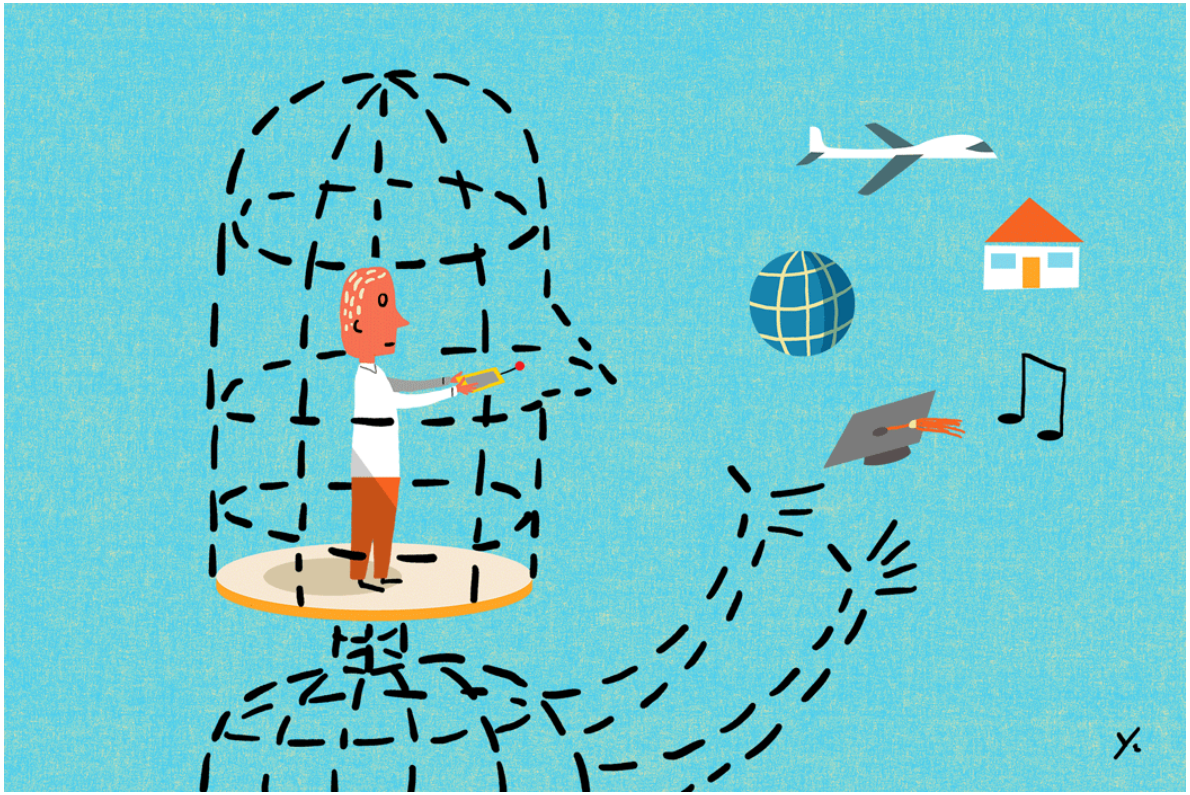
Theoretical Foundations of Pre-trained Models

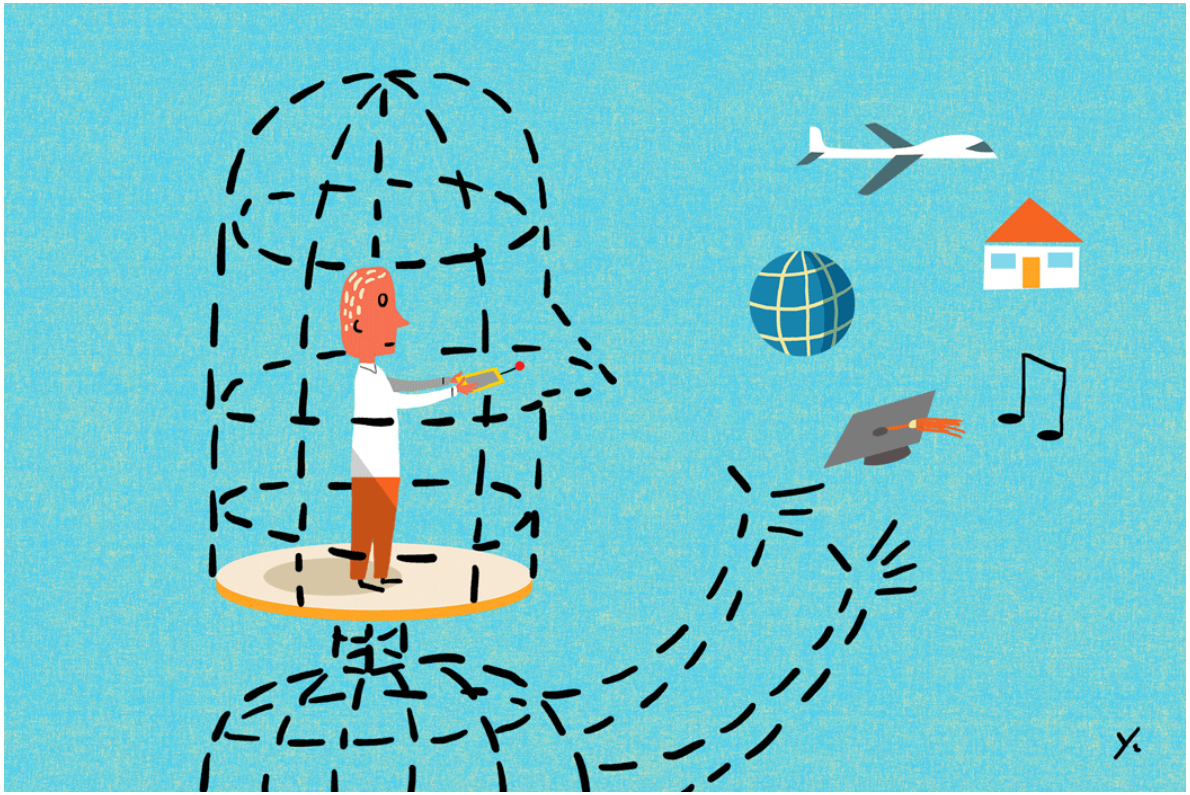
Qi Lei

Princeton University

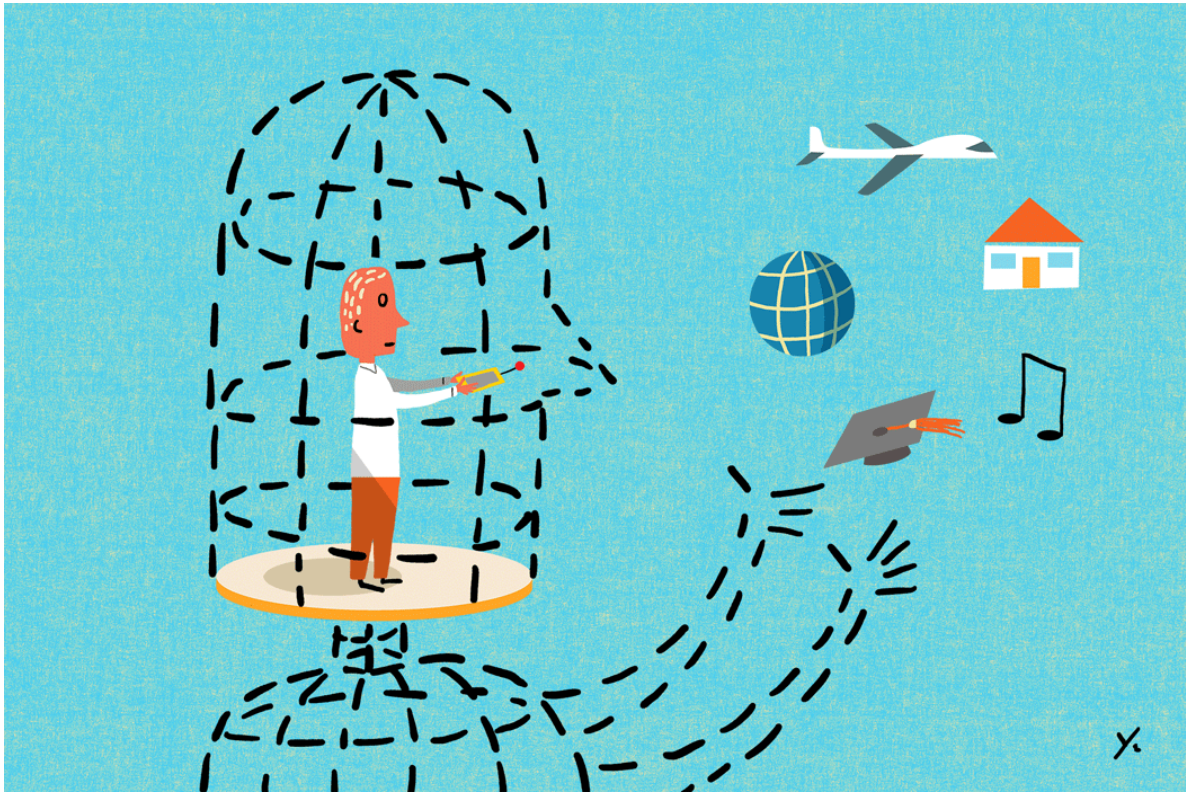
A.I. is Everywhere



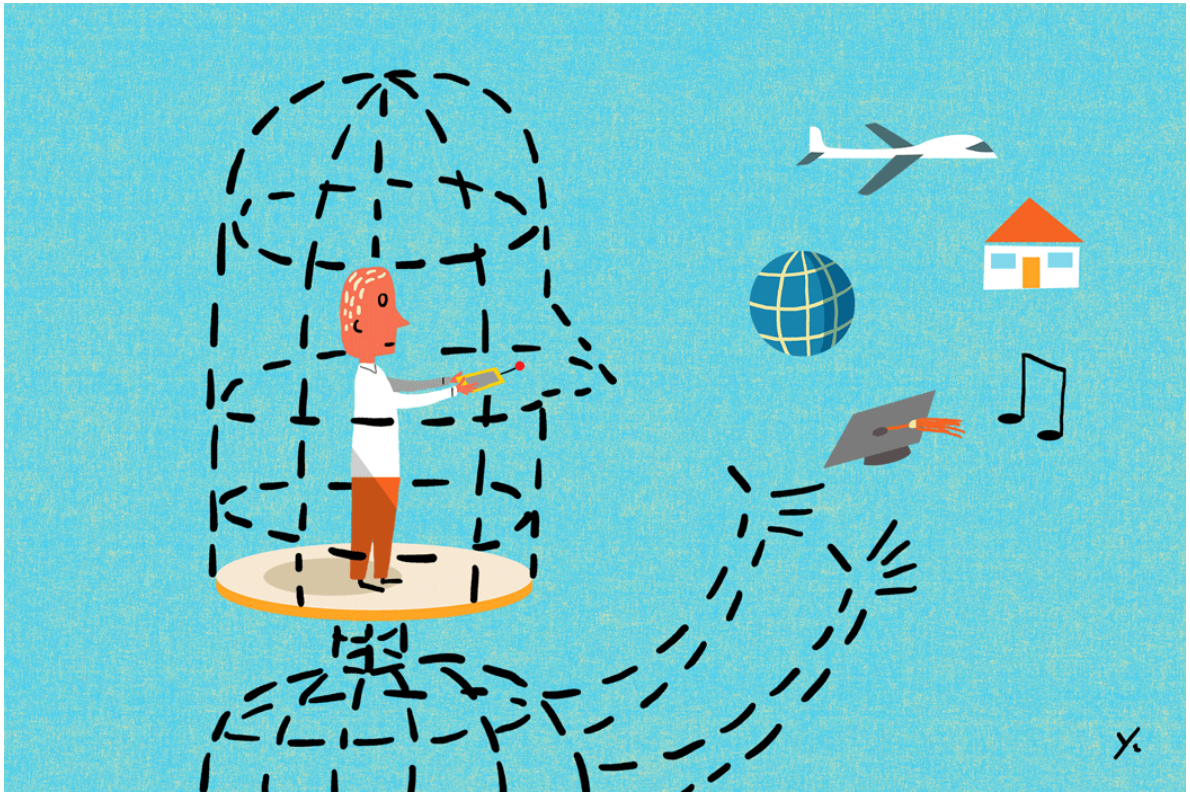
A.I. is Everywhere

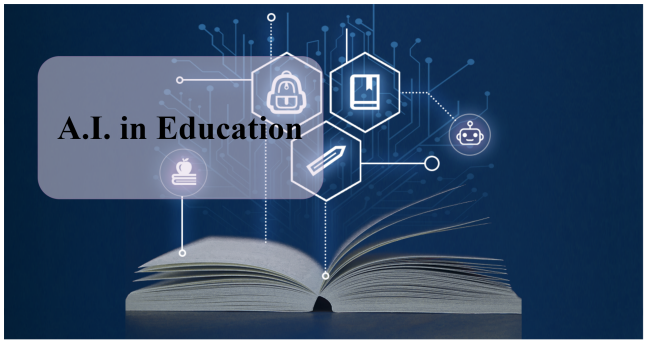


A.I. is Everywhere

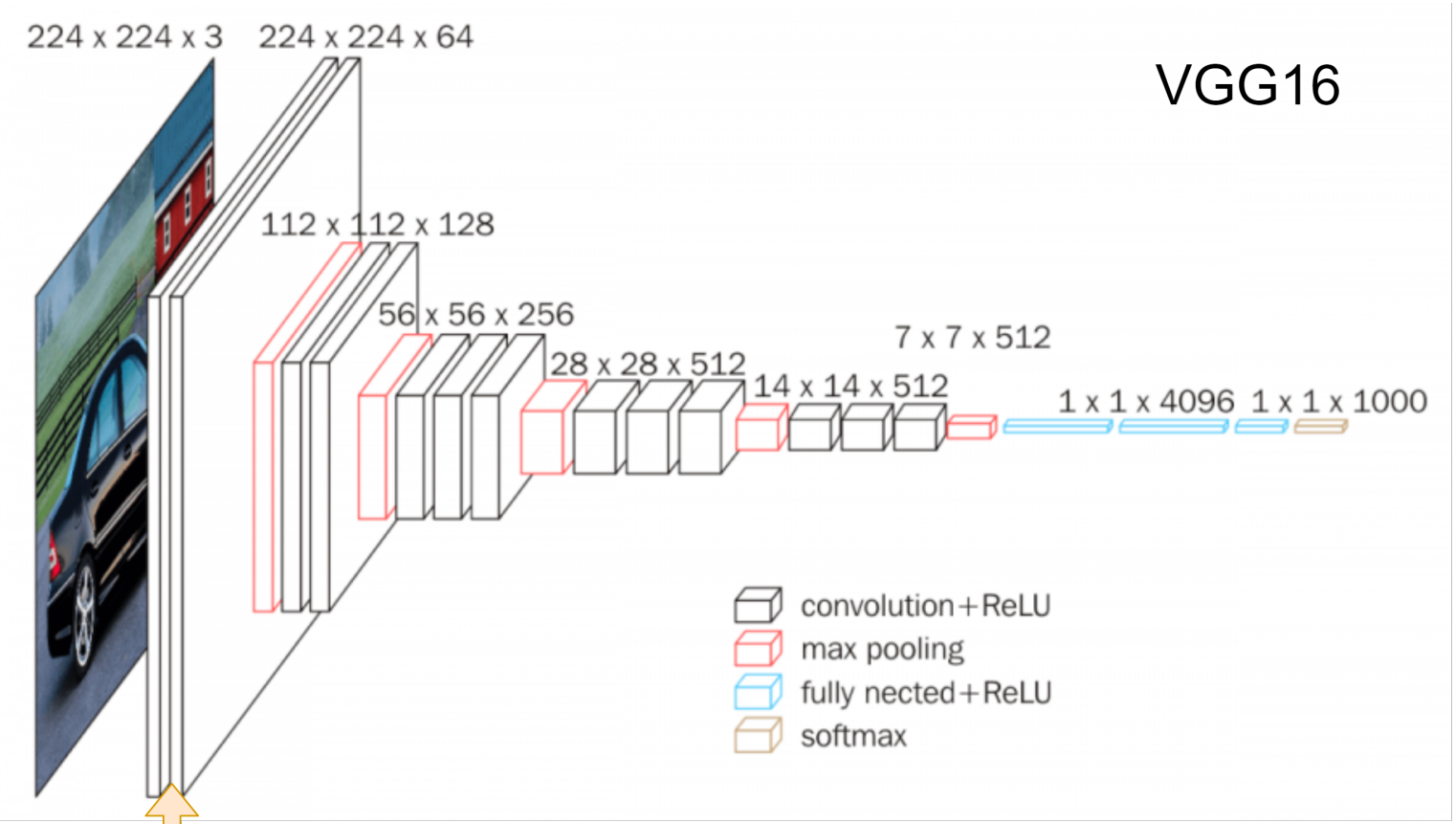


A.I. is Everywhere





Deep Learning

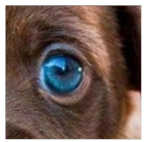


- Deep learning succeeds with abundant labeled data.

Human Learning

extract features

ear shape



eye color

coat color

Pre-trained Models

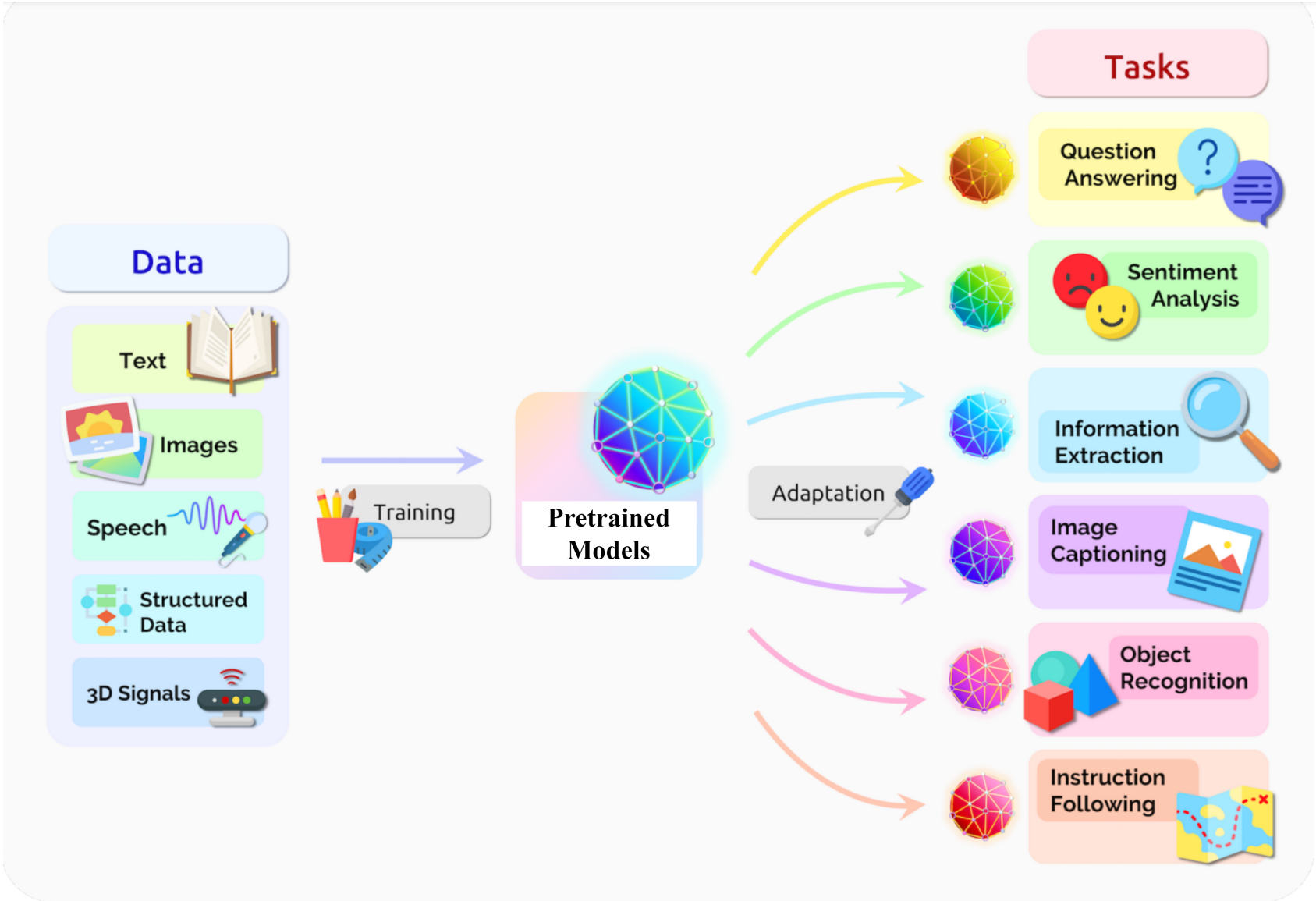
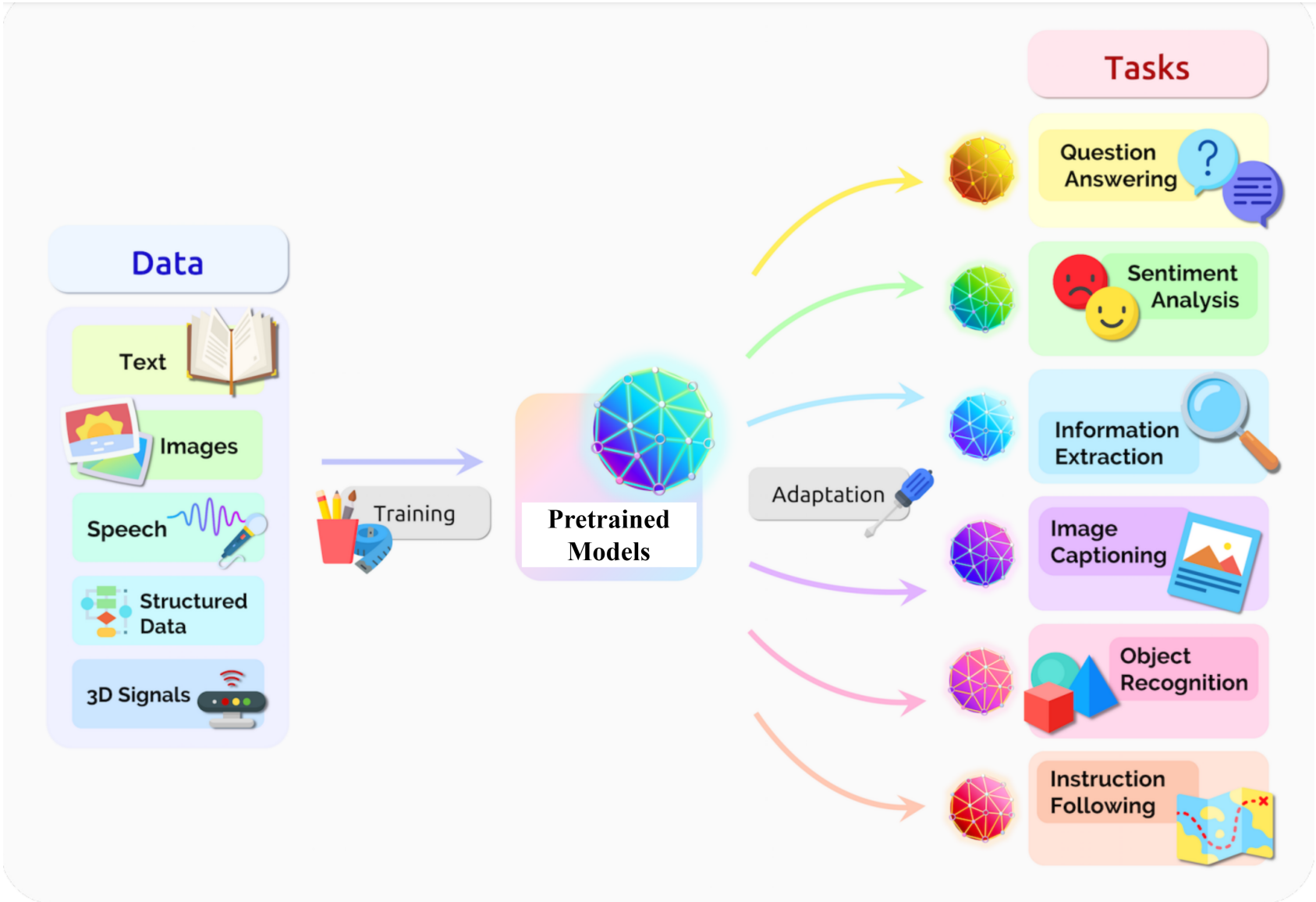


