Machine Learning under Distribution Shifts

Masashi Sugiyama

RIKEN Center for Advanced Intelligence Project/
The University of Tokyo, Japan

http://www.ms.k.u-tokyo.ac.jp/sugi/

Learning under Distribution Shifts

- **Given:**
  - Training data $\{(\boldsymbol{x}_i^{\mathrm{tr}}, y_i^{\mathrm{tr}})\}_{i=1}^{n_{\mathrm{tr}}} \overset{\text{i.i.d.}}{\sim} p_{\mathrm{tr}}(\boldsymbol{x}, y)$

$\boldsymbol{x}$: Input

y : Output

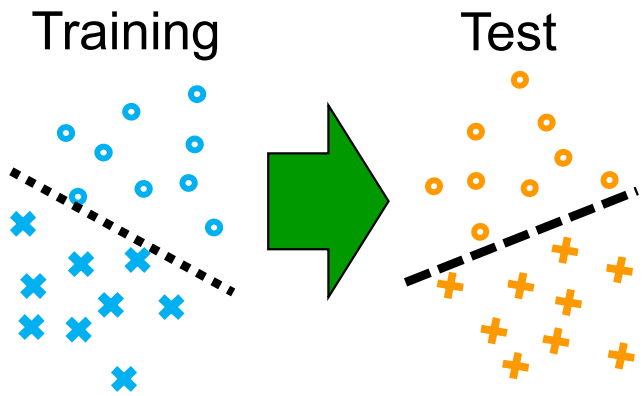
- **Goal:**
  - Learn predictor $y = f(\boldsymbol{x})$ minimizing the test risk (with some additional data from the test domain).

$$\min_f R(f) \qquad R(f) = \mathbb{E}_{p_{\mathrm{te}}(\boldsymbol{x},y)}[\ell(f(\boldsymbol{x}), y)]$$

ℓ : loss

- **Challenge:**
  - Overcome changing distributions!

$$p_{\mathrm{tr}}(\boldsymbol{x}, y) \neq p_{\mathrm{te}}(\boldsymbol{x}, y)$$



- Non-stationary of the environments.
- Sample selection bias due to privacy concerns.



NIPS Workshop on Learning when Test and Training Inputs Have Different Distributions, Whistler 2006

Learning when test and training inputs have different distributions

Workshop

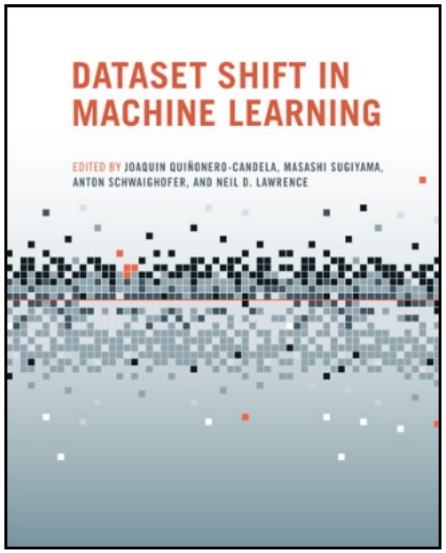
Joaquin Quiñonero Candela · Masashi Sugiyama · Anton Schwaighofer · Neil D Lawrence

Sat Dec 09 05:00 PM -- 05:00 PM (JST) @ Nordic

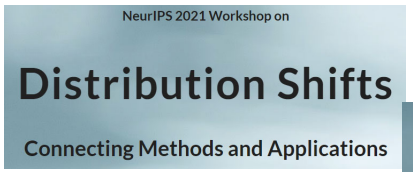
Event URL: http://ida.first.fraunhofer.de/projects/different06/ »

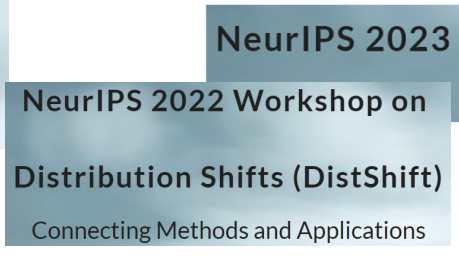
Many machine learning algorithms assume that the training and the test data are drawn from the same distribution. Indeed many of the proofs of statistical consistency, etc., rely on this assumption. However, in practice we are very often faced with the situation where the training and the test data both follow the same conditional distribution, p(y|x), but the input distributions, p(x), differ. For example, principles of experimental design dictate that training data is acquired in a specific manner that bears little resemblance to the way the test inputs may later be generated. The aim of this workshop will be to try and shed light on the kind of situations where explicitly addressing the difference in the input distributions is beneficial, and on what the most sensible ways of doing this are.

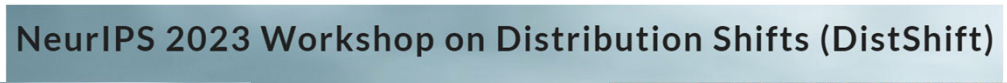
Quiñonero-Candela, Sugiyama, Schwaighofer & Lawrence (Eds.), Dataset Shift in Machine Learning, MIT Press, 2009.

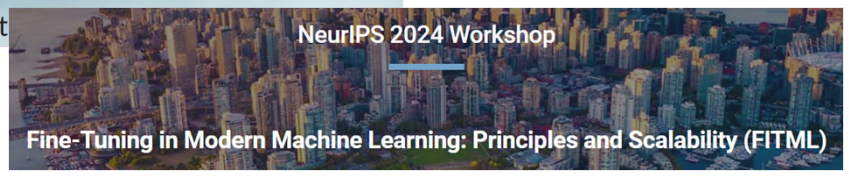


- "Distribution Shift" / "Finetuning" workshops at NeurIPS2021-2024:











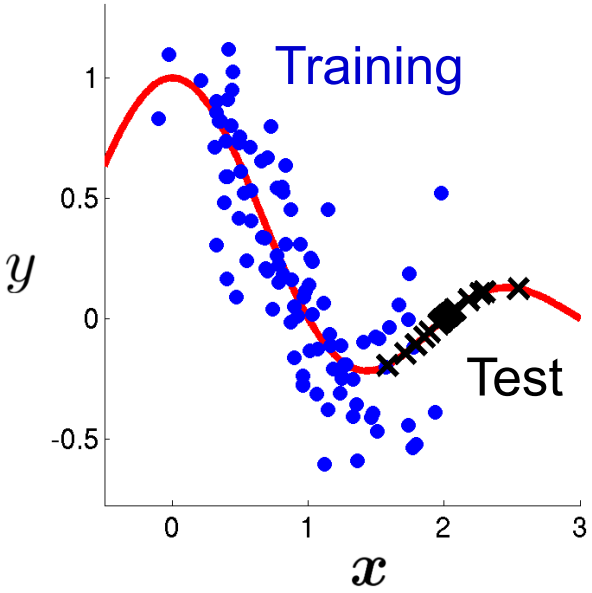
Contents

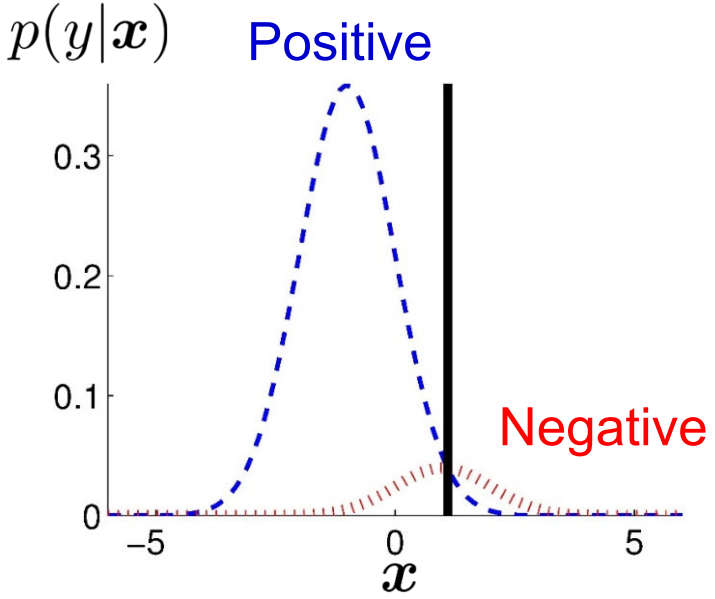
1. Importance Weighting
2. Sequential Shifts
3. Ongoing Challenges

