NEURAL INFORMATION PROCESSING SYSTEMS

Environment Inference for Learning Generalizable Dynamical System

Shixuan Liu[1], Yue He[2*], Haotian Wang[1], Wenjing Yang[1], Yunfei Wang[3]

Peng Cui[4*], Zhong Liu[5]

[1]College of Computer Science and Technology, NUDT

[2]School of Information, Renmin University of China

[3]College of Systems Engineering, NUDT

[4]Department of Computer Science and Technology, Tsinghua University

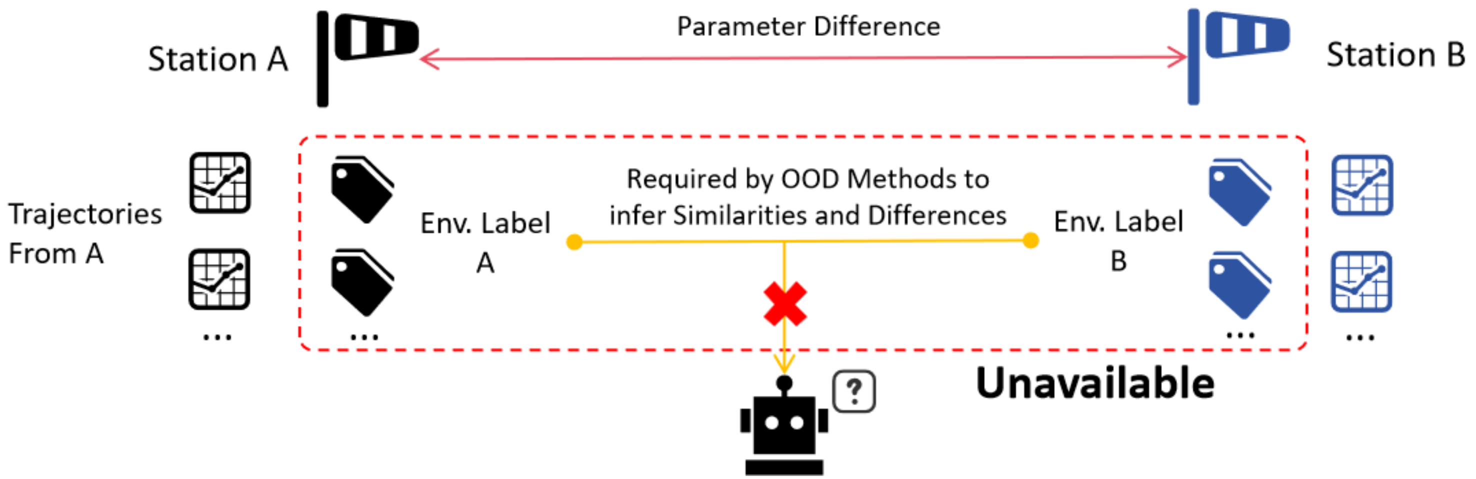
[5]Laboratory for Big Data and Decision, NUDT

Preliminary

- A dynamical system's state evolves over time according to a fixed rule, typically expressed as a set of ODE or PDE equations.
- Data-driven approaches provide a valuable alternative and complement to physics-based methods for modeling complex system dynamics.
- Most dynamical systems (DS) learning methods presuppose an idealized, static environment under the i.i.d. hypothesis.
- This idealized assumption drives the need for generalization techniques capable of handling environmental distribution shifts.