Figure credit to: "On the Opportunities and Risks of Foundation Models", CRFM, HAI, Stanford

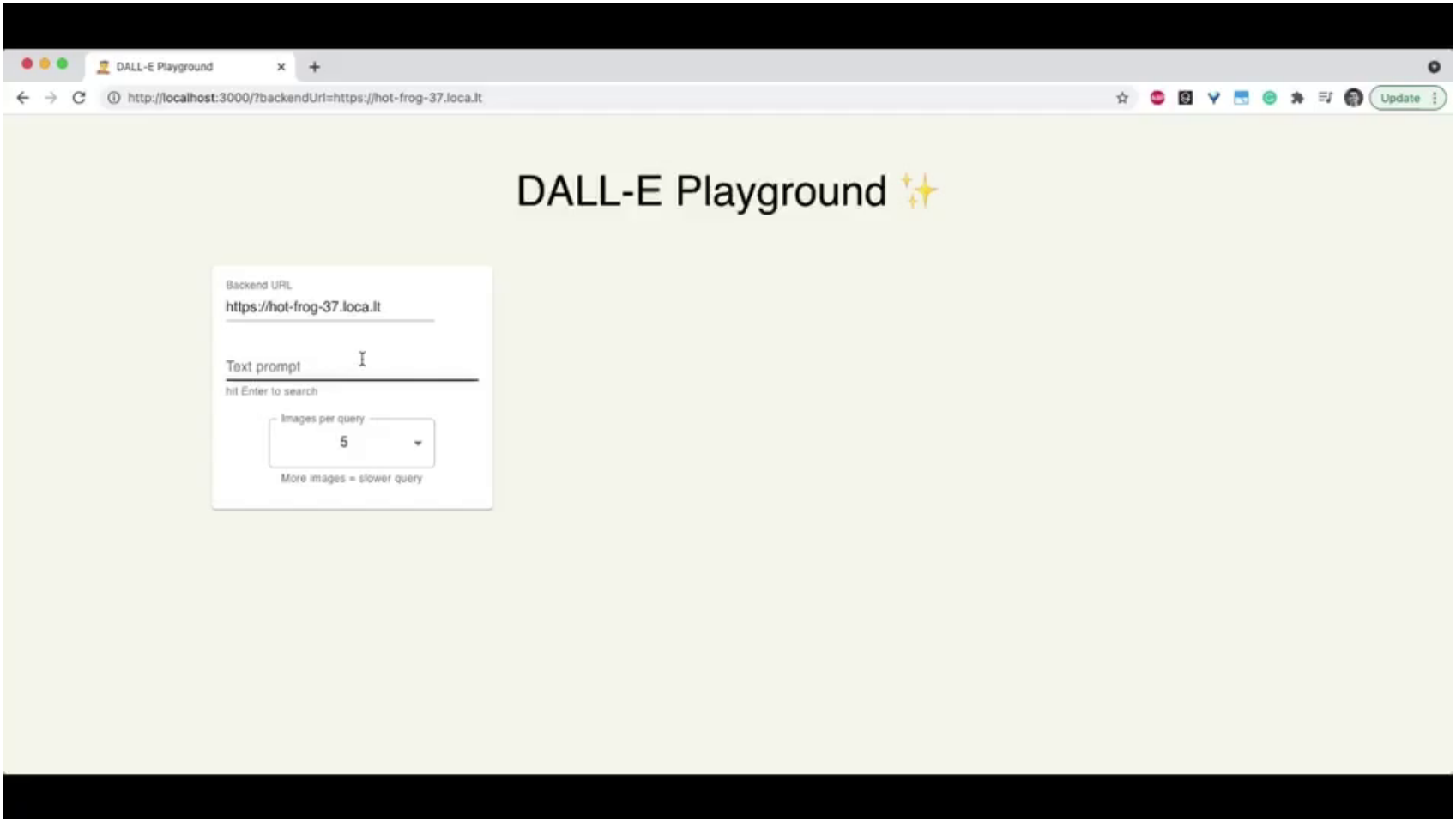
Pre-trained Models



BERT, GPT-3, Codex, SimCLR, MAE, DALL-E, CLIP

Example: DALL-E

DALL-E: create images from text captions



Example: DALL-E

DALL-E: create images from text captions

a pentagonal green clock. a green clock in the shape of a pentagon.

a cube made of porcupine. a cube with the texture of a porcupine.

a collection of glasses is sitting on a table

Figures source: OpenAI DALL-E playground

Example: CLIP

Zero shot image classifier

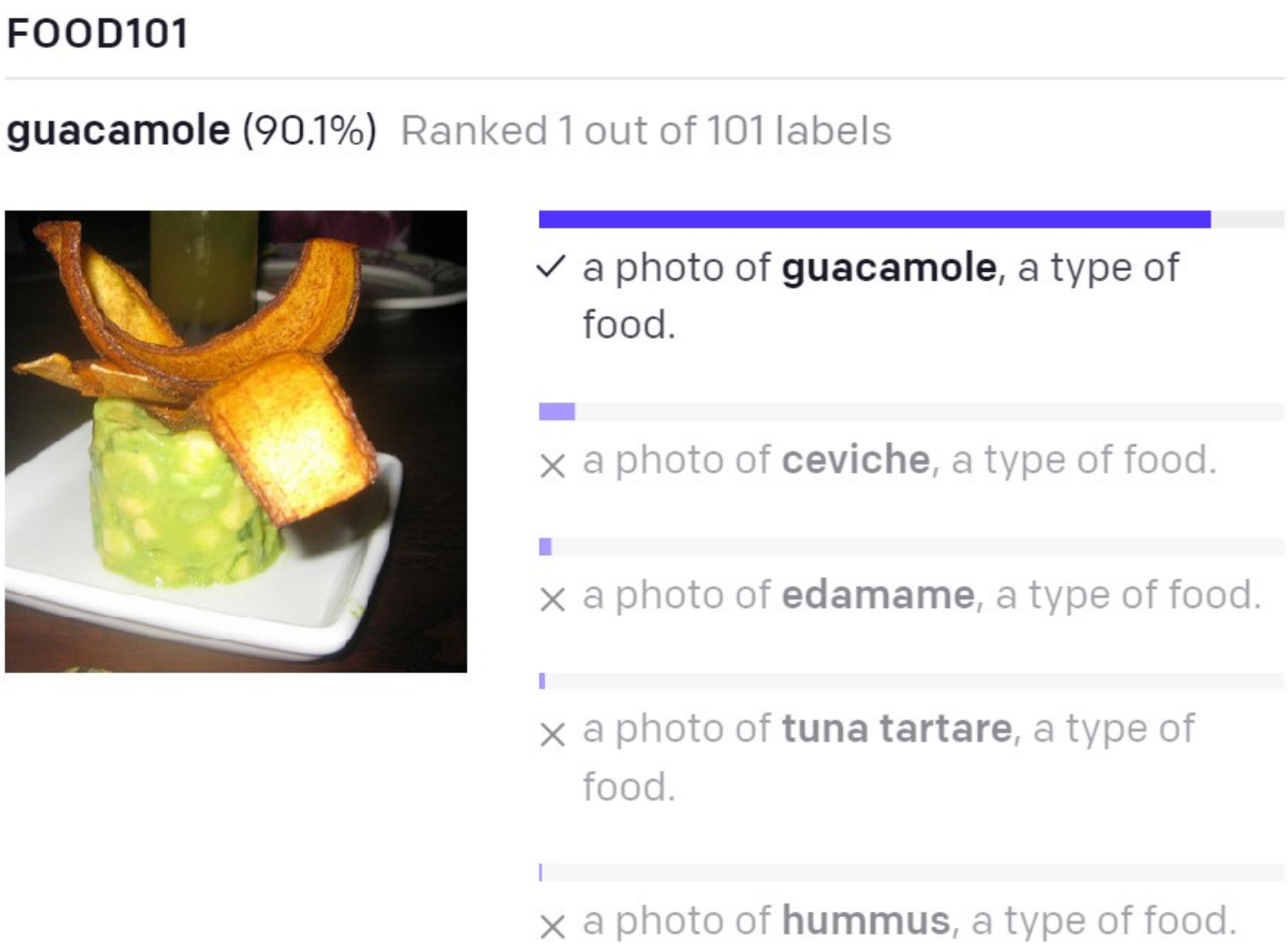
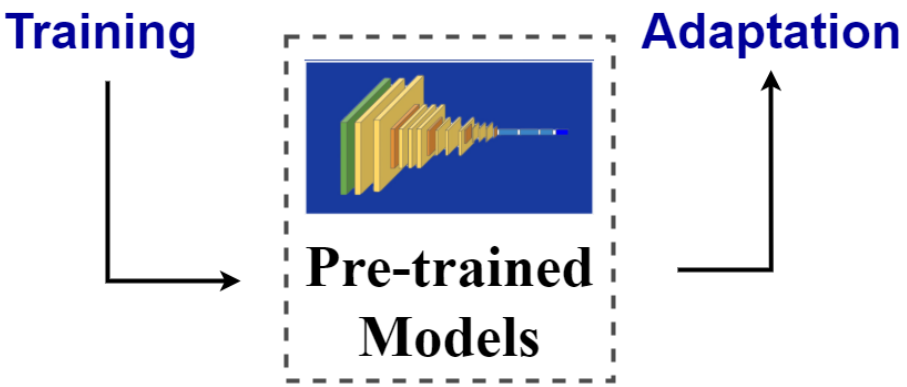


Figure source: OpenAI, `https://clip.backprop.co/`

Behind the Scenes



- What training tasks are useful for downstream tasks?
- What algorithm/architecture can identify the useful features?
- How many samples are required?

- guide technical decisions
- reduce trial and error
- forecast outcomes and risks
- inspire new methods

Challenge

Current learning theory is relatively more mature in **supervised learning**.

- Training and testing data follows the **same distribution**
- Complex model and **big labeled** data

Challenge

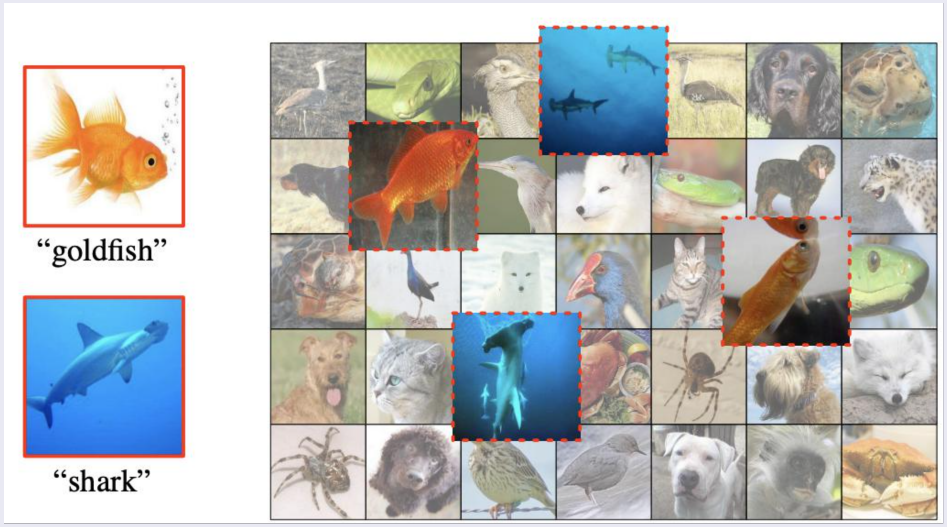
Current learning theory is relatively more mature in **supervised learning**.

- Training and testing data follows the **same distribution**
- Complex model and **big labeled** data

Our target

We want to understand how pre-trained model can

- adapt to new tasks quickly



5-shot learning on ImageNet

Challenge

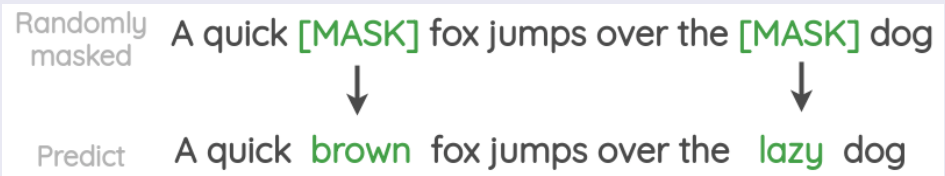
Current learning theory is relatively more mature in **supervised learning**.

- Training and testing data follows the **same distribution**
- Complex model and **big labeled** data

Our target

We want to understand how pre-trained model can

- adapt to new tasks quickly
- be possibly learned from unlabeled samples,



Challenge

Current learning theory is relatively more mature in **supervised learning**.

- Training and testing data follows the **same distribution**
- Complex model and **big labeled** data

Our target

We want to understand how pre-trained model can

- adapt to new tasks quickly
- be possibly learned from unlabeled samples,
- handle distributional shift from training to adaptation

Autonomous driving
Trained on sunny weather

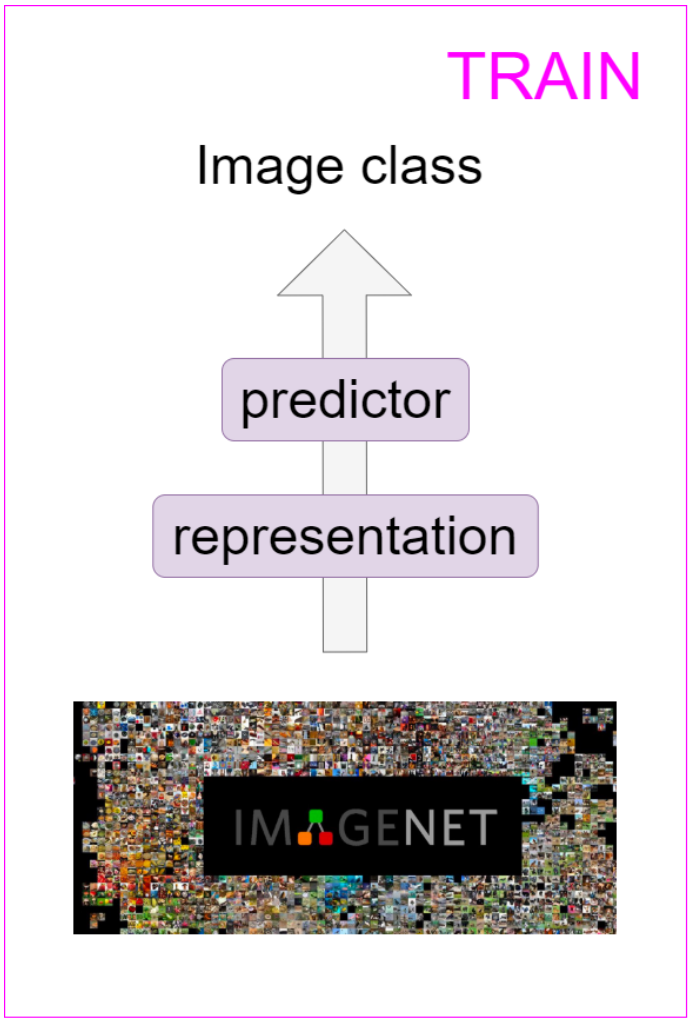
Transfer

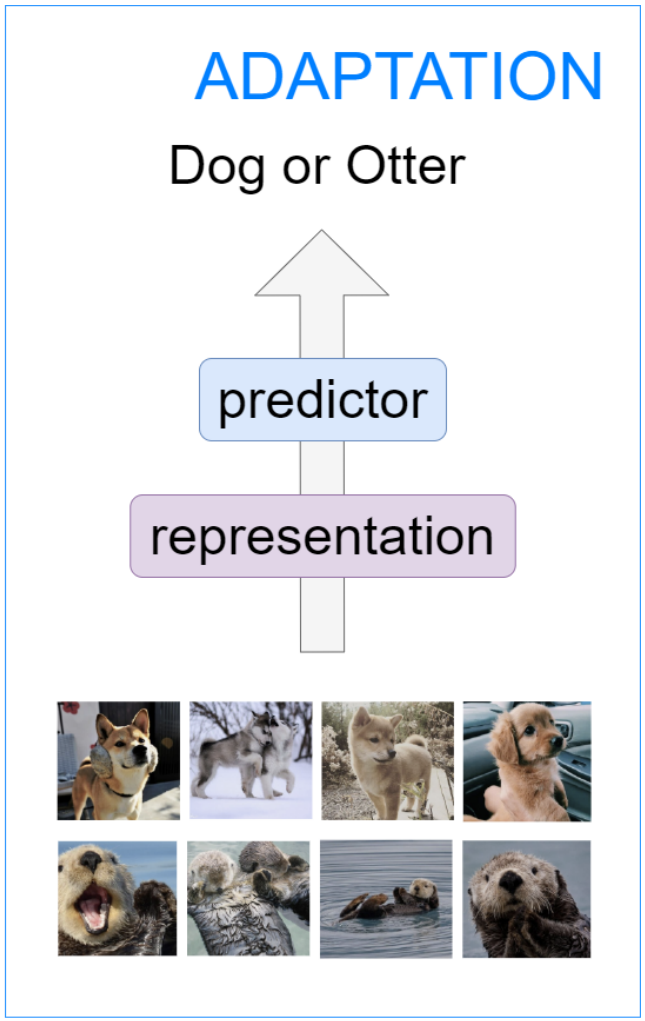
Tested on rainy weather

Outline

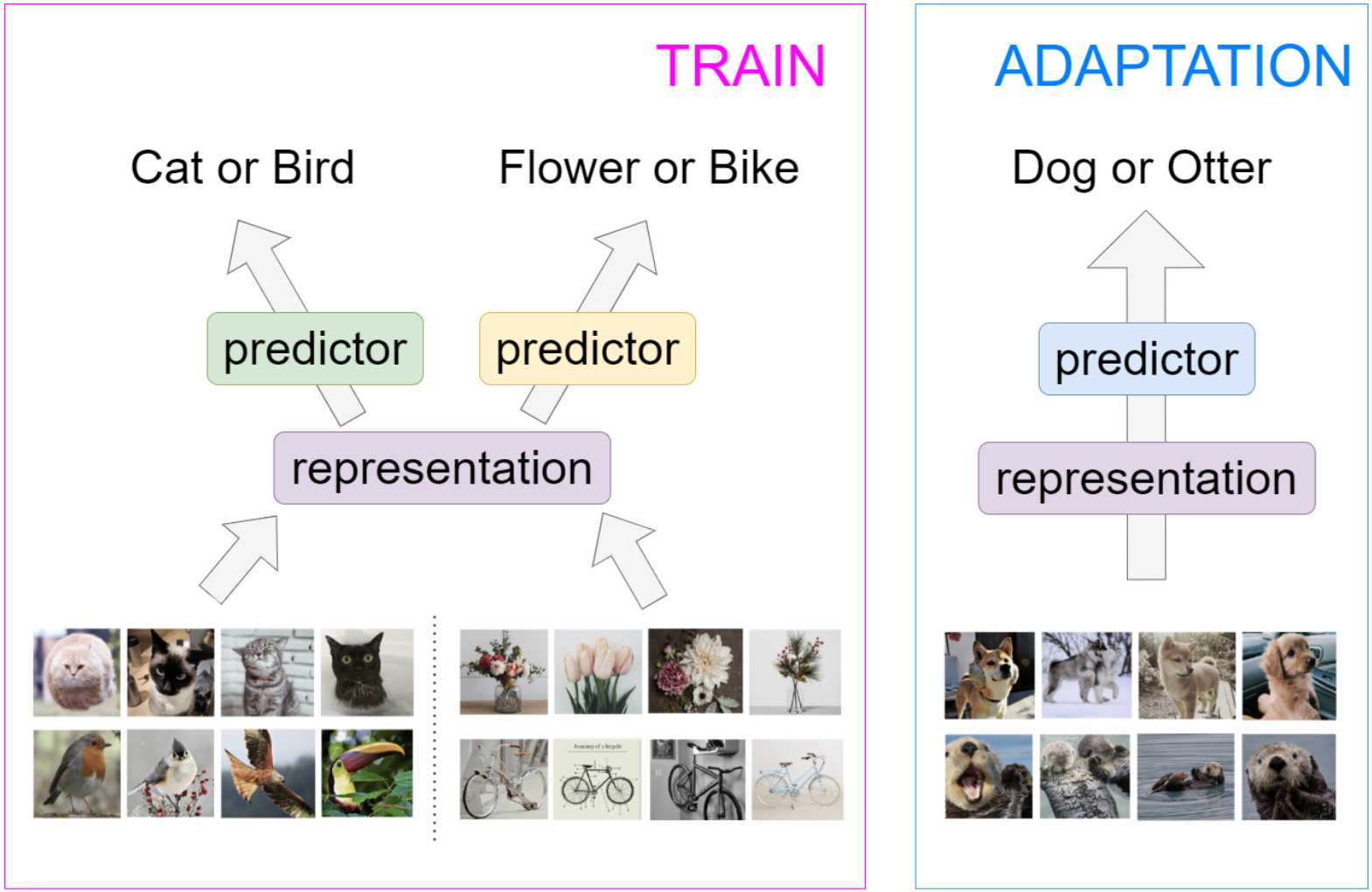
1. Supervised Pre-training and Meta-learning
   - Meta-learning Representation: Frozen Representation
   - Model-agnostic Meta-learning: Fine-tuned Representation
2. Self-Supervised Learning
3. Ongoing and Future Work
   - Domain Adaptation
   - Lifelong Learning
   - Meta Reinforcement Learning