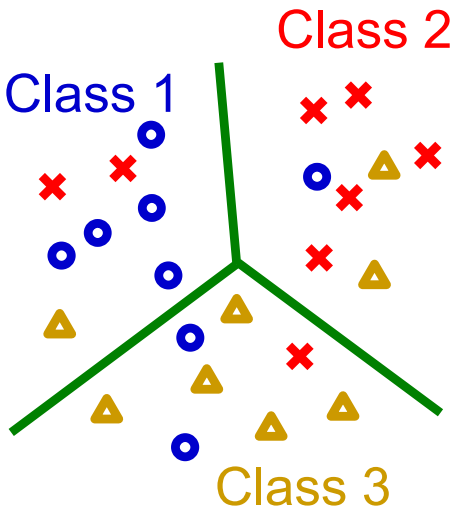
Types of Distribution Shifts

$\boldsymbol{x}$: Input y : Output

- Joint shift: $p_{\mathrm{tr}}(\boldsymbol{x}, y) \neq p_{\mathrm{te}}(\boldsymbol{x}, y)$
- Covariate shift: $p_{\mathrm{tr}}(\boldsymbol{x}) \neq p_{\mathrm{te}}(\boldsymbol{x})$
- Label shift: $p_{\mathrm{tr}}(y) \neq p_{\mathrm{te}}(y)$
- Output noise: $p_{\mathrm{tr}}(y|\boldsymbol{x}) \neq p_{\mathrm{te}}(y|\boldsymbol{x})$
- Class-conditional shift: $p_{\mathrm{tr}}(\boldsymbol{x}|y) \neq p_{\mathrm{te}}(\boldsymbol{x}|y)$







Covariate Shift Adaptation

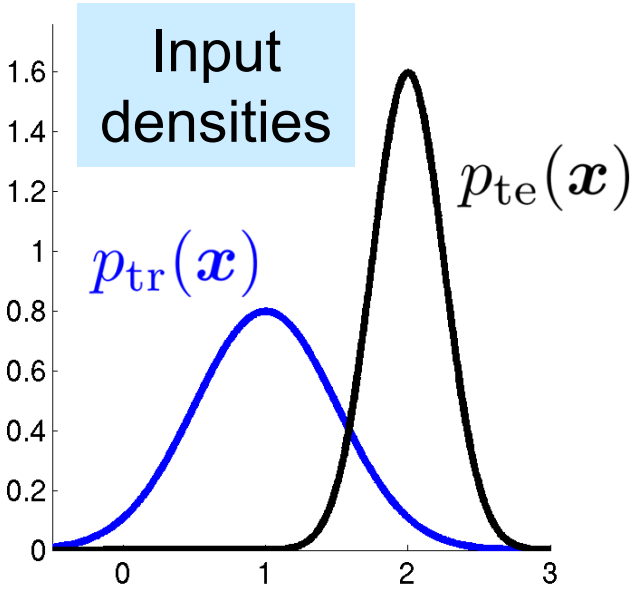
Shimodaira (JSPI2000)

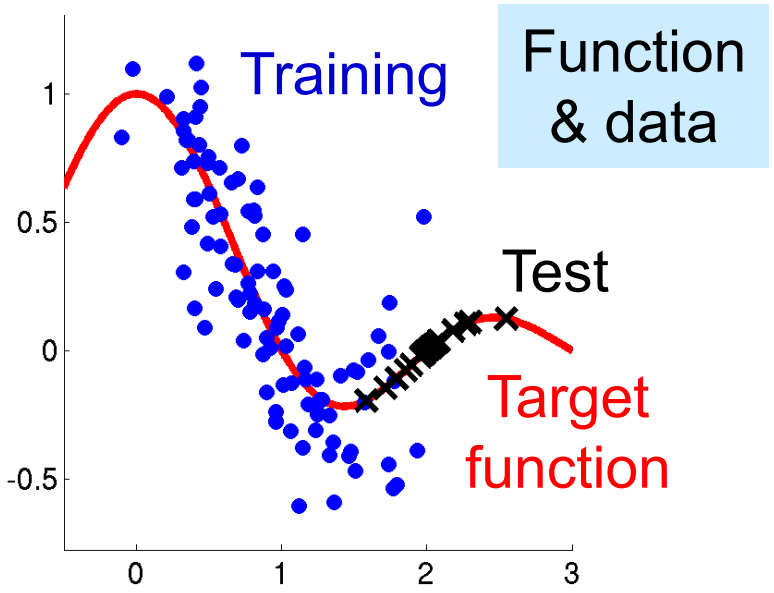
- Training and test input distributions are different,

$$p_{\mathrm{tr}}(\boldsymbol{x}) \neq p_{\mathrm{te}}(\boldsymbol{x})$$

but the output-given-input distribution is unchanged:

$$p_{\mathrm{tr}}(y|\boldsymbol{x}) = p_{\mathrm{te}}(y|\boldsymbol{x}) = p(y|\boldsymbol{x})$$





- Given:
  - Labeled training data: $\{(\boldsymbol{x}_i^{\mathrm{tr}}, y_i^{\mathrm{tr}})\}_{i=1}^{n_{\mathrm{tr}}} \overset{\text{i.i.d.}}{\sim} p_{\mathrm{tr}}(\boldsymbol{x}, y)$
  - Unlabeled test data: $\{\boldsymbol{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}} \overset{\text{i.i.d.}}{\sim} p_{\mathrm{te}}(\boldsymbol{x})$

Importance-Weighted Training

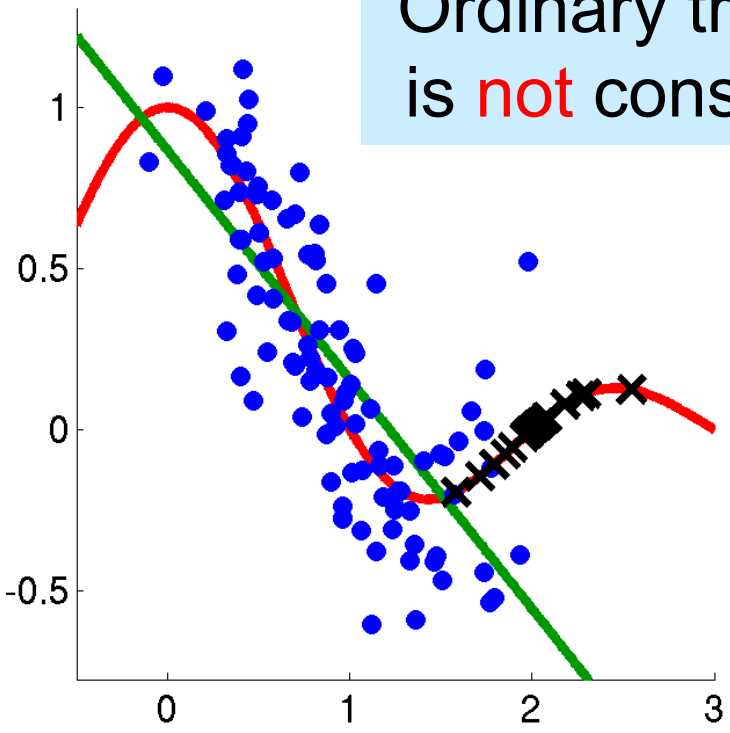
$$\underset{f}{\mathrm{argmin}} \left[\sum_{i=1}^{n_{\mathrm{tr}}} \ell(f(\boldsymbol{x}_i^{\mathrm{tr}}), y_i^{\mathrm{tr}}) \right]$$

$$\underset{f}{\mathrm{argmin}} \left[\sum_{i=1}^{n_{\mathrm{tr}}} \frac{p_{\mathrm{te}}(\boldsymbol{x}_i^{\mathrm{tr}})}{p_{\mathrm{tr}}(\boldsymbol{x}_i^{\mathrm{tr}})} \ell(f(\boldsymbol{x}_i^{\mathrm{tr}}), y_i^{\mathrm{tr}}) \right]$$

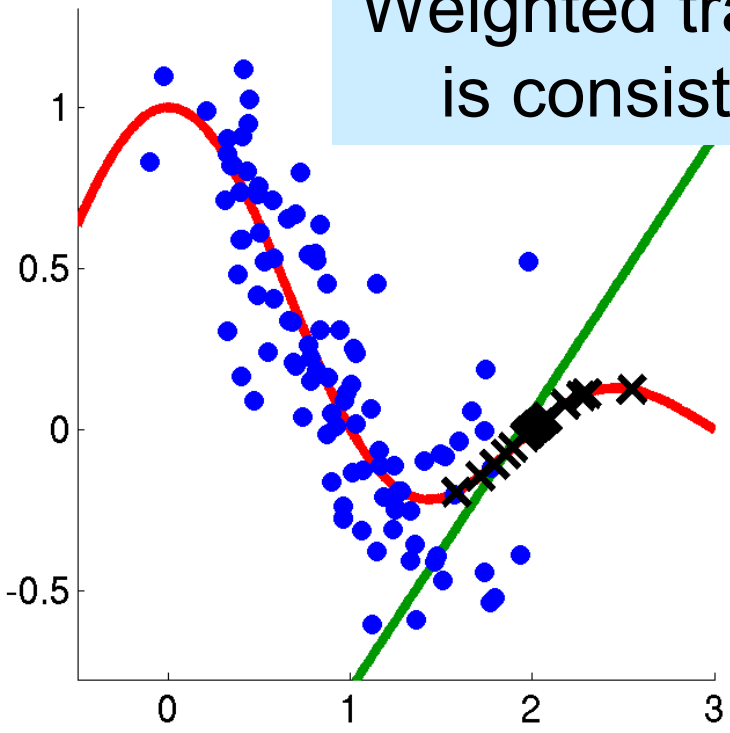
Importance

$$\{(\boldsymbol{x}_i^{\mathrm{tr}}, y_i^{\mathrm{tr}})\}_{i=1}^{n_{\mathrm{tr}}} \overset{\text{i.i.d.}}{\sim} p_{\mathrm{tr}}(\boldsymbol{x}, y)$$

Ordinary training is not consistent



Weighted training is consistent



- How do we estimate the importance?

