Learning to Stop While Learning to Predict

Chen Zhao

ICML 2020 https://arxiv.org/pdf/2006.05082.pdf

1 Motivation

Intuitively, for deep learning models, the optimal depth (or the optimal number of steps to operate on an input) can also be different for different input instances, either because we want to compute less for operations converged already, or we want to generalize better by avoiding "over-thinking".

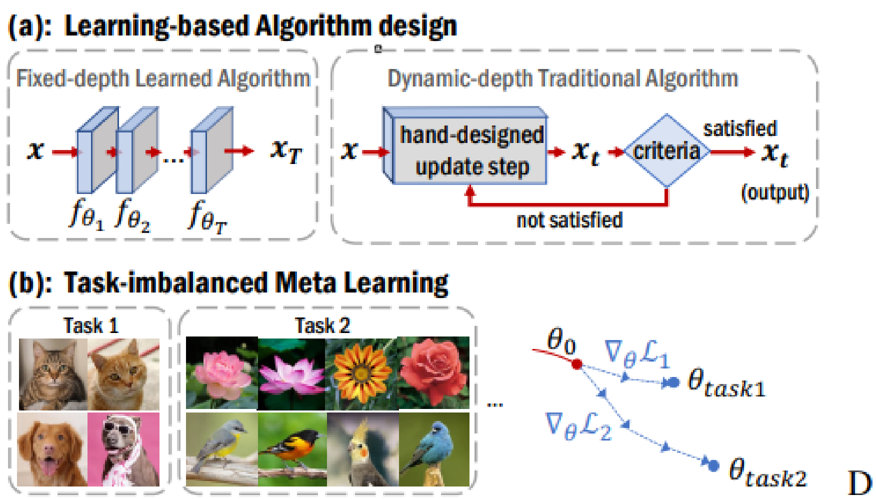


Figure 1: Motivation for learning to stop.

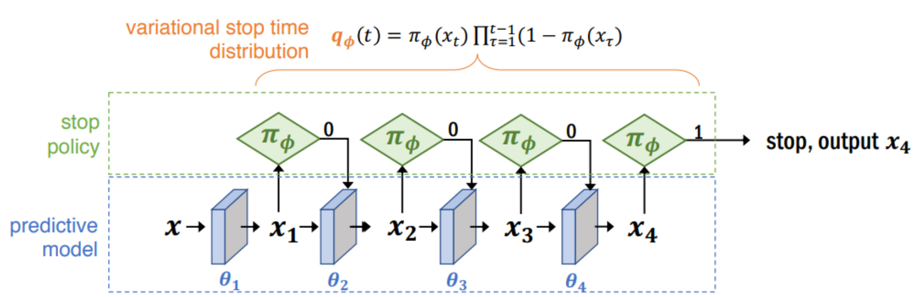
- Task-imbalanced Meta Learning: Need different numbers of gradient steps for adaptation.
- Data-driven Algorithm Design: Traditional algorithms have certain stop criteria to determine the number of iterations for each problem, *e.g.* iterate until convergence and early stopping to avoid over-fitting. Deep

learning based algorithms usually have a fixed number of iterations in the architecture.

- Image Denoising: Images with different noise levels may need different number of denoising steps.
- Image Recognition: 'early exits' is proposed to improve the computation efficiency and avoid 'over-thinking'.

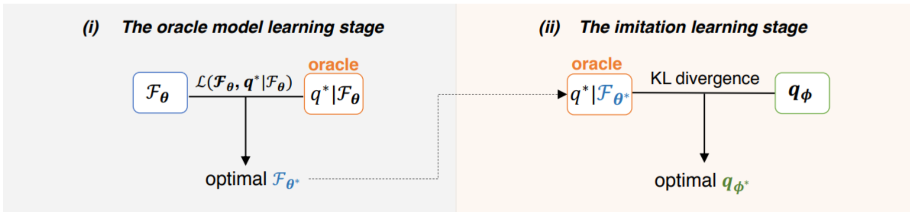
2 Predictive Model with Stopping Policy

- Predictive model $F(\theta)$
  - Transforms the input $\mathbf{x}$ to generate a path of states $\mathbf{x}_1, ..., \mathbf{x}_T$.
  - $\mathbf{x}_t = f_{\theta_t}(\mathbf{x}_{t-1})$, for $t = 1, 2, ..., T$
- Stopping Policy π_ϕ
  - Sequentially observes the states $\mathbf{x}_t$ and determines the probability of stop at layer t.
  - $\pi_t = \pi_\phi(\mathbf{x}, \mathbf{x}_t)$, for $t = 1, 2, ..., T$
- Variational stop time distribution q_ϕ
  - Stop time distribution induced by stopping policy π_ϕ.
  - $q_\phi(t) = \pi_t \prod_{\tau=1}^{t-1}(1 - \pi_\tau)$, for $t < T$