Supervised Pre-training



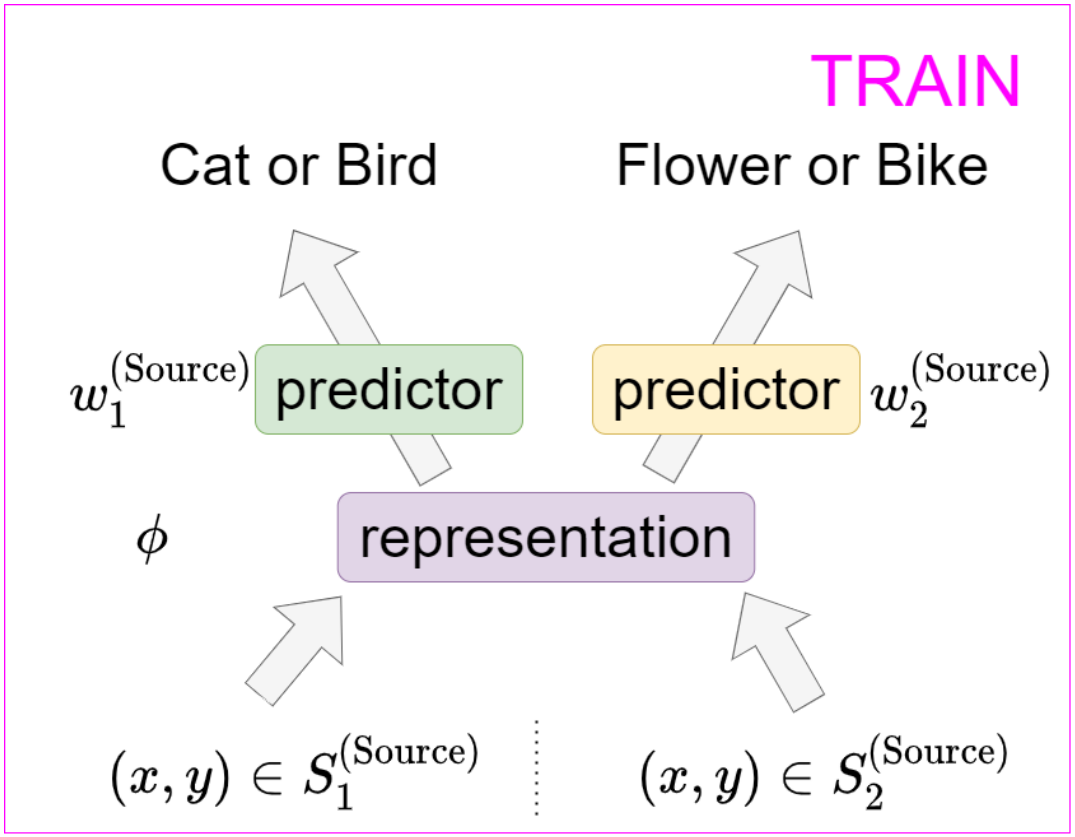


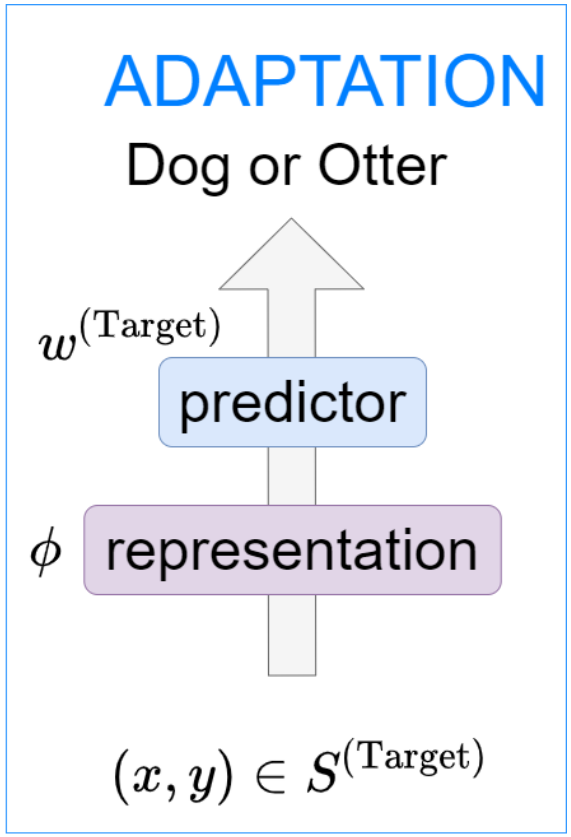
Meta-Learning Representation



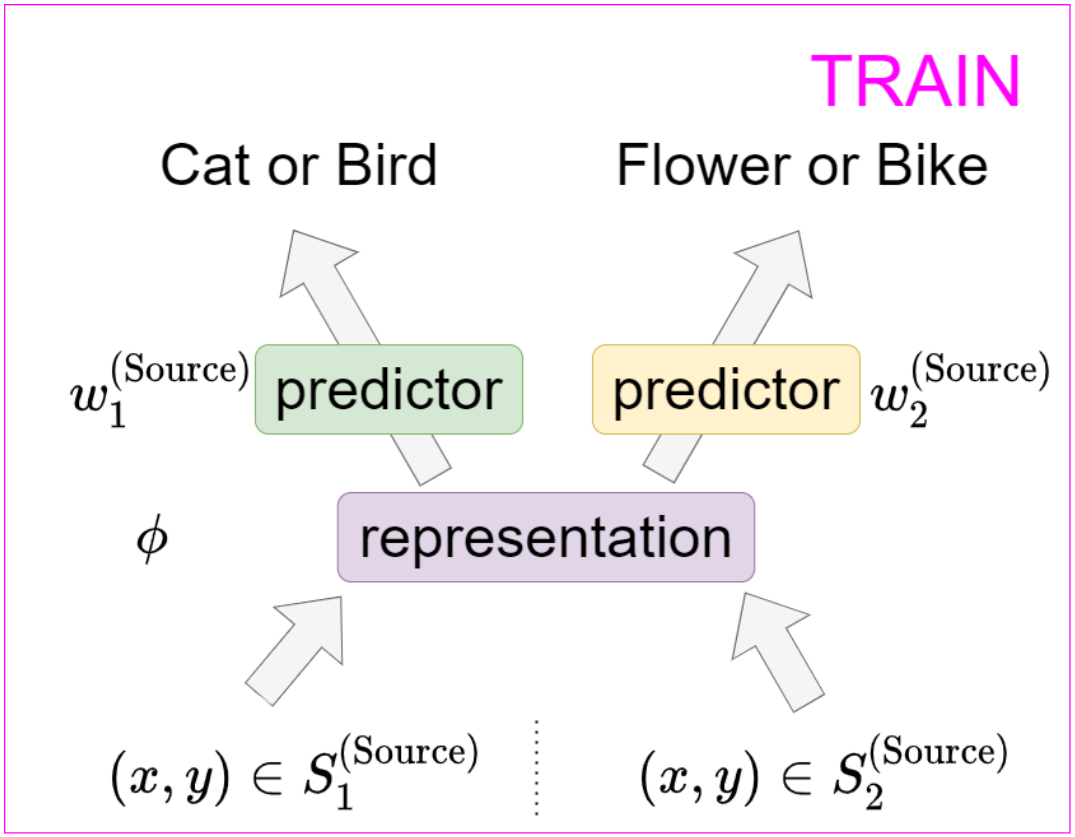
Prototypical network: (Snell et al. 2017), Meta-learning representation: (Javed and White, 2019)

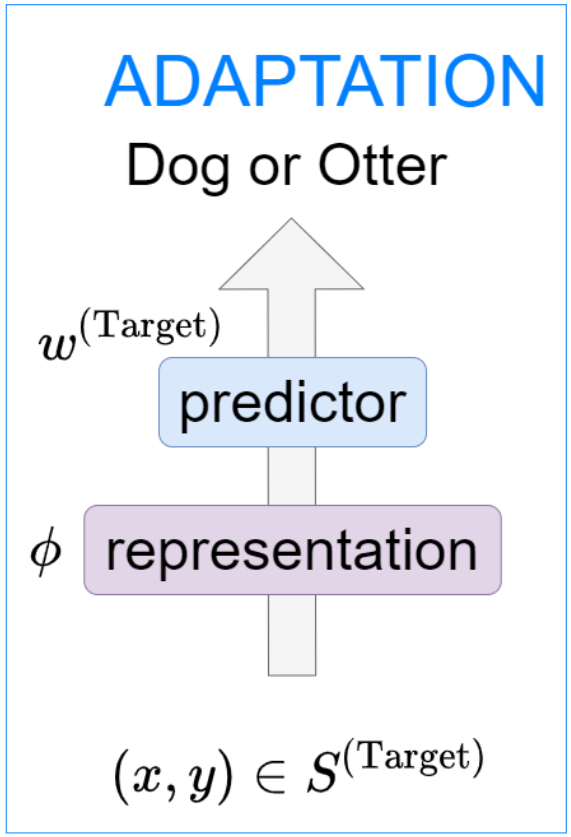
Algorithm


TRAIN
Cat or Bird
Flower or Bike
$w_1^{(\text{Source})}$
predictor
predictor
$w_2^{(\text{Source})}$
ϕ
representation
$(x, y) \in S_1^{(\text{Source})}$
$(x, y) \in S_2^{(\text{Source})}$


ADAPTATION
Dog or Otter
$w^{(\text{Target})}$
predictor
ϕ
representation
$(x, y) \in S^{(\text{Target})}$

Algorithm

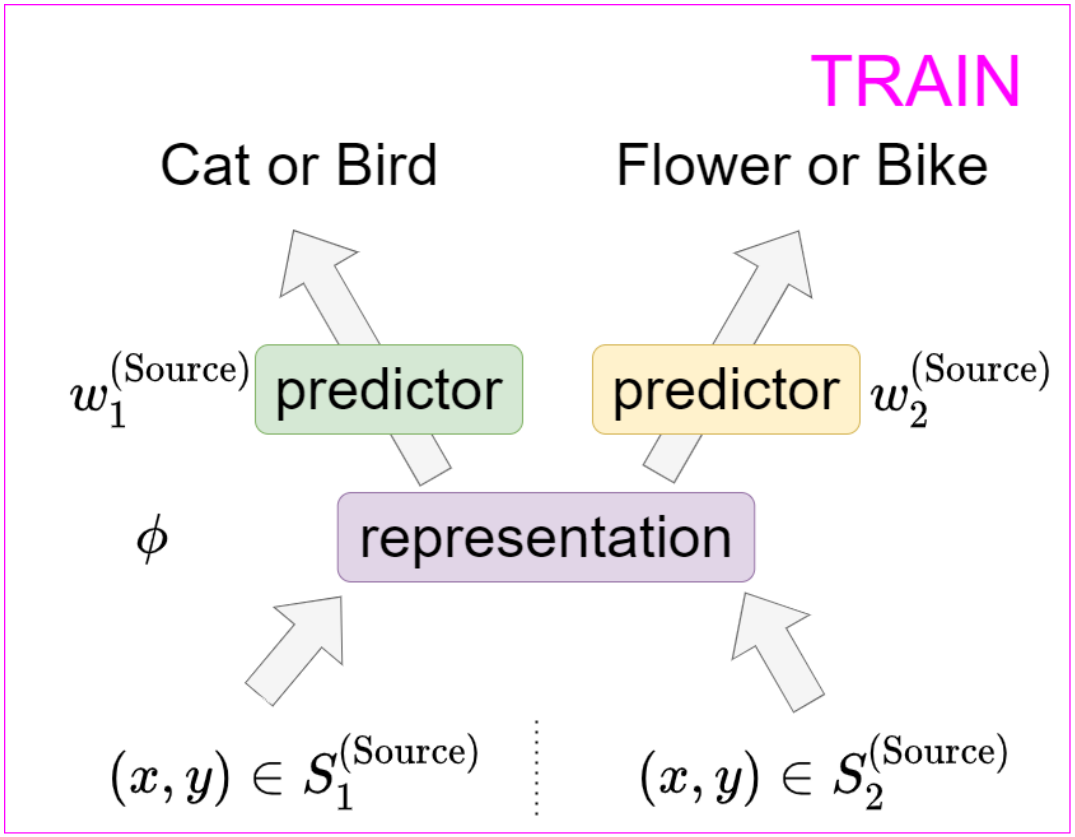


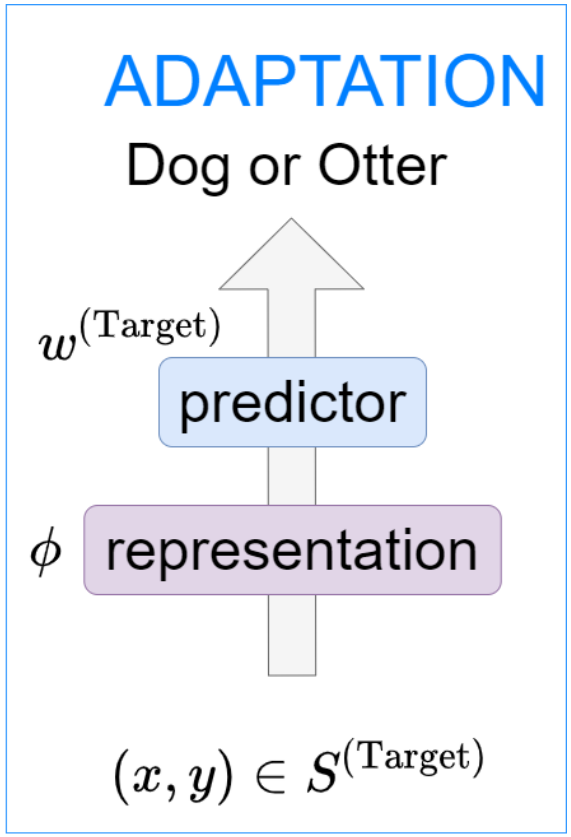


On source tasks:

$$\hat{\phi} \leftarrow \arg\min_{\phi \in \Phi} \sum_{t=1}^{\#\text{source tasks}} \left\{ \min_{w_t} \sum_{(x,y) \in S_t^{(\text{Source})}} \text{loss}\left(y, w_t \circ \phi(x)\right) \right\}.$$

Algorithm





On target task:

$$\hat{w}^{(\text{Target})} \leftarrow \arg\min_{w} \sum_{(x,y) \in S_t^{(\text{Target})}} \text{loss}\left(y, w \circ \hat{\phi}(x)\right).$$

How to Quantify Task Similarities

Assumptions:

1. Shared good representation across tasks:
   There exist predictors w_t, embedding function ϕ,
   $y_t = (w_t)^\top \phi(x)$+noise, $\phi \in \Phi$ for both source and target tasks.
2. Is that enough?

How to Quantify Task Similarities

Assumptions:

1. Shared good representation across tasks:
   There exist predictors w_t, embedding function ϕ,
   $y_t = (w_t)^\top \phi(x)$+noise, $\phi \in \Phi$ for both source and target tasks.
2. Is that enough?

Behind the scenes

1. Shared representation encodes what transfers across the tasks.
2. Need source tasks $\{w_t^{(\text{Source})}\}$ diverse enough to "cover" $w^{(\text{Target})}$.

Importance of Task Diversity

1. Shared representation encodes what transfers across the tasks.
2. Diversity of source tasks $\{w_t^{(\text{Source})}\}$ (at least needs to "cover" the target task.)

Task diversity

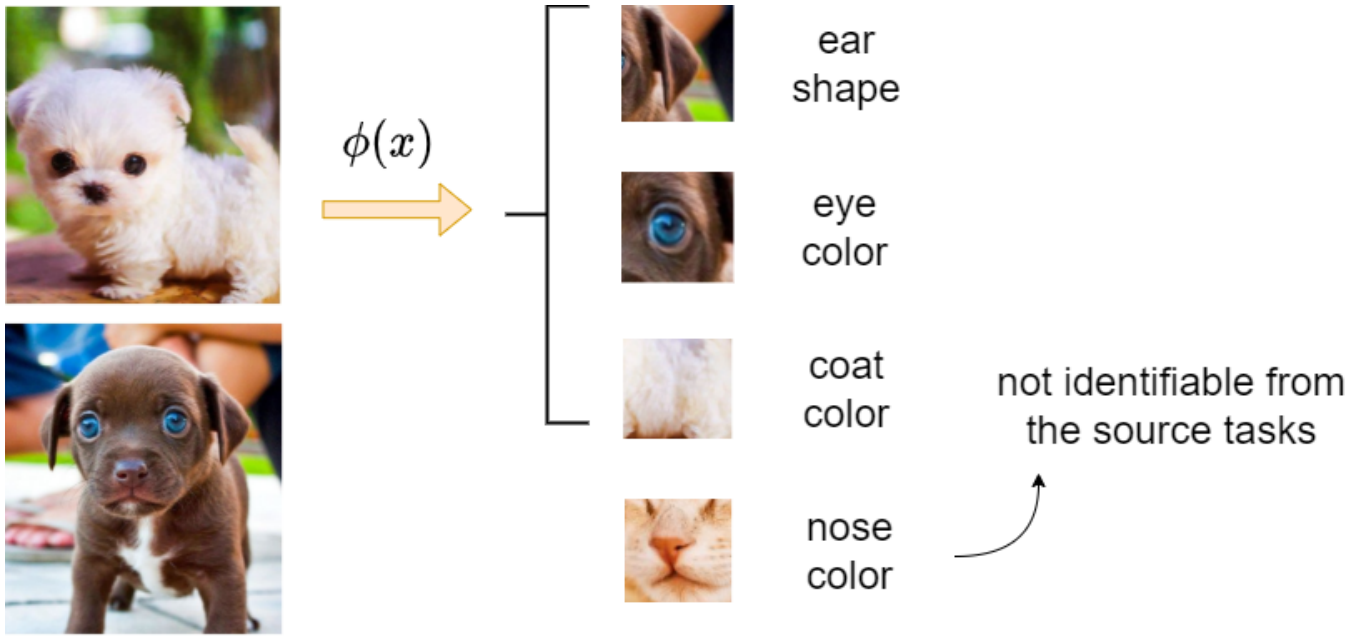
Source tasks:
Classify types of dogs



Target task:
Cat or dog?

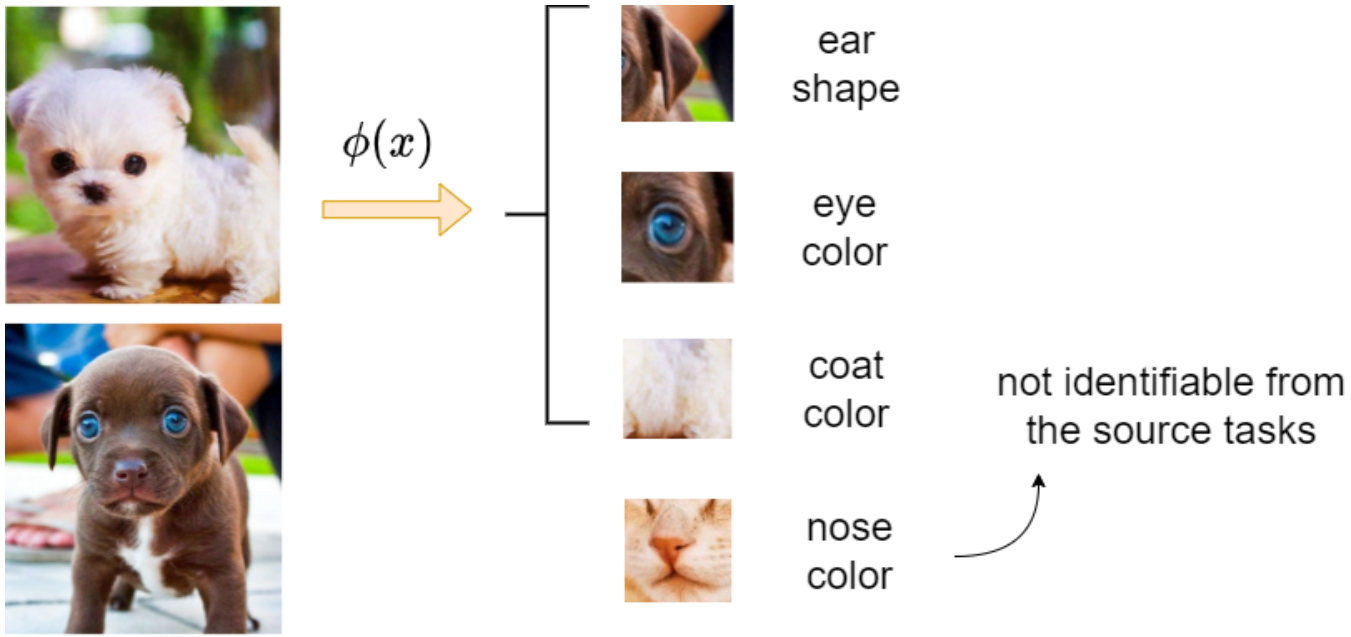

Importance of Task Diversity

1. Shared representation encodes what transfers across the tasks.
2. Diversity of source tasks $\{w_t^{(\text{Source})}\}$ (at least needs to "cover" the target task.)



Importance of Task Diversity

1. Shared representation encodes what transfers across the tasks.
2. Diversity of source tasks $\{w_t^{(\text{Source})}\}$ (at least needs to "cover" the target task.)