Direct Importance Estimation

- Given: training and test input data

$$\{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}} \overset{\text{i.i.d.}}{\sim} p_{\mathrm{tr}}(\boldsymbol{x}) \qquad \{\boldsymbol{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}} \overset{\text{i.i.d.}}{\sim} p_{\mathrm{te}}(\boldsymbol{x})$$

- Goal: estimate the density ratio $\hat{r}(\boldsymbol{x}) \approx \frac{p_{\mathrm{te}}(\boldsymbol{x})}{p_{\mathrm{tr}}(\boldsymbol{x})}$

- Kernel mean matching: Huang+ (NeurIPS2006)
  - Match the means of $p_{\mathrm{te}}(\boldsymbol{x})$ and $r(\boldsymbol{x})p_{\mathrm{tr}}(\boldsymbol{x})$ in RKHS $\mathcal{H}$.

$$\min_{r \in \mathcal{H}} \left\| \mathbb{E}_{p_{\mathrm{te}}(\boldsymbol{x})}[K(\boldsymbol{x}, \cdot)] - \mathbb{E}_{p_{\mathrm{tr}}(\boldsymbol{x})}[K(\boldsymbol{x}, \cdot) r(\boldsymbol{x})] \right\|_{\mathcal{H}}^2$$

$K(\cdot, \cdot)$: Characteristic kernel

- Least-squares importance fitting (LSIF):
  - Fit a model $r(\boldsymbol{x})$ to $\frac{p_{\mathrm{te}}(\boldsymbol{x})}{p_{\mathrm{tr}}(\boldsymbol{x})}$ by least squares: Kanamori, Hido & Sugiyama (NeurIPS2008)

$$\operatorname*{argmin}_r \mathbb{E}_{p_{\mathrm{tr}}(\boldsymbol{x})} \left[\left(r(\boldsymbol{x}) - \frac{p_{\mathrm{te}}(\boldsymbol{x})}{p_{\mathrm{tr}}(\boldsymbol{x})} \right)^2 \right] = \operatorname*{argmin}_r \mathbb{E}_{p_{\mathrm{tr}}(\boldsymbol{x})}[r(\boldsymbol{x})^2] - 2\mathbb{E}_{p_{\mathrm{te}}(\boldsymbol{x})}[r(\boldsymbol{x})]$$

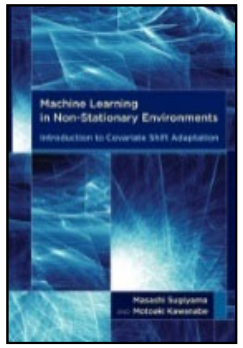
  - Extendable to Bregman divergences: Sugiyama, Suzuki & Kanamori (AISM2012)

$$\operatorname*{argmin}_r \mathrm{Breg}_\psi \left(\frac{p_{\mathrm{te}}(\boldsymbol{x})}{p_{\mathrm{tr}}(\boldsymbol{x})} \middle\| r \right) = \operatorname*{argmin}_r \mathbb{E}_{p_{\mathrm{tr}}(\boldsymbol{x})}[\partial\psi(r(\boldsymbol{x}))r(x) - \psi(r(\boldsymbol{x}))] - \mathbb{E}_{p_{\mathrm{te}}(\boldsymbol{x})}[\partial\psi(r(\boldsymbol{x}))]$$

Joint Importance-Predictor Estimation

- The classical approaches are two steps:

Sugiyama & Kawanabe (MIT Press 2012)



1. Importance weight estimation (e.g., LSIF):

$$\widehat{r} = \operatorname*{argmin}_{r} \widehat{J}_1(r) \qquad J_1(r) = \mathbb{E}_{p_{\mathrm{tr}}(\boldsymbol{x})}\left[\left(r(\boldsymbol{x}) - \frac{p_{\mathrm{te}}(\boldsymbol{x})}{p_{\mathrm{tr}}(\boldsymbol{x})}\right)^2\right]$$

2. Importance-weighted predictor training:

$$\widehat{f} = \operatorname*{argmin}_{f} \widehat{J}_2^{\ell}(f, \widehat{r}) \qquad J_2^{\ell}(f, r) = \mathbb{E}_{p_{\mathrm{tr}}(\boldsymbol{x}, y)}\left[r(\boldsymbol{x})\ell(f(\boldsymbol{x}), y)\right]$$

- For $\ell_{\mathrm{te}} \leq 1, \ell_{\mathrm{tr}} \geq \ell_{\mathrm{te}}, r \geq 0$, the test risk is bounded as

$$\frac{1}{2} R_{\ell_{\mathrm{te}}}(f)^2 \leq J_{\ell_{\mathrm{tr}}}(f, r) \qquad J_{\ell}(f, r) = J_1(r) + J_2^{\ell}(f, r)$$

$$R_{\ell}(f) = \mathbb{E}_{p_{\mathrm{te}}(\boldsymbol{x}, y)}[\ell(f(\boldsymbol{x}), y)]$$

Zhang, Yamane, Lu & Sugiyama (ACML2020, SNCS2021)

- Joint upper-bound minimization: $\widehat{f} = \operatorname*{argmin}_{f} \min_{r \geq 0} \widehat{J}_{\ell_{\mathrm{tr}}}(f, r)$

