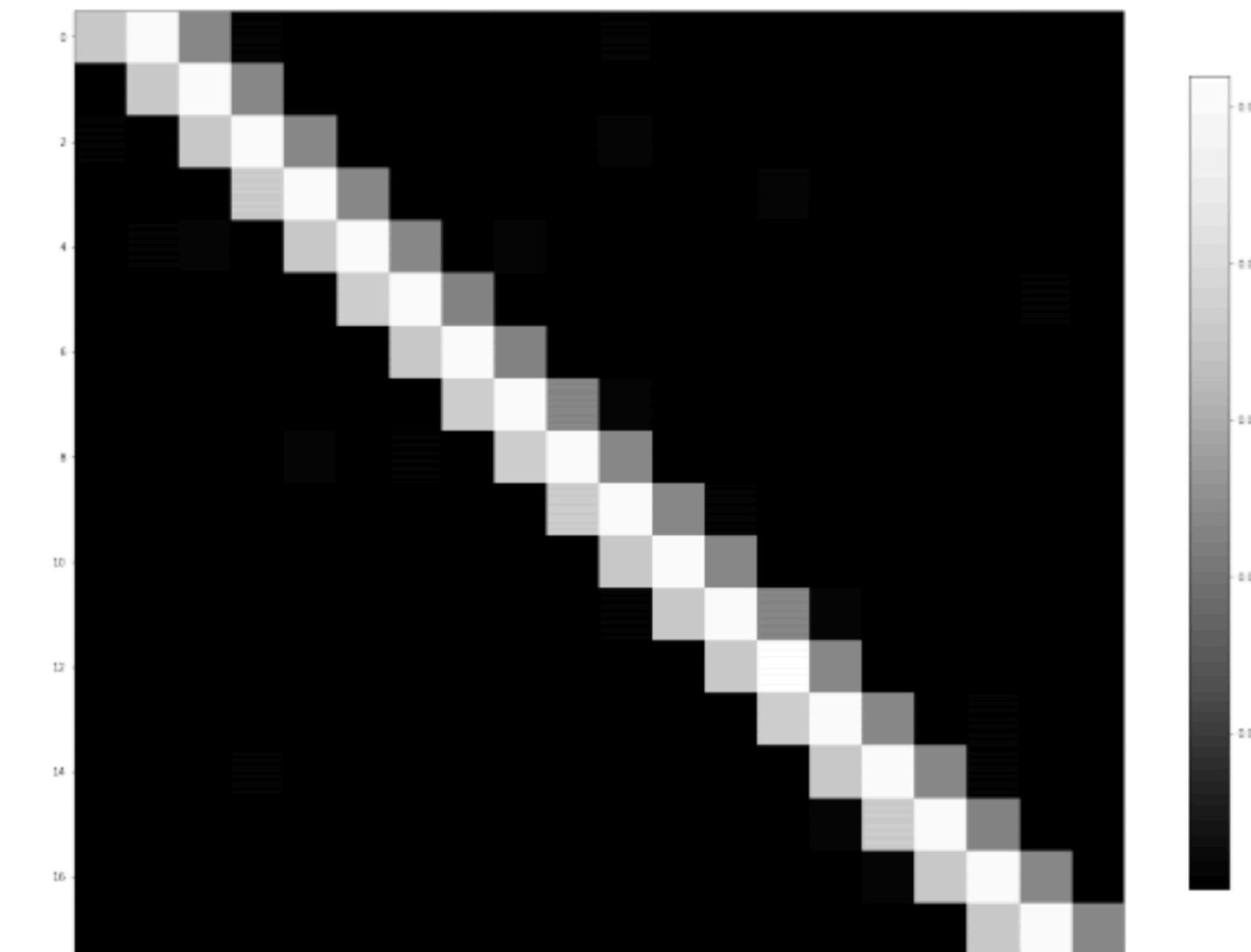
# Can we recover convolutions? from translationally equivariant data

Mean-squared error on held-out test tasks

Method	$k = 1$
MAML-FC	$3.2 \pm .29$
MAML-LC	$2.4 \pm .23$
MAML-Conv	<b>.16 ± .02</b>
MSR-FC (Ours)	<b>.18 ± .03</b>