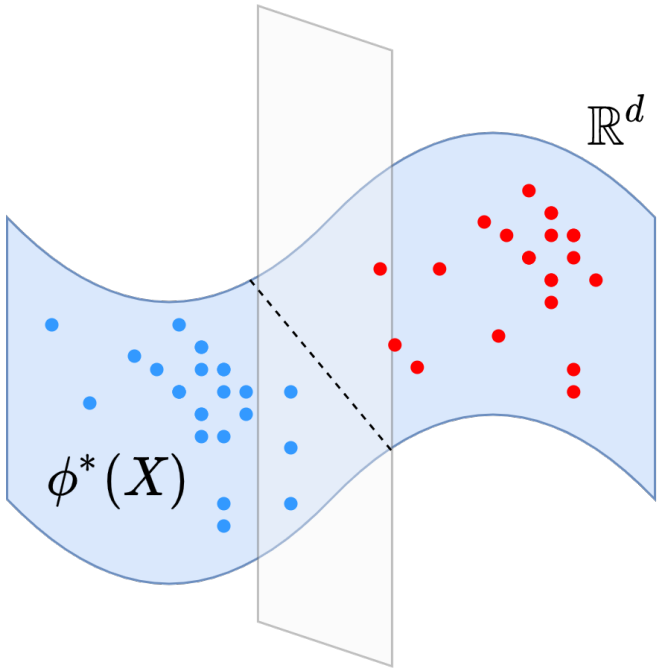


Mathematically speaking, $w^{(\text{Target})} \in \text{span}\{w_1^{(\text{Source})}, \cdots w_{n_e}^{(\text{Source})}\}$.

General Low-dimensional Meta Representation

Setup:

- Shared representation: $y_t = w_t^\top \phi(x_t) + \text{noise}$
- Representation layer is of dimension k (We assume k is small)



Theorem 1 (Informal)

We only need $O(k)$ labeled samples from target domain to get good **test error**.

In contrast, supervised learning requires samples up to the **complexity of the function class**.

Main Result on Meta Representation

Theorem 1

With shared representation and task diversity,
Test Error$(\hat{w}^{(\text{Target})} \circ \hat{\phi}) \leq$ Representation Error + Adaptation Error.

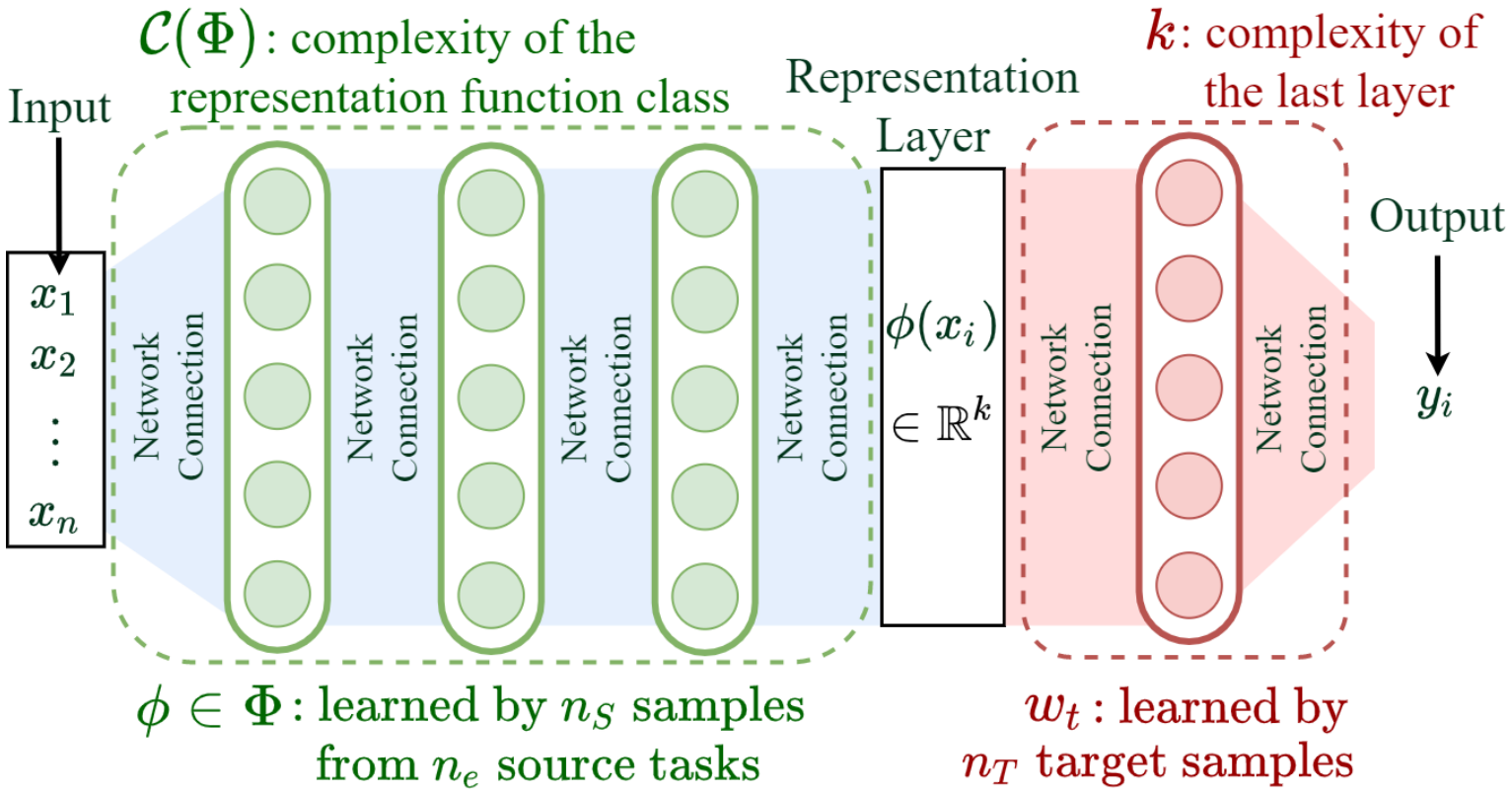
- Representation error: how well you learn representation layer ϕ
- Adaptation error: how well you learn target-task predictor $w^{(\text{Target})}$

Main Result on Meta Representation

Theorem 1

With shared representation and task diversity,

$$\text{Test Error}(\hat{w}^{(\text{Target})} \circ \hat{\phi}) \lesssim \underbrace{\frac{\mathcal{C}(\Phi)}{n_S n_e}}_{\text{representation error}} + \underbrace{\frac{k}{n_T}}_{\text{adaptation error}} .$$



Main Result on Meta Representation

Theorem 1

With shared representation and task diversity,

$$\text{Test Error}(\hat{w}^{(\text{Target})} \circ \hat{\phi}) \lesssim \underbrace{\frac{\mathcal{C}(\Phi)}{n_S n_e}}_{\text{representation error}} + \underbrace{\frac{k}{n_T}}_{\text{adaptation error}}.$$

Baselines:

- Supervised learning:

$$\text{Test error} \leq \frac{\mathcal{C}(w \circ \Phi)}{n_T}.$$

- (Baxter et al. 2014):

$$\text{Test error} \leq \frac{\mathcal{C}(\Phi)}{\sqrt{n_e}} + \frac{k}{n_T}.$$

Main Result on Meta Representation

Theorem 1

With shared representation and task diversity,

$$\text{Test Error}(\hat{w}^{(\text{Target})} \circ \hat{\phi}) \lesssim \underbrace{\frac{\mathcal{C}(\Phi)}{n_S n_e}}_{\text{representation error}} + \underbrace{\frac{k}{n_T}}_{\text{adaptation error}}.$$

Remark:

- Covariate shift is allowed: source and target data can come from different marginal distribution.
- Representation ϕ selects k most important features from the data.

Main Result on Meta Representation

Theorem 1

With shared representation and task diversity,

$$\text{Test Error}(\hat{w}^{(\text{Target})} \circ \hat{\phi}) \lesssim \underbrace{\frac{\mathcal{C}(\Phi)}{n_S n_e}}_{\text{representation error}} + \underbrace{\frac{k}{n_T}}_{\text{adaptation error}}.$$

Remark:

- Covariate shift is allowed: source and target data can come from different marginal distribution.
- Representation ϕ selects k most important features from the data.

Few-shot learning via learning the representation, provably. ICLR 2021

Outline

No Shared Representation?

- What if there is small misspecification in the representation? (Namely, the representation is only approximately shared.)

No Shared Representation?

- What if there is small misspecification in the representation? (Namely, the representation is only approximately shared.)
- Previously: $y_t = w_t^\top \phi(x) +$ noise.

No Shared Representation?

- What if there is small misspecification in the representation? (Namely, the representation is only approximately shared.)
- Previously: $y_t = w_t^\top \phi(x)$+noise.
- Now: $y_t = w_t^\top \phi_t(x)$ + noise, ϕ_t is δ_0-close to ϕ. δ_0 is small.

No Shared Representation?

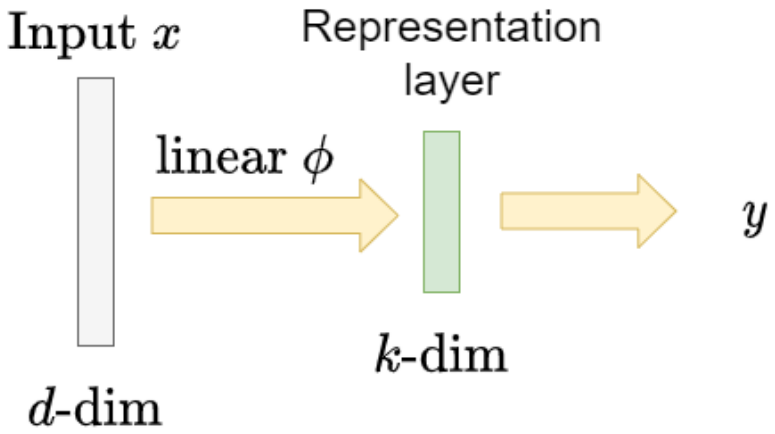
- What if there is small misspecification in the representation? (Namely, the representation is only approximately shared.)
- Previously: $y_t = w_t^\top \phi(x)$+noise.
- Now: $y_t = w_t^\top \phi_t(x)$ + noise, ϕ_t is δ_0-close to ϕ. δ_0 is small.

Question

Does MetaRep (previous algorithm) still work?
If not, how shall we adjust the algorithm?

Failure with Previous Algorithm: MetaRep

Instantiation with linear setting:



When $\delta_0 = 0$ (no misspecification), MetaRep requires at most $O(k)$ samples on the target task.

Failure with Previous Algorithm: MetaRep

Instantiation with linear setting:

When $\delta_0 = 0$ (no misspecification), MetaRep requires at most $O(k)$ samples on the target task.

However, when $\delta_0 > 0$ MetaRep requires at least $\Omega(d)$ samples on the target task

Remark: no improvement over traditional supervised learning that requires $O(d)$ samples.

Modified Algorithm: FINE-TUNEDREP

1. Use source tasks to find ϕ as an initialization.
2. Fine-tune each representation ϕ_t slightly from ϕ that tolerates small mis-specification.

Model-agnostic Meta-learning: Finn et al. 2017

Modified Algorithm: FINE-TUNEDREP

1. Use source tasks to find ϕ as an initialization.
2. Fine-tune each representation ϕ_t slightly from ϕ that tolerates small mis-specification.

$$\min_{\phi} \min_{\|\phi_t - \phi\| \le \delta_0, w_t} \sum_{\text{Task } t} \text{loss}(w_t, \phi_t),$$

Model-agnostic Meta-learning: Finn et al. 2017

Modified Algorithm: FINE-TUNEDREP

1. Use source tasks to find ϕ as an initialization.
2. Fine-tune each representation ϕ_t slightly from ϕ that tolerates small mis-specification.

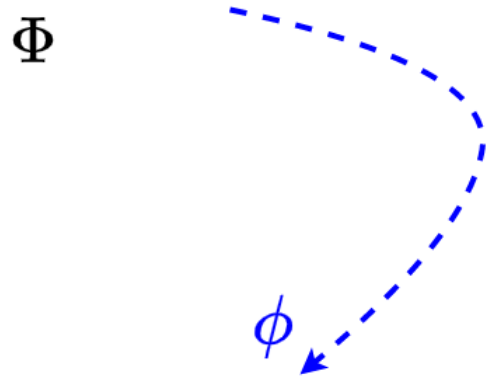
$$\Phi$$

$$\min_{\phi} \min_{\|\phi_t - \phi\| \le \delta_0, w_t} \sum_{\text{Task } t} \text{loss}(w_t, \phi_t),$$

Model-agnostic Meta-learning: Finn et al. 2017

Modified Algorithm: FINE-TUNEDREP

1. Use source tasks to find ϕ as an initialization.
2. Fine-tune each representation ϕ_t slightly from ϕ that tolerates small mis-specification.



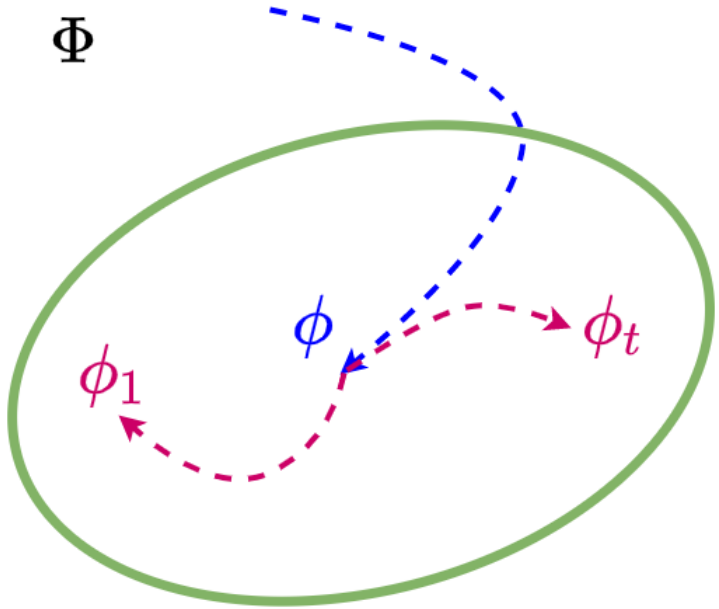
$$\min_{\phi} \min_{\|\phi_t - \phi\| \le \delta_0, w_t} \sum_{\text{Task } t} \text{loss}(w_t, \phi_t),$$

Model-agnostic Meta-learning: Finn et al. 2017

Modified Algorithm: FINE-TUNEDREP

1. Use source tasks to find ϕ as an initialization.
2. Fine-tune each representation ϕ_t slightly from ϕ that tolerates small mis-specification.

$$\min_{\phi} \min_{\|\phi_t - \phi\| \le \delta_0, w_t} \sum_{\text{Task } t} \text{loss}(w_t, \phi_t),$$

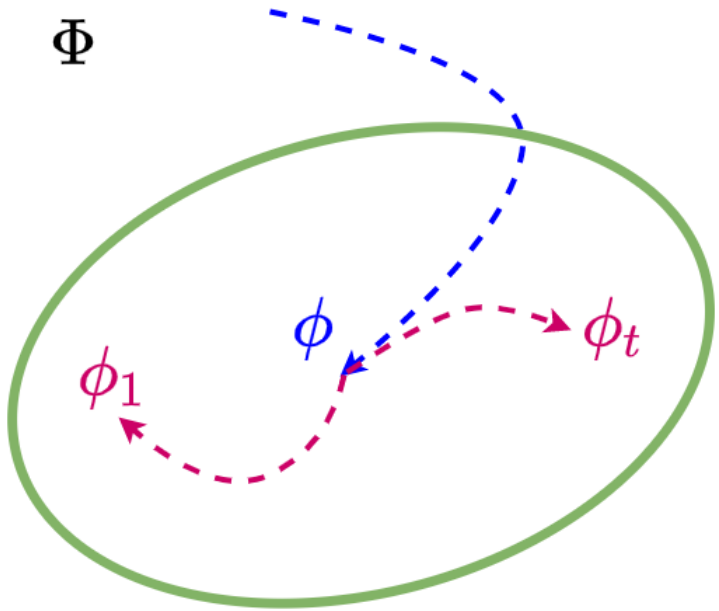


Model-agnostic Meta-learning: Finn et al. 2017

Modified Algorithm: FINE-TUNEDREP

1. Use source tasks to find ϕ as an initialization.
2. Fine-tune each representation ϕ_t slightly from ϕ that tolerates small mis-specification.

$$\min_{\phi} \min_{\|\phi_t - \phi\| \le \delta_0, w_t} \sum_{\text{Task } t} \text{loss}(w_t, \phi_t),$$



Theorem 2 (Special case on linear setting, informal)

When adapting ϕ to target task, it requires $O(k) + O(\delta_0^2)$ training samples from target domain.

Main Result on Fine-tuned Representation

Theorem 2

For general function classes, under similar settings,

$$\text{Test Error} \lesssim \text{Previous Error (when } \delta_0 = 0) + O(\frac{\delta_0}{\sqrt{n_T}}).$$

- δ_0 measures mis-specification in representation ϕ
- n_T: number of samples from target training set

How Fine-Tuning Allows for Effective Meta-Learning, NeurIPS 2021

Main Result on Fine-tuned Representation

Theorem 2

For general function classes, under similar settings,

$$\text{Test Error} \lesssim \text{Previous Error (when } \delta_0 = 0) + O(\frac{\delta_0}{\sqrt{n_T}}).$$

- We need $O(k) + O(\delta_0^2)$ samples from target domain.

Baselines:

- Supervised learning and previous method needs $\mathcal{C}(w \circ \Phi)$ samples from target domain.
- $\mathcal{C}(w \circ \Phi)$: Complexity of the function class for the whole network.

How Fine-Tuning Allows for Effective Meta-Learning, NeurIPS 2021

No labeled data?

Create your own labels

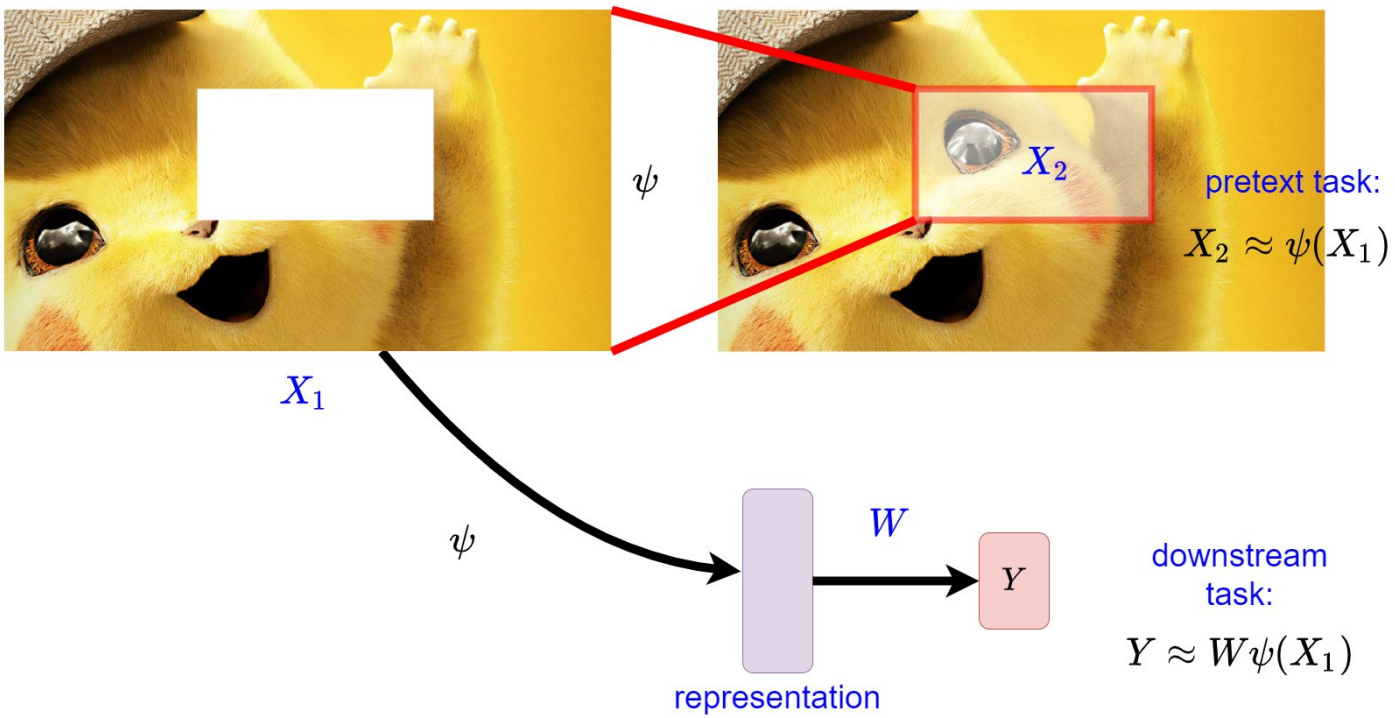
Supervised representation learning needs labels from related tasks. What if this isn't available?

Create pseudo-labels from the input data.

