Learning to Adapt to Evolving Domains

Hong Liu, Mingsheng Long, Jianmin Wang, Yu Wang

School of Software, Tsinghua University
Department of Electronic Engineering, Tsinghua University

https://github.com/Liuhong99/EAML

Neural Information Processing Systems 2020

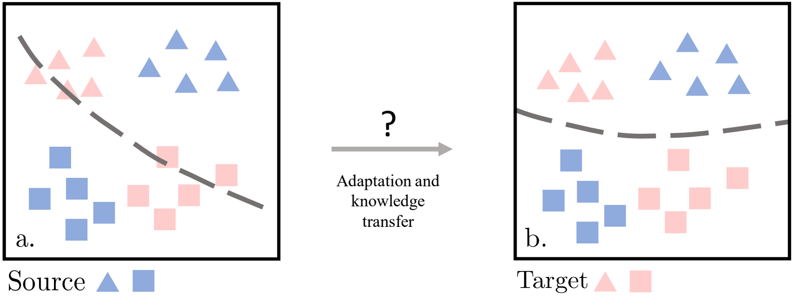
Outline

1. Domain Adaptation
   - Stationary Domain Adaptation
   - Evolving Domain Adaptation (EDA)
   - Analysis of EDA
2. Evolving Adaptive Meta Learning
   - Learn a Meta-Representation for Continually Evolving Target
   - Learning a Meta-Adapter to Overcome Forgetting
3. Experiments
   - Datasets
   - Results

Stationary Domain Adaptation

Transfer knowledge across different domains:

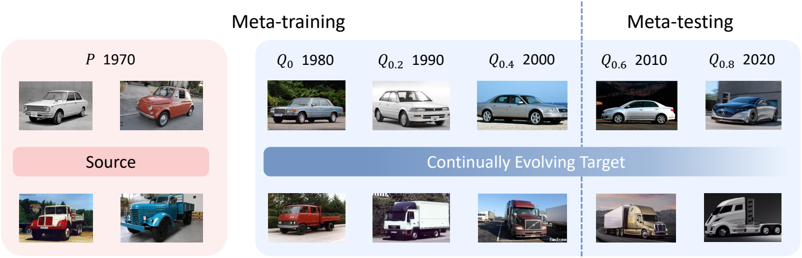
- The learner is provided with n_s i.i.d. observations $\{\mathbf{x}_s^{(i)}, \mathbf{y}_s^{(i)}\}_{i=1}^{n_s}$ from a source domain of distribution $P(\mathbf{x}_s, \mathbf{y}_s)$, and n_t i.i.d. observations $\{\mathbf{x}_t^{(i)}\}_{i=1}^{n_t}$ from a target domain of distribution $Q(\mathbf{x}_t, \mathbf{y}_t)$.
- Learn an accurate model for the target domain



Evolving Domain Adaptation (EDA)

A source distribution $P(x, y)$ and an evolving target distribution $Q_t(x, y)$, $t \in [0, 1]$. Typically $d(Q_{t_1}, Q_{t_2}) > 0$ for some distribution distance d. $\lim_{\Delta t \to 0} d(Q_t, Q_{t+\Delta t}) = 0$ as the continuity of the evolvement. Target unlabeled data points come in small batches sequentially, $T = \{\mathbf{X}_{t_1}, \mathbf{X}_{t_2} \cdots \mathbf{X}_{t_n}\}$, forming a trajectory of evolving target. Our goal is to minimize the expected risk on the evolving target.

$$\mathbb{E}_{t\sim U(0,1)}\mathbb{E}_{(x,y)\sim Q_t} L(f_\theta(x), y) = \int_0^1 \mathbb{E}_{(x,y)\sim Q_t} L(f_\theta(x), y)\mathrm{d}t. \tag{1}$$



Analysis of EDA

What factors are the performance of EDA related to?

- To solve the EDA problem with target trajectories, we need the target domain to evolve steadily to facilitate knowledge transfer.
- $d_{\mathcal{H}\Delta\mathcal{H}}(P, Q_t) = \sup_{\theta,\theta'} |\mathbb{E}_P L(f_\theta(x), f_{\theta'}(x)) - \mathbb{E}_{Q_t} L(f_\theta(x), f_{\theta'}(x))|$, which quantifies the discrepancy. The adaptability measuring the feature of cross-domain learning is $\lambda_t = \min_\theta [\mathbb{E}_P L(f_\theta(x), y) + \mathbb{E}_{Q_t} L(f_\theta(x), y)]$.