Problem: Unlabeled Environments

- Current DS generalization methods share a critical limitation: a strict dependency on environment labels during training. These labels, which identify which environment a trajectory sample belongs to, are essential for these methods to identify inter-domain similarities and differences.
- However, in practice, these labels are often unavailable. This can be due to privacy concerns, data aggregation from multiple sources, or simply a failure to record the environmental context.
- This reality presents a fundamental challenge: **How can we learn generalizable DS models when environment labels are unavailable?**



Generalization for DS

- We review existing DS generalization methods and summarize their objectives for the multi-environment learning problem with labeled environments $e = \{e_1, e_2, \ldots, e_N\}$, as follows:

$$R_e(\theta, \phi) = \sum_{i=1}^{N} \int_{t \in I} \left\| \frac{dx_t^i}{dt} - h(x_t^i; \theta, \phi_{e_i}) \right\|_2^2 dt + \lambda \sum_{e=1}^{M} \Omega(\phi_e)$$

- The dynamics is decomposed into two components: a global component shared across all environments, parameterized by θ, and an environment-specific component, parameterized by ϕ_{e_i} for each environment e_i.
- This parametrization entails a decomposition that can be implemented either in a functional form ($h\left(x_t^i; \theta, \phi_{e_i}\right) = f_\theta\left(x_t^i\right) + g_{\phi_e}\left(x_t^i\right)$), or in a parametric form ($h\left(x_t^i; \theta, \phi_{e_i}\right) = f_{\theta+\phi_e}\left(x_t^i\right)$).
- The key ingredient for multi-environment learning is that θ should encapsulate the maximal shared dynamics, whereas ϕ_e should exclusively reflect the unique characteristics of each environment e not described by θ. To do so, a regularization term $\Omega(\phi_e)$ is introduced to effectively penalize ϕ_e, thereby facilitating learning in the global component

Problem Formulation

- We aim to infer an environment assignment for each sample that maximizes the model's generalization ability across different environments.
- To achieve this goal, we reformulate the learning objective into an optimization problem contingent on a specific environment assignment e.
- Specifically, our aim is to learn the environment assignment $\hat{e} = \{\widehat{e^1}, \widehat{e^2}, \dots, \widehat{e^n}\} \in [M]^N$ for each trajectory, to optimize generalization loss effectively:

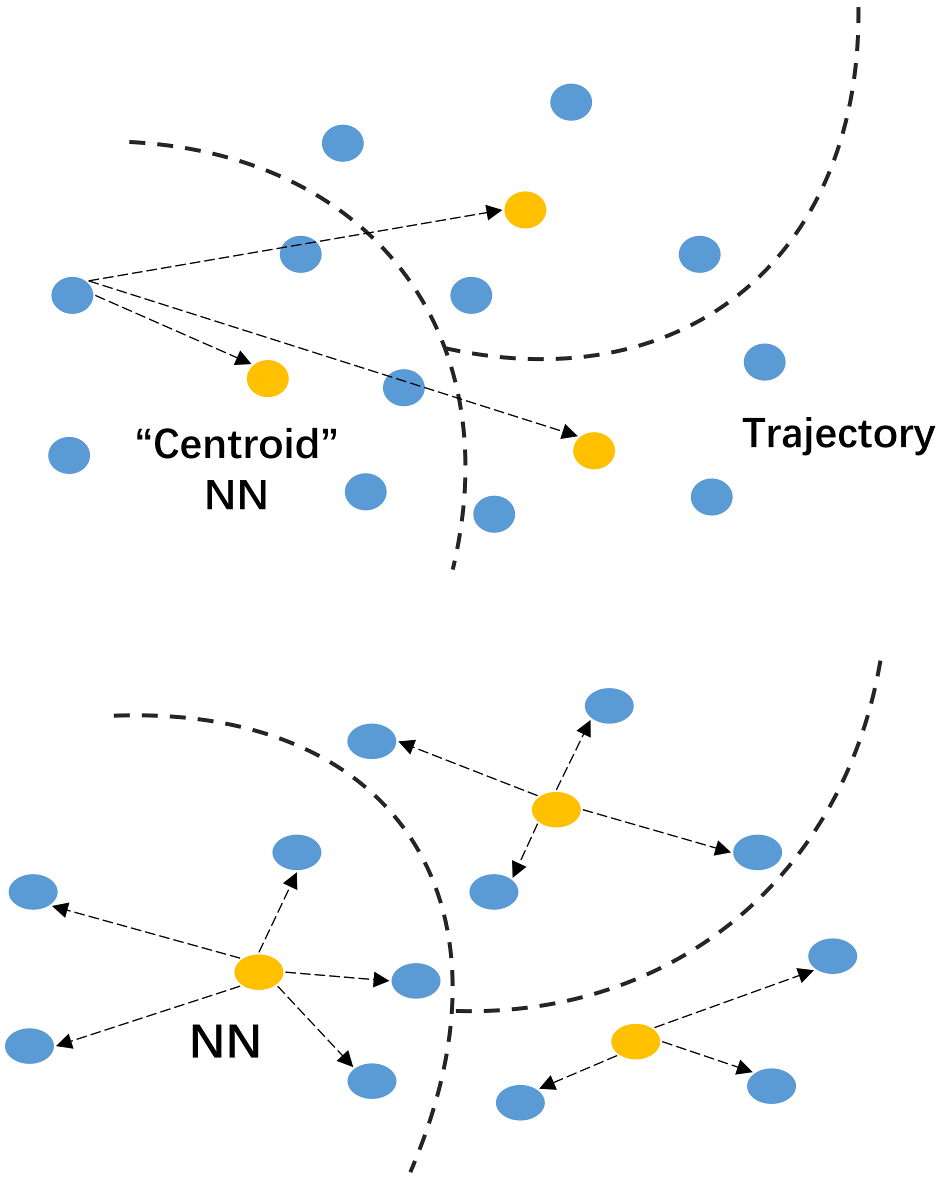
$$R_e(\theta, \phi) = \sum_{i=1}^{N} \int_{t \in I} \left\| \frac{dx_t^i}{dt} - h(x_t^i; \theta, \phi_{e_i}) \right\|_2^2 dt + \lambda \sum_{e=1}^{M} \Omega(\phi_e)$$

