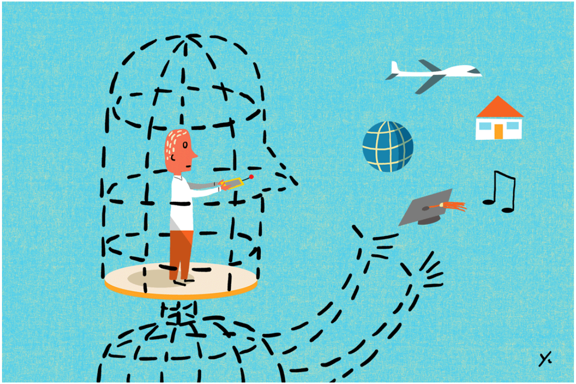
Theoretical Foundations of Pre-trained Models

Qi Lei

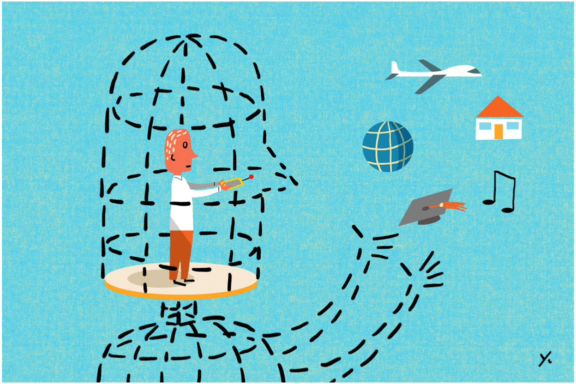
Princeton University

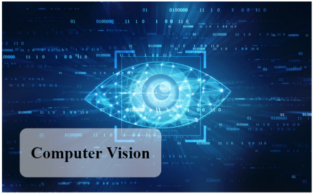
Slides available at: cecilialeiqi.github.io/job_talk.pdf

A.I. is Everywhere

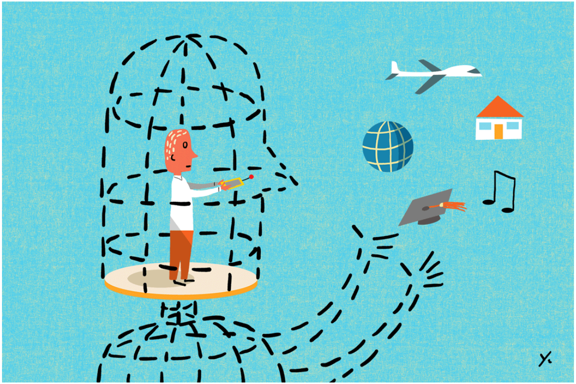


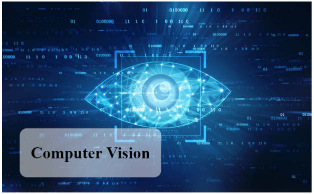
A.I. is Everywhere



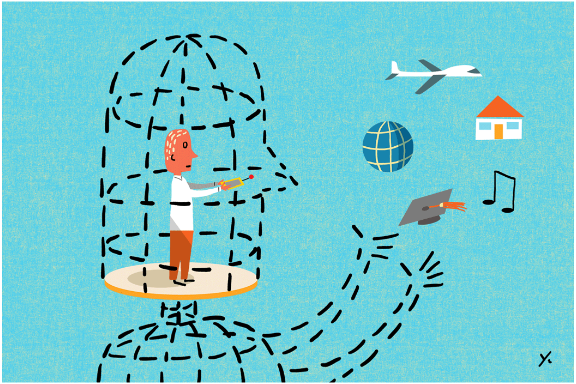


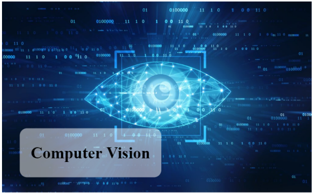
A.I. is Everywhere



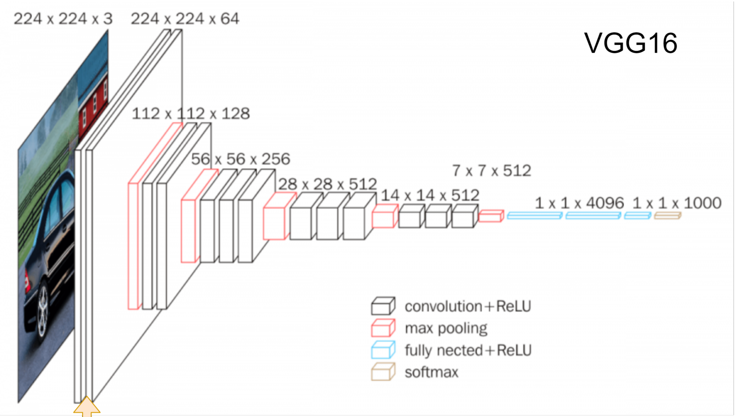


A.I. is Everywhere



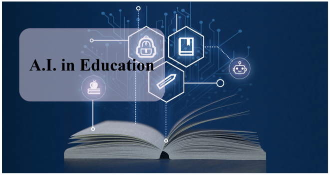


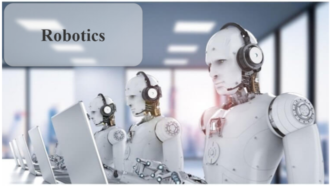
Deep Learning Requires Big Labeled Data



- Deep learning succeeds with abundant labeled data.

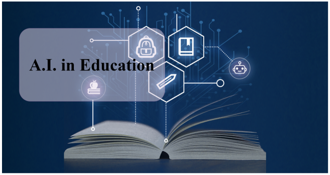
Emerging Application Domains Lack Data

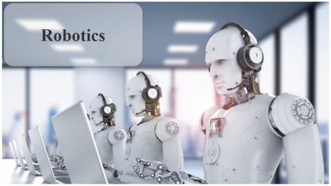




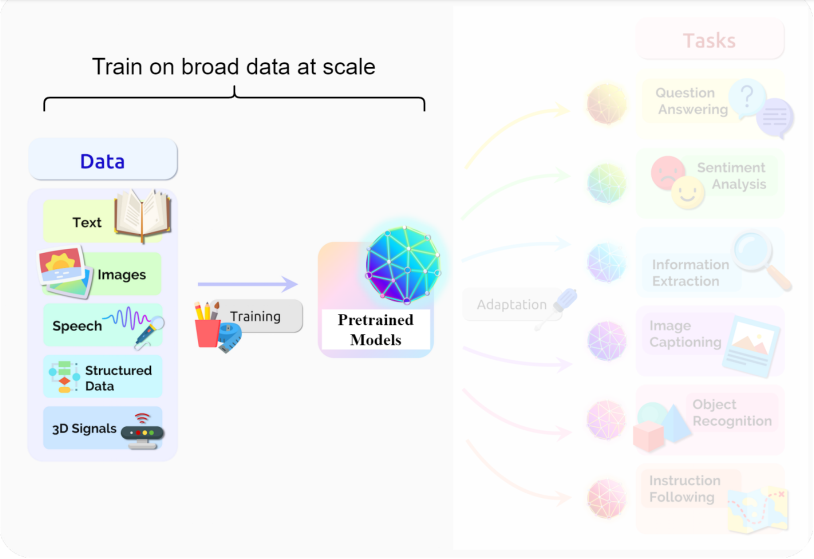
Emerging Application Domains Lack Data

Labeled data is lacking: significant costs in money and time.