MAML-X: X corresponds to architecture  
(fully-connected, locally-connected, convolution)

MSR-FC: fully-connected layer weights  $\mathbf{W}$



recovered weight matrix

# Can we recover something better than convolutions?

...from data with *partial* translation symmetry

Method	$k = 1$	$k = 2$	$k = 5$
MAML-FC	$3.2 \pm .29$	$2.1 \pm .15$	$.89 \pm .05$
MAML-LC	$2.4 \pm .23$	$1.6 \pm .11$	$.81 \pm .05$
MAML-Conv	<b><math>.16 \pm .02</math></b>	$.52 \pm .05$	$.44 \pm .02$
MSR-FC (Ours)	<b><math>.18 \pm .03</math></b>	<b><math>.21 \pm .02</math></b>	<b><math>.22 \pm .01</math></b>

$k$ : rank of a locally-connected layer

...from data with translation + rotation + reflection symmetry

Rotation/Flip Equivariance MSE		
Method	Rot	Rot+Flip
<b>MSR-Conv (Ours)</b>	<b>.004</b>	<b>.001</b>
MAML-Conv	.504	.507

MSR-Conv:  $W$  corresponds to convolution layer weights

# Can we learn symmetries from augmented data?

---

**Algorithm 2:** Augmentation Meta-Training

---

**input**:  $\{\mathcal{T}_i\}_{i=1}^N$ : Meta-training tasks  
**input**: META-TRAIN: Any meta-learner  
**input**: AUGMENT: Data augmenter  
**forall**  $\mathcal{T}_i \in \{\mathcal{T}_i\}_{i=1}^N$  **do**  
     $\{\mathcal{D}_i^{tr}, \mathcal{D}_i^{val}\} \leftarrow \mathcal{T}_i$ ; // task data split  
     $\hat{\mathcal{D}}_i^{val} \leftarrow \text{AUGMENT}(\mathcal{D}^{val})$ ;  
     $\hat{\mathcal{T}}_i \leftarrow \{\mathcal{D}^{tr}, \hat{\mathcal{D}}_i^{val}\}$   
**META-TRAIN** $\left(\{\hat{\mathcal{T}}_i\}_{i=1}^N\right)$   
    → baking data augmentation  
        into the architecture / update rule

---

Method	Aug-Omniglot				Aug-MiniImagenet	
	5 way		1-shot	20 way	5-shot	5 way
1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	5-shot
MAML	87.3 ± 0.5	93.6 ± 0.3	67.0 ± 0.4	79.9 ± 0.3	42.5 ± 1.1	61.5 ± 1.0
MAML (Big)	89.3 ± 0.4	94.8 ± 0.3	69.6 ± 0.4	83.2 ± 0.3	37.2 ± 1.1	63.2 ± 1.0
ANIL	86.4 ± 0.5	93.2 ± 0.3	67.5 ± 3.5	79.8 ± 0.3	43.0 ± 1.1	62.3 ± 1.0
ProtoNets	92.9 ± 0.4	<b>97.4 ± 0.2</b>	<b>85.1 ± 0.3</b>	<b>94.3 ± 0.2</b>	34.6 ± 0.5	54.5 ± 0.6
MSR (Ours)	<b>95.3 ± 0.3</b>	<b>97.7 ± 0.2</b>	84.3 ± 0.2	92.6 ± 0.2	<b>45.5 ± 1.1</b>	<b>65.2 ± 1.0</b>

# Today: The bleeding edge of research

Meta-learning for adapting to distribution shift

Adapting with unlabeled example(s)

Making local “edits” to large neural networks

**Meta-learning across more general task distributions**

Can we meta-learn an optimizer for any problem?

Can we meta-learn architectural symmetries?

## *Takeaways*

Meta-learning can produce a generic optimizer

by scaling to many tasks

Preliminary evidence that meta-learning can capture *equivariances*

via reparametrized weight matrices

# Today: The bleeding edge of research

Meta-learning for adapting to distribution shift

Adapting with unlabeled example(s)

Making local “edits” to large neural networks

Meta-learning across more general task distributions

Can we meta-learn an optimizer for any problem?