3 How to learn the optimal (F_θ, π_ϕ) efficiently?

- Design a joint training objective: $\mathcal{L}(F_\theta, q_\phi; x, y) = \mathbb{E}_{t\sim q_\phi} l(y, x_t; \theta) - \beta H(q_\phi)$.
- Introduce an oracle stop time distribution: $q^*|F_\theta := \arg\min_{q\in\delta^{T-1}} \mathcal{L}(F_\theta, q)$.
- Then we decompose the learning procedure into two stages: (1) the oracle model learning stage; (2) the imitation learning stage.



4 Training Algorithm – Stage I

Oracle stop time distribution:

$$q_\theta^*(\cdot|y,x) = \arg\max_{q\in\delta^{T-1}} \mathcal{J}_{\beta-VAE}(F_\theta, q; x, y) = \frac{p_\theta(y|t,x)^{1/\beta}}{\sum_{t=1}^T p_\theta(y|t,x)^{1/\beta}}$$

It is the optimal stop time distribution given a predictive model F_θ. When $\beta = 1$, the oracle is the true posterior, $q_\theta^*(t|y,x) = p_\theta(t|y,x)$. This posterior is computationally tractable, but it requires the knowledge of the true label y.

Stage I. Oracle model learning

$$\max_\theta \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \mathcal{J}_{\beta-VAE}(\mathcal{F}_\theta, q_\theta^*; x, y) = \max_\theta \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \sum_{t=1}^{T} q_\theta^*(t|y,x) \log \underbrace{p_\theta(y|t,x)}_{\text{likelihood of the output at } t\text{-th layer}}$$

5 Training Algorithm – Stage II

Recall the variational stop time distribution $q_\theta(t|x)$ induced by the sequential policy π_ϕ. We hope $q_\phi(t|x)$ can mimic the oracle distribution $q_{\theta*}^*(t|y,x)$, by optimizing the forward KL divergence.

Stage II. Imitation With Sequential Policy

forward KL divergence $\quad \mathrm{KL}(q_{\theta^*}^* || q_\phi) = -\sum_{t=1}^{T} q_{\theta^*}^*(t|y,x) \log q_\phi(t|x) - H(q_{\theta^*}^*)$

6 Advantages of the training procedure

- Principled: Two components are optimized towards a joint objective.

- Tuning-free: (1) Weights of different layers in the loss are given by the oracle distribution automatically; (2) For different input samples, the weights on the layers can be different.

- Efficient: Instead of updating θ and ϕ alternatively, θ is optimized in 1st stage, and then ϕ is optimized in 2nd stage.

- Generic: It can be applied to a diverse range of applications.

- Better understanding: (1) A variational Bayes perspective, for better understanding the proposed model and joint training; (2) A reinforcement learning perspective, for better understanding the learning of the stop policy.

Algorithm 1 Overall Algorithm

Randomly initialized θ and ϕ.

For $itr = 1$ *to #iterations* **do** ▷ Stage I.

  Sample a batch of data points $\mathcal{B} \sim \mathcal{D}$.

  Take an optimization step to update θ towards the marginal likelihood function defined in Eq. 9.

For $itr = 1$ *to #iterations* **do** ▷ Stage II.

  Sample a batch of data points $\mathcal{B} \sim \mathcal{D}$.

  Take an optimization step to update ϕ towards the reverse KL divergence defined in Eq. 10.

For $itr = 1$ *to #iterations* **do** ▷ Optional Step

  Sample a batch of data points $\mathcal{B} \sim \mathcal{D}$.

  Update both θ and ϕ towards β-VAE objective in Eq. 6.

return θ, ϕ