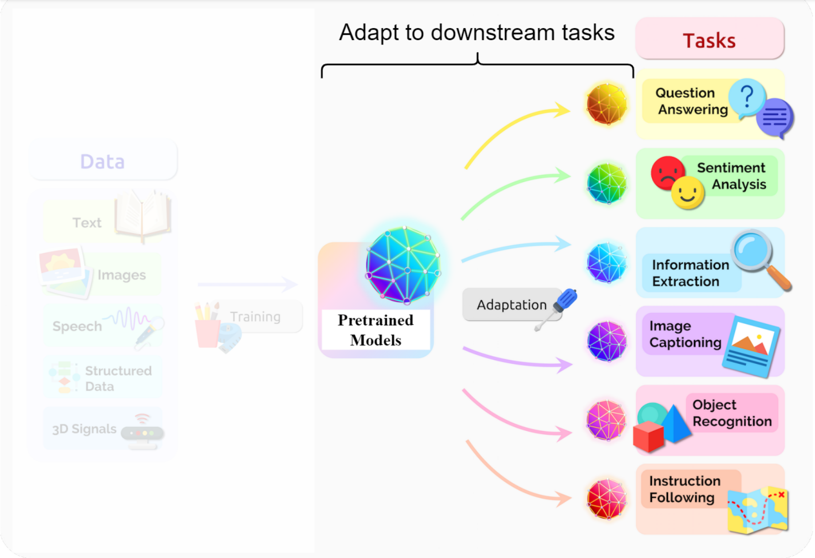




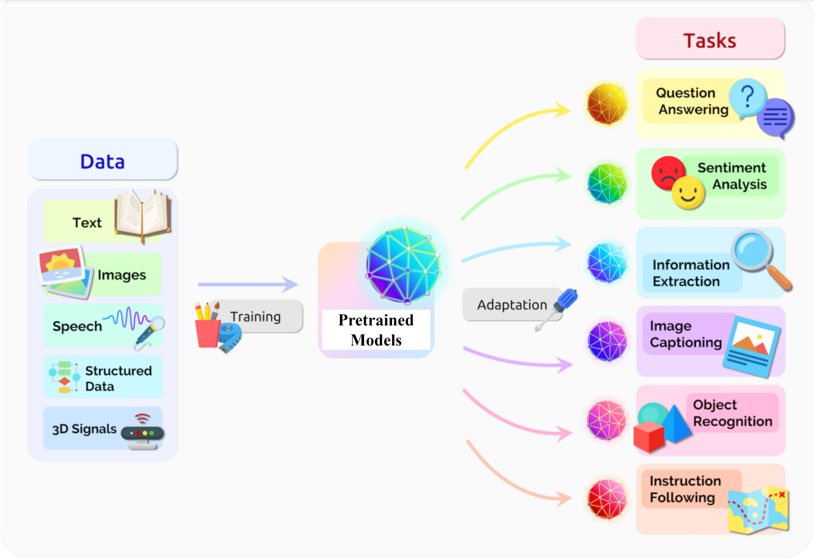
Pre-trained model: any models trained on broad data at scale and can be adapted to a wide range of downstream tasks.



Pre-trained model: any models trained on broad data at scale and can be adapted to a wide range of downstream tasks.



Pre-trained model: any models trained on broad data at scale and can be adapted to a wide range of downstream tasks.



AI is undergoing a paradigm shift with pre-trained models.

Examples:

- language models: BERT, GPT-3

SD Times

GPT-3 can now be customized to individual applications

Developers can now fine-tune GPT-3 on their own data, creating a custom version tailored to their application, which allows for faster and...

3 weeks ago

Figure source: Google News.

Pre-trained Model Expands the Capability of AI

AI is undergoing a paradigm shift with pre-trained models.

Examples:

- language models: BERT, GPT-3
- code generation: Codex, AlphaCode

Figure source: Google News.

Pre-trained Model Expands the Capability of AI

AI is undergoing a paradigm shift with pre-trained models.

Examples:

- language models: BERT, GPT-3
- code generation: Codex, AlphaCode
- multi-modal pre-trained models: DALL-E, CLIP

VB VentureBeat

OpenAI's text-to-image engine, DALL-E, is a powerful visual idea generator

DALL-E is a 12-billion parameter version of the 175 billion parameter GPT-3 natural language processing neural network. GPT-3 "learns" based on...

Jan 16, 2021

Figure source: Google News.

Power of Pre-trained Models

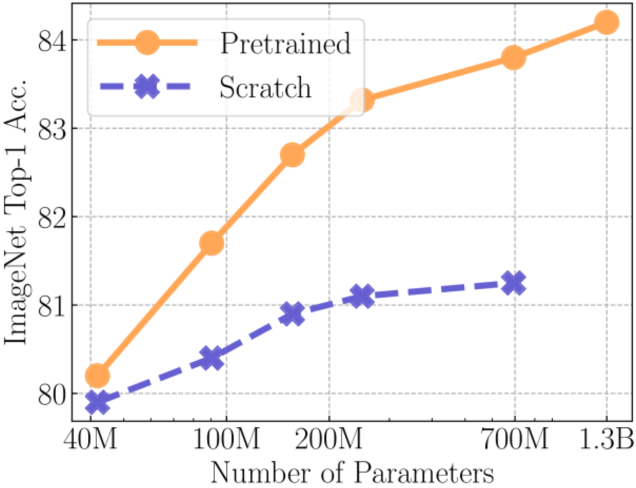


Figure source: Goyal et al. 2021.

Zero shot image classifier

FOOD101

guacamole (90.1%) Ranked 1 out of 101 labels

✓ a photo of **guacamole**, a type of food.

× a photo of **ceviche**, a type of food.

× a photo of **edamame**, a type of food.

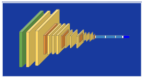
× a photo of **tuna tartare**, a type of food.

× a photo of **hummus**, a type of food.

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

Behind the Scenes

Training



Pre-trained Models

Adaptation

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

Theory's Role on Pre-trained models?

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

Challenge

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

- **Big labeled** dataset is necessary to fit deep networks.
- Training and testing data follow the same distribution.

Challenge

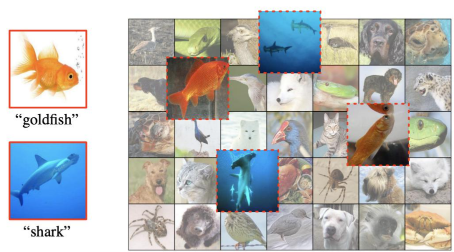
Current learning theory is more mature in **supervised learning**.

- **Big labeled** dataset is necessary to fit deep networks.
- Training and testing data follow the same distribution.

Our target

We want to understand how pre-trained model can

- adapt to new tasks **quickly**,



5-shot learning on ImageNet

Challenge

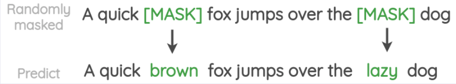
Current learning theory is more mature in **supervised learning**.

- **Big labeled** dataset is necessary to fit deep networks.
- Training and testing data follow the same distribution.

Our target

We want to understand how pre-trained model can

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



Challenge

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

- **Big labeled** dataset is necessary to fit deep networks.
- Training and testing data follow the same distribution.