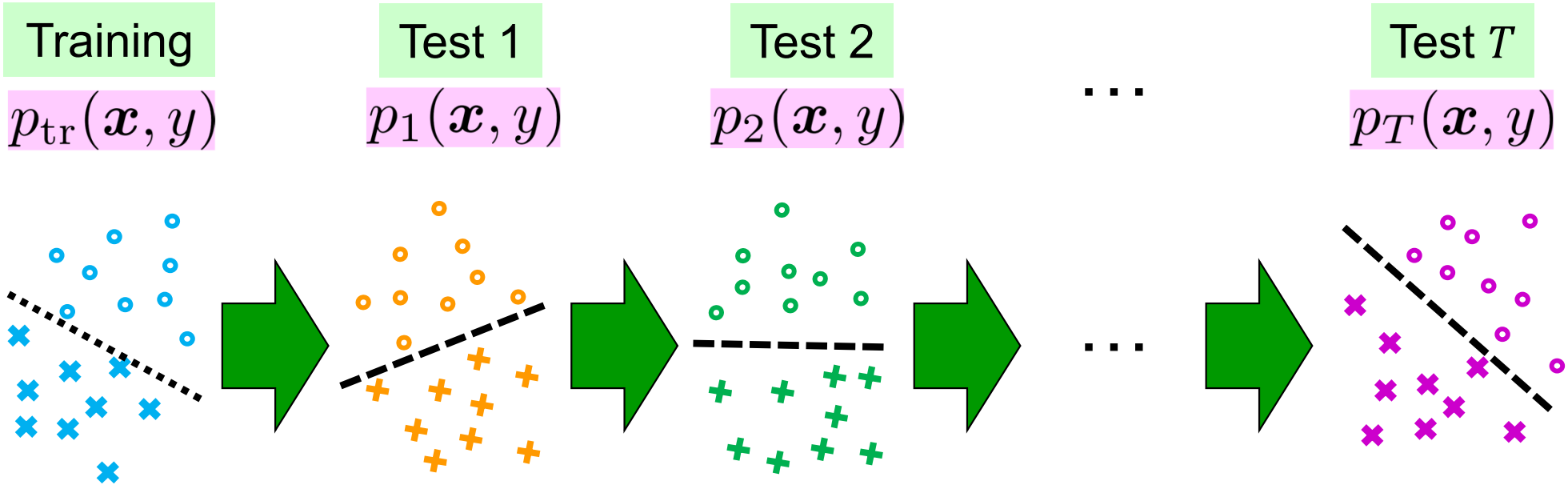


Contents

1. Importance Weighting
2. Sequential Shifts
3. Ongoing Challenges

Sequential Distribution Shifts

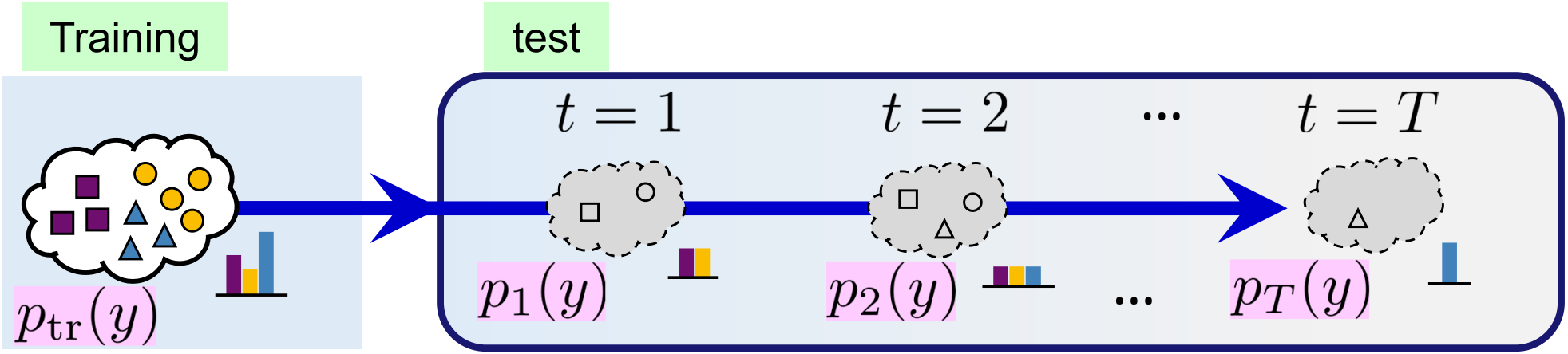
- So far, we focused on a fixed test domain:
  - We trained a predictor to match the test domain.
- However, test domains can change over time.



- Goal: Obtain predictor $\widehat{f}_t$ that works well for $p_t(\boldsymbol{x}, y)$.

$$R_t(f) = \mathbb{E}_{p_t(\boldsymbol{x},y)}[\ell(f(\boldsymbol{x}), y)] \quad t = 1, \ldots, T$$

Sequential Label Shift



- Class-priors $p_t(y)$ change arbitrarily over time, but class-conditionals stay unchanged:

$$p_t(\boldsymbol{x}|y) = p_{\mathrm{tr}}(\boldsymbol{x}|y) \quad t = 1, \ldots, T$$

- Given:
  - (Large) labeled training data: $\{(\boldsymbol{x}_i^{\mathrm{tr}}, y_i^{\mathrm{tr}})\}_{i=1}^{n_{\mathrm{tr}}} \overset{\text{i.i.d.}}{\sim} p_{\mathrm{tr}}(\boldsymbol{x}, y)$
  - (Small) unlabeled test data: $\{\boldsymbol{x}_i^{(t)}\}_{i=1}^{n_t} \overset{\text{i.i.d.}}{\sim} p_t(\boldsymbol{x})$

$$t = 1, \ldots, T$$

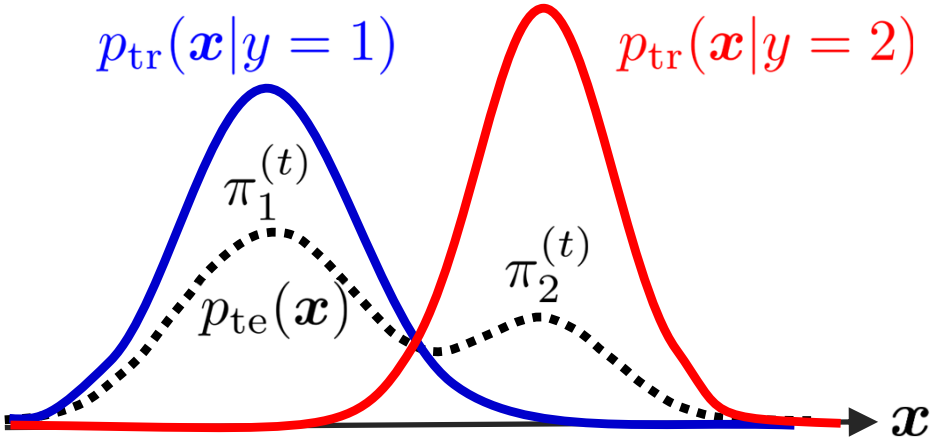
Batch Importance Weighting for Label Shift

du Plessis & Sugiyama (NN2014)

- At time step t, $p_t(y)$ can be estimated by implicit distribution matching (no density estimation is needed):

$$\min_{\boldsymbol{\pi}^{(t)} \in \Delta_{c-1}} \mathrm{Div}[p_{\mathrm{te}}(\boldsymbol{x}) \| q_t(\boldsymbol{x})]$$

$$q_t(\boldsymbol{x}) = \sum_{y=1}^{c} \pi_y^{(t)} p_{\mathrm{tr}}(\boldsymbol{x}|y)$$



- Perform importance-weighted training:

$$\min_f \sum_{i=1}^{n_{\mathrm{tr}}} \frac{\widehat{p}_t(y_i^{\mathrm{tr}})}{\widehat{p}_{\mathrm{tr}}(y_i^{\mathrm{tr}})} \ell(f(\boldsymbol{x}_i^{\mathrm{tr}}), y_i^{\mathrm{tr}})$$

ATLAS (Adapting To LAbel Shift)

Bai, Zhang, Zhao, Sugiyama & Zhou (NeurIPS2022)

$$\min_f \sum_{i=1}^{n_{\mathrm{tr}}} \frac{\widehat{p}_t(y_i^{\mathrm{tr}})}{\widehat{p}_{\mathrm{tr}}(y_i^{\mathrm{tr}})} \ell(f(\boldsymbol{x}_i^{\mathrm{tr}}), y_i^{\mathrm{tr}})$$

- Batch importance weighing requires retraining in each time step.
- Can we make it computationally more efficient?
  - Online learning!

Hazan (2016)

- We use online convex optimization, assuming
  - convex loss ℓ (e.g., logistic),
  - linear model $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x}, \quad \boldsymbol{w} \in \mathcal{W}$.

$$\boldsymbol{w}_t = \Pi_{\mathcal{W}} \left[\boldsymbol{w}_{t-1} - \eta \nabla \widehat{R}_{t-1}(\boldsymbol{w}_{t-1}) \right]$$

$\Pi_{\mathcal{W}}$: projection

$\eta > 0$: step size

- We use black box shift estimation for class priors.

Lipton+ (ICML2018)

Choice of Step Size η

$$\boldsymbol{w}_t = \Pi_{\mathcal{W}}\left[\boldsymbol{w}_{t-1} - \eta \nabla \widehat{R}_{t-1}(\boldsymbol{w}_{t-1})\right]$$

- If the speed of distribution shift is
  - slow, η should be small to keep the previous classifier.
  - fast, η should be large to quickly update the classifier.
- How do we choose η in practice?
  - Ensemble learning! Zhao+ (NeurIPS2020)
- For $0 < \eta_1 < \cdots < \eta_M$, we run M learners:

$$\boldsymbol{w}_t^{(m)} = \Pi_{\mathcal{W}}\left[\boldsymbol{w}_{t-1}^{(m)} - \eta_m \nabla \widehat{R}_{t-1}(\boldsymbol{w}_{t-1}^{(m)})\right] \quad \widehat{R}_t(\boldsymbol{w}) = \frac{1}{n_{\mathrm{tr}}} \sum_{i=1}^{n_{\mathrm{tr}}} \frac{\widehat{p}_t(y_i^{\mathrm{tr}})}{\widehat{p}_{\mathrm{tr}}(y_i^{\mathrm{tr}})} \ell(\boldsymbol{w}^\top \boldsymbol{x}_i^{\mathrm{tr}}, y_i^{\mathrm{tr}})$$

- Final output is the weighted average (cf. Hedge):

Freund+ (JCSS1997)

$$\boldsymbol{w}_t = \sum_{m=1}^{M} p_t^{(m)} \boldsymbol{w}_t^{(m)} \quad p_t^{(m)} \propto \exp\left(-\varepsilon \sum_{s=1}^{t-1} \widehat{R}_s(\boldsymbol{w}_s^{(m)})\right) \quad \varepsilon = \Theta\left(\sqrt{\frac{\ln M}{T}}\right)$$