Outline

Type I: reconstruction-based SSL

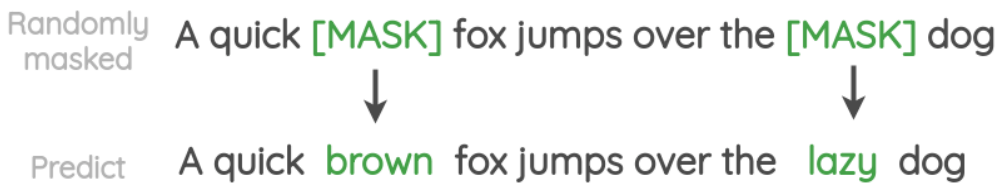
Reconstructing part of the input from the other part



Context encoder: (Pathak et al. 2016)

Type I: reconstruction-based SSL

Reconstructing part of the input from the other part



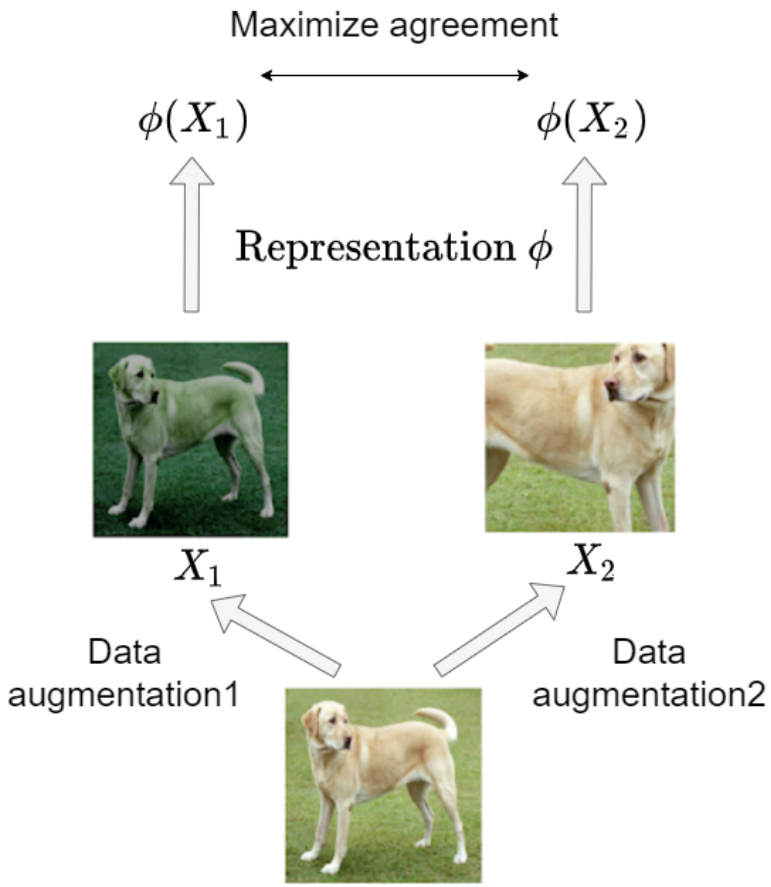
[1]

[1] BERT: (Devlin et al., 2018)

Other examples: Masked Autoencoder: (He et al., 2021), Colorization: (Zhang et al., 2016)

Type II: similarity-based SSL

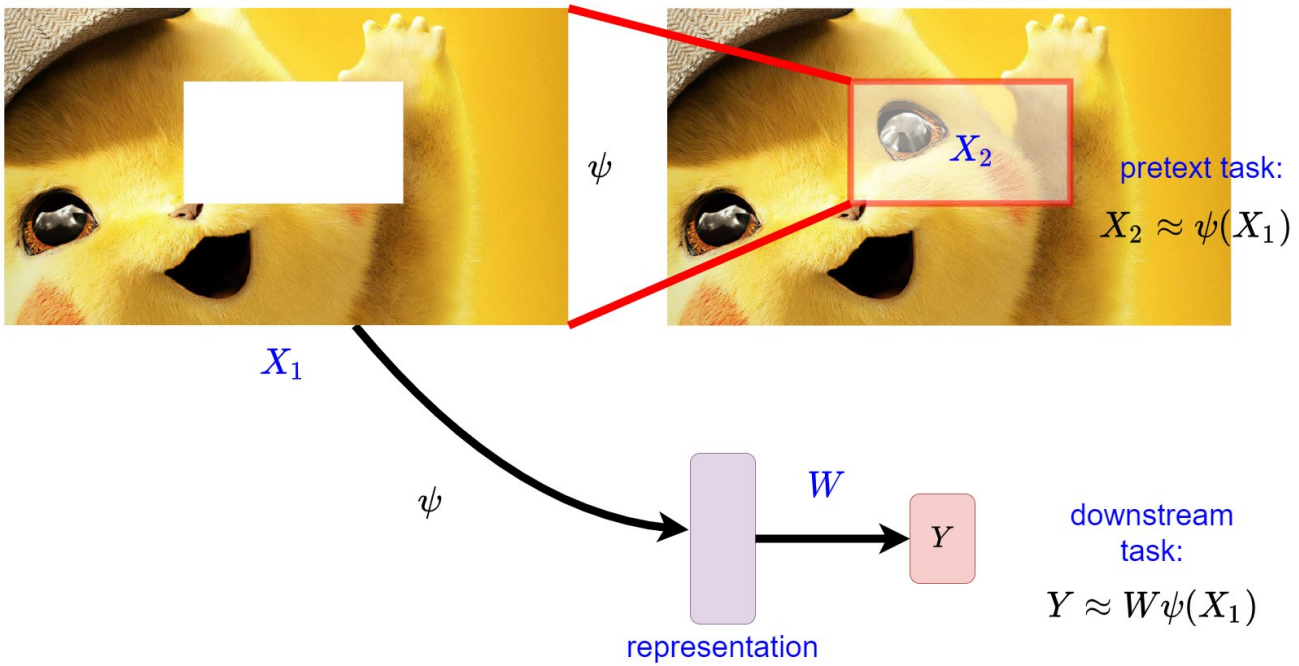
Enforcing two views of the same data to have similar representation



Examples: SimSiam: (Chen et al., 2021), CLIP: (Radford et al., 2021) , SimCLR: (Chen et al., 2020)

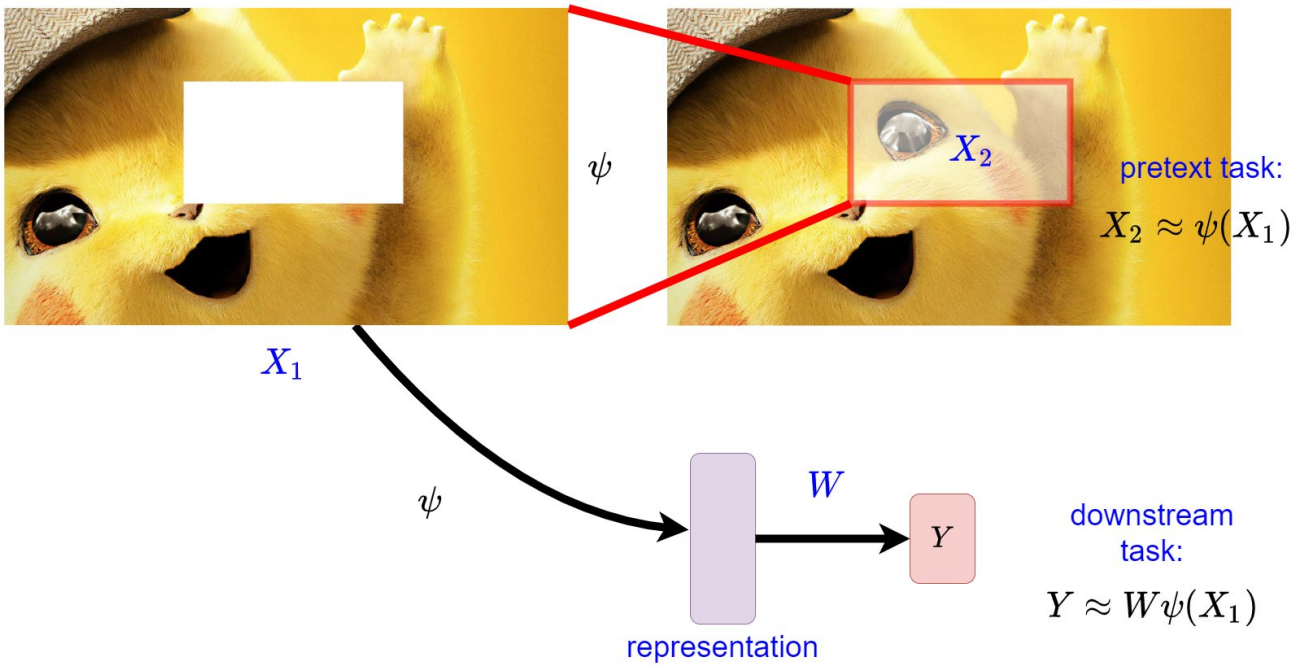
Ideal Scenario

1. Label Y with k classes.
2. Unmasked image X_1 and masked image X_2.



Ideal Scenario

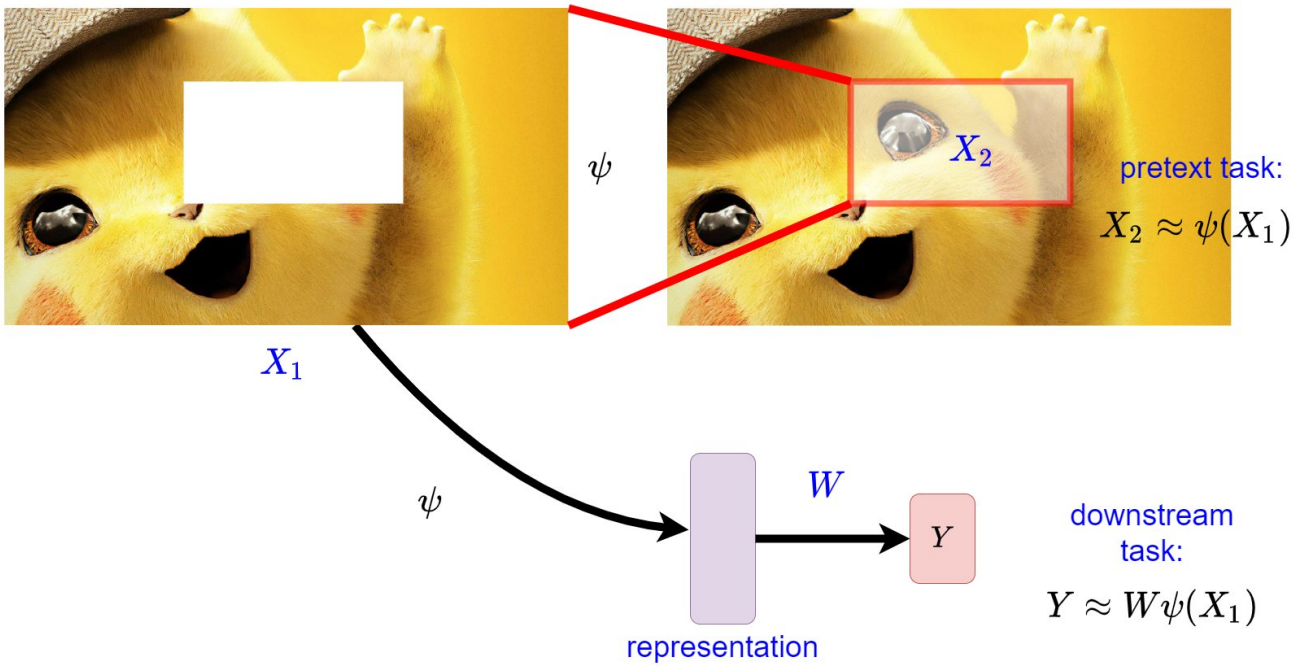
1. Label Y with k classes.
2. Unmasked image X_1 and masked image X_2.



3. Key intuition: Pretext tasks should help us reduce irrelevant features

Ideal Scenario

1. Label Y with k classes.
2. Unmasked image X_1 and masked image X_2.



3. Ideal scenario: $X_1 \to Y \to X_2$
$\iff X_1 \perp X_2 | Y$

Our Result: Ideal Scenario

Setting:

1. k-class labels Y.
2. Representation ψ, last layer W^* (Learn from population loss.)

Compare this procedure to ground truth classifier f^*:

Theorem 3

1. (No representation error.) If $X_1 \perp X_2 | Y$,

$$f^* = W^* \psi(X_1).$$

2. Only need $O(k)$ labeled samples.

Remark: Only need k samples instead of Rademacher complexity of function class.

Linear case was studied in (Foster, Kakade, Zhang, 2008)

Proof sketch

No representation error: if $X_1 \to Y \to X_2, f^* = W^* \psi(X_1)$.

Lemma is proved by law of total expectation:

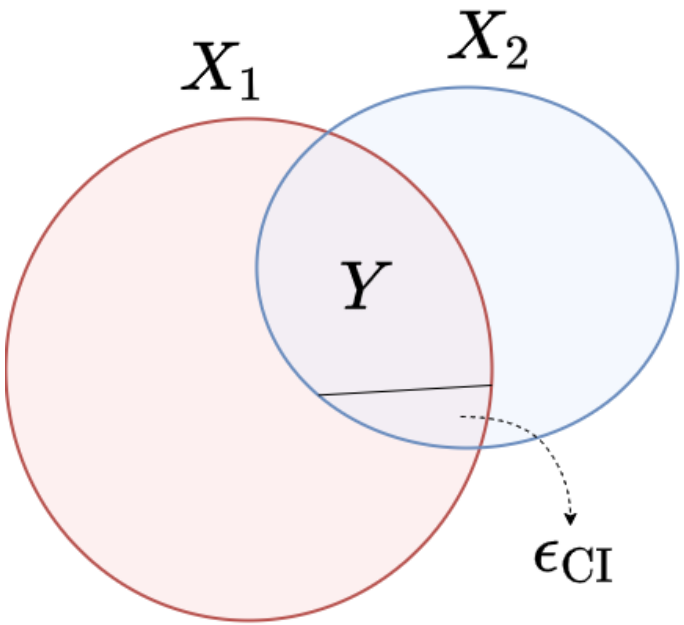
$$\psi^*(\cdot) := \mathbb{E}[X_2|X_1] = \mathbb{E}[\mathbb{E}[X_2|X_1, Y]|X_1] = \mathbb{E}[\mathbb{E}[X_2|Y]|X_1]$$

(uses CI)

$$= \sum_y P(Y = y|X_1)\, \mathbb{E}[X_2|Y = y] =: f(X_1)^\top \mathbf{A},$$

Here $f(x_1)_y := P(Y = y|X_1 = x_1)$
$\mathbf{A}$ satisfies $\mathbf{A}_{y,:} = \mathbb{E}[X_2|Y = y]$.

Main Results on Reconstruction-based SSL



Characterizing Approximate Conditional Independence

Define $\epsilon_{\text{CI}} = \mathbb{E}_{X_1} \left\| \mathbb{E}[X_2|X_1] - \mathbb{E}_Y[\mathbb{E}[X_2|Y]|X_1] \right\|^2$.

- This is 0 if $X_1 \perp X_2 \mid Y$.
- ϵ_{CI} can be viewed as quantification of extra shared features between X_1 and X_2 are not captured by Y (spurious feature not reducible by SSL)

Main Results on Reconstruction-based SSL

Characterizing Approximate Conditional Independence

Define $\epsilon_{\mathsf{CI}} = \mathbb{E}_{X_1} \left\| \mathbb{E}[X_2|X_1] - \mathbb{E}_Y[\mathbb{E}[X_2|Y]|X_1] \right\|^2$.

- This is 0 if $X_1 \perp X_2 \mid Y$.
- ϵ_{CI} can be viewed as quantification of extra shared features between X_1 and X_2 are not captured by Y (spurious feature not reducible by SSL)

$$\text{Test Error} \lesssim \underbrace{\frac{k}{n_L}}_{\text{adaptation error}} + \underbrace{\epsilon_{\mathsf{CI}}}_{\text{Representation error}}$$

Main Results on Reconstruction-based SSL

Characterizing Approximate Conditional Independence

Define $\epsilon_{\mathsf{CI}} = \mathbb{E}_{X_1} \left\| \mathbb{E}[X_2|X_1] - \mathbb{E}_Y[\mathbb{E}[X_2|Y]|X_1] \right\|^2$.

- This is 0 if $X_1 \perp X_2 \mid Y$.
- ϵ_{CI} can be viewed as quantification of extra shared features between X_1 and X_2 are not captured by Y (spurious feature not reducible by SSL)

$$\text{Test Error} \lesssim \underbrace{\frac{k}{n_L}}_{\text{adaptation error}} + \underbrace{\epsilon_{\mathsf{CI}}}_{\text{Representation error}}$$

- ERM would need $n_L \asymp$ Complexity of Function Class.

Main Results on Reconstruction-based SSL

Characterizing Approximate Conditional Independence

Define $\epsilon_{\mathsf{CI}} = \mathbb{E}_{X_1} \left\| \mathbb{E}[X_2|X_1] - \mathbb{E}_Y[\mathbb{E}[X_2|Y]|X_1] \right\|^2$.

- This is 0 if $X_1 \perp X_2 \mid Y$.
- ϵ_{CI} can be viewed as quantification of extra shared features between X_1 and X_2 are not captured by Y (spurious feature not reducible by SSL)

$$\text{Test Error} \lesssim \underbrace{\frac{k}{n_L}}_{\text{adaptation error}} + \underbrace{\epsilon_{\mathsf{CI}}}_{\text{Representation error}}$$

- ERM would need $n_L \asymp$ Complexity of Function Class.
- Both terms are tight in $\frac{k}{n_L} + \epsilon_{\mathsf{CI}}$ in some scenarios

Main Results on Reconstruction-based SSL

Characterizing Approximate Conditional Independence

Define $\epsilon_{\mathsf{CI}} = \mathbb{E}_{X_1} \left\| \mathbb{E}[X_2|X_1] - \mathbb{E}_Y[\mathbb{E}[X_2|Y]|X_1] \right\|^2$.

- This is 0 if $X_1 \perp X_2 \mid Y$.
- ϵ_{CI} can be viewed as quantification of extra shared features between X_1 and X_2 are not captured by Y (spurious feature not reducible by SSL)

$$\text{Test Error} \lesssim \underbrace{\frac{k}{n_L}}_{\text{adaptation error}} + \underbrace{\epsilon_{\mathsf{CI}}}_{\text{Representation error}}$$

- ERM would need $n_L \asymp$ Complexity of Function Class.
- Both terms are tight in $\frac{k}{n_L} + \epsilon_{\mathsf{CI}}$ in some scenarios
- Also applies to similarity-based SSL

Empirical Implication

Implications on pretext selection

- Design pretext tasks such that X_1 and X_2 have smaller dependence (given Y)

Predicting what you already know helps: Provable self-supervised learning, NeurIPS 2021

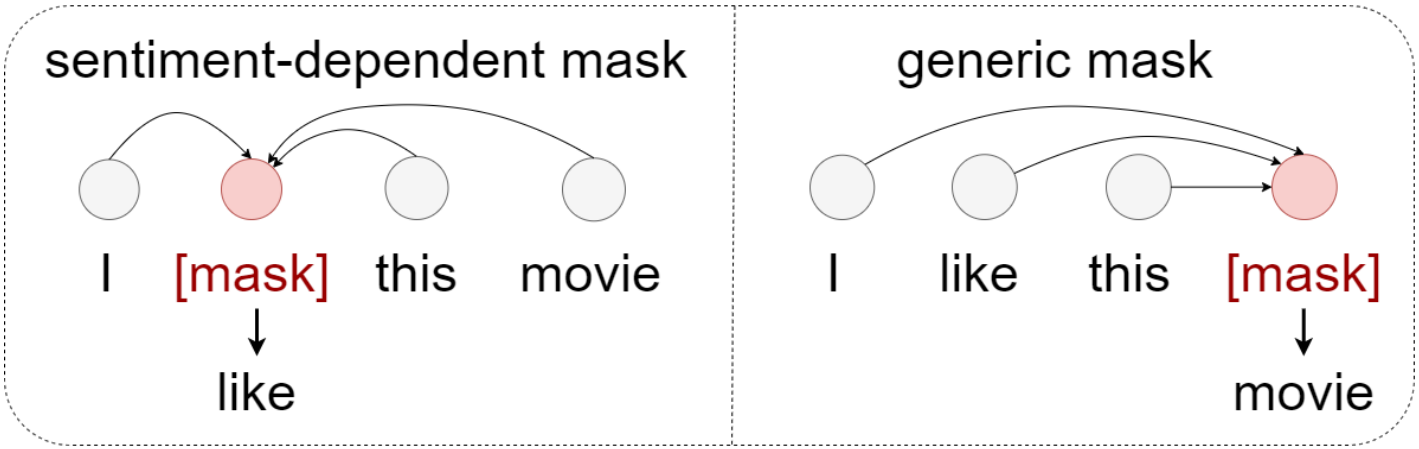
Empirical Implication

Implications on pretext selection

- Design pretext tasks such that X_1 and X_2 have smaller dependence (given Y)

Applications:

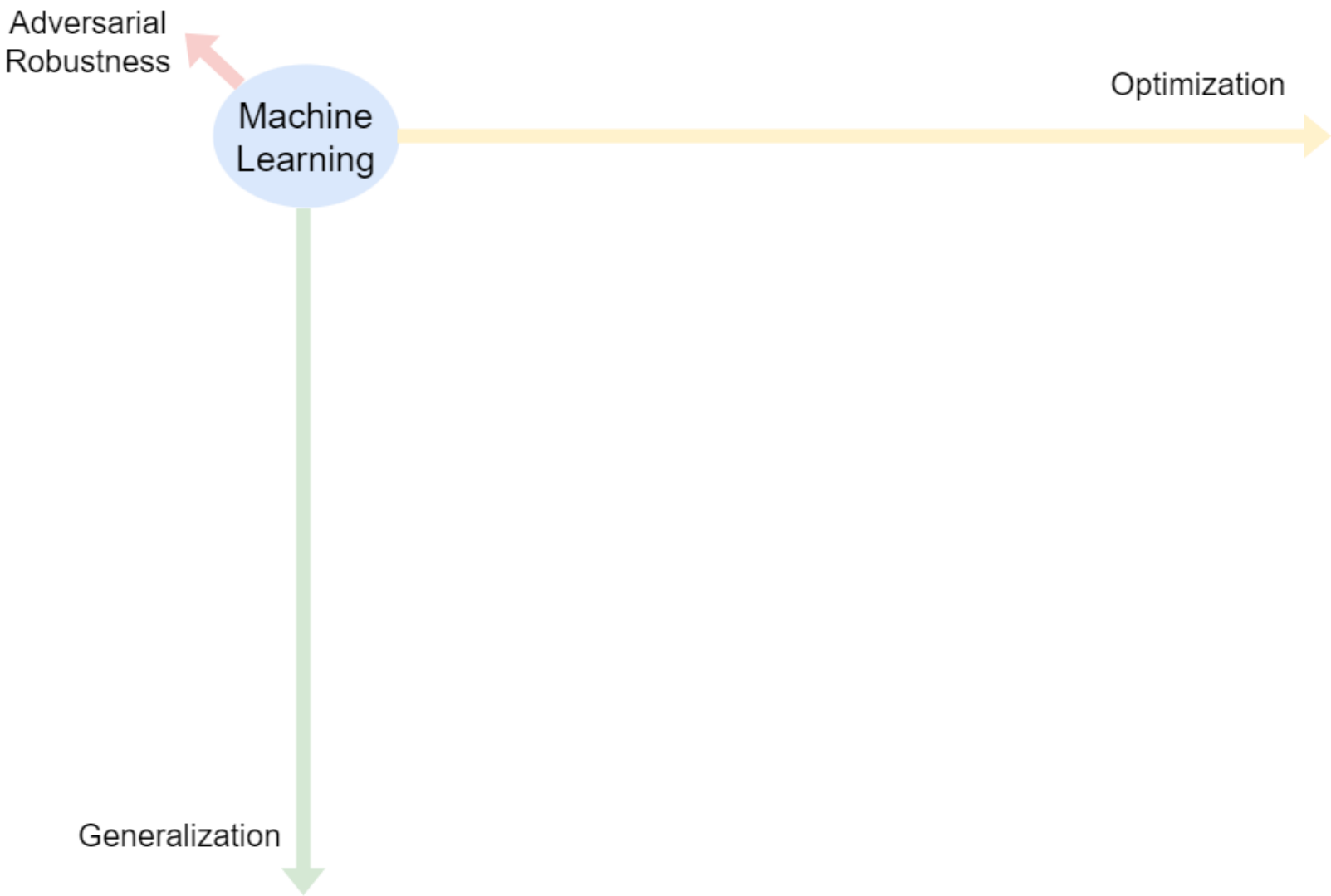
- Image: image classification [He et al. 2021]
- Text: sentiment analysis [Zhang and Hashimoto, 2021]