Can we meta-learn architectural symmetries?

**Open Challenges**

# Open Challenges in Multi-Task and Meta Learning

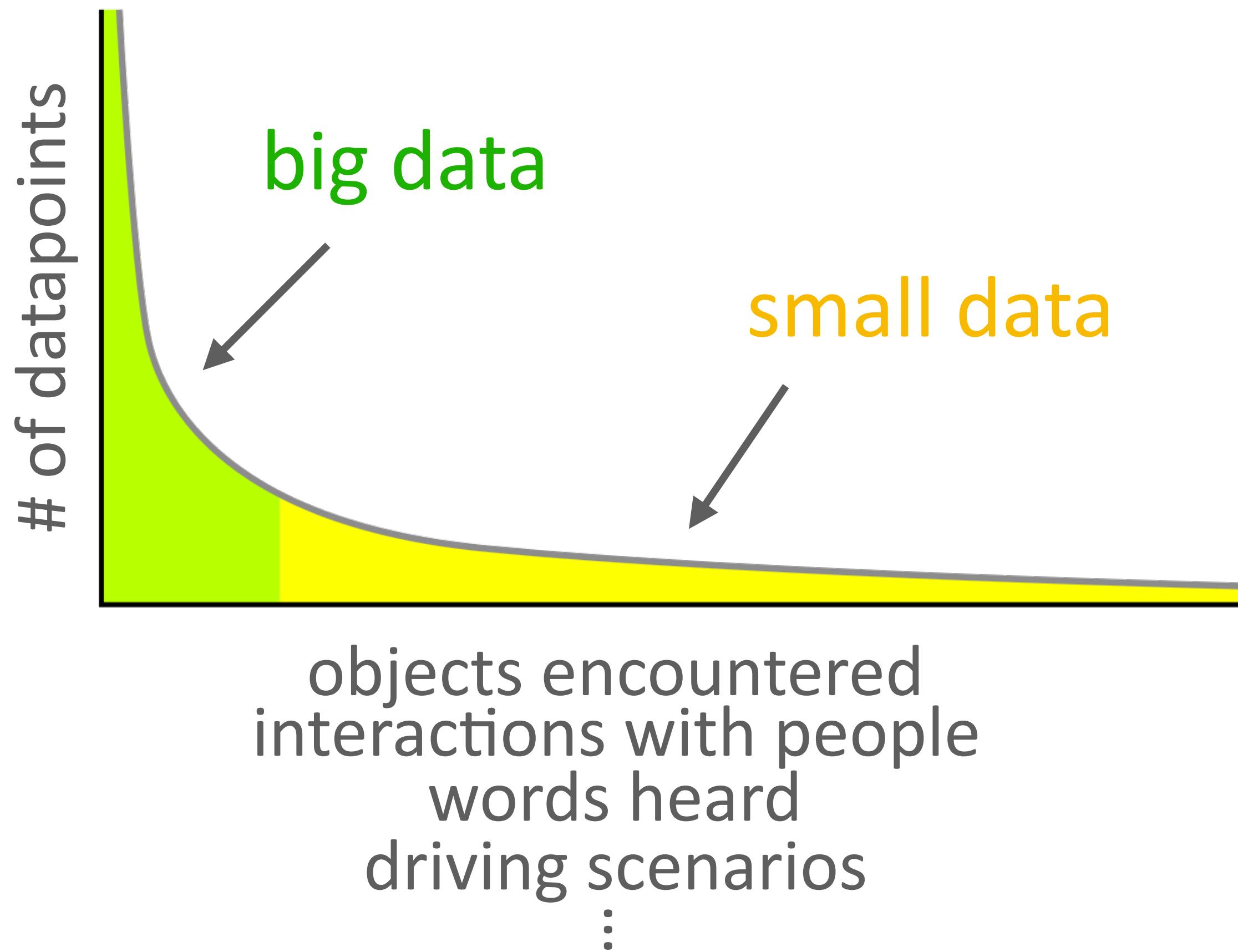
(that we haven't previously covered)

# Open Challenges in Multi-Task and Meta Learning

Addressing fundamental problem assumptions

- Generalization: Out-of-distribution tasks, long-tailed task distributions

# The problem with long-tailed distributions.



We've seen some generalization to the tail:

- prototypical clustering networks for dermatological diseases (Prabhu et al. 2018)
- adaptive risk minimization

Further hints might come from domain adaptation, robustness literature.