Theoretical Analysis

- Shift intensity: $V_T = \sum_{t=2}^{T} \sum_{y=1}^{c} |p_t(y) - p_{t-1}(y)| \geq \Theta(T^{-\frac{1}{2}})$
- When V_T is known:
  - Dynamic regret is minimized with step size $\eta = \Theta(V_T^{\frac{1}{3}} T^{-\frac{1}{3}})$:

$$\mathbb{E}\left[\sum_{t=1}^{T} R_t(\boldsymbol{w}_t) - \sum_{t=1}^{T} \min_{\boldsymbol{w} \in \mathcal{W}} R_t(\boldsymbol{w})\right] = \mathcal{O}(V_T^{\frac{1}{3}} T^{\frac{2}{3}})$$

Risk of our model — Risk of the best model at each iteration

$$R_t(\boldsymbol{w}) = \mathbb{E}_{p_{\mathrm{tr}}(\boldsymbol{x}|y)p_t(y)}[\ell(\boldsymbol{w}^\top \boldsymbol{x}, y)]$$

- Even when V_T is unknown:
  - ATLAS still achieves the same dynamic regret!
    - Number of learners: $M = 1 + \lceil \frac{1}{2} \log_2(1 + 2T) \rceil$
    - Step size: $\eta_m = 2^{m-1} Z / \sqrt{T}, \quad m = 1, \ldots, M$

ATLAS with Hints

- If we have some hints, can we perform better?
  - Hint function: $H_t(\boldsymbol{w}) \approx R_t(\boldsymbol{w}) = \mathbb{E}_{p_{\mathrm{tr}}(\boldsymbol{x}|y)p_t(y)}[\ell(\boldsymbol{w}^\top \boldsymbol{x}, y)]$
- ATLAS-ADA:
  - $\boldsymbol{w}_{t-\frac{1}{2}} = \Pi_{\mathcal{W}}\left[\boldsymbol{w}_{t-1} - \eta \nabla \widehat{R}_{t-1}(\boldsymbol{w}_{t-1})\right]$ ← Same as ATLAS
  - $\boldsymbol{w}_t = \operatorname*{argmin}_{\boldsymbol{w}\in\mathcal{W}}\left[\eta H_t(\boldsymbol{w}) + \frac{1}{2}\|\boldsymbol{w} - \boldsymbol{w}_{t-\frac{1}{2}}\|^2\right]$ ← Use the hint to match the next loss
- Dynamic regret: $\mathbb{E}\left[\sum_{t=1}^{T} R_t(\boldsymbol{w}_t) - \sum_{t=1}^{T} \min_{\boldsymbol{w}\in\mathcal{W}} R_t(\boldsymbol{w})\right] = \mathcal{O}\left(V_T^{\frac{1}{3}} G_T^{\frac{1}{3}} T^{\frac{1}{3}}\right)$
  - Reusability: $G_T = \sum_{t=1}^{T} \mathbb{E}\left[\sup_{\boldsymbol{w}\in\mathcal{W}} \|\nabla \widehat{R}_t(\boldsymbol{w}) - \nabla H_t(\boldsymbol{w})\|^2\right] \leq \mathcal{O}(T)$
  - ATLAS-ADA is better and safe! $\mathcal{O}(V_T^{\frac{1}{3}} T^{\frac{2}{3}})$

Sequential Covariate Shift

- Input density $p_t(\boldsymbol{x})$ changes arbitrarily over time, but output-given-input is unchanged:

$$p_t(y|\boldsymbol{x}) = p_{\mathrm{tr}}(y|\boldsymbol{x}) \quad t = 1, \ldots, T$$

- Given:
  - (Large) labeled training data: $\{(\boldsymbol{x}_i^{\mathrm{tr}}, y_i^{\mathrm{tr}})\}_{i=1}^{n_{\mathrm{tr}}} \overset{\text{i.i.d.}}{\sim} p_{\mathrm{tr}}(\boldsymbol{x}, y)$
  - (Small) unlabeled test data: $\{\boldsymbol{x}_i^{(t)}\}_{i=1}^{n_t} \overset{\text{i.i.d.}}{\sim} p_t(\boldsymbol{x})$

$$t = 1, \ldots, T$$

- Batch importance weighting with Bregman density ratio estimation (DRE):

Sugiyama, Suzuki & Kanamori (AISM2012)

$$\min_f \widehat{R}_t(f) \quad \widehat{R}_t(f) = \frac{1}{n_{\mathrm{tr}}} \sum_{i=1}^{n_{\mathrm{tr}}} \widehat{r}_t(\boldsymbol{x}_i^{\mathrm{tr}}) \ell(f(\boldsymbol{x}_i^{\mathrm{tr}}), y_i^{\mathrm{tr}}) \quad \widehat{r}_t(\boldsymbol{x}) = \operatorname*{argmin}_{r \in \mathcal{R}} \widehat{L}_t^{\psi}(r)$$

$$\widehat{L}_t^{\psi}(r) = \frac{1}{n_{\mathrm{tr}}} \sum_{i=1}^{n_{\mathrm{tr}}} (\partial\psi(r(\boldsymbol{x}_i^{\mathrm{tr}}))r(x) - \psi(r(\boldsymbol{x}_i^{\mathrm{tr}}))) - \frac{1}{n_t} \sum_{i=1}^{n_t} [\partial\psi(r(\boldsymbol{x}_i^{\mathrm{te}}))]$$

Risk-Regret Bound via DRE-Regret

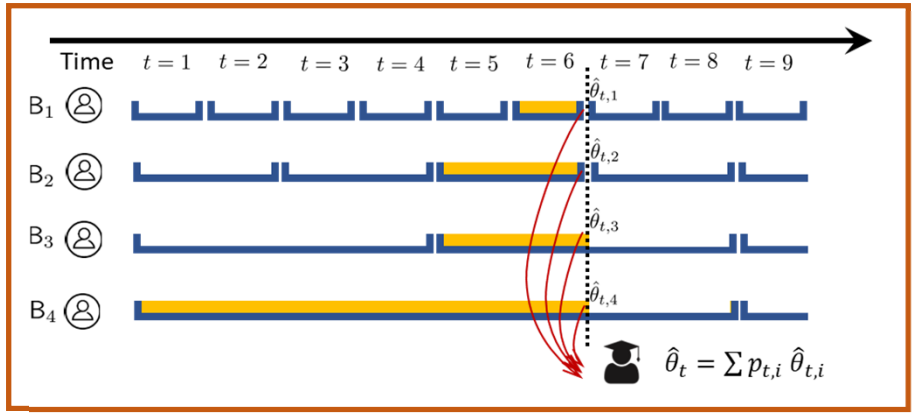
Zhang, Zhang, Zhao & Sugiyama (NeurIPS2023)

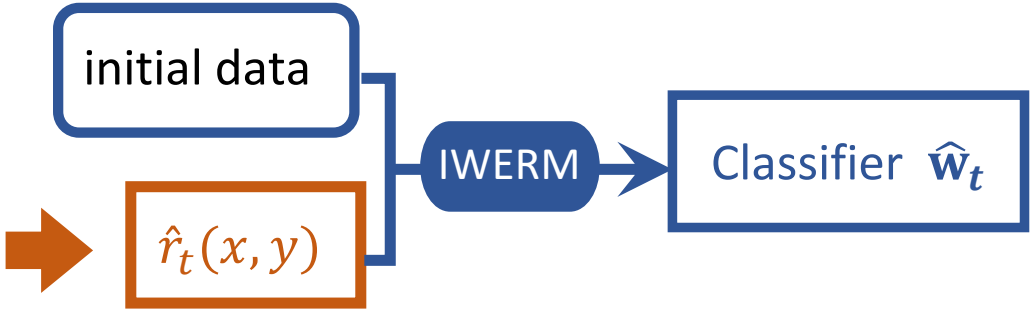
- For μ-strongly convex ψ, $\forall \widehat{r}_t \in \mathcal{R}$, with prob. $1-\delta$,

$$\sum_{t=1}^{T} R_t(\widehat{\boldsymbol{w}}_t) - \sum_{t=1}^{T} \min_{\boldsymbol{w}\in\mathcal{W}} R_t(\boldsymbol{w}) \leq \sqrt{\frac{8T}{\mu}\left[\sum_{t=1}^{T} \widetilde{L}_t^{\psi}(\widehat{r}_t) - \sum_{t=1}^{T} \widetilde{L}_t^{\psi}(\tfrac{p_t}{p_{\mathrm{tr}}})\right]} + \mathcal{O}\left(\frac{T\log(T/\delta)}{\mu\sqrt{n_{\mathrm{tr}}}}\right)$$

$$\widetilde{L}_t^{\psi}(r) = \frac{1}{n_{\mathrm{tr}}}\sum_{i=1}^{n_{\mathrm{tr}}}(\partial\psi(r(\boldsymbol{x}_i^{\mathrm{tr}}))r(x) - \psi(r(\boldsymbol{x}_i^{\mathrm{tr}}))) - \mathbb{E}_{p_t(\boldsymbol{x})}[\partial\psi(r(\boldsymbol{x}))]$$

- How do we approximate $\mathbb{E}_{p_t(\boldsymbol{x})}$?
  - With only $\{\boldsymbol{x}_i^{(t)}\}_{i=1}^{n_t}$: High variance
  - With all $\{\boldsymbol{x}_i^{(t')}\}_{i=1}^{n_{t'}}$ for $t' = 1,\ldots,t$: High bias
  - Use an ensemble!







