ICLR2023, Kigali, Rwanda

May 2, 2023

Importance-Weighting Approach to Distribution Shift Adaptation

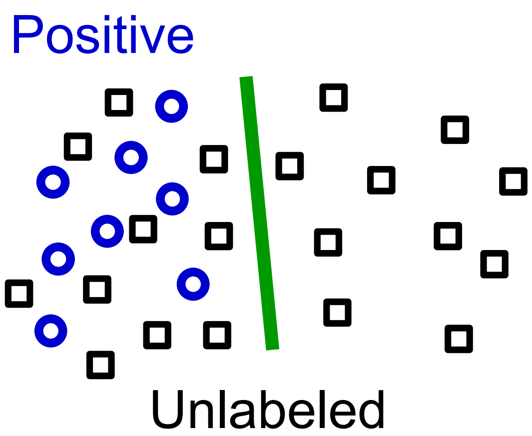
Masashi Sugiyama

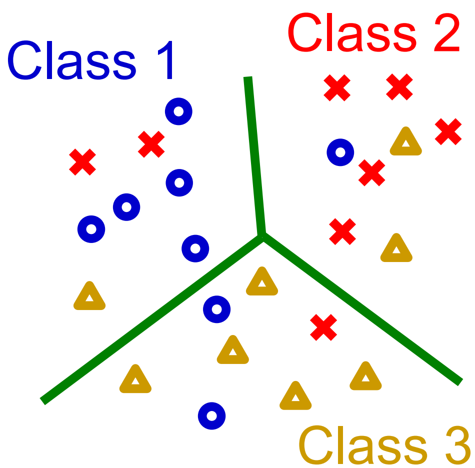
RIKEN Center for Advanced Intelligence Project/
The University of Tokyo, Japan

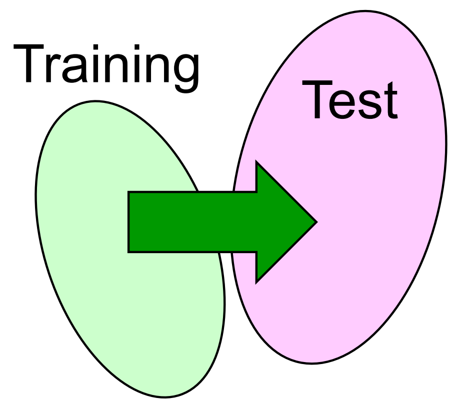
Slides→

Reliable Machine Learning

- Reliability of machine learning systems can be degraded by various factors:
  - Insufficient information: weak supervision.
  - Label noise: human error, sensor error.
  - Data bias: changing environments, privacy.
- Improving the reliability is an urgent challenge!







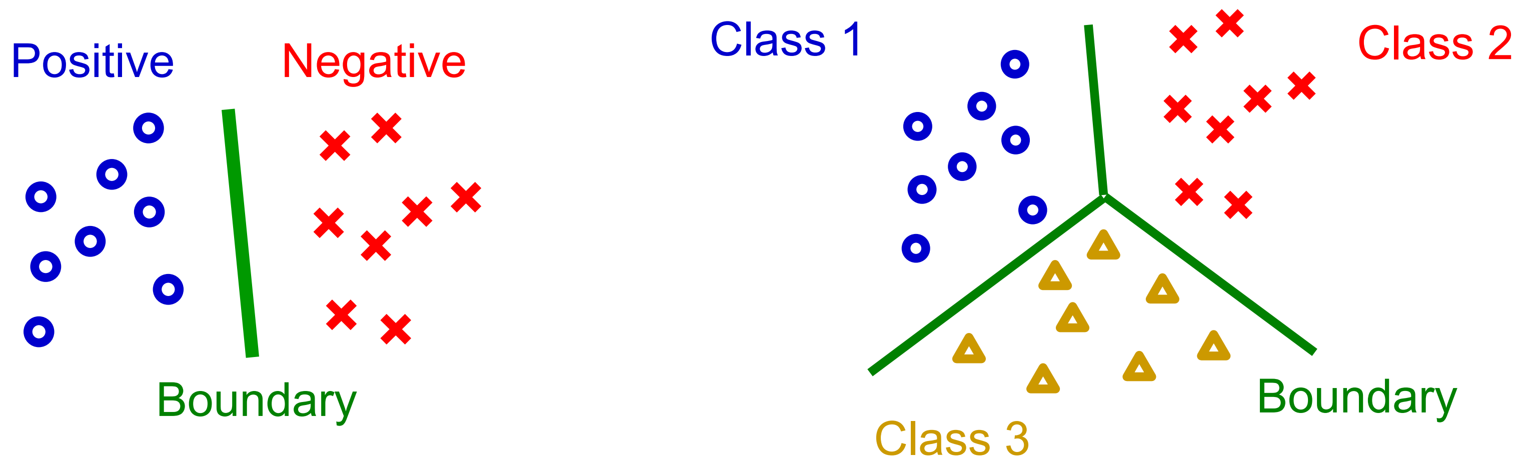
Contents

1. Weakly Supervised Learning
   A) Positive-Unlabeled Classification
   B) Various Extensions
2. Noisy-Label Learning
3. Transfer Learning
4. Towards More Reliable Learning

Slides→

Weakly Supervised Classification

- Supervised classification from big labeled data is successful: speech, image, language, …



- However, there are many applications where big labeled data is not available:
  - Medicine, disaster, robot, brain, …

- We want to utilize "weak" supervision that can be collected easily!