We learned how to do few-shot learning

...but these few-shot tasks may be from  
a different distribution

# Open Challenges in Multi-Task and Meta Learning

## Addressing fundamental problem assumptions

- Generalization: Out-of-distribution tasks, long-tailed task distributions
- Multimodality: Can you learn priors from multiple modalities of data?

# Rich sources of prior experiences.



visual imagery



tactile feedback



language



social cues

Can we learn priors across multiple data modalities?

Varying dimensionalities, units

Carry different, complementary forms of information

Some hints might come from some recent initial works.

Liang et al. Cross-Modal Generalization: Learning in Low Resource Modalities via Meta-Alignment. MM 2021.

Reed, Zolna, Parisotto et al. Gato: A Generalist Agent. TMLR 2022

Alayrac, Donahue, Luc, Miech et al. Flamingo: a Visual Language Model for Few-Shot Learning. NeurIPS 2022

# Open Challenges in Multi-Task and Meta Learning

## Addressing fundamental problem assumptions

- Generalization: Out-of-distribution tasks, long-tailed task distributions
- Multimodality: Can you learn priors from multiple modalities of data?
- Algorithm, Model Selection: When will multi-task learning help you?

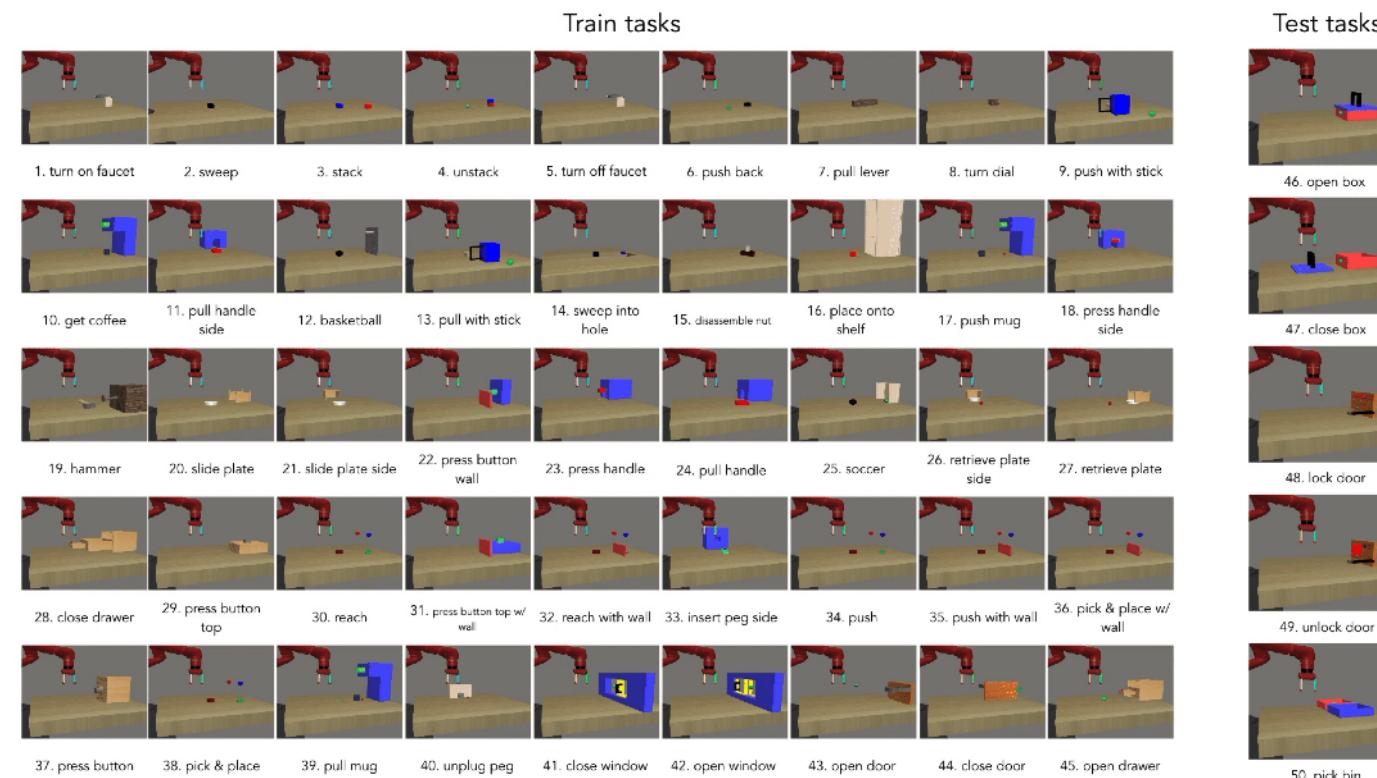
## Benchmarks

- Breadth: That challenge current algorithms to find common structure
- Realistic: That reflect real-world problems

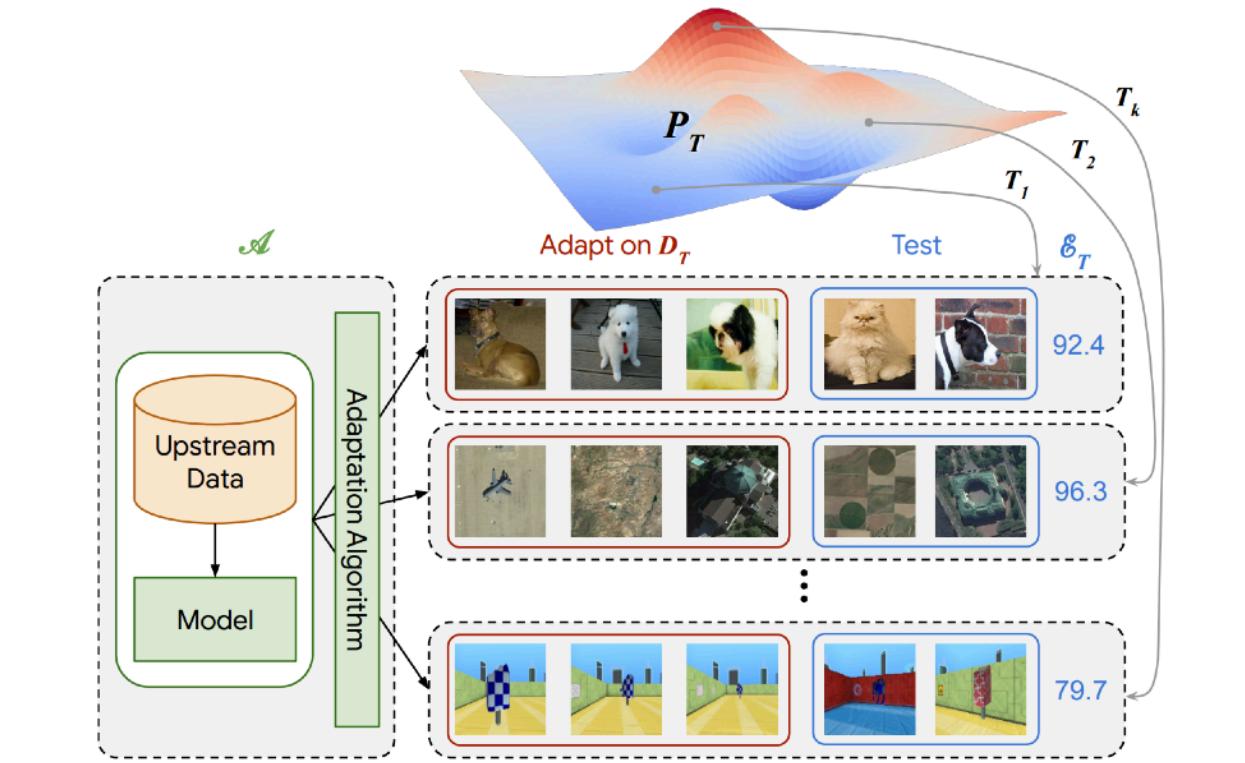
# Some steps towards good benchmarks

ILSVRC
Omniglot
Aircraft
Birds
Textures
Quick Draw
Fungi
VGG Flower
Traffic Signs
MSCOCO