- The overall objective is defined as follows:

$$\hat{e}^*, \theta^*, \phi^* = \arg\min_{\hat{e}, \theta, \phi} R_{\hat{e}}(\theta, \phi)$$

Key Insights

- Trajectories from the same environment have similar prediction errors under the same model.
- Ultimately, each trajectory has a uniquely best-fit NN. This NN generalize to a cluster of trajectories from the same underlying environment.
- Solution: Use prediction loss to infer environment labels. The NN that achieves the minimum loss for a given trajectory is designated as its environment.
- Alternate between:
  - Inferring environments from current NNs
  - Updating NNs with inferred environments



DynaInfer

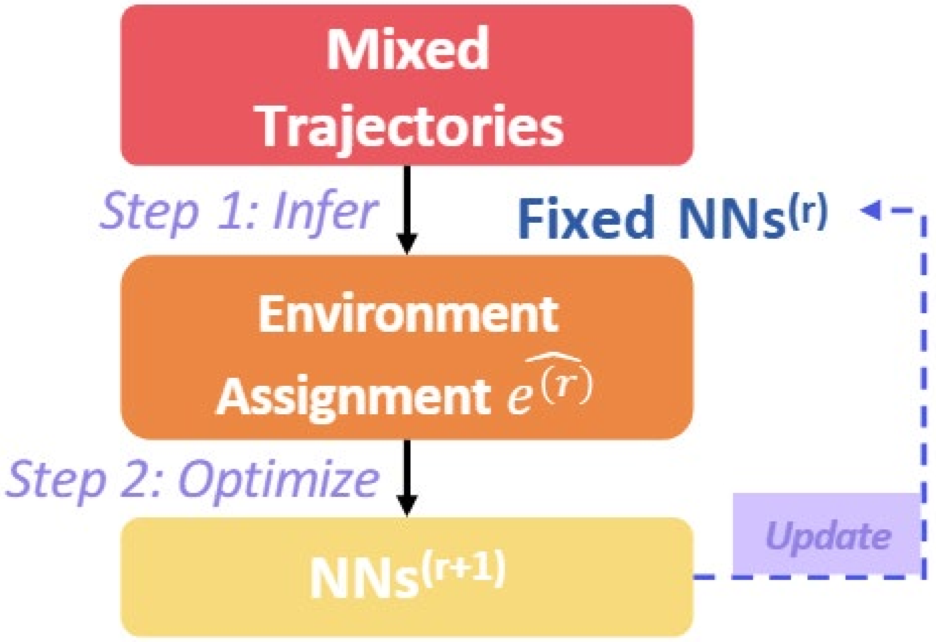
- Bi-level Optimization with a min-min scheme
- ① Inference Stage:
  - Each trajectory is assigned to the NN with the minimal prediction loss:

$$\hat{e}_i^{(r)} = \arg\min_{e\in[M]} \int_{t\in I} \left\| \frac{dx_t^i}{dt} - h(x_t^i; \theta^{(r-1)}, \phi_{e_i}^{(r-1)}) \right\|_2^2 dt$$

- ② Optimization Stage:
  - Each NN is then optimized using its assigned trajectories:

$$\theta^{(r)}, \phi^{(r)} = \arg\min_{\theta,\phi} R_{\hat{e}^{(r)}}(\theta, \phi)$$

$$R_e(\theta,\phi) = \sum_{i=1}^{N} \int_{t\in I} \left\| \frac{dx_t^i}{dt} - h(x_t^i; \theta, \phi_{e_i}) \right\|_2^2 dt + \lambda \sum_{e=1}^{M} \Omega(\phi_e)$$



Theoretical Property

Proposition 3.1. *For all rounds* $1 \leq r < T_r$*, we must have*

$$R_{\hat{\mathbf{e}}^{(r+1)}}\left(\theta^{(r+1)}, \boldsymbol{\phi}^{(r+1)}\right) \leq R_{\hat{\mathbf{e}}^{(r)}}\left(\theta^{(r)}, \boldsymbol{\phi}^{(r)}\right).$$

Furthermore, suppose the space of $\arg\min_{\theta,\boldsymbol{\phi}} R_{\hat{\mathbf{e}}}(\theta, \boldsymbol{\phi})$ *is finite for all* $\hat{\mathbf{e}} \in [M]^N$*. Then there exists a constant* $C > 0$ *such that if* $r > 1$ *and* $R_{\hat{\mathbf{e}}^{(r+1)}}(\theta^{(r+1)}, \boldsymbol{\phi}^{(r+1)}) < R_{\hat{\mathbf{e}}^{(r)}}(\theta^{(r)}, \boldsymbol{\phi}^{(r)})$*, we must have*

$$R_{\hat{\mathbf{e}}^{(r+1)}}\left(\theta^{(r+1)}, \boldsymbol{\phi}^{(r+1)}\right) \leq R_{\hat{\mathbf{e}}^{(r)}}\left(\theta^{(r)}, \boldsymbol{\phi}^{(r)}\right) - C.$$

Remark 3.1. Given the assumptions made in prior works [43, 17] that $h(\cdot; \theta, \phi_e)$ is linear with respect to θ and ϕ_e, and that $\Omega(\phi_e)$ is strictly convex with respect to ϕ_e, it follows logically that the space of $\arg\min_{\theta,\boldsymbol{\phi}} R_{\hat{\mathbf{e}}}(\theta, \boldsymbol{\phi})$ is finite for all $\hat{e} \in [M]^N$, as is evident from Equation (2). The proof is provided in Appendix A.