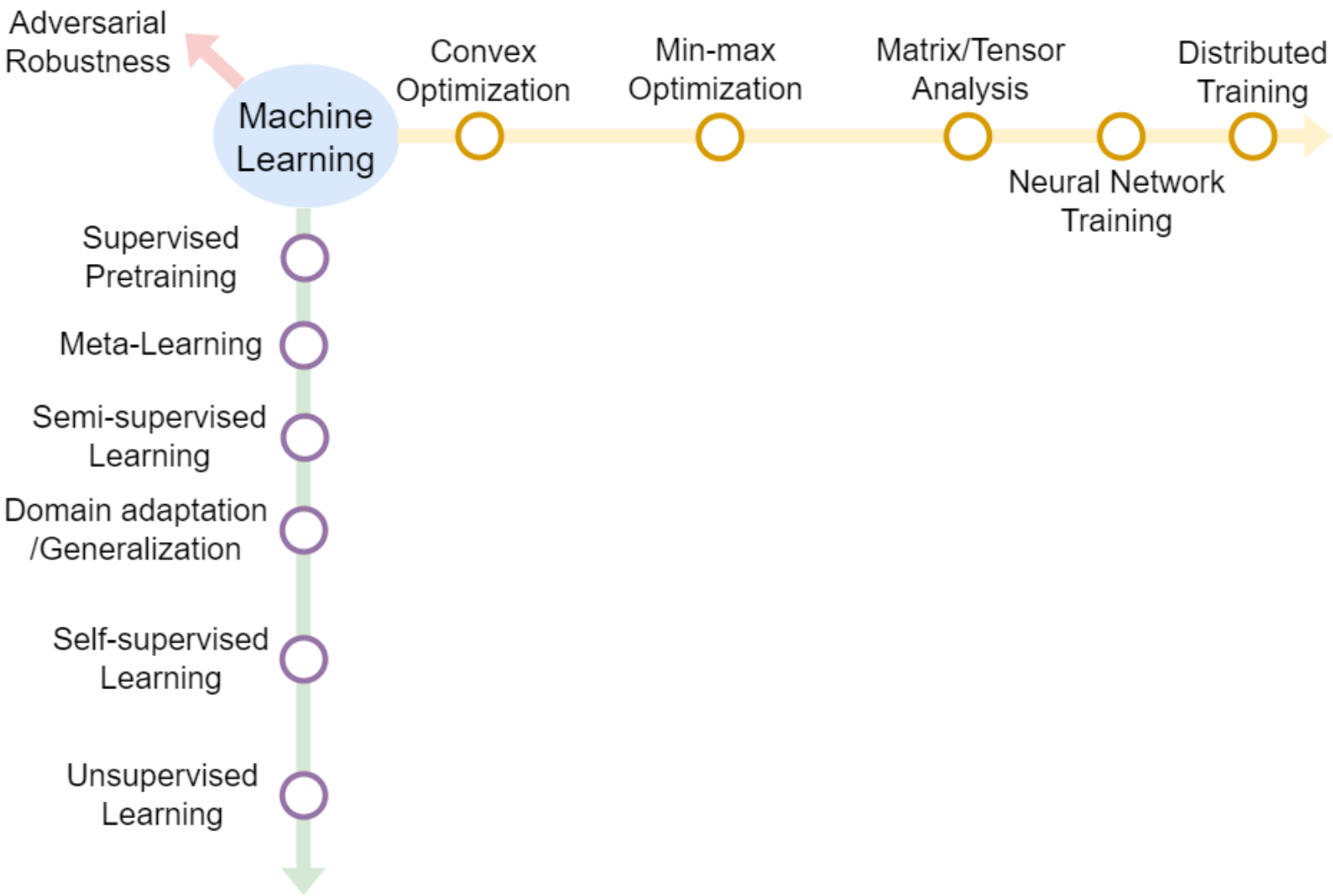
- Audio: speaker recognition [Zaiem et al., 2021]

Predicting what you already know helps: Provable self-supervised learning, NeurIPS 2021

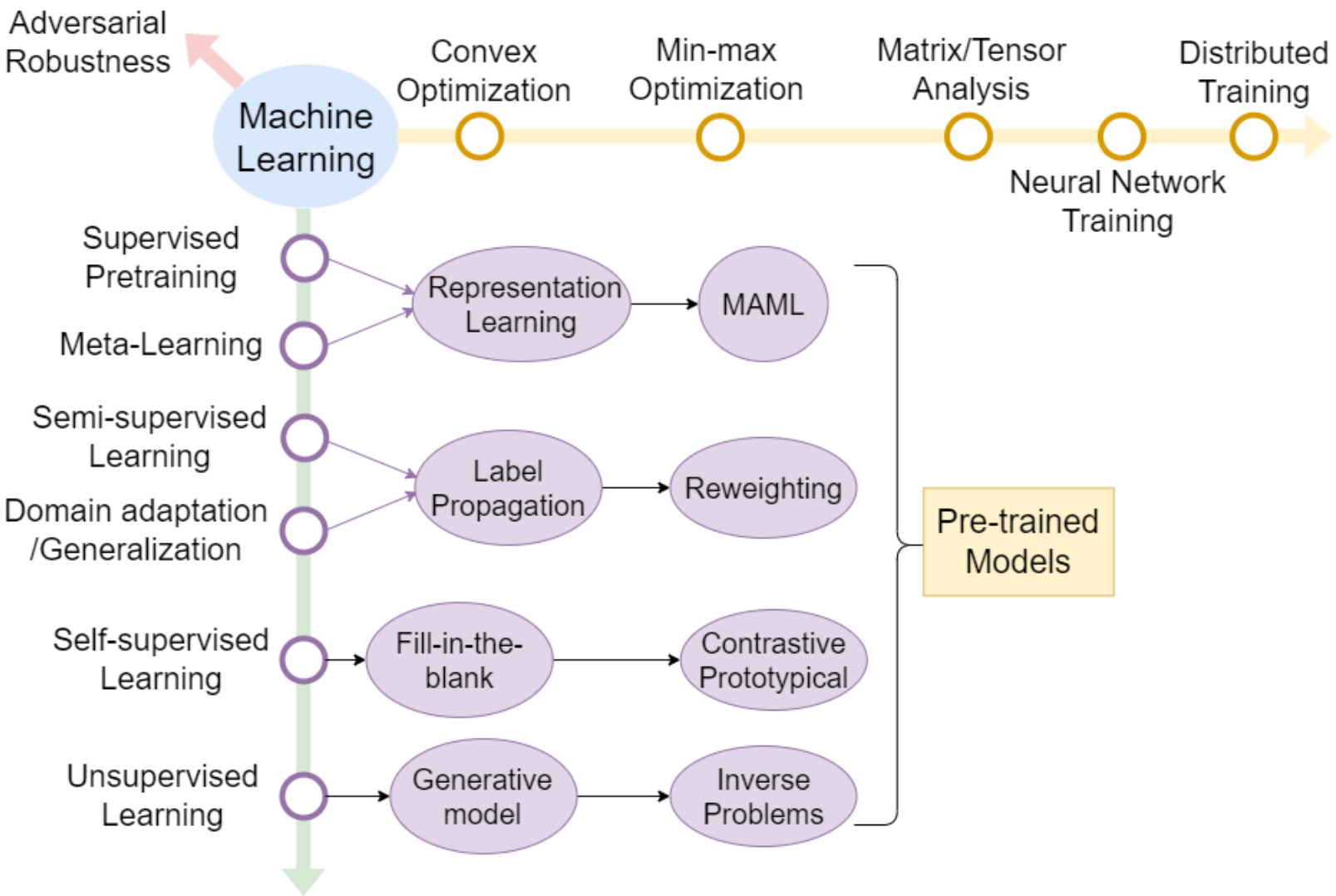
Overview of My Research Contributions



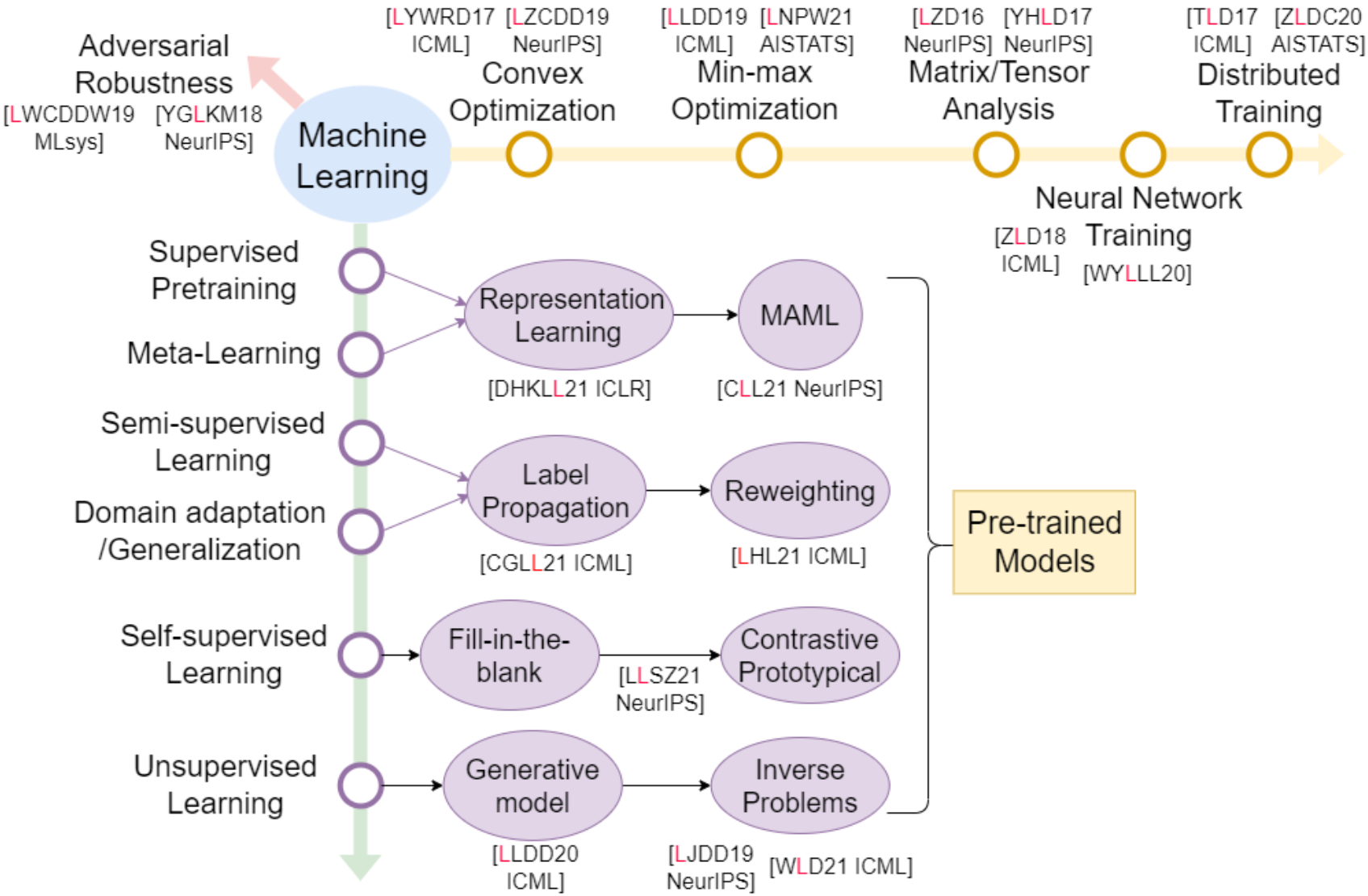
Overview of My Research Contributions



Overview of My Research Contributions



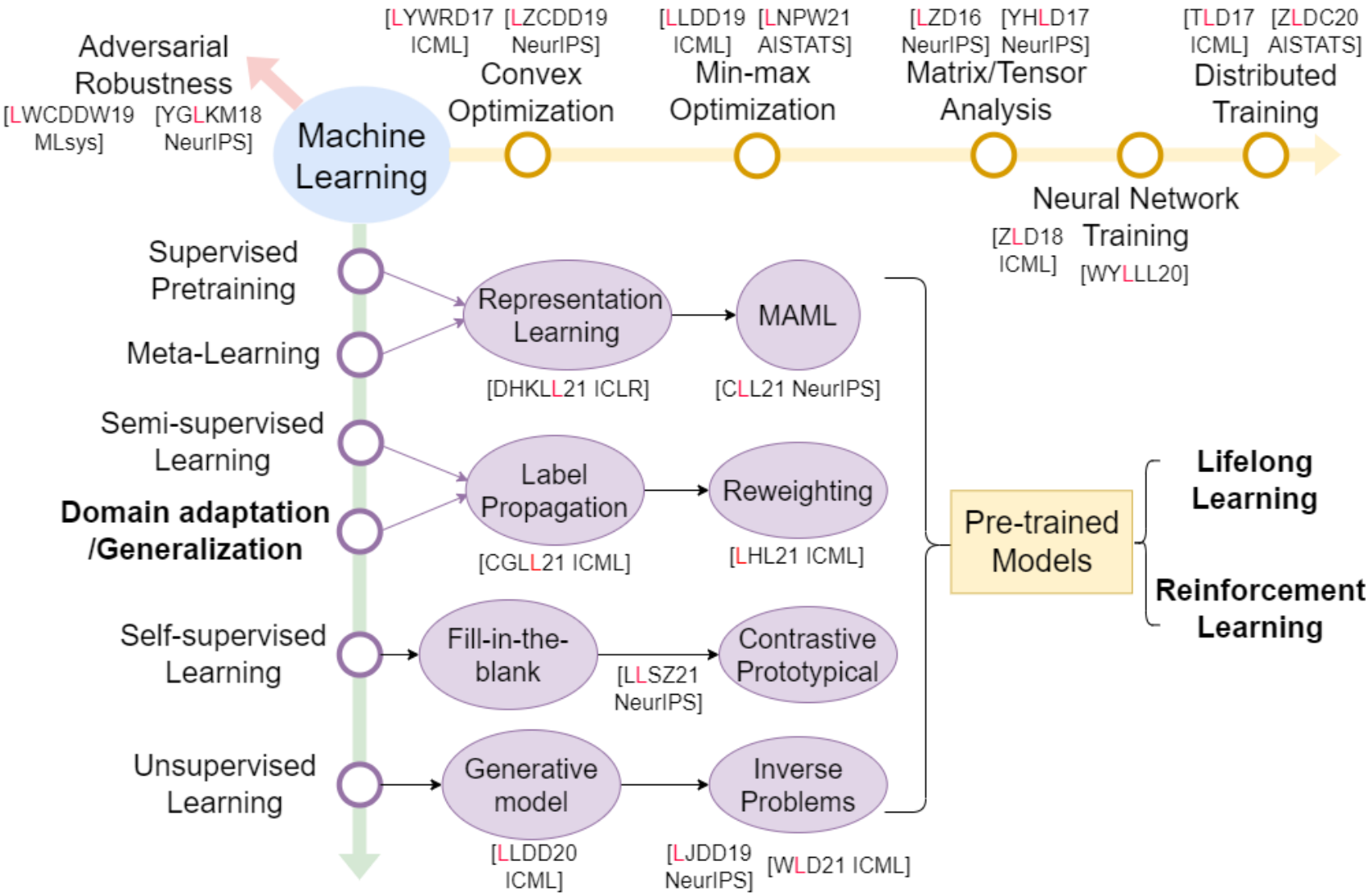
Overview of My Research Contributions



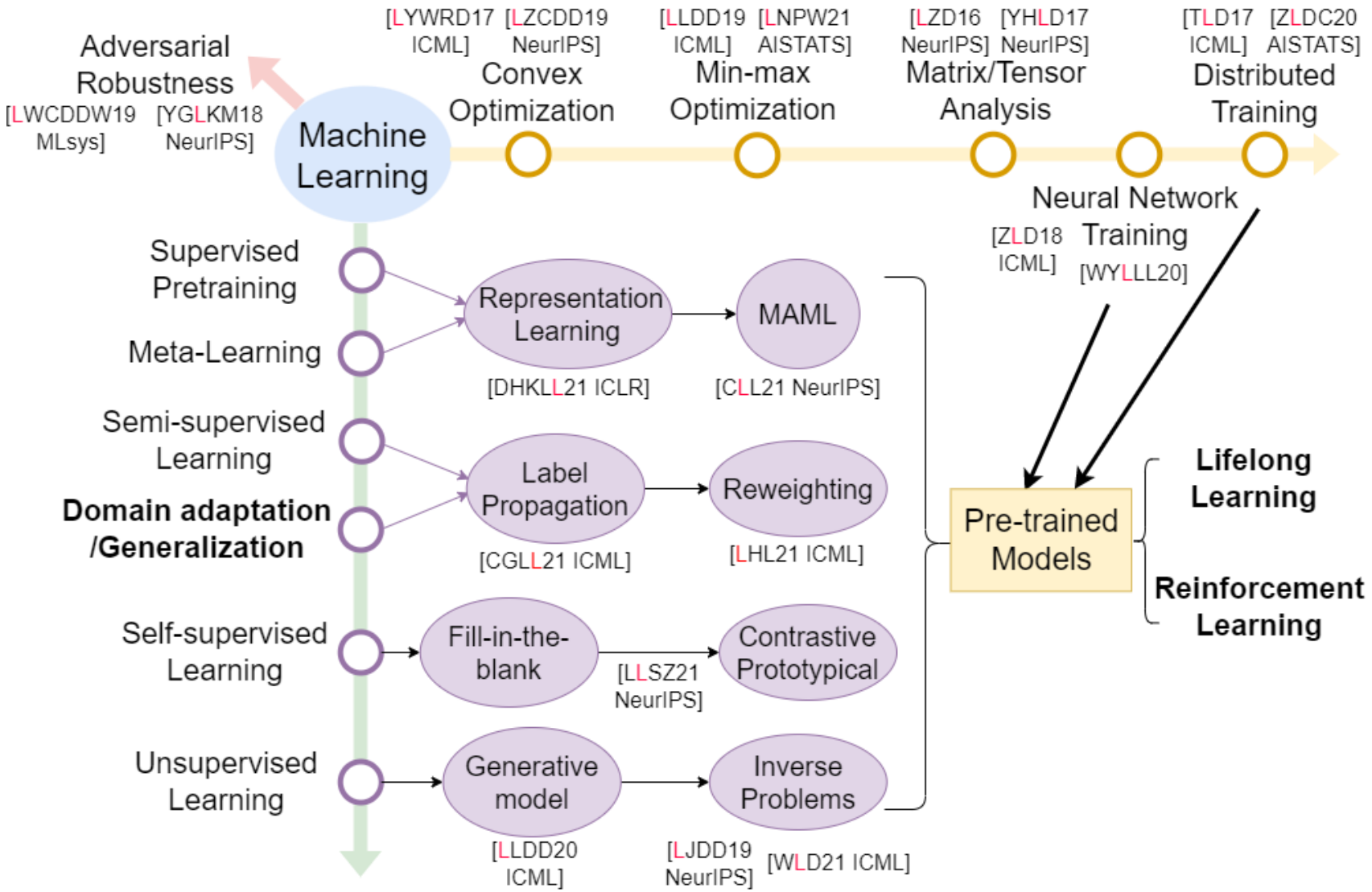
Outline

1. Supervised Pre-training and Meta-learning
   - Meta-learning Representation: Frozen Representation
   - Model-agnostic Meta-learning: Fine-tuned Representation
2. Self-Supervised Learning
3. Ongoing and Future Work
   - Domain Adaptation
   - Lifelong Learning
   - Meta Reinforcement Learning

Future Work


Adversarial Robustness
[LWCDDW19 MLsys]
[YGLKM18 NeurIPS]
Machine Learning
[LYWRD17 ICML]
[LZCDD19 NeurIPS]
Convex Optimization
[LLDD19 ICML]
[LNPW21 AISTATS]
Min-max Optimization
[LZD16 NeurIPS]
[YHLD17 NeurIPS]
Matrix/Tensor Analysis
[TLD17 ICML]
[ZLDC20 AISTATS]
Distributed Training
Neural Network Training
[ZLD18 ICML]
[WYLLL20]
Supervised Pretraining
Meta-Learning
Representation Learning
[DHKLL21 ICLR]
MAML
[CLL21 NeurIPS]
Semi-supervised Learning
Domain adaptation /Generalization
Label Propagation
[CGLL21 ICML]
Reweighting
[LHL21 ICML]
Self-supervised Learning
Fill-in-the-blank
[LLSZ21 NeurIPS]
Contrastive Prototypical
Unsupervised Learning
Generative model
[LLDD20 ICML]
Inverse Problems
[LJDD19 NeurIPS]
[WLD21 ICML]
Pre-trained Models
Lifelong Learning
Reinforcement Learning

Future Work


Adversarial Robustness
[LWCDDW19 MLsys]
[YGLKM18 NeurIPS]
Machine Learning
[LYWRD17 ICML]
[LZCDD19 NeurIPS]
Convex Optimization
[LLDD19 ICML]
[LNPW21 AISTATS]
Min-max Optimization
[LZD16 NeurIPS]
[YHLD17 NeurIPS]
Matrix/Tensor Analysis
[TLD17 ICML]
[ZLDC20 AISTATS]
Distributed Training
Neural Network Training
[ZLD18 ICML]
[WYLLL20]
Supervised Pretraining
Meta-Learning
Representation Learning
[DHKLL21 ICLR]
MAML
[CLL21 NeurIPS]
Semi-supervised Learning
Domain adaptation /Generalization
Label Propagation
[CGLL21 ICML]
Reweighting
[LHL21 ICML]
Self-supervised Learning
Fill-in-the-blank
[LLSZ21 NeurIPS]
Contrastive Prototypical
Unsupervised Learning
Generative model
[LLDD20 ICML]
Inverse Problems
[LJDD19 NeurIPS]
[WLD21 ICML]
Pre-trained Models
Lifelong Learning
Reinforcement Learning

Summary

We investigate sufficient conditions that guarantee:

- Deep networks can **learn a good pre-trained model**.
- Good pre-trained models drastically **reduce sample complexity**.
- Pre-trained models can be **learned on unlabeled data**.
- Pre-trained models **transfer to other tasks/domains with covariate shift**.

Summary

We investigate sufficient conditions that guarantee:

- Deep networks can **learn a good pre-trained model**.
- Good pre-trained models drastically **reduce sample complexity**.
- Pre-trained models can be **learned on unlabeled data**.
- Pre-trained models **transfer to other tasks/domains with covariate shift**.