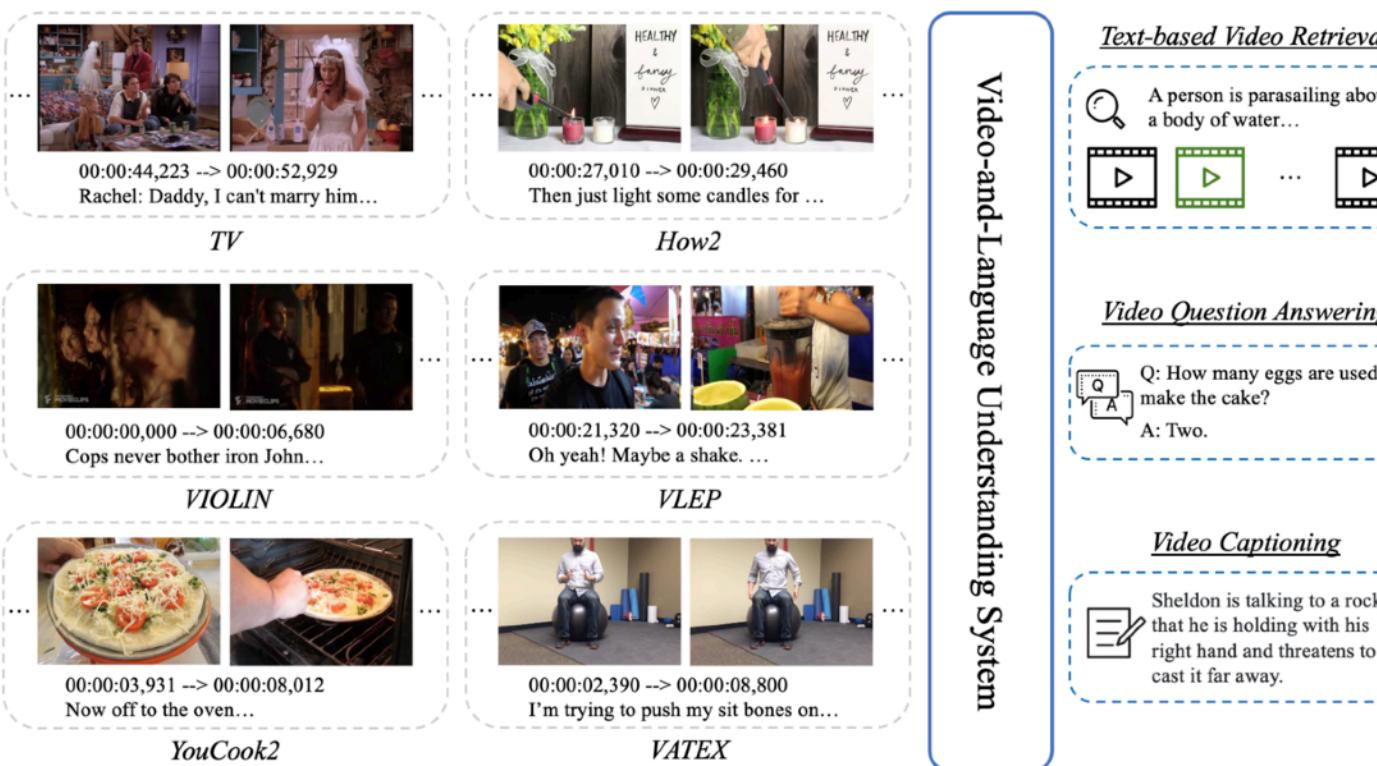
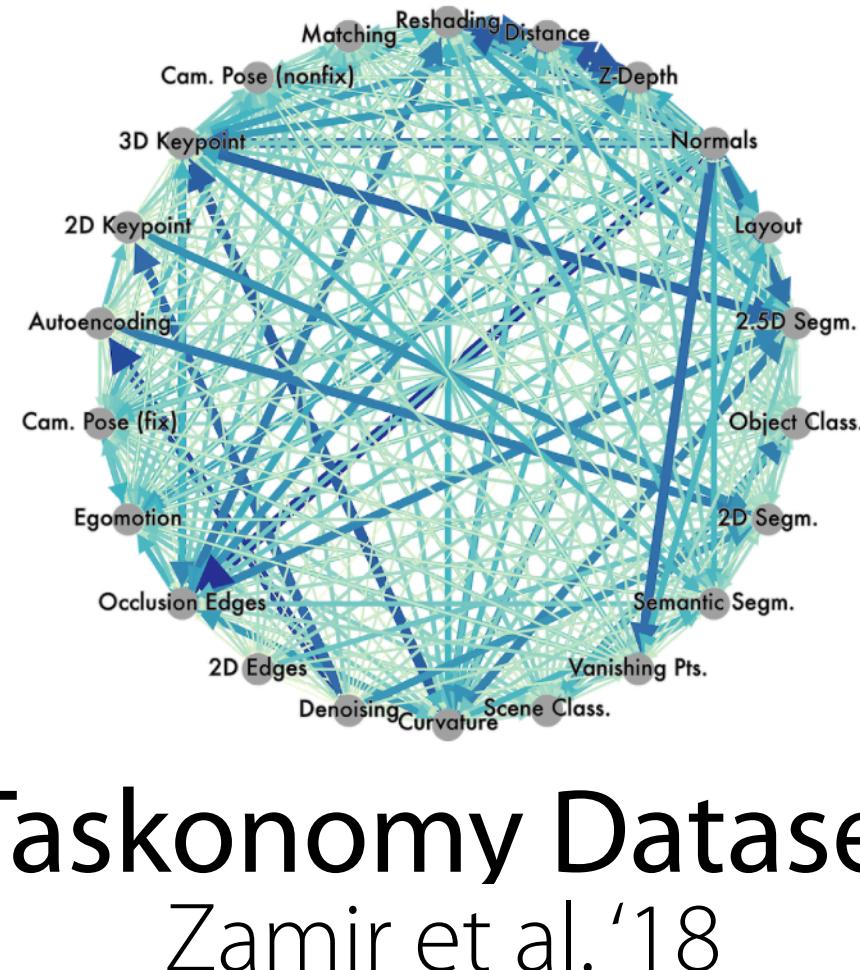
Meta-Dataset  
Triantafillou et al.'19