Theorem 1. Assume $d_{\mathcal{H}\Delta\mathcal{H}}(Q_{t_1}, Q_{t_2}) \leq \alpha|t_1 - t_2|$ holds with constant α for $t_1, t_2 \geq 0$. Then for any θ, with probability at least $1-\delta$ over the sampling of target trajectory $t_1, t_2 \cdots t_n$,

$$\mathbb{E}_t \mathbb{E}_{Q_t} L(f_\theta(x), y) \leq \mathbb{E}_P L(f_\theta(x), y) + \frac{1}{n}\sum_{i=1}^{n} [d_{\mathcal{H}\Delta\mathcal{H}}(P, Q_{t_i})] + \mathbb{E}_t \lambda_t + O\left(\frac{\alpha}{\delta n}\right).$$

Analysis of EDA

Learn representations to capture and harness the evolvement of target domain, i.e. make α sufficiently small.

- α indicates the rate of evolvement of the target domain Q_t.
- A reasonably small α means Q_t is evolving evenly, and neighboring target data share knowledge.
- n is the length of target trajectory T. With small α and large n, we can solve the EDA problem by adapting source and target trajectories.

Design architectures to mitigate catastrophic forgetting in EDA.

- Target data come online and cannot be stored in meta-testing.
- Adapting to current target inevitably results in forgetting knowledge on previous target.

Outline

1. Domain Adaptation
   - Stationary Domain Adaptation
   - Evolving Domain Adaptation (EDA)
   - Analysis of EDA
2. Evolving Adaptive Meta Learning
   - Learn a Meta-Representation for Continually Evolving Target
   - Learning a Meta-Adapter to Overcome Forgetting
3. Experiments
   - Datasets
   - Results

Learn a Meta-Representation for Continually Evolving Target

Recall that Model-Agnostic Meta-Learning (MAML) learns transferable features by training on the support set in the inner loop and minimizing error on the query set in the outer loop. We can learn a representation for adapting to evolving target similarly.

- h_θ: the representation function parametrized by θ
- g_ϕ the adapter parametrized by ϕ
- c_W: the task classifier with parameters W
- $T_{\text{spt}} = \{\mathbf{X}_{t_1}, \mathbf{X}_{t_2} \cdots \mathbf{X}_{t_n}\}$: target support set
- $T_{\text{qry}} = \{\mathbf{X}'_{t_1}, \mathbf{X}'_{t_2} \cdots \mathbf{X}'_{t_n}\}$: target query set

In the inner loop, we train g_ϕ and c_W to adapt to evolving target sequentially on representation f_θ. Thus, f_θ is fixed in the inner loop.

$$(\phi, W)_{i+1} \leftarrow (\phi, W)_i - \eta_{\text{in}} \nabla_{(\phi,W)} \left[L(f_{\theta,\phi_i,W_i}(\mathbf{X}_s), \mathbf{Y}_s) + d(f_{\theta,\phi_i,W_i}(\mathbf{X}_s), f_{\theta,\phi_i,W_i}(\mathbf{X}_{t_i})) \right], \quad (2)$$

Learn a Meta-Representation for Continually Evolving Target

Train the meta-representation in the outer loop.

- We require the representation to make the adaptation in the inner loop more effective.
- Update the representation f_θ to minimize the EDA loss on the query set following Equation (1)? – no access to target labels.
- Replace the EDA loss with its upper bound, and update θ to control the upper bound.
- Take α into consideration. We use $\max_i d(f_{\theta,\phi_n,W_n}(\mathbf{X}'_{t_{i-1}}), f_{\theta,\phi_n,W_n}(\mathbf{X}'_{t_i}))$ as an approximation of α

$$\begin{aligned}\theta \leftarrow \theta - \eta_{\text{out}} \nabla_\theta L[(f_{\theta,\phi_n,W_n}(\mathbf{X}_s), \mathbf{Y}_s) + \frac{1}{n}\sum_{i=1}^{n} d(f_{\theta,\phi_n,W_n}(\mathbf{X}_s), f_{\theta,\phi_n,W_n}(\mathbf{X}'_{t_i})) \\ + \max_i d(f_{\theta,\phi_n,W_n}(\mathbf{X}'_{t_{i-1}}), f_{\theta,\phi_n,W_n}(\mathbf{X}'_{t_i}))],\end{aligned} \tag{3}$$

where η_{out} denotes the learning rate of the outer loop.