In the future, I will explore other aspects of pre-trained model:

- How to handle different types of **distribution shift** for domain adaptation?
- How to utilize **latent structure** for meta reinforcement learning?
- How to improve the training algorithms?

Thank you!

Thank you!

Link back to: MetaRep FinetunedRep SSL
Back-up slides: FinetunedRep SSL Example Experiments

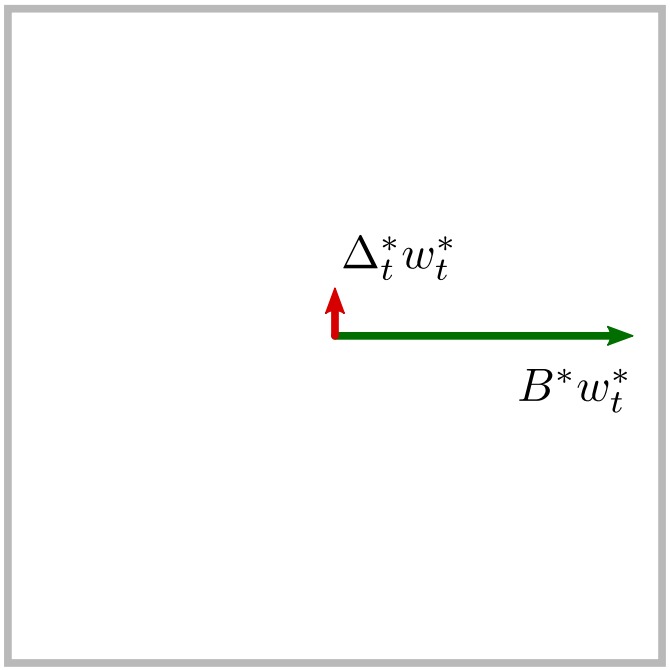
Failure with Previous Algorithm: MetaRep

MetaRep (Fixed feature) has $\Omega\left(\frac{d}{n_T}\right)$ minimax rate on the target task

Failure with Previous Algorithm: MetaRep

MetaRep *chooses representation based on prediction space norm, not parameter space norm!*

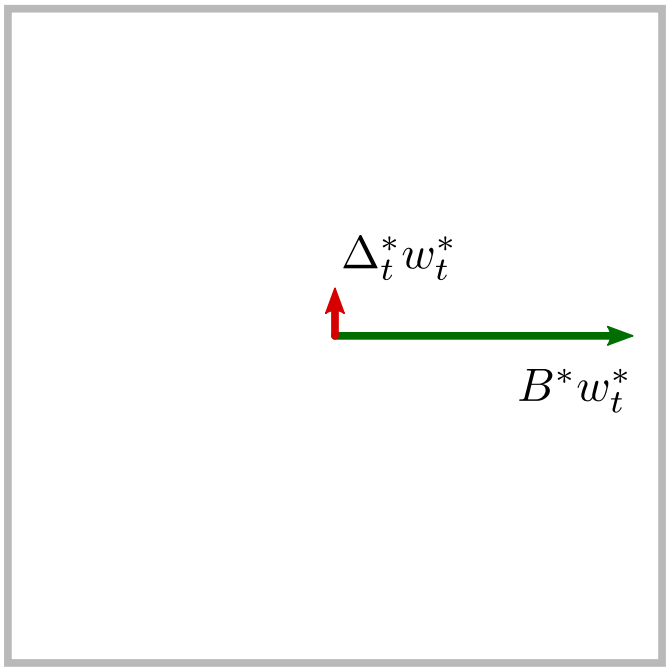
Parameter space (L_2-norm)



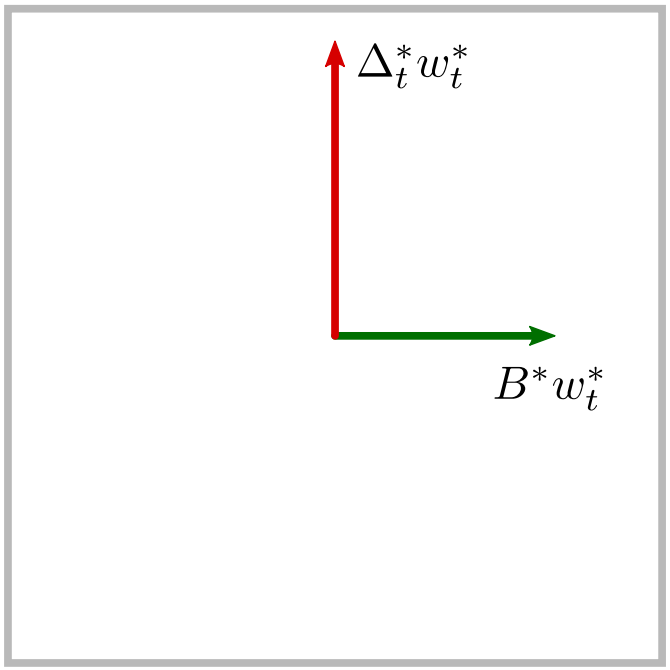
Failure with Previous Algorithm: MetaRep

MetaRep *chooses representation based on prediction space norm, not parameter space norm!*

Parameter space (L_2-norm)



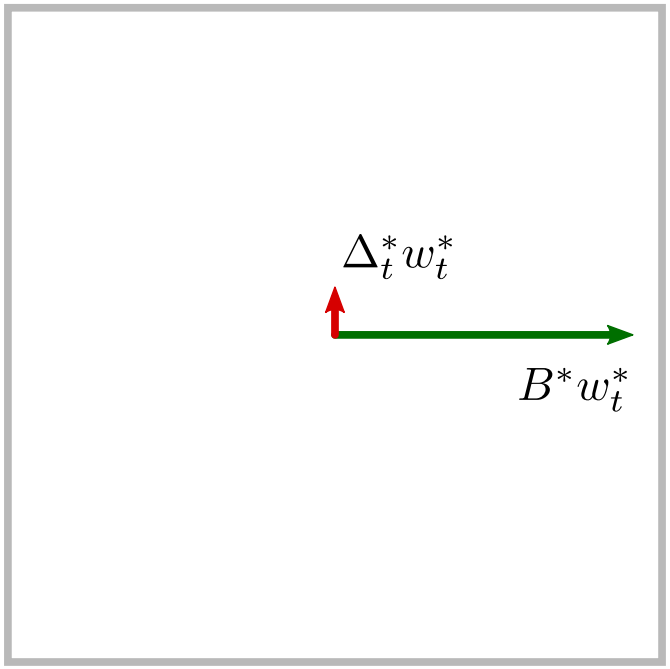
Prediction space (Σ-norm)



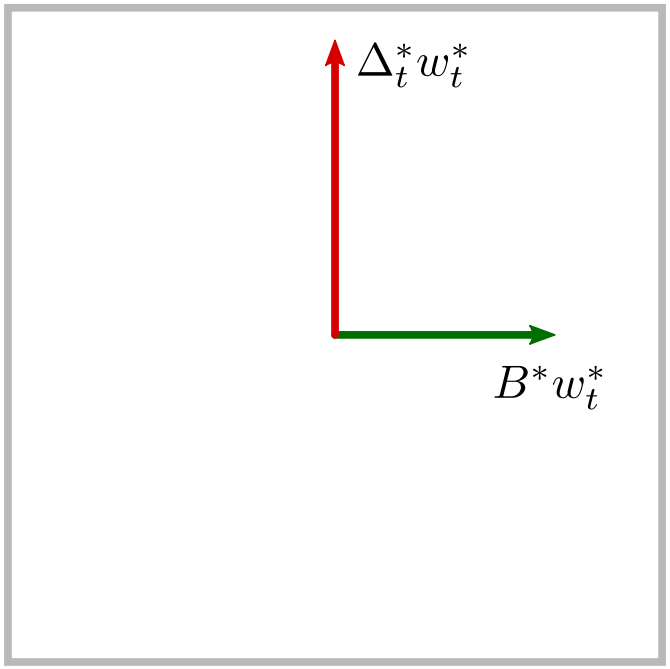
Failure with Previous Algorithm: MetaRep

MetaRep *chooses representation based on prediction space norm, not parameter space norm!*

Parameter space (L_2-norm)



Prediction space (Σ-norm)



Intuition: Trick MetaRep into learning subspace of δ_t^*'s, and pay cost of having to fine-tune to learn large-norm $B^* w_{\text{test}}^*$.

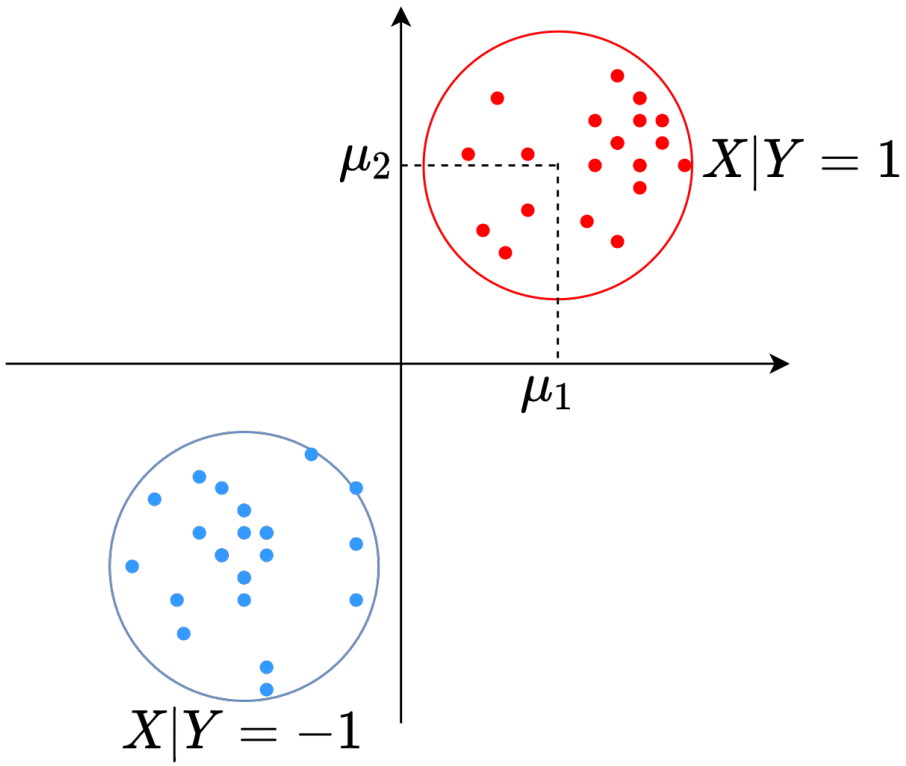
General Functions

Theoretical results:

1. If $X_1 \perp X_2 | Y$,
$$f^* = W^* \psi(X_1).$$
2. Only need k (dimension of Y) samples.

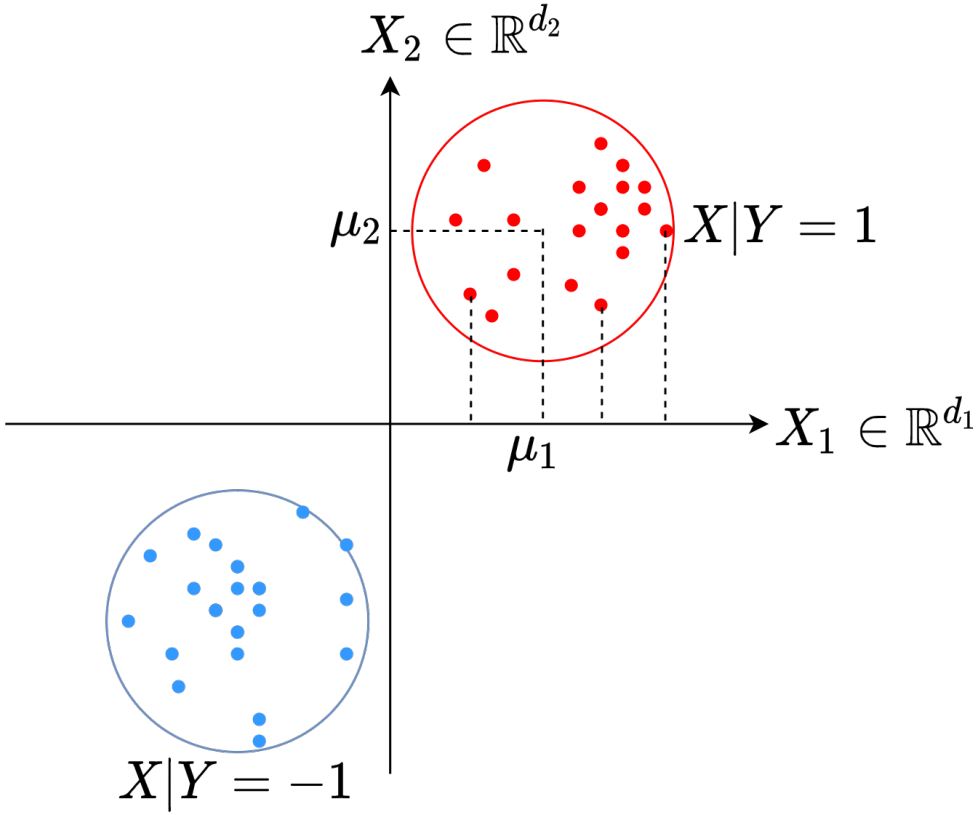
How can $\psi(X_1)$ be better than X_2 to predict Y?

Example: Gaussian Mixture



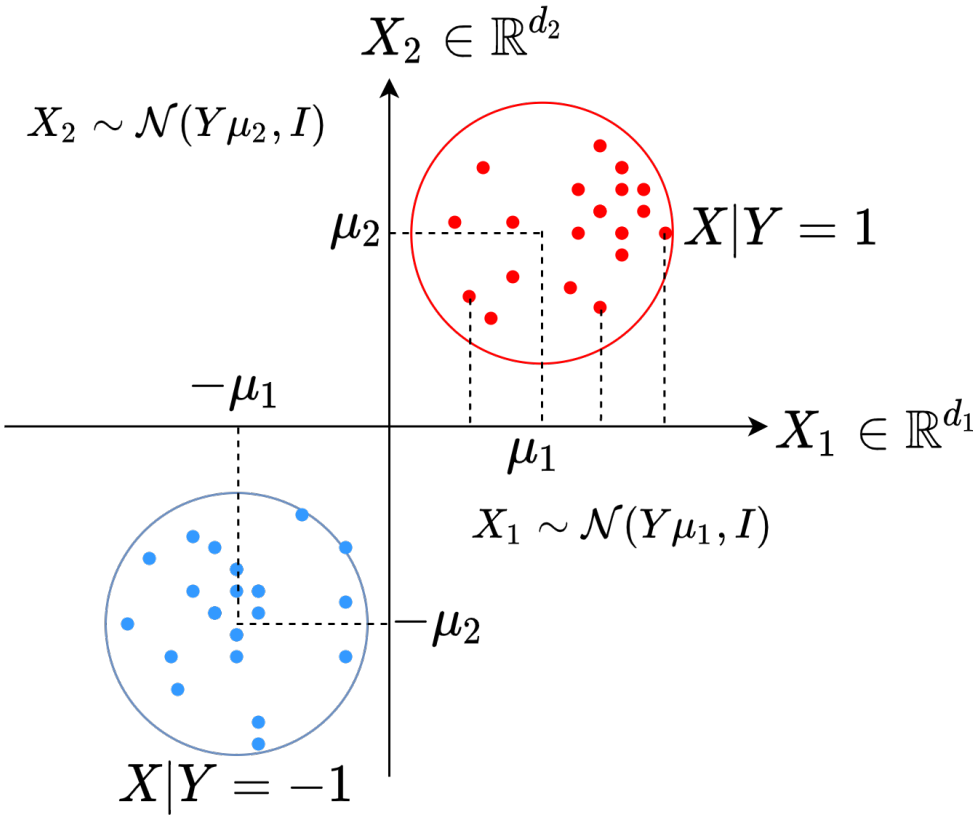
$$Y \in \{-1, 1\}$$
$$X \sim \mathcal{N}(Y[\mu_1, \mu_2], I_{d_1+\mathrm{d}_2})$$

Example: Gaussian Mixture



$$Y \in \{-1, 1\}$$
$$X \sim \mathcal{N}(Y[\mu_1, \mu_2], I_{d_1+\mathrm{d}_2})$$
$$X_1 \sim \mathcal{N}(Y\mu_1, I_{d_1}),$$
$$X_2 \sim \mathcal{N}(Y\mu_2, I_{d_2})$$
$$X_1 \perp X_2 | Y$$

Example: Gaussian Mixture



$Y \in \{-1, 1\}$

$X \sim \mathcal{N}(Y[\mu_1, \mu_2], I_{d_1+\mathrm{d}_2})$

$X_1 \sim \mathcal{N}(Y\mu_1, I_{d_1}),$

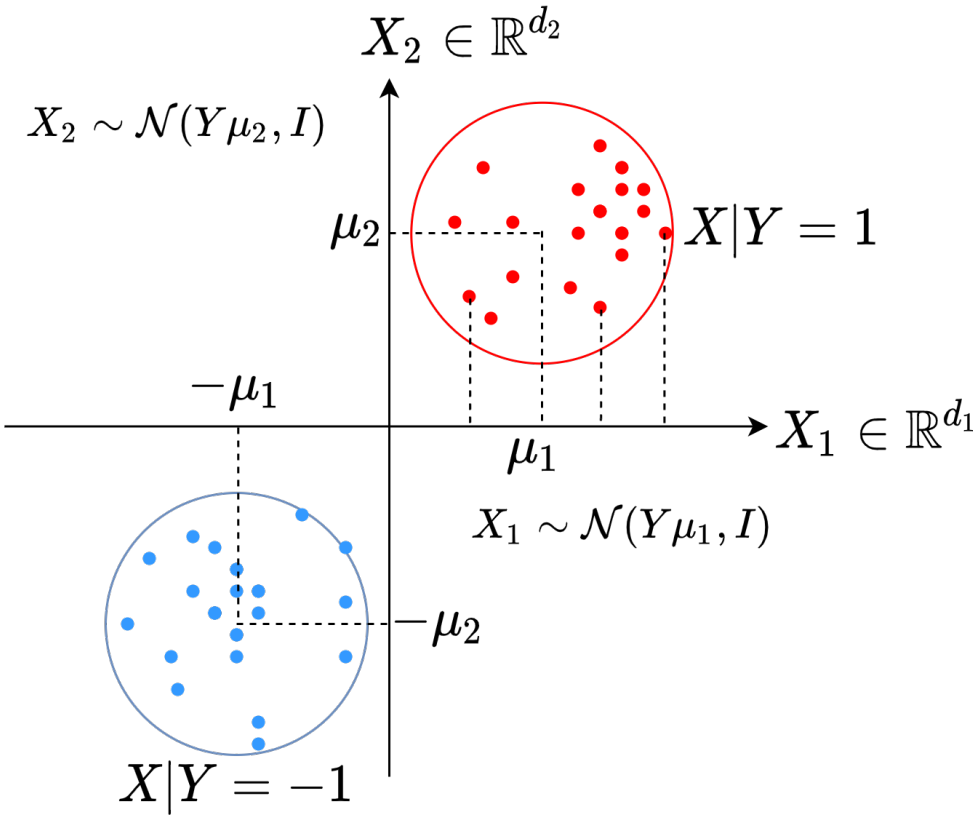
$X_2 \sim \mathcal{N}(Y\mu_2, I_{d_2})$

$X_1 \perp X_2 | Y$

$\mathbb{E}[Y|X_2]$ is not linear, but

$\mathbb{E}[Y|\psi]$ is linear:

Example: Gaussian Mixture



$Y \in \{-1, 1\}$

$X \sim \mathcal{N}(Y[\mu_1, \mu_2], I_{d_1+d_2})$

$X_1 \sim \mathcal{N}(Y\mu_1, I_{d_1})$,

$X_2 \sim \mathcal{N}(Y\mu_2, I_{d_2})$

$X_1 \perp X_2 | Y$

$\mathbb{E}[Y|X_2]$ is not linear, but

$\mathbb{E}[Y|\psi]$ is linear:

- $\psi(x_1) = \mathbb{E}[X_2|X_1 = x_1]$
  $= p_1(x_1)\mu_2 + p_{-1}(x_1)(-\mu_2)$
- $\mathbb{E}[Y|X_1] = p_1(x_1) -$
  $p_{-1}(x_1) = \mu_2^\top \psi(X_1)/\|\mu_2\|^2$.
- $p_y(x_1) := P(Y = y|X_1 =$
  $x_1)$

Algorithm (SimSiam):

$$\max_{\psi,\eta,\text{normalized}} \mathbb{E}[\psi(X_1)^\top \eta(X_2)]$$

New measure of conditional independence:

$$\epsilon_{\text{CI}} := \max_{\|g\|_{L^2(X_2)}=1} \mathbb{E}_{X_1}(\mathbb{E}[g(X_2)|X_1] - \mathbb{E}[\mathbb{E}[g(X_2)|Y]|X_1])^2.$$

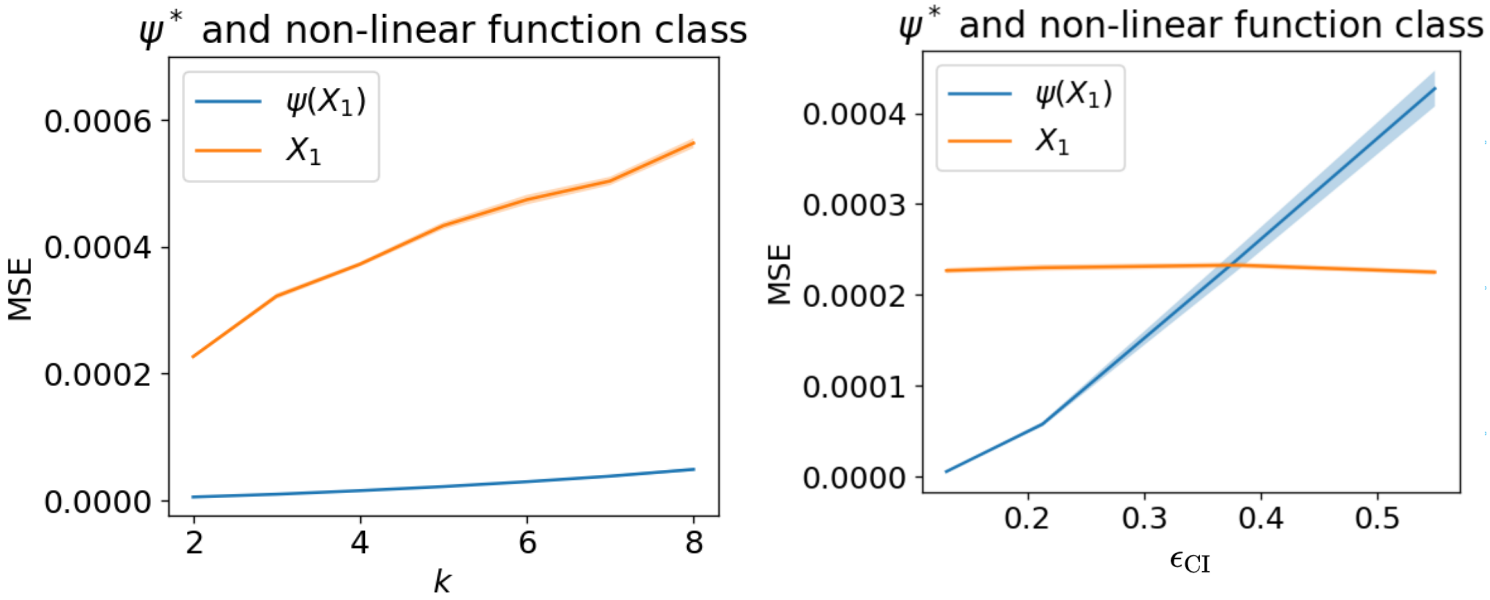
- Extension of previous result:

$$\textit{test error} \lesssim \epsilon_{\textit{CI}} + \frac{k}{n_L}.$$

- If both X_2 and X_1 can well predict Y, i.e., $P_{X_1,Y}(g^*(x_1) \neq y) \leq \alpha$ (same for X_2), we have:

$$\textit{test error} \lesssim \frac{\alpha}{1-\epsilon_{\textit{CI}}} + \frac{k}{n_L}.$$

Simulations: Both Terms Tight in $\frac{k}{n_L} + \epsilon_{\mathsf{CI}}$



Left: Class Conditional Gaussian $X \sim \mathcal{N}(\mu_Y, I), \mu_Y \in \mathbb{R}^{90}, Y \in \{1, 2, \cdots k\}, X_1 = X_{1:50}, X_2 = X_{51:90}$. $X_1 \perp X_2 | Y$

Right: Similar mixture of Gaussian: $X \sim \mathcal{N}(\mu_Y, \Sigma_{\epsilon_{\mathsf{CI}}})$, $\alpha \propto \epsilon_{\mathsf{CI}}$ controls the dependence of X_1 and X_2: $\epsilon_{\mathsf{CI}} = 0 \Rightarrow$ exact CI, and $\epsilon_{\mathsf{CI}} = 1 \Rightarrow X_2$ fully depends on X_1.