Meta-World Benchmark  
Yu et al.'19



Visual Task Adaptation Benchmark  
Zhai et al.'19



computer code alignment grammar  
creativity theory of mind non-language  
humor text simplification semantic parsing  
syntax memorization sufficient information truthfulness question generation  
multilingual dialogue system religious bias  
mathematics morphology instructions  
visual reasoning narrative understanding game play  
human-like behavior domain specific social reasoning  
logical reasoning causal reasoning toxicology  
reading comprehension analogical reasoning word sense disambiguation fallacy  
comprehension out of distribution translation intent recognition  
programmatic contextual question-answering social bias  
implicit reasoning paraphrase repeated interaction decomposition  
algorithms paragraph conversational question answering  
algebra self evaluation negation emotional intelligence  
arithmetic probabilistic reasoning figurative language  
low-resource language numerical response gender bias physical reasoning  
mathematical proof self play medicine  
probabilistic reasoning conversational question answering  
numerical response gender bias physical reasoning  
self play medicine

**common sense free response programmatic logical reasoning reading comprehension**

**BIG Bench**  
Srivastava et al.'22

Goal: reflection of real world problems + appropriate level of difficulty + ease of use

# Open Challenges in Multi-Task and Meta Learning

## Addressing fundamental problem assumptions

- Generalization: Out-of-distribution tasks, long-tailed task distributions
- Multimodality: Can you learn priors from multiple modalities of data?
- Algorithm, Model Selection: When will multi-task learning help you?

