DS-GA 3001: Modern Topics on Statistical Learning Theory Spring 2024

Lecture 8 Meta Representation Learning — Mar. 14, 2024

Prof. Qi Lei *Scribe: Marco Li*

1 Motivation: Why Meta learning?

For average human, it is often easy to use prior knowledge in learning to perform new task, like we do when distinguishing objects using features extracted from visual information. As this is shown an advantage of human in learning, it is a problem how we can use prior information/ prior task to train AI/ML models to be more sample efficient.

The answer is we can do this through Meta learning, where we train a model on different learning tasks such that it can resolve new tasks using only a few number of samples. In this lecture, we will lay our focus on Meta-Representation Learning, which transfers knowledge of feature representation (more details in section 3).

2 The Problem

In this lecture, we focus on the few-shot classification problem:

$$Input : \{D_1^S, D_2^S, ..., D_e^S, D^T\}$$

S refers to source tasks (prior tasks), T refers to a target task (new task), and e the number of source tasks.

$$D_i^S = \{(x_j^i, y_j^i)\}_{j=1}^{n_S}$$

$$D^T = \{(x_j^T, y_j^T)\}_{j=1}^{n_T}$$

n_S and n_T are respectively the number of samples per source/target task. In few shot problem, it is often the case that $n_S >> n_T$. We have $y_i \in [L]$ for some integer L as the set of predictions and call it a L-way classification problem.

3 How to utilize source tasks?

How to use the structure learned from D^S to solve the target task more sample-efficiently? We can learn meta-parameter θ such that

$$f_i = \mathcal{A}(D_i^S, \theta)$$

with $\mathcal{A}$ the base learner is a good task predictor for D_i^S, and $\mathcal{A}(D^T, \theta)$ is good for D^T.

Procedures:

1. Train on source tasks.
$$\hat{\theta} = \arg\min \Sigma_{i=1}^{e} \hat{L}(D_i^S, f_i)$$
$$f_i = \mathcal{A}(D_i^S, \theta)$$
2. Train and test on target task.
$$f^T = \mathcal{A}(D^T, \hat{\theta})$$

Methods:

- **Find a shared initialization**

  There are two popular ways to do this.

  1. **Pretrain + Finetune**. Pretrain a model first and finetuned the learned θ for specific tasks (most common in transfer learning).
  2. **MAML** (model-agnostic meta learning) [3]. While training the meta parameter θ on the outer loop, spend a few gradient steps (maybe 5 or 10) on the inner loop to train the specific predictors and gather the loss.

- **Find a shared representation**

  This is what we call Meta Representation Learning. An example of this is the **prototypical network** [4], further discussed in the next section.

4 Meta Representation Learning: Prototypical network [4]

Training: There are two types of parameters that we train here.

- shared ϕ intended for learning representation
- task specific parameters h_t for different tasks

Prediction: The prediction is given by the composite function $h_t \circ \phi$.

In terms of neural networks, ϕ are often the early layers and h are the latter layers (or prediction head). There is one h_i for each task.

Some theory:

- **Source tasks**
$$\hat{\phi} \leftarrow \min_{\phi} \min_{h_1, h_2, \ldots, h_e} \Sigma_{t=1}^{e} \hat{L}_t^S(h_t \circ \phi)$$
  We want to minimise the sum of loss from all source tasks.

- **Target Task**
$$\hat{\phi} \leftarrow fixed$$
$$\hat{h} \leftarrow \min_{h} \hat{L}^T(h \circ \phi)$$
  Then, we fix $\hat{\phi}$, which represents the prior knowledge, and train $\hat{h}$ for prediction of the target task.

Remark:

- The purpose of learned $\hat{\phi}$ is to make it adapt to new task quickly (sample-efficiently).
- This depends on whether the optimal prediction function (for target) f^T can be expressed as one simple function (h^T) composed with $\hat{\phi}$.

  eg. With ϕ the identity function, it is not useful because it doesn't change x. With ϕ a constant, it is not expressive since we lost all the information from x. Both are bad functions for getting f^T with $h \circ \phi$.