Our target

We want to understand how pre-trained model can

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

Autonomous driving
Trained on sunny weather

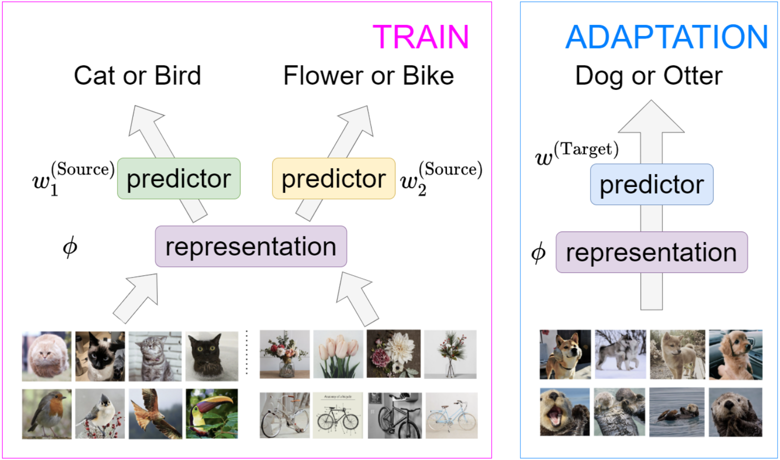
Transfer

Tested on rainy weather

Outline

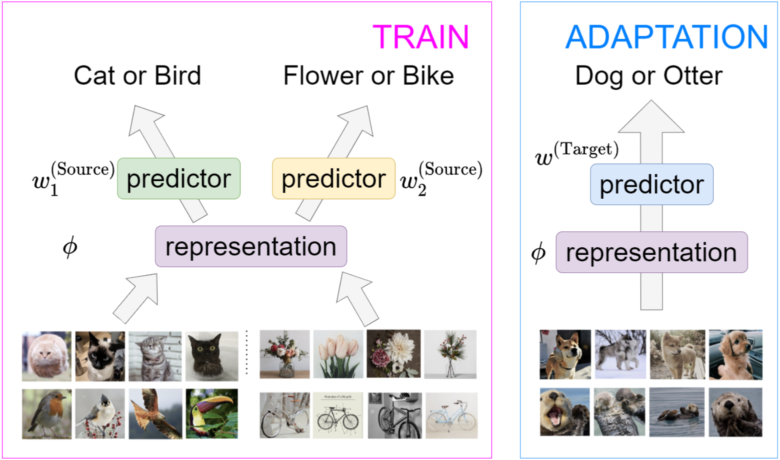
1. Meta-learning
   - Meta-Learning with Frozen Representation
   - Meta-learning with Fine-tuned Representation
2. Self-Supervised Learning
3. Ongoing and Future Work
   - Domain Adaptation
   - Lifelong Learning
   - Meta Reinforcement Learning

Learning the Meta-Representation



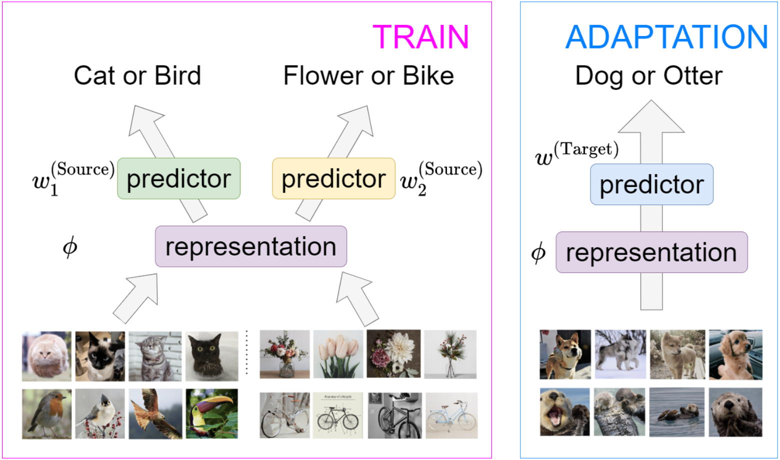
Prototypical network: (Snell et al. 2017), Meta-learning representation: (Javed and White, 2019)

Learning the Meta-Representation



On source tasks: $\hat{\phi} \leftarrow \arg\min_{\phi \in \Phi} \sum_{\text{source tasks } t} \left\{ \min_{w_t} \text{loss}(w_t \circ \phi) \right\}.$

Learning the Meta-Representation



$$\text{On target task: } \hat{w}^{(\text{Target})} \leftarrow \arg\min_{w} \text{loss}(w \circ \hat{\phi}).$$

How to Quantify Task Similarities?

Why does the learned representation transfer to target task?

How to Quantify Task Similarities?

Why does the learned representation transfer to target task?

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

How to Quantify Task Similarities?

Why does the learned representation transfer to target task?

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

Behind the scenes

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

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.)

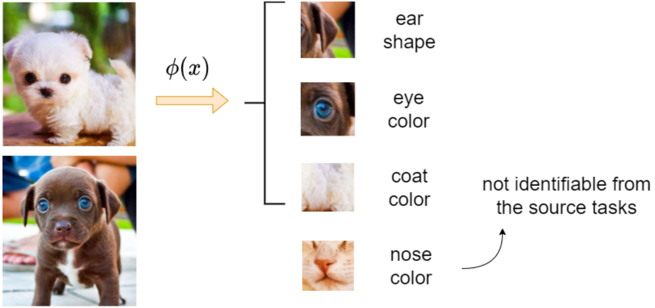
Task diversity

Source tasks:
Classify types of dogs

Target task:
Cat or dog?

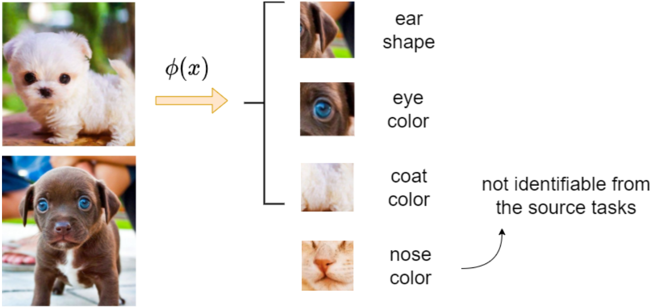
Importance of Task Diversity

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.)

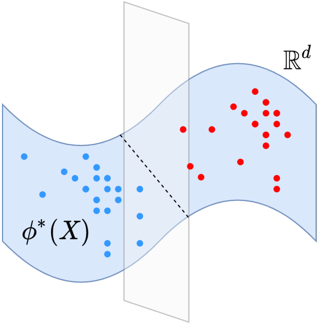


Mathematically speaking, $w^{\text{(Target)}} \in \text{span}\{w_1^{\text{(Source)}}, \cdots w_{n_e}^{\text{(Source)}}\}$.

General Low-dimensional Meta Representation

Setup:

- Shared representation: $y_t = w_t^\top \phi(x_t)$+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 small **test error**.

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

E.g., VGG19: 10^3 vs. 10^7, from Arora et al. 2018.