Connection to SimSiam method

- Before, we learn $\psi(x_1) = \mathbb{E}[X_2|X_1 = x_1]$, which naturally requires X_2 and Y to be linearly correlated
- We can actually predict any $g(X_2)|X_1$, or even $p(X_2|X_1)$

ACE and nonlinear CCA

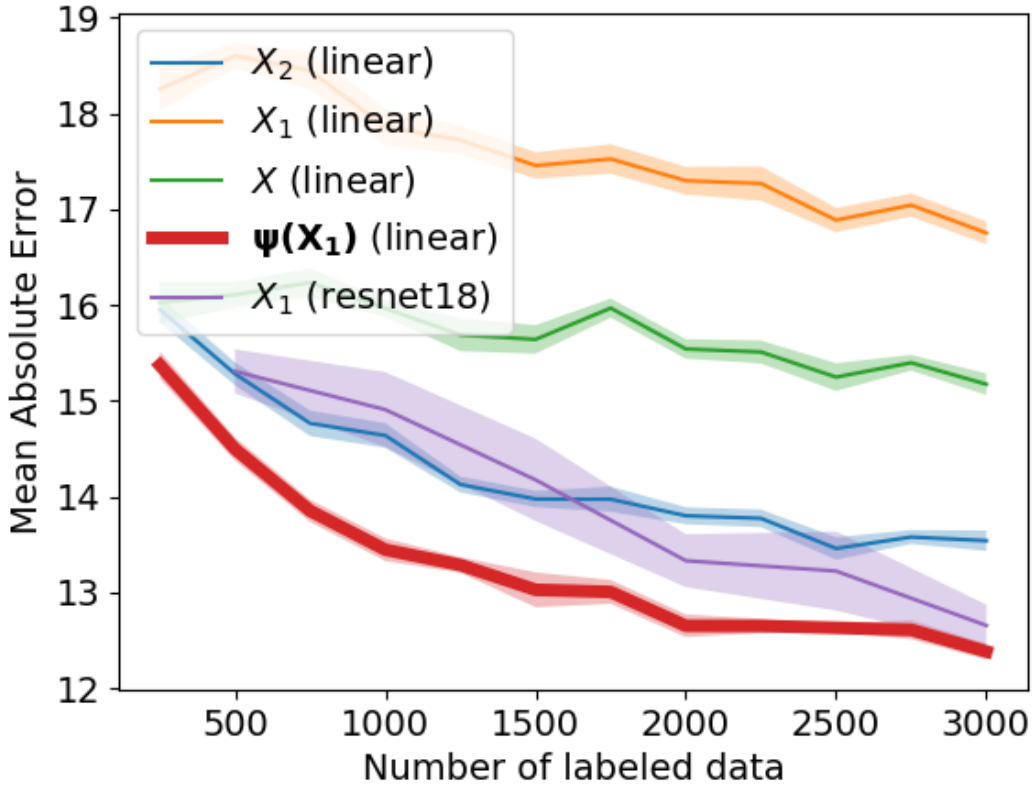
- Alternating conditional expectation (ACE):

$$\min_{\psi,\eta} L_{ACE}(\psi,\eta) = \mathbb{E}_{X_1,X_2}\left[\|\psi(X_1) - \eta(X_2)\|^2\right],$$
$$\text{s.t. } \Sigma_{\psi,\psi} = \Sigma_{\eta,\eta} = I_k.$$

- This is equivalent to the following canonical correlation analysis (CCA):

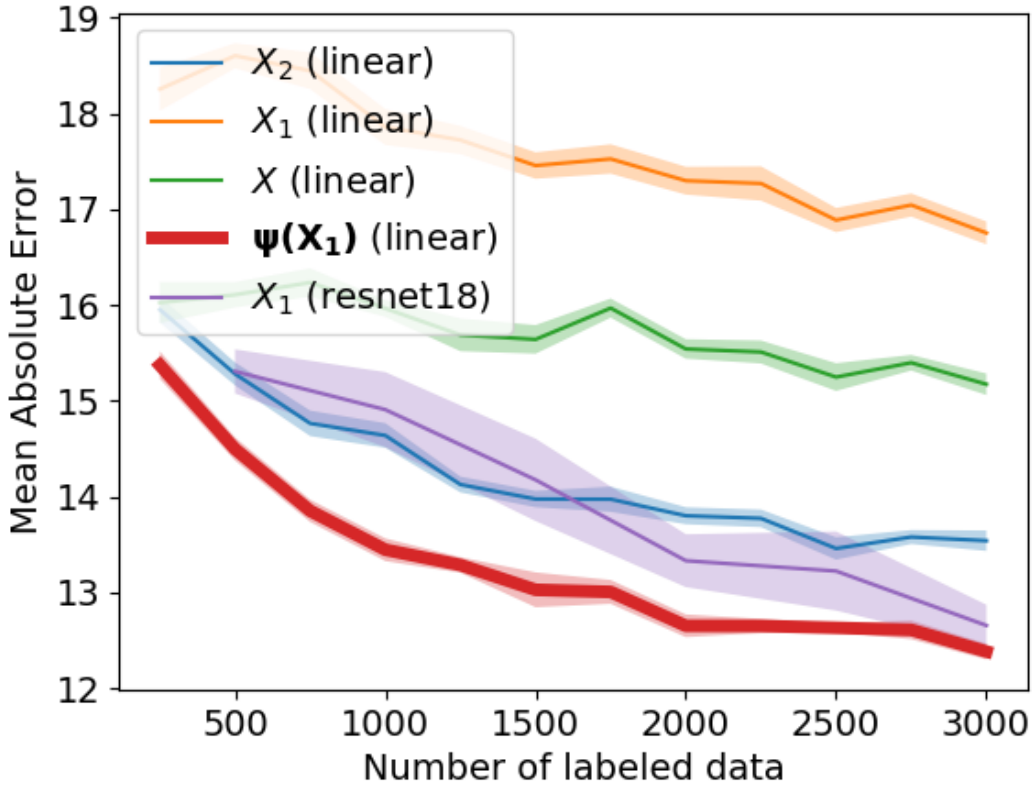
$$\max_{\psi,\eta} L_{CCA}(\psi,\eta) = \mathbb{E}_{X_1,X_2}\left[\psi(X_1)^\top \eta(X_2)\right],$$
$$\text{s.t. } \Sigma_{\psi,\psi} = \Sigma_{\eta,\eta} = I_k.$$

Experiments



- Yearbook: portraits date from 1905 to 2013.

Experiments



- Yearbook: portraits date from 1905 to 2013.
- Predicting what you already know helps: provable self-supervised learning. NeurIPS 2021

Ongoing Work: Distribution Shift

Source Target

Entity30 - Passerine

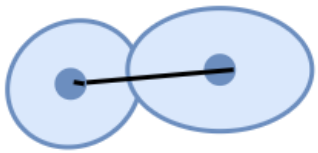
Source Target

Entity30 - Tableware

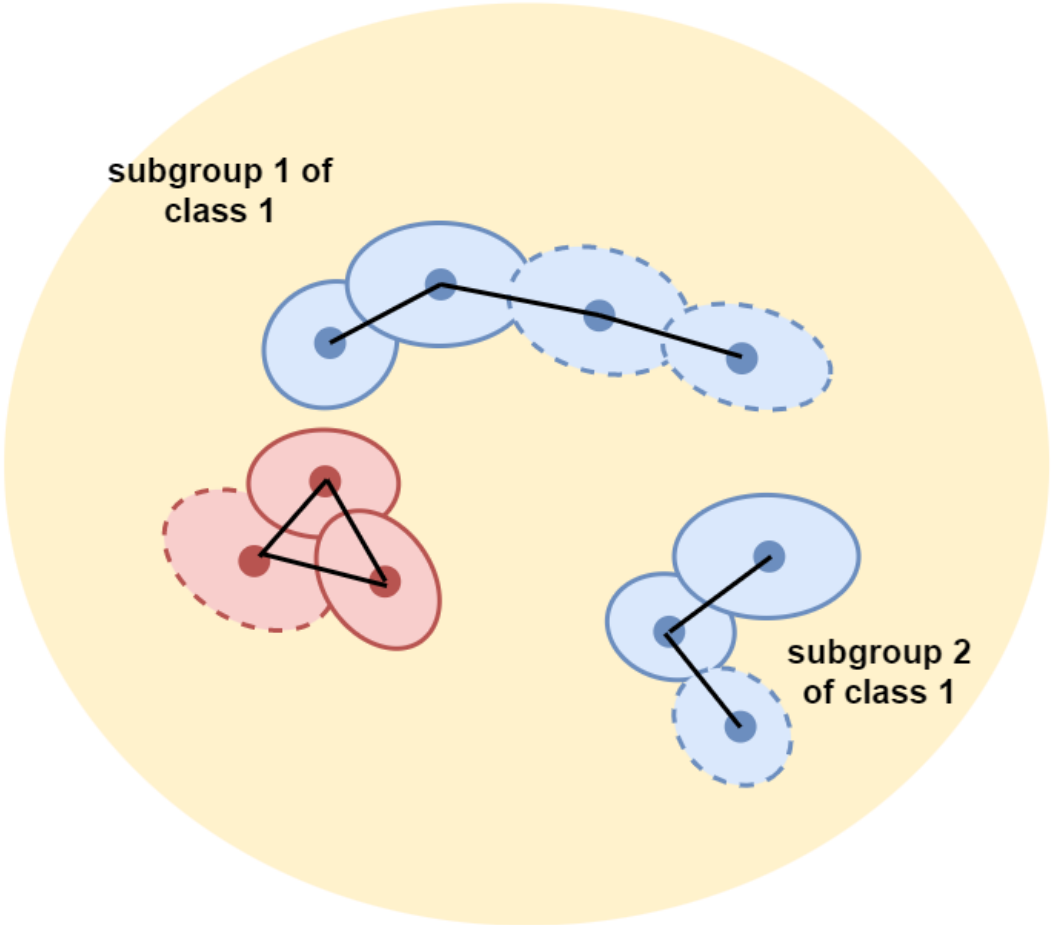
BREEDS dataset: (Santurkar et al., 2021)

Our New Framework: Subpopulation Shift

	Class 1	Class -1
Source		
Target		

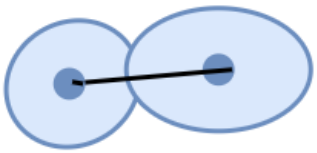


Components connected through data augmentation

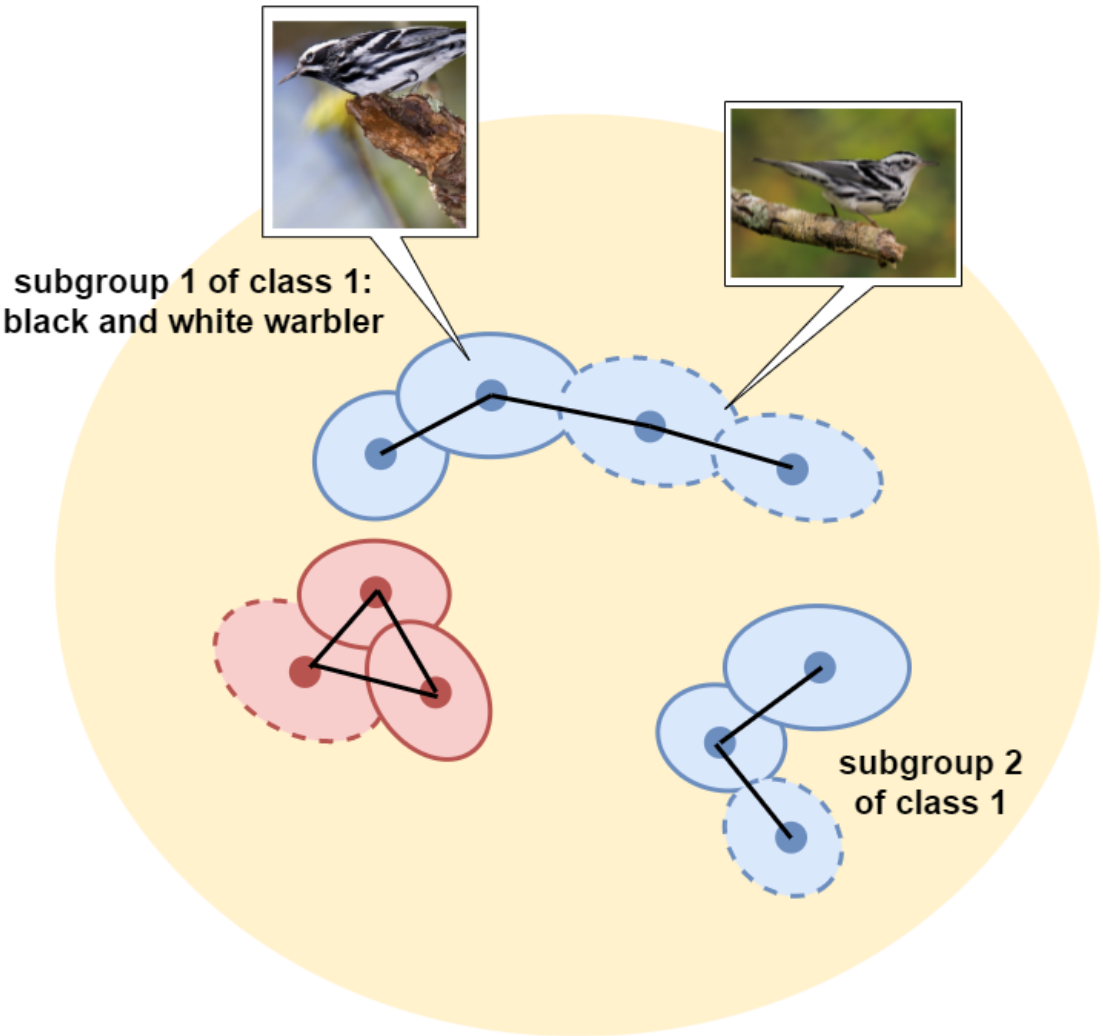


Our New Framework: Subpopulation Shift

	Class 1	Class -1
Source		
Target		

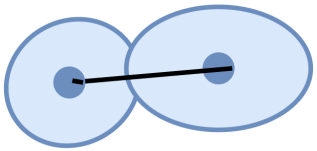


Components connected through data augmentation

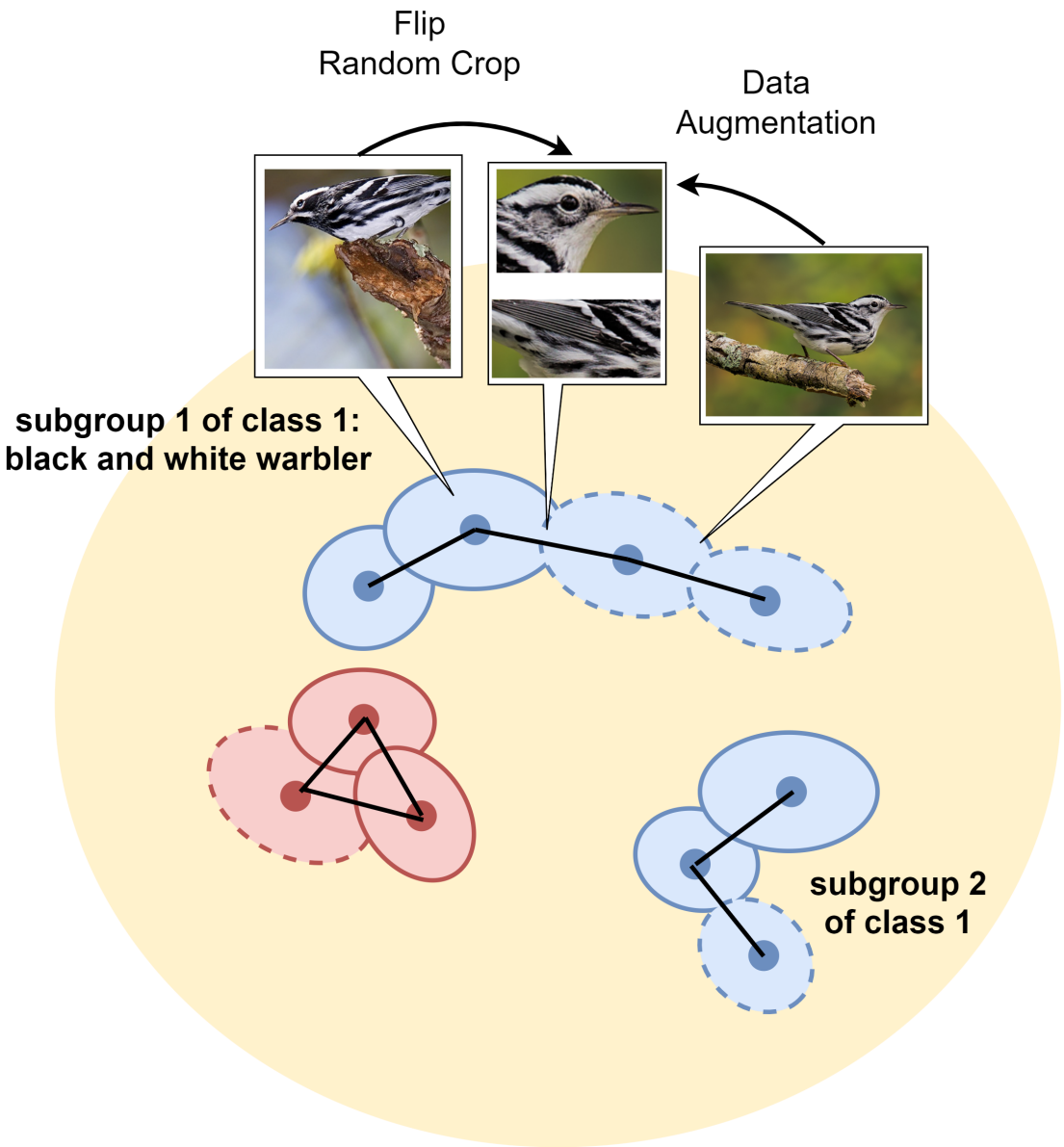


Our New Framework: Subpopulation Shift

	Class 1	Class -1
Source		
Target		



Components connected through data augmentation



Experiments

Method	A → W	D → W	W → D	A → D	D → A	W → A	Average
MDD	94.97±0.70	98.78±0.07	100±0	92.77±0.72	75.64±1.53	72.82±0.52	89.16
Ours	95.47±0.95	98.32±0.19	100±0	93.71±0.23	76.64±1.91	74.93±1.15	**89.84**

Performance of MDD[2] and our method on Office-31 dataset.

Method	Ar → Cl	Ar → Pr	Ar → Rw	Cl → Ar	Cl → Pr	Cl → Rw	Pr → Ar
MDD	54.9±0.7	74.0±0.3	77.7±0.3	60.6±0.4	70.9±0.7	72.1±0.6	60.7±0.8
Ours	55.1±0.9	74.7±0.8	78.7±0.5	63.2±1.3	74.1±1.8	75.3±0.1	63.0±0.6

Method	Pr → Cl	Pr → Rw	Rw → Ar	Rw → Cl	Rw → Pr	Average
MDD	53.0±1.0	78.0±0.2	71.8±0.4	59.6±0.4	82.9±0.3	68.0
Ours	53.0±0.6	80.8±0.4	73.4±0.1	59.4±0.7	84.0±0.5	**69.6**

Performance of MDD and our method on Office-Home dataset.

[2]MDD: (Zhang et al. 2019)
A Theory of Label Propagation for Subpopulation Shift, ICML 2021

Experiments: Subpopulation Shift Dataset

- ENTITY-30 task from BREEDS tasks.
- We use FixMatch, an existing consistency regularization method. We also leverage SwAV, an existing unsupervised representation learned from ImageNet, where there can be a better structure of subpopulation shift. We compare with popular distribution matching methods like DANN and MDD.

Method	Source Acc	Target Acc
Train on Source	91.91±0.23	56.73±0.32
DANN (Ganin et al., 2016)	92.81±0.50	61.03±4.63
MDD (Zhang et al., 2019)	92.67±0.54	63.95±0.28
FixMatch (Sohn et al., 2020)	90.87±0.15	72.60±0.51