5 Generalizer gap(excess risk)

We are interested in how good this method by evaluating its distance from the best possible function for this task. Previously, we have bias-variance reduction:

$$L^T(\hat{f}) - L_*^T = \underset{\text{estimation error}}{L^T(\hat{f}) - \min_{f \in \mathcal{F}} L^T(f)} + \underset{\text{approximation error}}{\min_{f \in \mathcal{F}} L^T(f) - L_*^T}$$

Now with $\hat{\phi}$ fixed, $f = h \circ \hat{\phi}$. We have

$$L^T(\hat{f}) - L_*^T = \underset{\text{III}}{L^T(\hat{h \circ \hat{\phi}}) - \min_{h \in \mathcal{H}} L^T(h \circ \hat{\phi})} + \underset{\text{II}}{\min_{h \in \mathcal{H}} L^T(h \circ \hat{\phi}) - \min_{f \in \mathcal{F}} L^T(f)} + \underset{\text{I}}{\min_{f \in \mathcal{F}} L^T(f) - L_*^T}$$

where III is called the estimation error or adaptation error, II is called approximation error due to fixed representation or representation error, and I is called approximation error due to model architecture.

”Few Shot Learning via Learning the Representation, Provably”[1] provides a study of the specific value:

Assumption:

1. Shared representation exists

   $$\exists \phi^* \in \Phi \text{ s.t. } f_i^* = h_i \circ \phi^*,\ h_i \in \mathcal{H}$$

2. Task diversity: If $h_1, ..., h_e$ are linear, it requires $h^T \in span(h_1, ..., h_e)$. In other words, all useful features for the target task T need to be trained while training for the source tasks S.

   e.g. If $h_1, ...h_e \in span(e_1, ...e_{10})$, but $h^T \in span(e_{11})$, there is no way we can learn the direction of $\phi_1 1$ from the source tasks.

   On the other hand, h^T shouldn't just use a small proportion of $\hat{\phi}$ learned.

   In other words, $\sigma_{min}([h_1, ..., h_e])$ and $\sigma_{max}([h_1, ..., h_e])$ should be of similar order.

Thus,

$$\text{the Generalizer gap} = \text{I} + \text{II} + \text{III}$$

$$= 0 + \frac{\mathcal{C}(\Phi)}{n_s \cdot e} + \frac{\mathcal{C}(\mathcal{H})}{n_T}$$

$n_s \cdot e$ is the number of all samples from the source tasks. n_T is the number of samples from target tasks. I is 0 based on the assumptions. Compared to $\frac{\mathcal{C}(\mathcal{H}\circ\Phi)}{n_T}$ the gap of learning from scratch, this is much better.

6 Problem

It's a problem that assumption one in the previous section is too ideal, especially when there is some "mis-specification", meaning ϕ doesn't perfectly represent f_i.

If we learn the representation as before despite misspecification, the loss for the source:

$\min_{\phi} \min_{h_1,\ldots,h_e} \Sigma_{t=1}^{e} \hat{L_t^S}(h_t \circ \phi)$ can be arbitrarily bad. Thus, it is essitial to use MAML [3] and allow adaptivity in the source learning steps. We have the source training as follows:

$$\hat{\phi} \leftarrow \min_{\phi} \min_{h_1,\ldots,h_e} \Sigma_{t=1}^{e} \hat{L_t^S}(h_t \circ \phi_t)$$

where for each task we have an adapted ϕ_t and $\|\phi_t - \phi\| \leq \delta$ for some δ. In this way, we have

$$\text{generalizer gap} \leq 0 + \frac{\mathcal{C}(\Phi)}{n_s \cdot e} + \frac{\mathcal{C}(\mathcal{H})}{n_T} + \frac{\delta}{\sqrt{n_T}}$$

[2]

References

[1] Kurtland Chua, Qi Lei, and Jason D. Lee. How fine-tuning allows for effective meta-learning, 2021. https://arxiv.org/abs/2105.02221 `arXiv:2105.02221`.

[2] Simon S. Du, Wei Hu, Sham M. Kakade, Jason D. Lee, and Qi Lei. Few-shot learning via learning the representation, provably, 2021. https://arxiv.org/abs/2002.09434 `arXiv:2002.09434`.

[3] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017. https://arxiv.org/abs/1703.03400 `arXiv:1703.03400`.

[4] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning, 2017. https://arxiv.org/abs/1703.05175 `arXiv:1703.05175`.