Main Result on Meta Representation

Theorem 1

With shared representation and task diversity,

$$\begin{aligned}&\text{Test Error}(\hat{w}^{(\text{Target})} \circ \hat{\phi})\\ \leq&\text{Representation Error} + \text{Adaptation Error}\end{aligned}$$

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

Main Result on Meta Representation

Theorem 1

With shared representation and task diversity,

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

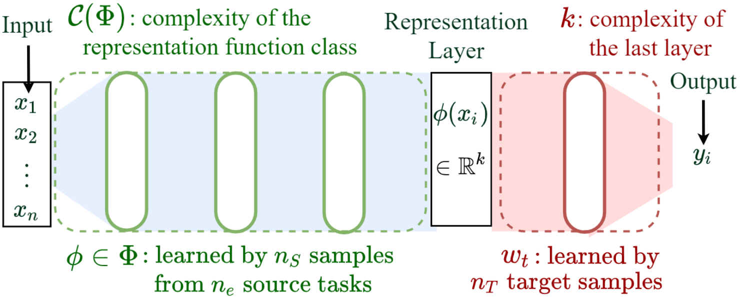
- Representation error: how well you learn representation layer ϕ
- Adaptation error: how well you learn target predictor $w^{(\text{Target})}$

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:

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

- Maurer et al. 2016:

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

Main Result on Meta Representation

Theorem 1

With shared representation and task diversity,

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

Meta-learning handles distributional shift:

- Covariate shift is allowed.
  Source and target data can come from different marginal distribution.

Theorem 1

With shared representation and task diversity,

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

Meta-learning handles distributional shift:

- Covariate shift is allowed.
  Source and target data can come from different marginal distribution.

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

Link to appendix

Outline

1. Meta-learning
   - Meta-Learning with Frozen Representation
   - Meta-learning with Fine-tuned Representation
2. Self-Supervised Learning
3. Ongoing and Future Work
   - Domain Adaptation
   - Lifelong Learning
   - Meta Reinforcement Learning

No Shared Representation?

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

No Shared Representation?

What if there is 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 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 γ-close to ϕ.

No Shared Representation?

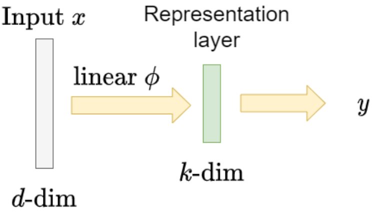
What if there is misspecification in the representation?
(Namely, the representation is only approximately shared.)

- Previously: $y_t = w_t^\top \phi(x)$+noise.
- Now: $y_t = w_t^\top \phi_t(x)$ + noise, ϕ_t is γ-close to ϕ.

Question

Does MetaRep (previous algorithm) still work?
If not, how should we modify the algorithm?

Instantiation in linear setting:



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

Failure with Previous Algorithm: MetaRep

Instantiation in linear setting:

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

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

Failure with Previous Algorithm: MetaRep

Instantiation in linear setting:

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

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

- No improvement over supervised learning that requires $O(d)$ samples.

Failure with Previous Algorithm: MetaRep

Instantiation in linear setting:

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

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

- No improvement over supervised learning that requires $O(d)$ samples.
- Previous algorithm is extremely fragile!

Modified Algorithm: FINE-TUNEDREP

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

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

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

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

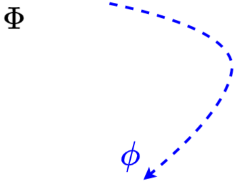
Φ

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

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 starting from ϕ that tolerates mis-specification.



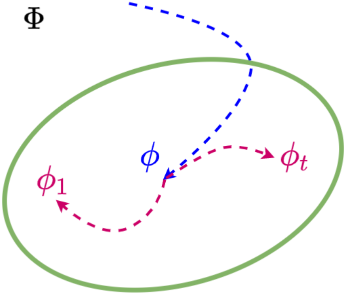
$$\min_{\phi} \min_{\|\phi_t - \phi\| \leq \gamma, w_t} \sum_{\text{Task } t} \text{loss}(w_t, \phi_t),$$

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 starting from ϕ that tolerates mis-specification.

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

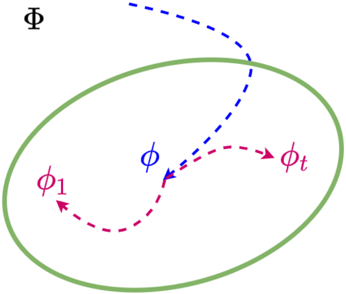


Model-agnostic Meta-learning: Finn et al. 2017

Modified Algorithm: Fine-tunedRep

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

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



Theorem 2 (Informal)

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

Main Result on Fine-tuned Representation

Theorem 2

For **general function classes**, under similar settings,

$$\text{Test Error} \lesssim \textsc{MetaRep}\ \text{Error (when } \gamma = 0) + O(\frac{\gamma}{\sqrt{n_T}}).$$

- γ 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

Link to appendix

Main Result on Fine-tuned Representation

Theorem 2

For **general function classes**, under similar settings,

$$\text{Test Error} \lesssim \textsc{MetaRep} \text{ Error (when } \gamma = 0) + O\left(\frac{\gamma}{\sqrt{n_T}}\right).$$

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

Baselines:

- Supervised learning and $\textsc{MetaRep}$ need $\mathcal{C}(w \circ \Phi)$ 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

Link to appendix

Outline

No labeled data?

Create your own labels

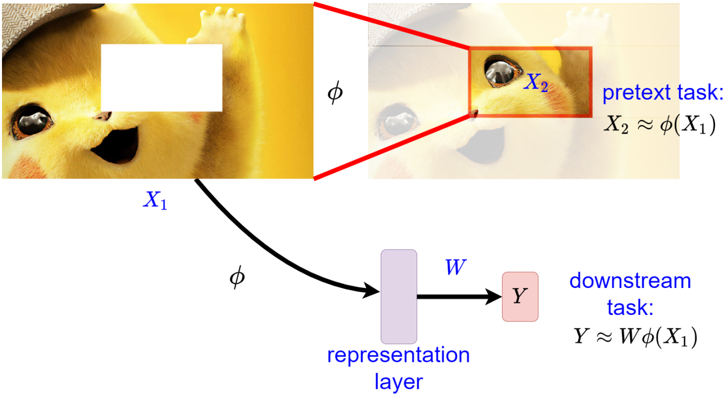
Supervised representation learning needs labels from related tasks. What if this isn't available?

Create pseudo-labels from the input data.

Self-supervised Learning

Type I: reconstruction-based SSL

Reconstructing part of the input from the other part



Context encoder: (Pathak et al. 2016)

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

Type I: reconstruction-based SSL

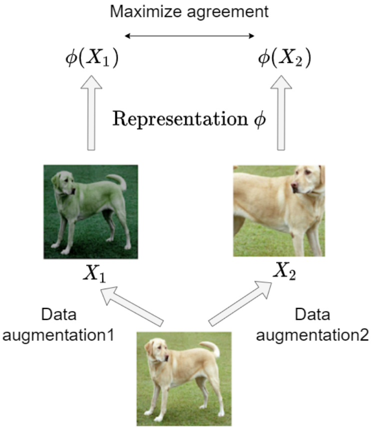
Reconstructing part of the input from the other part

BERT: (Devlin et al., 2018)