## Benchmarks

- Breadth: That challenge current algorithms to find common structure
- Realistic: That reflect real-world problems

## Improving core algorithms

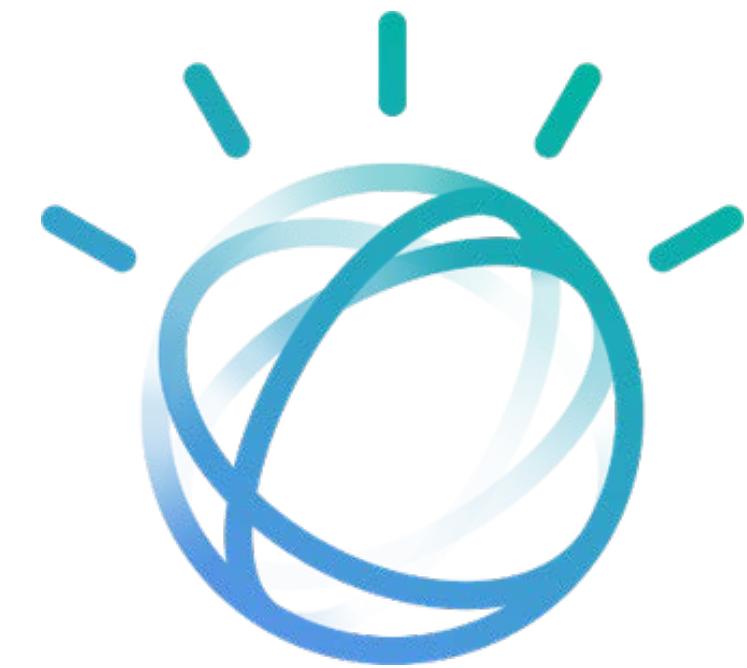
- Computation & Memory: Making large-scale bi-level optimization practical
- Theory: Develop a theoretical understanding of the performance of these algorithms

+ the challenges you discovered in your homework & final projects!

# The Bigger Picture



TD Gammon

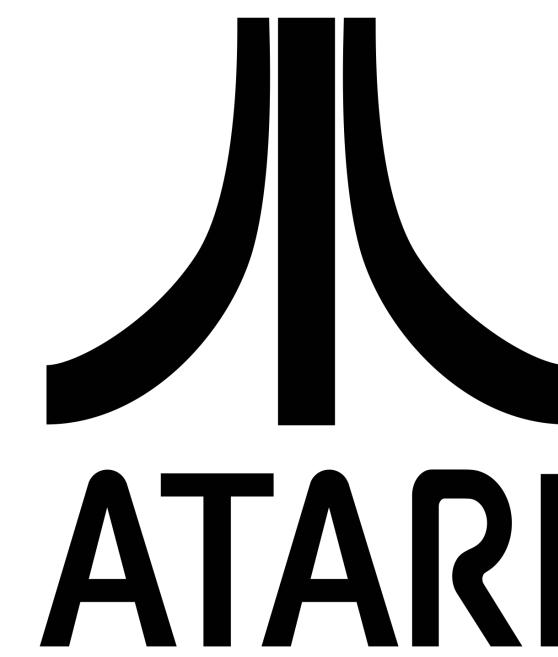


IBM Watson

helicopter acrobatics



machine translation



DQN



Machines are *specialists*.

Watch the following video <https://youtu.be/8vNxjwt2AqY>

Humans are *generalists*.

# Steps Towards Generalists

**Some of what we covered in CS330:**

- learn multiple tasks in a single model (multi-task learning)
- leverage prior experience when learning new things (pre-training, meta-learning)
- leveraging *unlabeled* prior data (contrastive, generative pre-training)
- leveraging data from different domains (domain adaptation & generalization)
- learn continuously (lifelong learning)

**What's missing?**