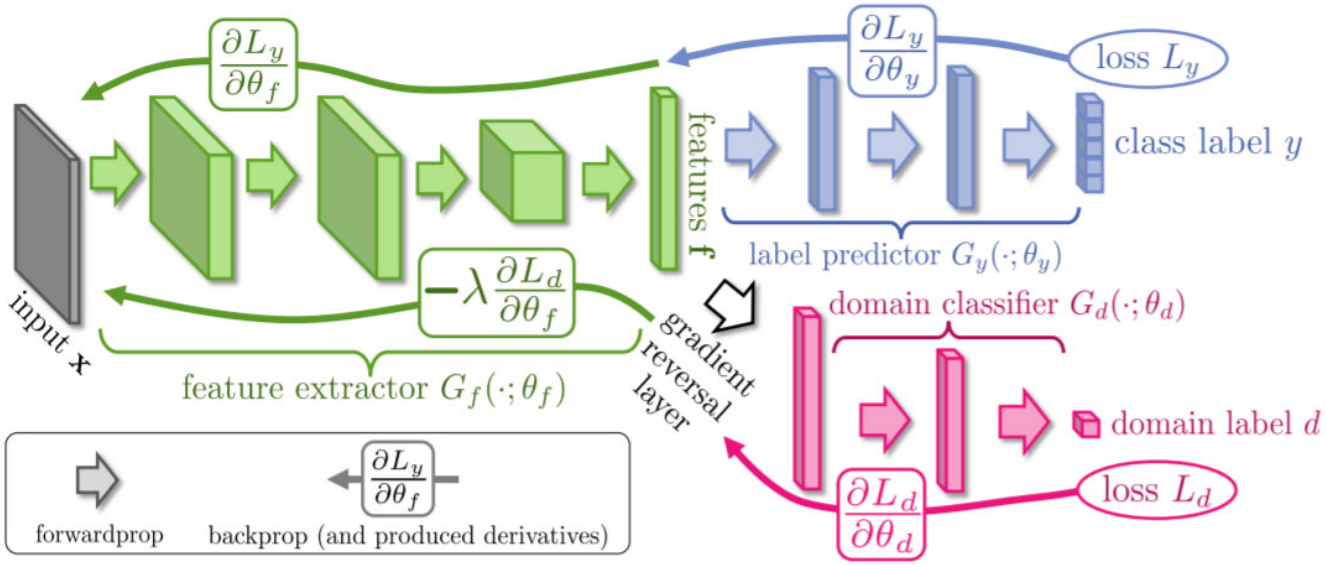
Contents

1. Importance Weighting
2. Sequential Shifts
3. Ongoing Challenges

Domain Matching

- Domain matching is another popular approach for transfer learning in deep learning:

Ganin & Lempitsky (ICML2015)
Ben-David+ (NIPS2006)



- The covariate shift assumption is essential:

$$p_{\mathrm{tr}}(\boldsymbol{x}) \neq p_{\mathrm{te}}(\boldsymbol{x}) \quad p_{\mathrm{tr}}(y|\boldsymbol{x}) = p_{\mathrm{te}}(y|\boldsymbol{x})$$

Joint Distribution Shifts

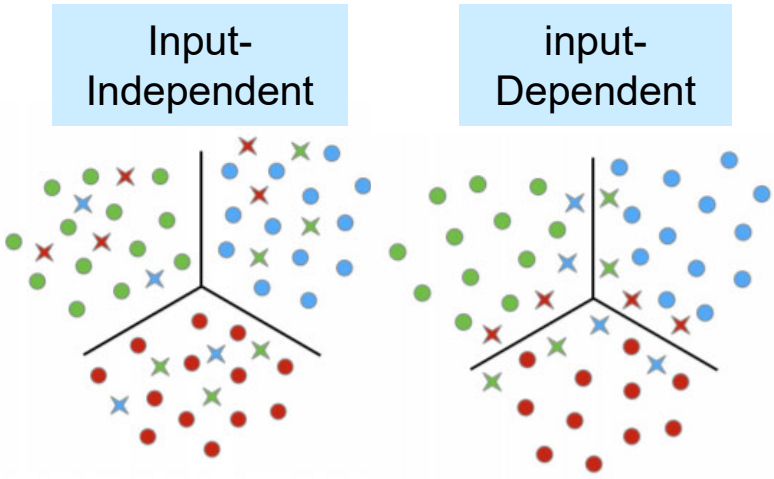
- Many distribution shift works considers a specific shift type (e.g., covariate shift).
  - However, identification of the shift type is challenging.
- Label noise is also a type of distribution shift:

$$p_{\mathrm{tr}}(\bar{y}|\boldsymbol{x}) = \sum_{y} \underbrace{p(\bar{y}|y,\boldsymbol{x})}_{\text{Noise transition}} p_{\mathrm{te}}(y|\boldsymbol{x})$$

$\bar{y}$: Noisy class label

  - Nice theory for input-independent noise. Li, Liu, Han, Niu & Sugiyama (ICML2021)
  - But input-dependent noise is hard to handle.

Xia, Liu, Han, Wang, Gong, Liu, Niu, Tao & Sugiyama (NeurIPS2020)
Berthon, Han, Niu, Liu & Sugiyama (ICML2021)
Cheng, Liu, Ning, Wang, Han, Niu, Gao & Sugiyama (CVPR2022)



- Let's consider joint shift:

$$p_{\mathrm{tr}}(\boldsymbol{x}, y) \neq p_{\mathrm{te}}(\boldsymbol{x}, y)$$

Correction with Noise Transition

- Noise transition matrix T:
  - Clean-to-noisy flipping probability.

$$T = \begin{bmatrix} 1 & 0 & 0 \\ 0.1 & 0.8 & 0.1 \\ 0.5 & 0.5 & 0 \end{bmatrix}$$

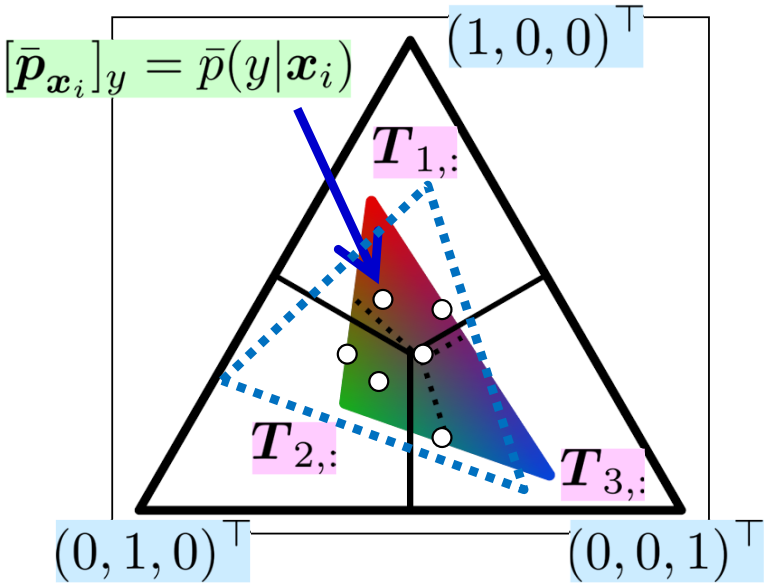
- Major approaches: Patrini+ (CVPR2017)
  - Classifier adjustment by $T^\top$ to simulate noise.
  - Loss correction by T^{-1} to eliminate noise.