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

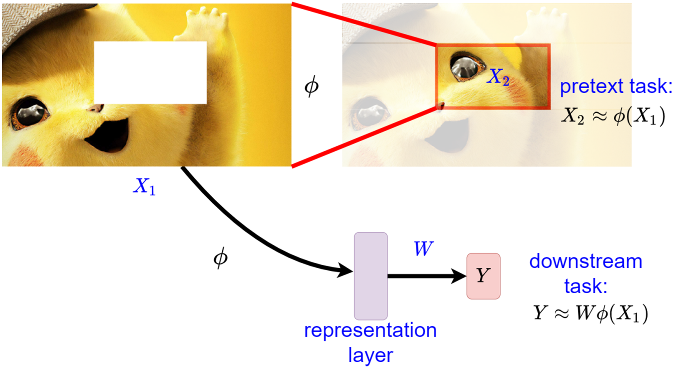
Type II: similarity-based SSL

Enforcing two views of the same data to have similar representation



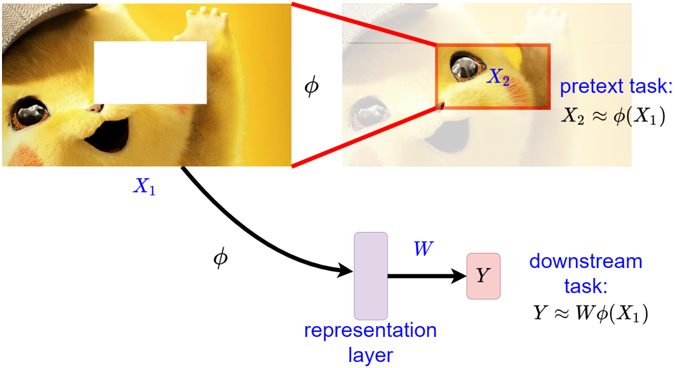
Examples: SimSiam: (Chen et al., 2021), CLIP: (Radford et al., 2021) , SimCLR: (Chen et al., 2020)

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



Setup

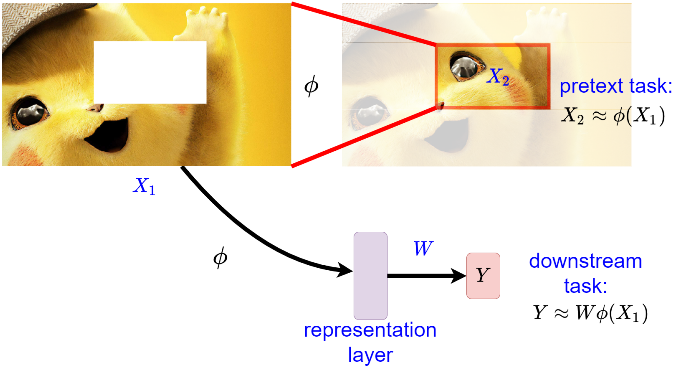
1. Label Y with k classes.
2. Unmasked image X_1 and masked image X_2.



3. Key intuition: Pretext tasks should help us reduce irrelevant features/forget information that is not necessary to predict Y

Ideal Scenario

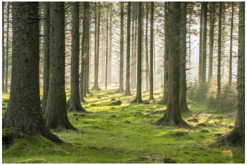
1. Label Y with k classes.
2. Unmasked image X_1 and masked image X_2.



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

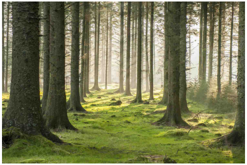
A Thought Experiment

- $X_1 \perp X_2 | Y$
- Image colorization for photos of desert, forest, sea

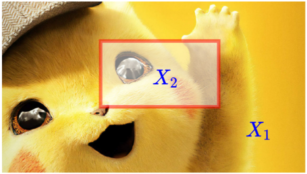


A Thought Experiment

- $X_1 \perp X_2 | Y$
- Image colorization for photos of desert, forest, sea



- Image inpainting:



Our Result: Ideal Scenario

Setting:

1. k-class labels Y.
2. Representation ϕ, last layer W^*.

Compare this procedure to ground truth classifier f^*:

Theorem 3

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

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

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.

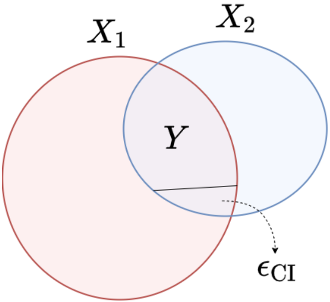
Proof sketch

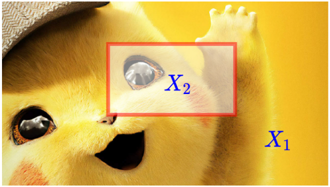
No representation error: if $X_1 \to Y \to X_2, $(i.e.,$X_1 \perp X_2 | Y$), then $f^* = W^* \phi(X_1)$.

$$\phi(\cdot) := \mathbb{E}[X_2|X_1] \overbrace{=}^{\text{tower property}} \mathbb{E}[\mathbb{E}[X_2|X_1, Y]|X_1] \overbrace{=}^{\text{CI}} \mathbb{E}[\mathbb{E}[X_2|Y]|X_1]$$

$$= \sum_{y=1}^{k} \mathbb{E}[X_2|Y=y] P(Y=y|X_1) =: \mathbf{A}^\top f(X_1),$$

Here $f(x_1)_y := P(Y=y|X_1=x_1), y = 1 \cdots, k$
$\mathbf{A}$ satisfies $\mathbf{A}_{y,:} = \mathbb{E}[X_2|Y=y]$.





Characterizing Approximate Conditional Independence

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

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

- 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)

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

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.

Characterizing Approximate Conditional Independence

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

- 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)

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

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

Main Results on Reconstruction-based SSL

Characterizing Approximate Conditional Independence

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

- 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 Implications

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

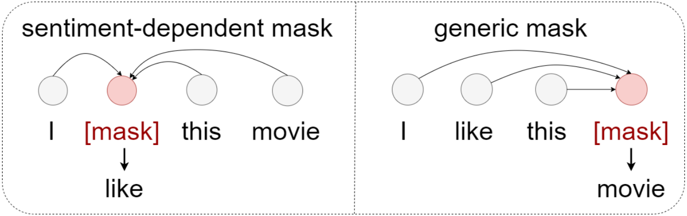
Empirical Implications

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]

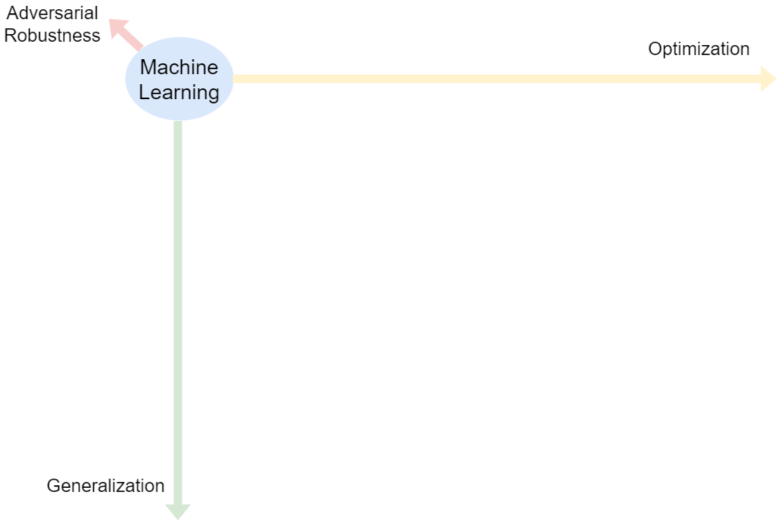


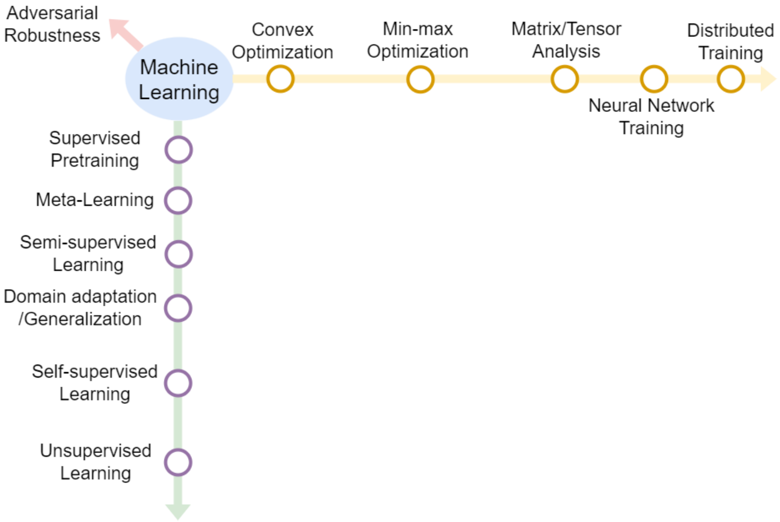
- Audio: speech recognition [Zaiem et al., 2021]