Learning a Meta-Adapter to Overcome Forgetting

Overcome catastrophic forgetting

- Design the adapter g_ϕ to avoid overwriting knowledge
- Introducing a meta-adapter $g'_{\phi'}$ to the original adapter g_ϕ

Mimicking intermediate features with meta-adapter in the inner loop

- A well-trained adapter contains useful knowledge for that target data
- Matching the intermediate features when adapting to new target helps overcome forgetting of the old ones

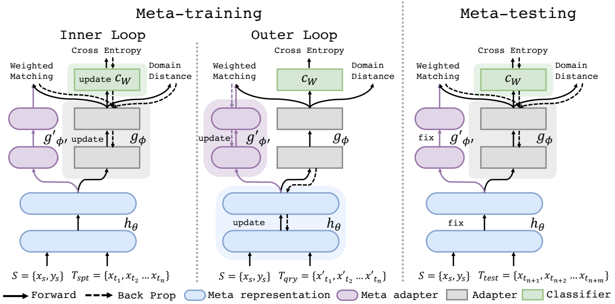
Then the weighted feature matching loss is $\sum_{j=1}^{n} w_j^\top \|g_{l,\phi_l}(x_j) - g_{l,\phi_l^{\text{prev}}}(x_j)\|$, where $w_j \in \mathcal{R}^m$ is the weight of sample x_j. To facilitate knowledge retaining, we introduce the meta-adapter g'_{l,ϕ'_l}, $w_j = g'_{l,\phi'_l}(x_j)$. Thus, the total loss is the sum of the weighted feature matching loss in each layer,

$$L_m(\phi', \phi, \phi^{\text{prev}}) = \sum_l \sum_j g'_{l,\phi'_l}(x_j)^\top \|g_{l,\phi_l}(x_j) - g_{l,\phi_l^{\text{prev}}}(x_j)\|.$$

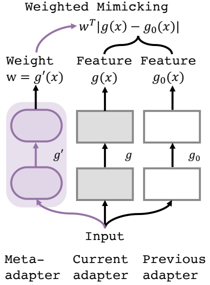
Learning a Meta-Adapter to Overcome Forgetting

In the outer loop, update the meta-adapter to make the weighted feature mimicking preserve more knowledge on previous target data. We use the same loss function as Equation (3), but we calculate the gradient w.r.t. both f_θ and $g'_{\phi'}$.

Summary of the proposed method:



(a) The training procedure of EAML



(b) The meta-adapter

Outline

1. Domain Adaptation
   - Stationary Domain Adaptation
   - Evolving Domain Adaptation (EDA)
   - Analysis of EDA
2. Evolving Adaptive Meta Learning
   - Learn a Meta-Representation for Continually Evolving Target
   - Learning a Meta-Adapter to Overcome Forgetting
3. Experiments
   - Datasets
   - Results

Datasets

Rotated MNIST, Evolving Vehicles, and Caltran

(c) Evolving Vehicles

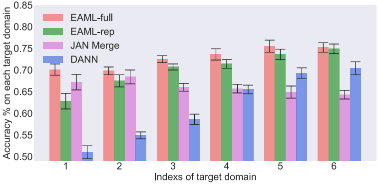
(d) Caltran

Results

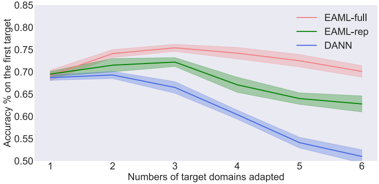
Table: Classification Accuracy (%) on rotated MNIST dataset.

Method	120°	126°	132°	138°	144°	150°	156°	162°	168°	174°	Avg.
Source Only	17.60	19.29	22.50	24.14	26.49	29.48	31.06	32.26	33.73	33.25	26.98
DANN	18.92	21.65	24.32	27.63	29.76	32.01	33.92	36.23	36.68	36.93	29.81
JAN Merge	20.20	21.71	25.75	29.16	33.27	37.19	40.04	40.39	39.67	38.71	32.60
MAML	22.75	25.11	28.90	30.40	32.62	34.56	35.14	36.55	37.31	38.30	32.16
CMA	21.82	23.65	26.48	29.48	32.05	34.99	35.08	36.34	38.33	39.25	31.75
DANN+EWC	20.39	24.19	28.50	30.10	32.48	35.75	36.23	38.47	38.63	38.05	32.27
EAML	**24.69**	**27.48**	**30.16**	**32.79**	**34.88**	**37.35**	**39.25**	**40.96**	**42.45**	**42.27**	**35.23**
Replay Oracle	24.35	26.21	30.33	31.89	33.02	35.87	37.98	39.66	41.40	42.21	34.28

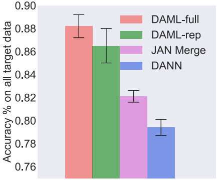
Results



(e) Each target on Evolving Vehicles



(f) First target on Evolving Vehicles



(g) Targets on Caltran

(h) Source Only

(i) JAN Merge

(j) EAML

Summary

- A novel and practical domain adaptation setting: Evolving domain adaptation
- Analyze the factor of EDA performance: features tailored to EDA and forgetting
- A meta-learning method to solve EDA efficiently
- Outperform well-established baselines

Thanks!

Contact: h-l17@mails.tsinghua.edu.cn
mingsheng@tsinghua.edu.cn

Code: `github.com/Liuhong99/EAML`