- We want to estimate T only from noisy data:
  - Use human cognition as a "mask" for T. Han+ (NeurIPS2018)
  - Reduce estimation error of T. Xia+ (NeurIPS2019) Yao+ (NeurIPS2020)
  - Learn T and classifier simultaneously. Zhang+ (ICML2021)
  - Estimate T under weaker conditions. Li+ (ICML2021)

Volume Minimization

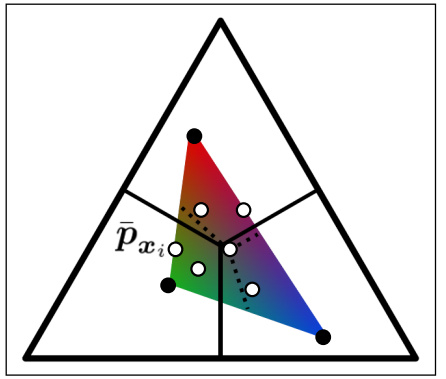
Li+ (ICML2021)

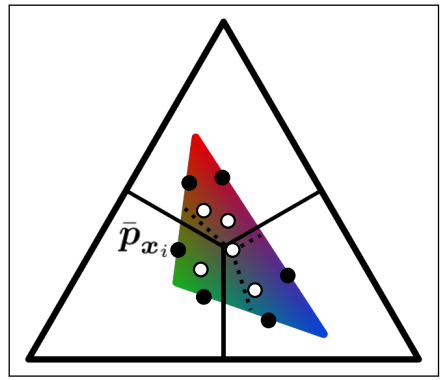
- Noise transition matrix $\boldsymbol{T}$ forms a simplex.
- Noisy training data $\{(\boldsymbol{x}_i, \bar{y}_i)\}_{i=1}^n$ can be mapped in the simplex.



- Find a minimum volume simplex that contains all training data:

$$\min_{\boldsymbol{T}', \boldsymbol{g}} \sum_{i=1}^{n} \ell(\bar{y}_i, \boldsymbol{T}'^\top \boldsymbol{g}(\boldsymbol{x}_i)) + \lambda \log\det(\boldsymbol{T}') \qquad \lambda > 0$$





  - With noiseless labels, we can find the true $\boldsymbol{T}$.
  - Even without noiseless labels, "sufficiently scattered" training data allow identification of the true $\boldsymbol{T}$!

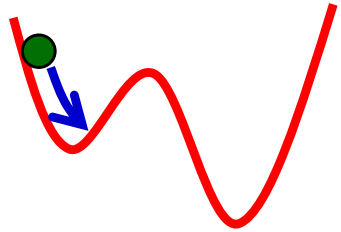
Mini-Batch-Wise Loss Matching

- Given:

Fang, Lu, Niu & Sugiyama (NeurIPS2020)

  - (Large) labeled training data: $\{(\boldsymbol{x}_i^{\mathrm{tr}}, y_i^{\mathrm{tr}})\}_{i=1}^{n_{\mathrm{tr}}} \overset{\text{i.i.d.}}{\sim} p_{\mathrm{tr}}(\boldsymbol{x}, y)$
  - (Small) labeled test data: $\{(\boldsymbol{x}_j^{\mathrm{te}}, y_j^{\mathrm{te}})\}_{j=1}^{n_{\mathrm{te}}} \overset{\text{i.i.d.}}{\sim} p_{\mathrm{te}}(\boldsymbol{x}, y)$

- We try to learn the importance weight dynamically in the mini-batch-wise manner.



$$f \leftarrow f - \eta \nabla \widehat{R}(f)$$

$\eta > 0$: step size

- For each mini-batch $\{(\tilde{\boldsymbol{x}}_i^{\mathrm{tr}}, \tilde{y}_i^{\mathrm{tr}})\}_{i=1}^{\tilde{n}_{\mathrm{tr}}}, \{(\tilde{\boldsymbol{x}}_j^{\mathrm{te}}, \tilde{y}_j^{\mathrm{te}})\}_{j=1}^{\tilde{n}_{\mathrm{te}}}$, importance weights are estimated by kernel mean matching for loss values:

Huang+ (NeurIPS2006)

$$\frac{1}{\tilde{n}_{\mathrm{tr}}} \sum_{i=1}^{\tilde{n}_{\mathrm{tr}}} r_i \ell(f(\tilde{\boldsymbol{x}}_i^{\mathrm{tr}}), \tilde{y}_i^{\mathrm{tr}}) \approx \frac{1}{\tilde{n}_{\mathrm{te}}} \sum_{j=1}^{\tilde{n}_{\mathrm{te}}} \ell(f(\tilde{\boldsymbol{x}}_j^{\mathrm{te}}), \tilde{y}_j^{\mathrm{te}})$$

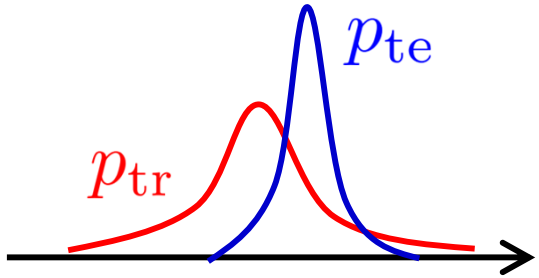
$$r_i \approx \frac{p_{\mathrm{te}}(\tilde{\boldsymbol{x}}_i^{\mathrm{tr}}, \tilde{y}_i^{\mathrm{tr}})}{p_{\mathrm{tr}}(\tilde{\boldsymbol{x}}_i^{\mathrm{tr}}, \tilde{y}_i^{\mathrm{tr}})}$$

Out-Of-Domain Extension

- Limitation of importance weighting:
  - The training domain must cover the test domain.

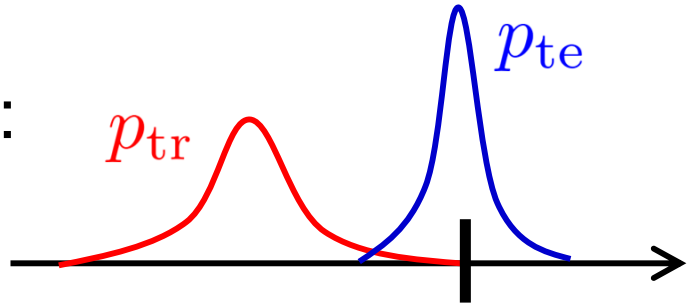
$$\frac{p_{\mathrm{te}}(\boldsymbol{x}, y)}{p_{\mathrm{tr}}(\boldsymbol{x}, y)} < \infty$$

- What if the test domain sticks out from the training domain?



- Out-of-domain extension: Fang, Lu, Niu & Sugiyama (NeurIPS2023)
  - Split training data into in-/out-domains by outlier detection (e.g., 1-class SVM):

$$\{(\boldsymbol{x}_j^{\mathrm{te_{in}}}, y_j^{\mathrm{te_{in}}})\}_{j=1}^{n_{\mathrm{te_{in}}}} \qquad \{(\boldsymbol{x}_j^{\mathrm{te_{out}}}, y_j^{\mathrm{te_{out}}})\}_{j=1}^{n_{\mathrm{te_{out}}}}$$

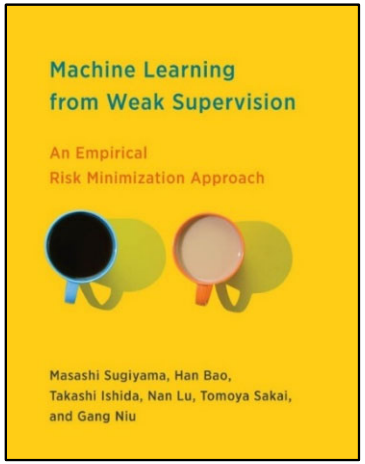


  - Compute the loss separately:

$$\frac{n_{\mathrm{te_{in}}}}{n_{\mathrm{tr}} n_{\mathrm{te}}} \sum_{i=1}^{n_{\mathrm{tr}}} \frac{p_{\mathrm{te}}(\boldsymbol{x}_i^{\mathrm{tr}}, y_i^{\mathrm{tr}})}{p_{\mathrm{tr}}(\boldsymbol{x}_i^{\mathrm{tr}}, y_i^{\mathrm{tr}})} \ell(f(\boldsymbol{x}_i^{\mathrm{tr}}), y_i^{\mathrm{tr}}) + \frac{1}{n_{\mathrm{te}}} \sum_{j=1}^{n_{\mathrm{te_{out}}}} \ell(f(\boldsymbol{x}_j^{\mathrm{te_{out}}}), y_j^{\mathrm{te_{out}}})$$

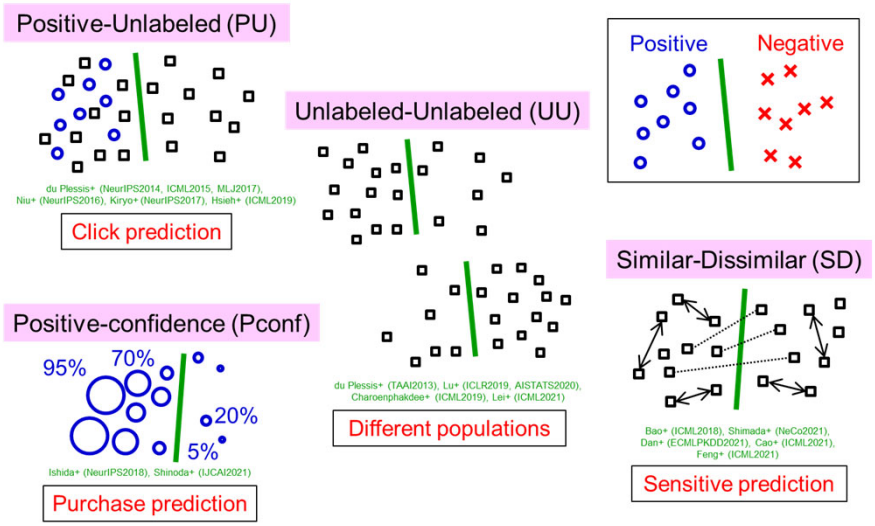
Joint Shift Adaptation from Weak Supervision

- For joint shift adaptation, requiring labeled test data is too strong.
- Can we use weakly supervised learning?
  - Unbiased risk estimation from weakly supervised data.
  - Any loss, classifier, and optimizer can be used.