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

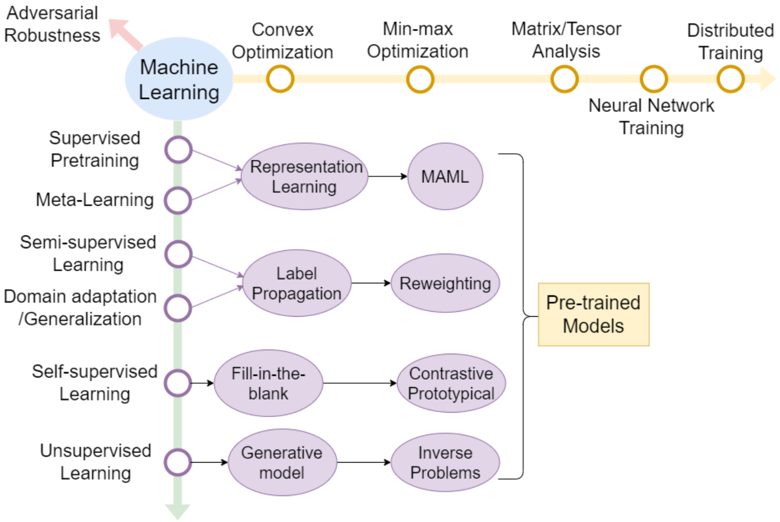
wrap-up

Overview of My Research Contributions

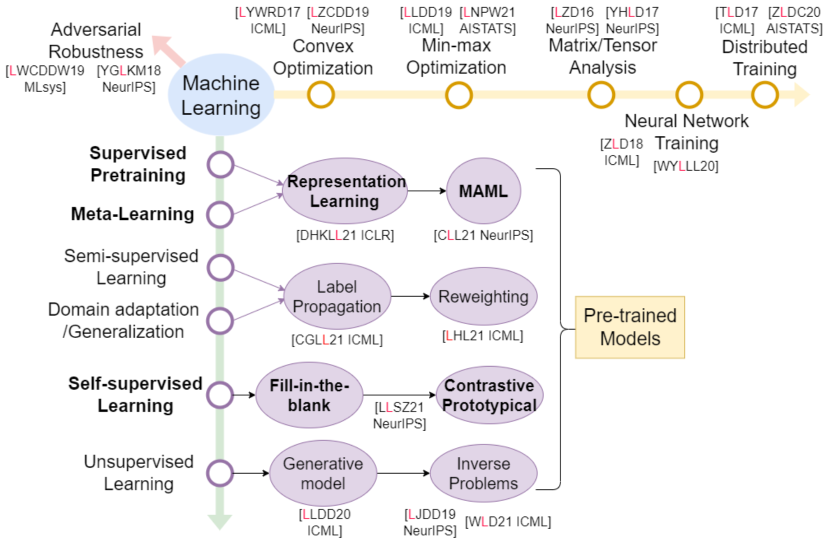



Adversarial Robustness
Machine Learning
Convex Optimization
Min-max Optimization
Matrix/Tensor Analysis
Neural Network Training
Distributed Training
Supervised Pretraining
Meta-Learning
Semi-supervised Learning
Domain adaptation /Generalization
Self-supervised Learning
Unsupervised Learning

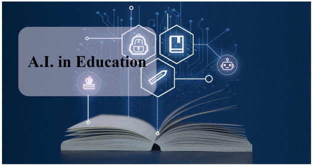
Overview of My Research Contributions



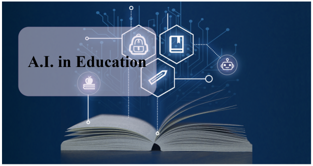
Overview of My Research Contributions



Outlook



Pre-trained model:
Expanding the capability of AI to broader disciplines.



Pre-trained model:
Expanding the capability of AI to broader disciplines.

I will work towards accelerating us to their "ImageNet" moment.

Remaining challenges:

- Optimization
  - Convergence analysis
  - Training and testing time speed up
- Robustness
  - Handle different types of distribution shift
  - Adversarial robustness (Security issues)
- Applications
  - Reinforcement learning
  - Medical image
- Trustworthy AI
  - Fairness, safety, privacy
  - Interpretability
- Evaluation
  - How to evaluate the model without target data
- Other technical details

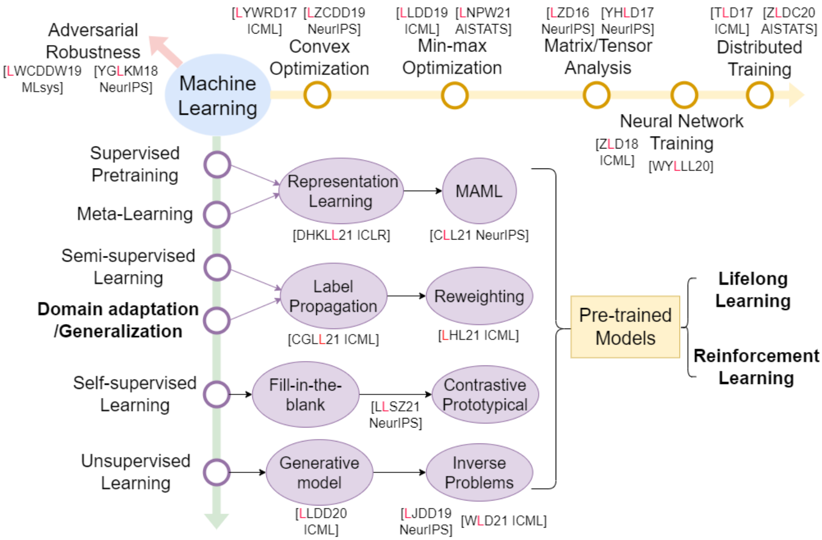
Remaining challenges:

- Optimization
  - Convergence analysis
  - Training and testing time speed up
- Robustness
  - Handle different types of distribution shift
  - Adversarial robustness (Security issues)
- Applications
  - Reinforcement learning
  - Medical image
- Trustworthy AI
  - Fairness, safety, privacy
  - Interpretability
- Evaluation
  - How to evaluate the model without target data
- Other technical details