Experiments

Domain Generalization Experiment

Data	Assignment	LEADS			CoDA-l_1			CoDA-l_2		
		Train	Test		Train	Test		Train	Test	
		MSE	MSE	MAPE	MSE	MSE	MAPE	MSE	MSE	MAPE
LV	All in One	7.17 E-2	7.41±0.02 E-2	49.22±1.84	7.14 E-2	7.40±0.01 E-2	49.44±3.15	7.17 E-2	7.41±0.00 E-2	39.26±22.13
	One per Env	4.15 E-4	4.91±3.50 E-4	6.68±2.44	8.68 E-4	9.14±0.41 E-4	5.67±1.01	8.18 E-4	8.43±0.39 E-4	5.73±1.19
	Random	7.20 E-2	7.38±0.02 E-2	50.01±1.05	7.12 E-2	7.39±0.01 E-2	48.87±1.81	7.09 E-2	7.39±0.00 E-2	48.86±2.54
	DynaInfer	**4.74 E-5**	**7.93±2.49 E-5**	**2.83±1.62**	**9.57 E-5**	**1.83±3.40 E-4**	**3.27±2.36**	**1.71 E-4**	**1.82±3.07 E-4**	**2.02±1.66**
	Oracle	4.55 E-5	7.02±0.76 E-5	1.78±0.10	1.78 E-5	3.19±0.24 E-5	1.26±0.06	1.99 E-5	2.72±0.18 E-5	1.21±0.08
GS	All in One	8.73 E-3	9.60±0.02 E-3	3008.80±892.20	9.24 E-3	9.61±0.03 E-3	4115.80±223.54	9.25 E-3	9.60±0.00 E-3	3723.00±713.85
	One per Env	1.38 E-3	1.65±0.54 E-3	173.44±59.16	1.56 E-3	1.91±0.06 E-3	185.23±61.43	1.52 E-3	1.87±0.02 E-3	174.08±57.82
	Random	8.78 E-3	9.36±0.20 E-3	1403.50±119.50	9.25 E-3	9.59±0.03 E-3	3958.25±682.38	8.77 E-3	9.35±0.02 E-3	3919.88±157.54
	DynaInfer	**3.60 E-5**	**4.14±0.21 E-5**	**117.57±33.90**	**9.22 E-5**	**1.23±0.41 E-4**	**122.93±22.05**	**6.69 E-5**	**7.25±2.11 E-5**	**112.52±14.15**
	Oracle	7.73 E-5	1.34±0.76 E-4	97.77±12.09	6.04 E-5	9.60±3.91 E-5	163.38±47.89	4.69 E-5	7.04±1.84 E-5	138.86±16.55
NS	All in One	5.34E-02	6.71±0.11 E-2	239.70±14.78	5.79E-02	6.64±0.11 E-2	251.38±8.54	6.17E-02	6.64±0.03 E-2	255.26±9.11
	One per Env	2.24E-02	4.11±0.14 E-2	169.48±9.68	3.45E-02	3.88±0.22 E-2	161.04±10.73	2.31E-02	4.04±0.22 E-2	158.26±9.35
	Random	3.06E-02	6.58±0.05 E-2	233.80 ± 8.44	5.04E-02	6.58±0.05 E-2	247.95±4.59	5.78E-02	6.66±0.04 E-2	254.47±6.40
	DynaInfer	**6.10E-04**	**7.05±0.34 E-3**	**77.29±10.18**	**1.23E-02**	**1.62±0.18 E-2**	**108.17±10.30**	**8.92E-04**	**1.19±0.12 E-2**	**96.57±12.75**
	Oracle	2.59E-04	6.55±1.34 E-3	67.58±9.37	1.36E-02	1.73±0.29 E-2	124.22±12.35	7.11E-04	9.46±0.51 E-3	91.06±5.85