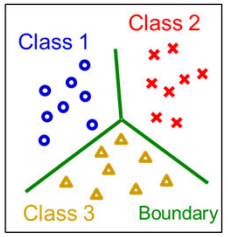
Sugiyama, Bao, Ishida, Lu, Sakai & Niu (MIT Press, 2022)

Weakly Supervised Classification (Binary)



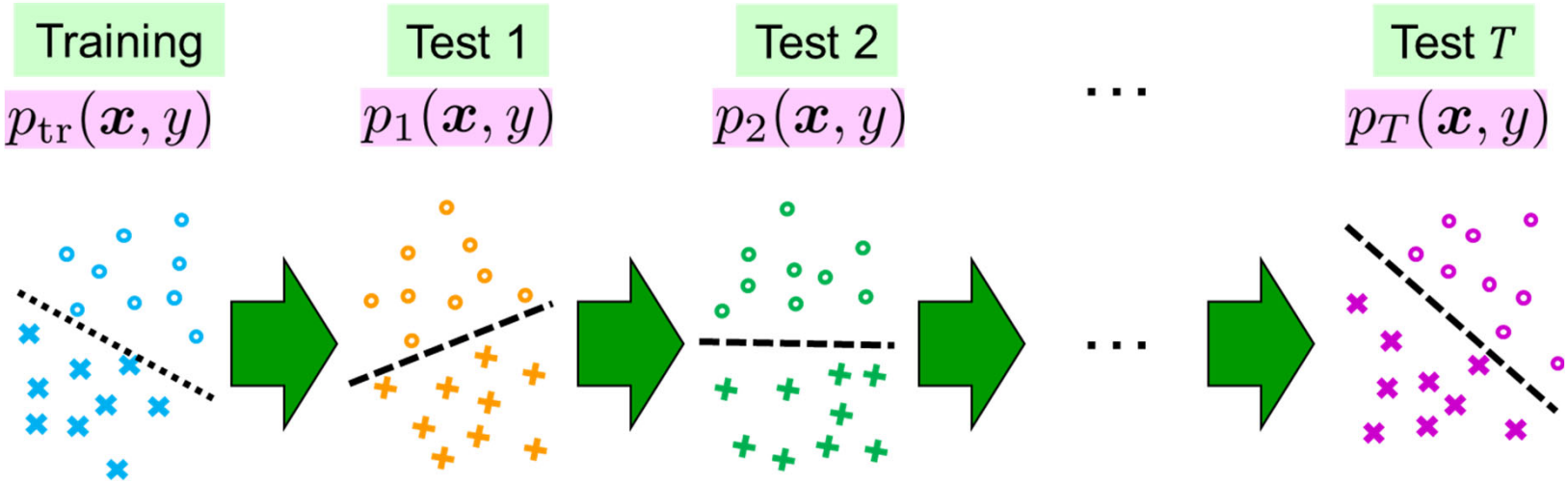
Weakly Supervised Classification (Multiclass)

- Multi-class weak-labels:
  - Complementary labels: Specify a class that a pattern does not belong to (“not 1”).
    Ishida et al. (NIPS2017, ICML2019), Chou et al. (ICML2020)
  - Partial labels: Specify a subset of classes that contains the correct one (“1 or 2”).
    Feng et al. (ICML2020, NeurIPS2020), Lv et al. (ICML2020)
  - Single-class confidence: One-class data with full confidence (“1 with 60%, 2 with 30%, and 3 with 10%”)
    Cao et al. (arXiv2021)



Extensions to Online Learning

- Can we handle sequential joint shift?



- Can we handle fixed-memory/no-memory scenarios?

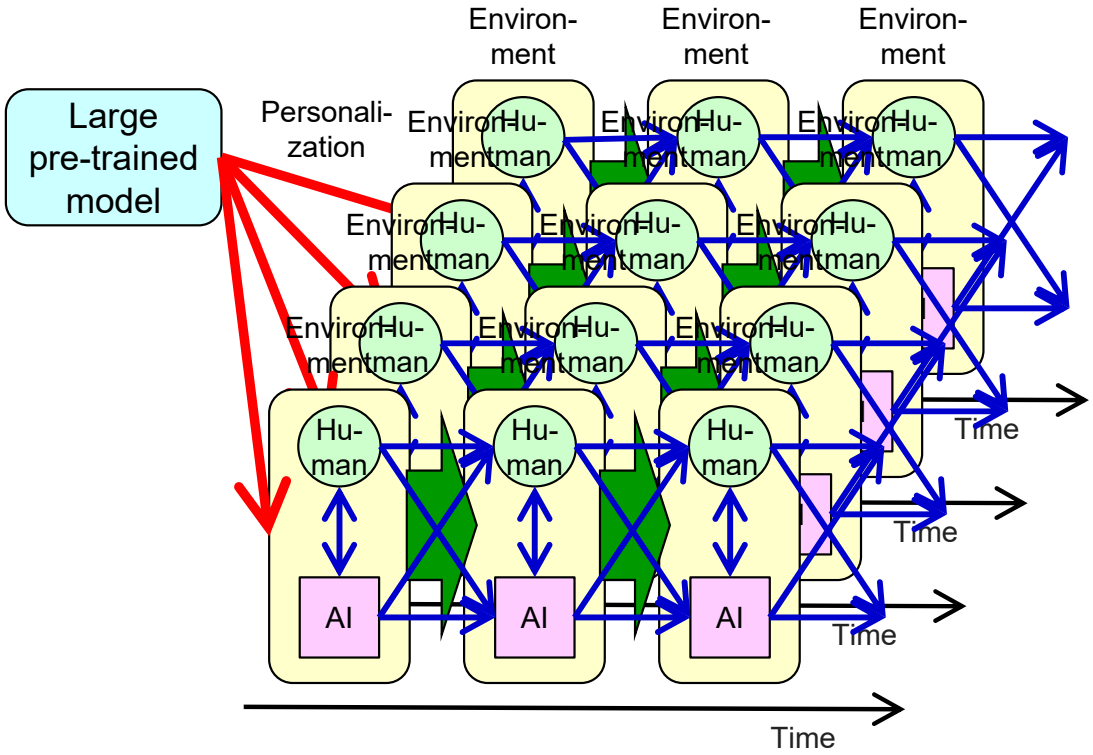
Importance Weighting for Overparameterized Models

$$\underset{f}{\mathrm{argmin}} \left[\sum_{i=1}^{n_{\mathrm{tr}}} \frac{p_{\mathrm{te}}(\boldsymbol{x}_i^{\mathrm{tr}}, y_i^{\mathrm{tr}})}{p_{\mathrm{tr}}(\boldsymbol{x}_i^{\mathrm{tr}}, y_i^{\mathrm{tr}})} \ell(f(\boldsymbol{x}_i^{\mathrm{tr}}), y_i^{\mathrm{tr}}) \right]$$

- For overparameterized models, the training loss is (nearly) zero, making the importance weights ineffective.
- However, in practice, importance weighting seems to be still useful.

Prospect: ML Research in the Era of Large Pre-Trained Models

- Generative AI changed our lifestyle:
  - Just as people have their own personalities, a large pre-trained model should also be personalized.



- Learning in dynamic environments:
  - Online/continual learning
  - Imperfect/structured learning
  - Sequential decision making
  - Probabilistic inference
- Learning in diverse environments:
  - Multi-agent learning/interaction
  - Multi-modal/multi-task learning
  - Privacy-preserving learning
  - Adversarial learning
  - Fairness-aware learning
- Computational/energy efficiency:
  - Large model/big data learning
  - Distributed optimization
  - Compressed data learning
  - Quantum computation