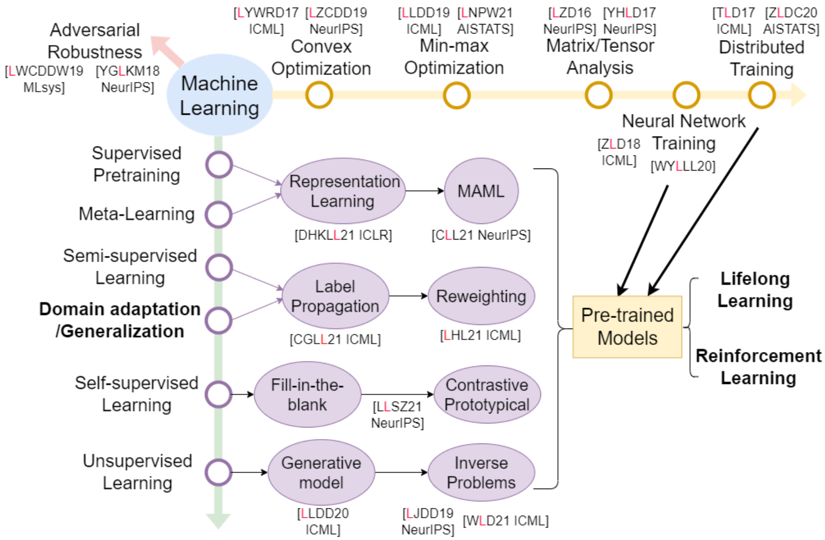
Outline

1. Meta-learning
   - Meta-Learning with Frozen Representation
   - Meta-learning with 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


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 and Future Work

We investigate sufficient conditions that guarantee:

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

Summary and Future Work

We investigate sufficient conditions that guarantee:

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

Future Work

Explore the following directions centered at pre-trained models:

- Domain Adaptation/Generalization
- Continual (Lifelong) Learning
- Meta Reinforcement Learning
- Optimization: Traning and Testing Time Speed-ups

Thank you!

Thank you!

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

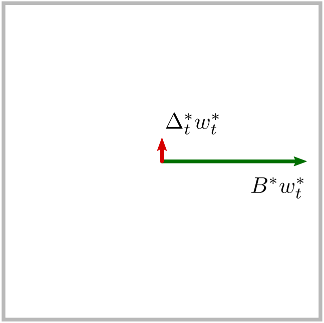
Failure with Previous Algorithm: MetaRep

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

Failure with Previous Algorithm: MetaRep

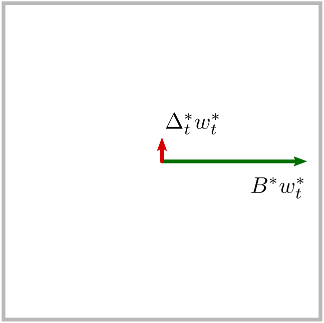
MetaRep chooses representation based on prediction space norm, not parameter space norm!

Parameter space (L_2-norm)

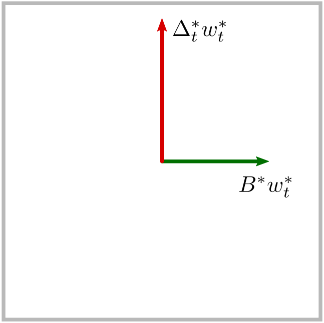


> MetaRep *chooses representation based on prediction space norm, not parameter space norm!*

Parameter space (L_2-norm)



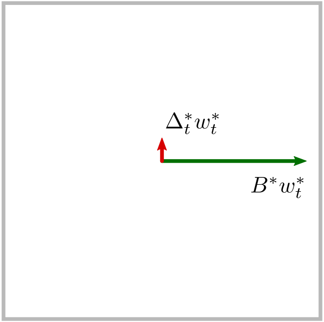
Prediction space (Σ-norm)



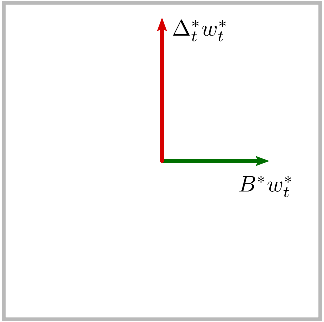
Failure with Previous Algorithm: MetaRep

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 $B^* w_{\text{test}}^*$.

Technical Assumptions for Theorem 1

Assumptions:

- I.i.d. Samples
- Input data is light-tail
- Shared representation
- Task diversity

Back

Assumptions:

- I.i.d. Samples
  - (x, y_t) generated i.i.d from each t-th task
- Input data is light-tail
  - P_X^t is sub-Gaussian
- Shared representation
  - $y_t = w_t \circ \phi(x) + \epsilon$, noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$
- Task diversity
  1. $w^{\text{target}} \in \text{span}\{w_t^{\text{source}}\}$
  2. $W = [w_1^{\text{source}}, \cdots w_{n_e}^{\text{source}}]$ is well-conditioned

Theory helps to diagnose the failure issue and determine the solution.

Potential problems:

1. No shared structure

2. Learning objective for meta-learning is not effective

3. Optimization issues

> Theory helps to diagnose the failure issue and determine the solution.

Potential problems:

1. No shared structure
   - No supervised pre-training algorithm will work. Need to switch to self-supervised pre-training.
2. Learning objective for meta-learning is not effective
3. Optimization issues

Empirical Implication

Theory helps to diagnose the failure issue and determine the solution.

Potential problems:

1. No shared structure
2. Learning objective for meta-learning is not effective
   - Switch from learning the meta-representation to MAML-like algorithms.
3. Optimization issues

Back

Empirical Implication

Theory helps to diagnose the failure issue and determine the solution.

Potential problems:

1. No shared structure
2. Learning objective for meta-learning is not effective
3. Optimization issues
   - Typical optimization tricks.

Back

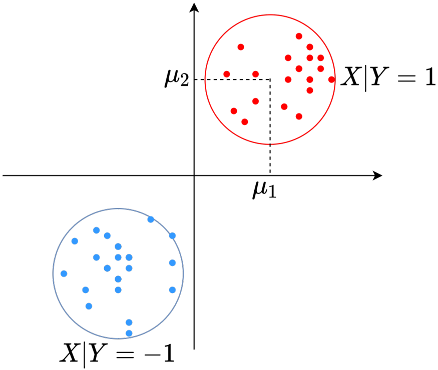
General Functions

Theoretical results:

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

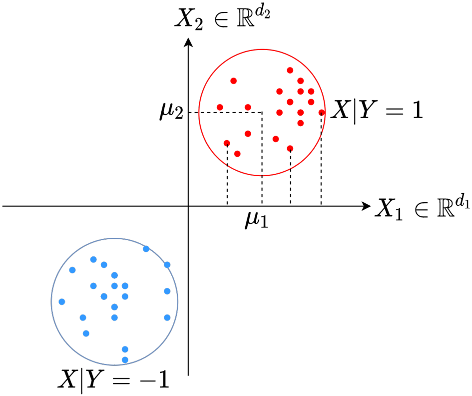
How can $\phi(X_1)$ be better than X_2 to predict Y?