Domain Adaption Experiment

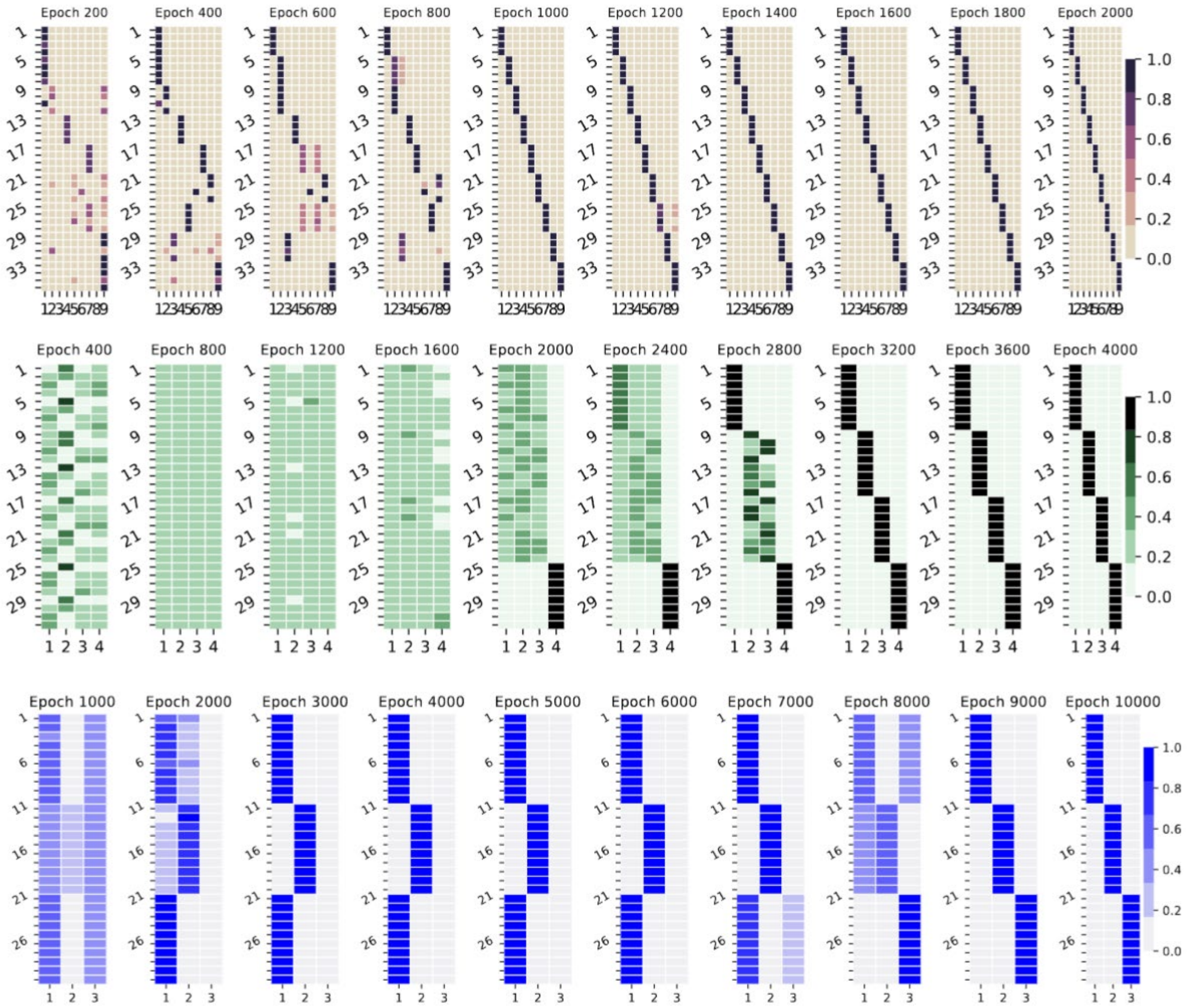
Data	Assignment	LEADS		CoDA-l_1		CoDA-l_2	
		MSE	MAPE	MSE	MAPE	MSE	MAPE
LV	All in One	4.16±8.61 E-2	9.92±3.55	4.01±6.43 E-2	26.90±10.65	4.10±7.61 E-2	27.80±8.81
	One per Env	2.28±1.81 E-3	5.41±0.50	1.72±0.53 E-3	27.63±7.81	1.66±0.82 E-3	25.51±7.72
	Random	1.72±0.53 E-3	6.87±0.13	1.14±0.61 E-3	27.56±6.36	1.05±0.54 E-3	29.65±6.31
	DynaInfer	**5.77±1.46 E-4**	**2.84±0.13**	**8.37±0.94 E-5**	**10.16±0.04**	**8.49±2.07 E-5**	**10.30±0.08**
	Oracle	1.67±2.26 E-3	3.16±0.37	5.85±1.24 E-5	10.24±0.06	5.12±3.17 E-5	10.24±0.04
GS	All in One	4.59±1.18 E-4	721.25±197.54	2.85±0.55 E-3	6658.20±1651.77	2.76±0.72 E-3	7355.20±335.45
	One per Env	3.59±2.71 E-4	450.20±368.36	1.08±0.82 E-3	6247.34±817.74	1.19±0.97 E-3	5948.57±935.71
	Random	5.73±0.86 E-4	1261.00±979.30	2.92±0.80 E-3	7292.88±312.54	2.84±0.89 E-3	4499.25±844.62
	DynaInfer	**1.00±0.32 E-4**	**378.73±182.12**	**2.41±0.91 E-4**	**220.54±65.60**	**2.13±0.41 E-4**	**207.96±46.49**
	Oracle	2.21±0.93 E-4	434.73±432.38	2.66±0.79 E-4	302.68±188.25	2.10±0.89 E-4	230.62±118.29
NS	All in One	1.25±2.04 E-2	67.17±3.33	1.25±0.20 E-2	218.66±27.26	1.29±0.29 E-2	214.44±17.08
	One per Env	2.78±2.08 E-2	96.23±4.54	2.04±0.68 E-2	214.38±15.19	2.43±0.48 E-2	209.68±18.98
	Random	1.32±0.53 E-2	81.18±7.43	4.66±1.04 E-2	215.21±19.39	4.37±0.99 E-2	191.03±16.38
	DynaInfer	**7.52±0.76 E-3**	**50.93±8.83**	**9.27±1.81 E-3**	**101.38±15.77**	**9.71±2.10 E-3**	**101.04±16.27**
	Oracle	1.16±0.68 E-2	57.35±17.90	7.46±0.72 E-3	154.86±41.00	7.32±0.81 E-3	100.04±24.93

