Few-shot Learning via Learning the Representation, Provably

Qi Lei
Princeton University

Joint work with
Simon Du, Wei Hu, Sham Kakade and Jason Lee.

https://arxiv.org/abs/2002.09434

Representation Learning

Representation Learning

Deep Learning's success is due to learning useful representations.

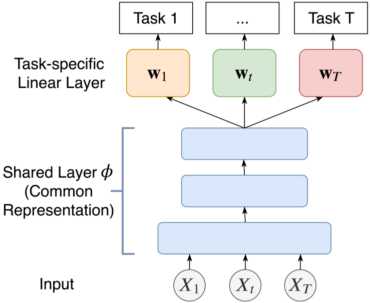
- Deep networks **learns the feature representation**.
- Feature representations transfer to other tasks.
- Competing methods lack transferrability.

Competing methods are unable to do this (random forests, kernel machines, gradient boosting)

How is deep representation learning done?

Simplest algorithm:

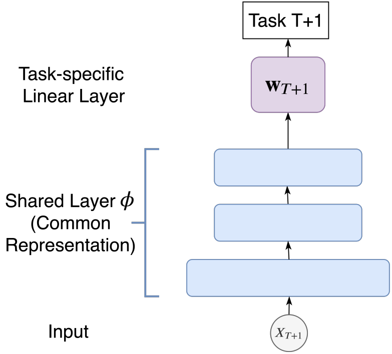
1. Train a deep network on some task (can be on Imagenet or self-supervised task).
2. Keep only the body, and discard the head.



How is deep representation learning done?

Simplest algorithm:

1. Train a deep network on some task (can be on Imagenet or self-supervised task).
2. Keep only the body, and discard the head.
3. Retrain the head (or finetune) using labeled data from target domain.



Ideal Representation Learning

- Learn $f_t(x) = g_t \circ \phi(\boldsymbol{x})$ with $g_t \in \mathcal{G}, \phi \in \Phi$ for each task $t = 1, 2, \cdots T$.
- Task-specific layer $g_t(\boldsymbol{x}) = \boldsymbol{w}_t^\top \boldsymbol{x}$
- Representation class Φ can be large and complex
- n_S samples from each source task, n_T samples for the target task

Goal

Under natural assumptions,

$$\text{Risk} \lesssim \frac{\mathcal{C}(\Phi)}{n_S T} + \frac{\mathcal{C}(\mathcal{G})}{n_T}.$$

$\mathcal{C}(\mathcal{F})$ is the complexity of the function class $\mathcal{F}$.
Without these assumptions, such a rate is not attainable.

How do attain ideal rate

Assumptions:

1. Shared good representation across tasks: $y_t = w_t^\top \phi^*(x) +$noise.
2. Diversity of source tasks $\{w_t\}$.

Why?

1. Shared representation encodes what transfers across the tasks.
2. Diversity of the $\{w_t\}$ (at least needs to "cover" w_{T+1}.)

Notation:

- $\phi(x) = \mathbb{R}^d \to \mathbb{R}^k$ selects k features.
- $y_t \approx (w_t^*)^\top \phi^*(x)$

Algorithm:

- For Source Tasks:

$$\hat{\phi} \leftarrow \underset{\phi\in\Phi,\boldsymbol{w}_1,\ldots,\boldsymbol{w}_T}{\text{minimize}} \frac{1}{2n_S T}\sum_{t=1}^{T}\left\|\boldsymbol{y}_t - \phi(X_t)\boldsymbol{w}_t\right\|^2.$$

- For Target Task: train a linear predictor on top of $\hat{\phi}$:

$$\hat{\boldsymbol{w}}_{T+1} \leftarrow \underset{\boldsymbol{w}_{T+1}\in\mathbb{R}^k}{\text{minimize}} \frac{1}{2n_T}\left\|\boldsymbol{y}_{T+1} - \hat{\phi}(X_{T+1})\boldsymbol{w}_{T+1}\right\|^2.$$

General Low-dim Representation Learning

Theorem

With shared representation and task diversity assumptions,

$$\textit{Risk} \lesssim \frac{\mathcal{C}(\Phi)}{n_S T} + \frac{k}{n_T}.$$

- Covariate shift is allowed.
- $\phi : \mathbb{R}^d \to \mathbb{R}^k$ selects k most important features

Two-layer Neural Network Representations

- Representation is no longer low dimensional: $\phi : \mathbb{R}^d \to \mathbb{R}^m : \boldsymbol{x} \to (B^\top \boldsymbol{x})_+$
- Complexity of function class is controlled by the parameters' norm
- Source Tasks with Weight decay:

$$\hat{B}, \hat{W} = \underset{\substack{B \in \mathbb{R}^{d\times m}, \\ W = [\boldsymbol{w}_1, \cdots, \boldsymbol{w}_T] \in \mathbb{R}^{m \times T}}}{\arg\min} \frac{1}{2n_S T} \sum_{t=1}^{T} \|\boldsymbol{y}_t - (X_t B)_+ \boldsymbol{w}_t\|^2 + \frac{\lambda}{2}\|B\|_F^2 + \frac{\lambda}{2}\|W\|_F^2.$$

- Training on target task:

$$\hat{\boldsymbol{w}}_{T+1} \leftarrow \underset{\|\boldsymbol{w}\| \le r}{\arg\min} \frac{1}{2n_T} \|\boldsymbol{y}_{T+1} - (X_{T+1}\hat{B})_+ \boldsymbol{w}\|^2.$$

Neural Network Excess Risk

Theorem

With shared representation and task coverage assumption:

$$\textit{Risk} \leq \sqrt{\frac{\textit{Rademacher}(\Phi)}{n_S T}} + \sqrt{\frac{\|w^*\|}{n_T}}$$

Future Directions.

- Provable **algorithms** for representation learning (preferably SGD).
- Finetuning, refine the representation when n_T is larger.

Thank you!