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



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

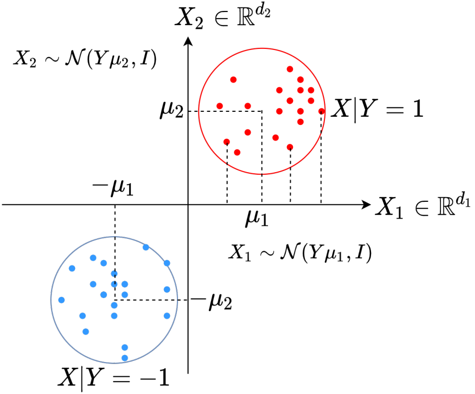
$X \sim \mathcal{N}(Y[\mu_1, \mu_2], I_{d_1+\mathrm{d}_2})$

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

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

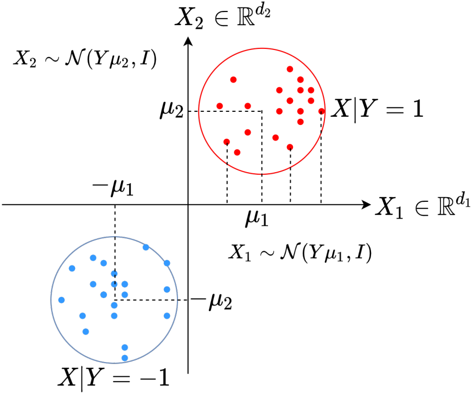
$X_1 \perp X_2 | Y$

Example: Gaussian Mixture



$Y \in \{-1, 1\}$
$X \sim \mathcal{N}(Y[\mu_1, \mu_2], I_{d_1+d_2})$
$X_1 \sim \mathcal{N}(Y\mu_1, I_{d_1})$,
$X_2 \sim \mathcal{N}(Y\mu_2, I_{d_2})$
$X_1 \perp X_2 | Y$
$\mathbb{E}[Y|X_2]$ is not linear, but
$\mathbb{E}[Y|\phi]$ is linear:

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

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

Connection to SimSiam method

- Before, we learn $\phi(x_1) = \mathbb{E}[X_2|X_1 = x_1]$, 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):

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

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

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

Algorithm (SimSiam):

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

New measure of conditional independence:

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

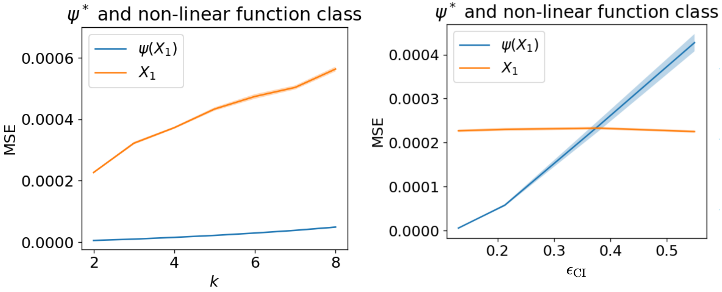
- Extension of previous result:

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

- 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:

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

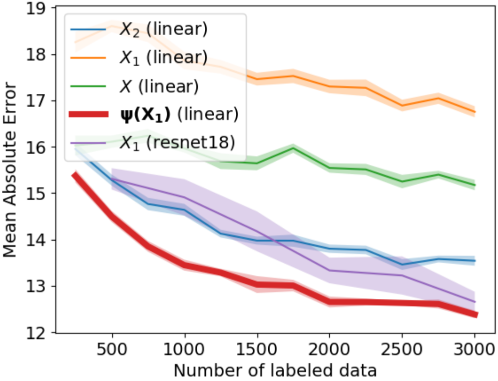
Simulations: Both Terms Tight in $\frac{k}{n_L} + \epsilon_{\text{CI}}$



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

Experiments



- Yearbook: portraits date from 1905 to 2013.

Source

Target

Entity30 - Passerine

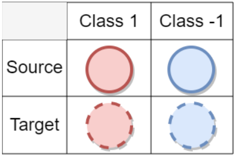
Source

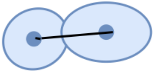
Target

Entity30 - Tableware

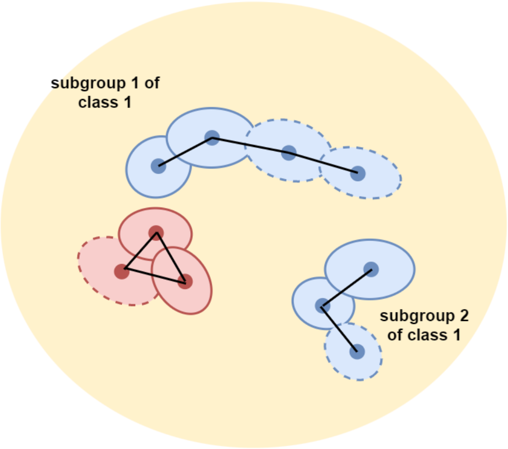
BREEDS dataset: (Santurkar et al., 2021)

Our New Framework: Subpopulation Shift

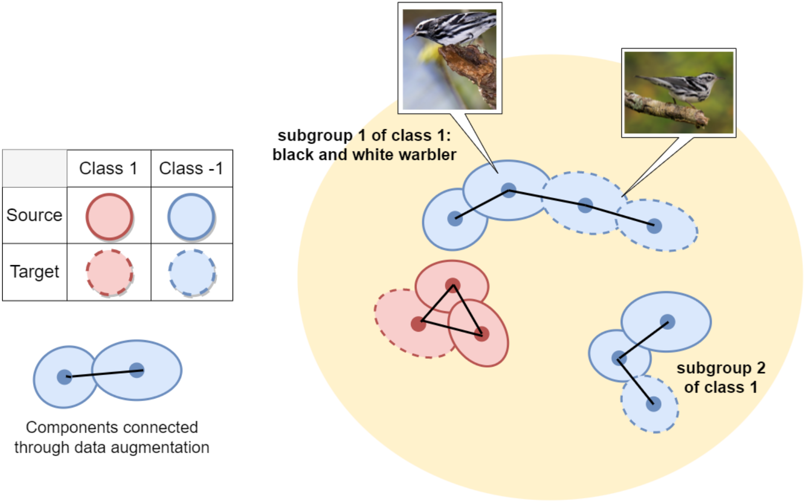




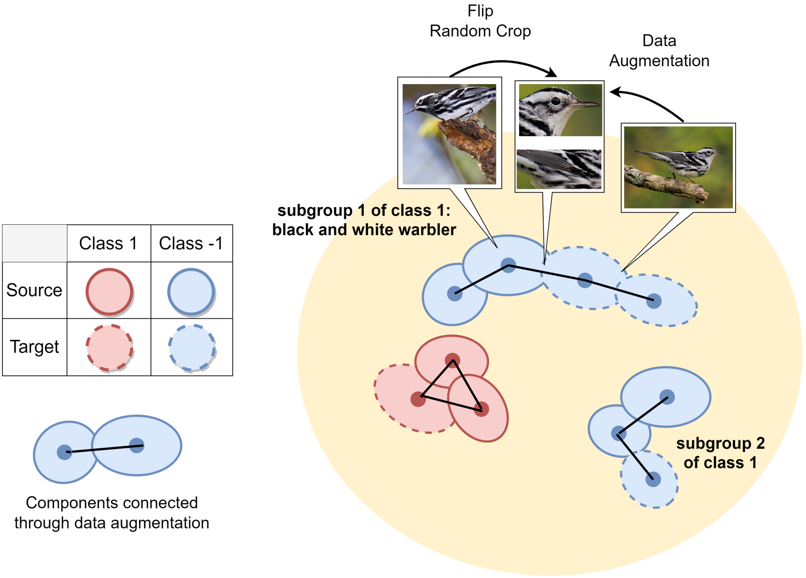
Components connected through data augmentation



Our New Framework: Subpopulation Shift



Our New Framework: Subpopulation Shift



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[1] 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.

[1] MDD: (Zhang et al. 2019)

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