- We evaluate three strategies for assigning environment labels without environmental data: grouping all samples into one environment (**All in One**), giving each sample a unique label (**One per Env**), and random assignment (**Random**). We also include an **Oracle** baseline with known labels.
- We consider three base models for dynamic system generalization: LEADS, CoDA-l1, and CoDA-l2. Three dynamic systems are tested under domain generalization and domain adaption settings.

- Across all datasets, DynaInfer significantly outperforms other assignment strategies.
- Notably, DynaInfer either matches or exceeds Oracle performance

Experiments

- We illustrate the probability of environment assignments with DynaInfer over training time.
- Initially, our model may default to random assignments due to unoptimized neural networks. However, the assignments quickly converge to the true labels.
- Notably, systems with simpler dynamics, like LV compared to NS, enable faster convergence of environment assignments



Contact

- Bibtex:

```
@inproceedings{liuenvironment,
  title={Environment Inference for Learning Generalizable Dynamical System},
  author={Liu, Shixuan and He, Yue and Wang, Haotian and Yang, Wenjing and Wang, Yunfei and Cui, Peng and Liu, Zhong},
  booktitle={The Thirty-ninth Annual Conference on Neural Information Processing Systems}
}
```

- Code: https://github.com/shixuanliu-andy/DynaInfer
- Mail: szftandy@hotmail.com
- Wechat: 13924663583