Contents

1. Weakly Supervised Learning
   A) Positive-Unlabeled Classification
   B) Various Extensions
2. Noisy-Label Learning
3. Transfer Learning
4. Towards More Reliable Learning

Slides→

Positive-Unlabeled (PU) Classification

Li+ (IJCAI2003)

- **Given:** PU samples (no N samples).

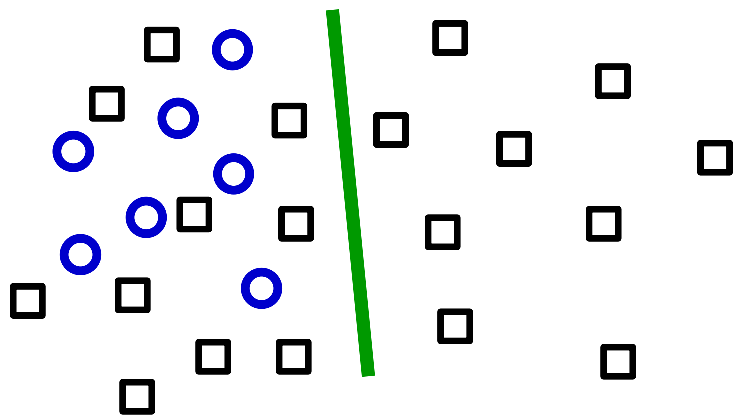
$$\{\boldsymbol{x}_i^{\mathrm{P}}\}_{i=1}^{n_{\mathrm{P}}} \overset{\mathrm{i.i.d.}}{\sim} p(\boldsymbol{x}|y=+1) \qquad \{\boldsymbol{x}_j^{\mathrm{U}}\}_{j=1}^{n_{\mathrm{U}}} \overset{\mathrm{i.i.d.}}{\sim} p(\boldsymbol{x})$$

- **Goal:** Obtain a classifier minimizing the PN risk.

$$\min_f R(f) \qquad R(f) = \mathbb{E}_{p(\boldsymbol{x},y)}\left[\ell\Big(y, f(\boldsymbol{x})\Big)\right]$$

$\mathbb{E}$: expectation $\quad$ ℓ : loss $\quad$ $y = \{+1, -1\}$

Positive [Negative]



Unlabeled (mixture of positives and negatives)

Example: Ad click prediction

- Clicked ad: User likes it → P
- Unclicked ad: User dislikes it or User likes it but doesn't have time to click it → U (=P or N)

PU Unbiased Risk Estimation

du Plessis+ (NeurIPS2014, ICML2015)

- Decompose the risk:

$$R(f) = \underbrace{\pi \mathbb{E}_{p(\boldsymbol{x}|y=+1)}\Big[\ell\Big(+1, f(\boldsymbol{x})\Big)\Big]}_{\text{Risk for P data}} + \underbrace{(1-\pi)\mathbb{E}_{p(\boldsymbol{x}|y=-1)}\Big[\ell\Big(-1, f(\boldsymbol{x})\Big)\Big]}_{\text{Risk for N data } R^{-}(f)}$$

$\pi = p(y=+1)$: Class prior (assumed known) →

Scott+ (AISTATS2009)
Ramaswamy+ (ICML2016)
du Plessis+ (MLJ2017)

- Without N data, $R^{-}(f)$ can not be estimated directly:
  - Eliminate the expectation over N data as

$$R^{-}(f) = \mathbb{E}_{p(\boldsymbol{x})}\Big[\ell\Big(-1, f(\boldsymbol{x})\Big)\Big] - \pi \mathbb{E}_{p(\boldsymbol{x}|y=+1)}\Big[\ell\Big(-1, f(\boldsymbol{x})\Big)\Big]$$

$$p(\boldsymbol{x}) = \pi p(\boldsymbol{x}|y=+1) + (1-\pi)p(\boldsymbol{x}|y=-1)$$

- Unbiased risk estimator:

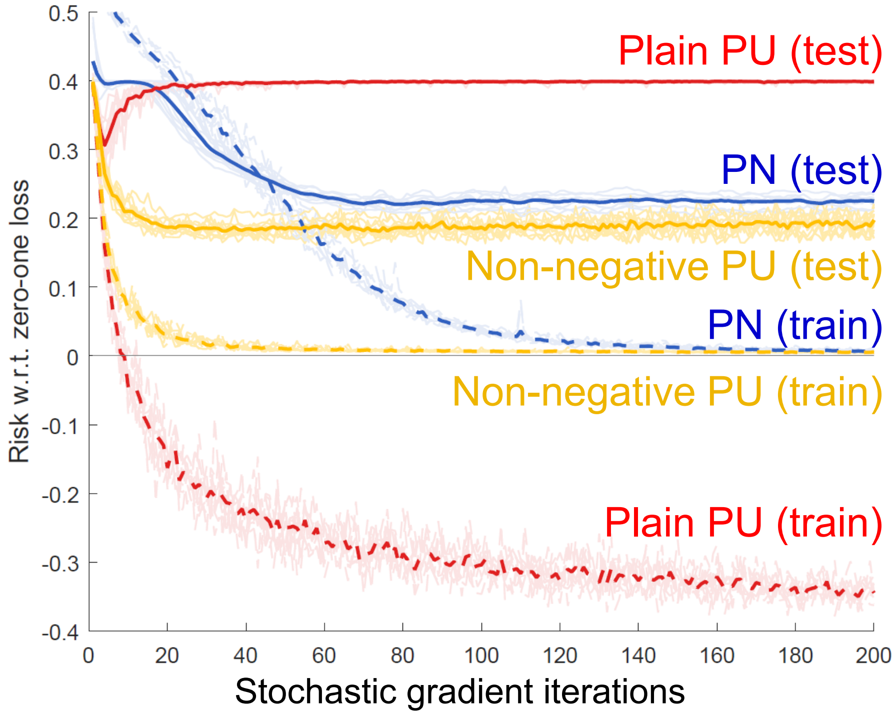
$$\widehat{R}_{\mathrm{PU}}(f) = \frac{\pi}{n_{\mathrm{P}}}\sum_{i=1}^{n_{\mathrm{P}}}\ell\Big(+1, f(\boldsymbol{x}_i^{\mathrm{P}})\Big) + \frac{1}{n_{\mathrm{U}}}\sum_{j=1}^{n_{\mathrm{U}}}\ell\Big(-1, f(\boldsymbol{x}_j^{\mathrm{U}})\Big) - \frac{\pi}{n_{\mathrm{P}}}\sum_{i=1}^{n_{\mathrm{P}}}\ell\Big(-1, f(\boldsymbol{x}_i^{\mathrm{P}})\Big)$$

Non-Negative Risk Correction

Kiryo+ (NeurIPS2017) , Lu+ (AISTATS2020)

$$R(f) = \underbrace{\pi\mathbb{E}_{p(\boldsymbol{x}|y=+1)}\Big[\ell\Big(+1, f(\boldsymbol{x})\Big)\Big]}_{\text{Risk for P data}} + \underbrace{(1-\pi)\mathbb{E}_{p(\boldsymbol{x}|y=-1)}\Big[\ell\Big(-1, f(\boldsymbol{x})\Big)\Big]}_{\text{Risk for N data } R^-(f)}$$

- Risk for N data: $R^-(f) = \mathbb{E}_{p(\boldsymbol{x})}\Big[\ell\Big(-1, f(\boldsymbol{x})\Big)\Big] - \pi\mathbb{E}_{p(\boldsymbol{x}|y=+1)}\Big[\ell\Big(-1, f(\boldsymbol{x})\Big)\Big]$
- Empirical estimate: $\widehat{R}^-_{\mathrm{PU}}(f) = \frac{1}{n_\mathrm{U}}\sum_{i=1}^{n_\mathrm{U}}\ell\Big(-1, f(\boldsymbol{x}_i^\mathrm{U})\Big) - \frac{\pi}{n_\mathrm{P}}\sum_{i=1}^{n_\mathrm{P}}\ell\Big(-1, f(\boldsymbol{x}_i^\mathrm{P})\Big)$
- When loss is non-negative:
  - True $R^-(f)$ is non-negative.
  - But empirical estimate can be negative!



- Non-negative correction:

$$\widetilde{R}_{\mathrm{PU}}(f) = \frac{\pi}{n_\mathrm{P}}\sum_{i=1}^{n_\mathrm{P}}\ell\Big(f(\boldsymbol{x}_i^\mathrm{P})\Big) + \max\left\{0,\ \widehat{R}^-_{\mathrm{PU}}(f)\right\}$$

Contents

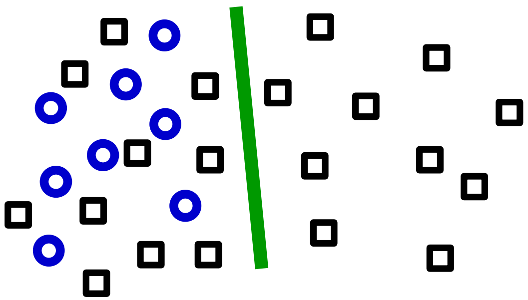
1. Weakly Supervised Learning
   A) Positive-Unlabeled Classification
   B) Various Extensions
2. Noisy-Label Learning
3. Transfer Learning
4. Towards More Reliable Learning

Slides→

Various Extensions (Binary)

- Similar unbiased risk estimation is possible!

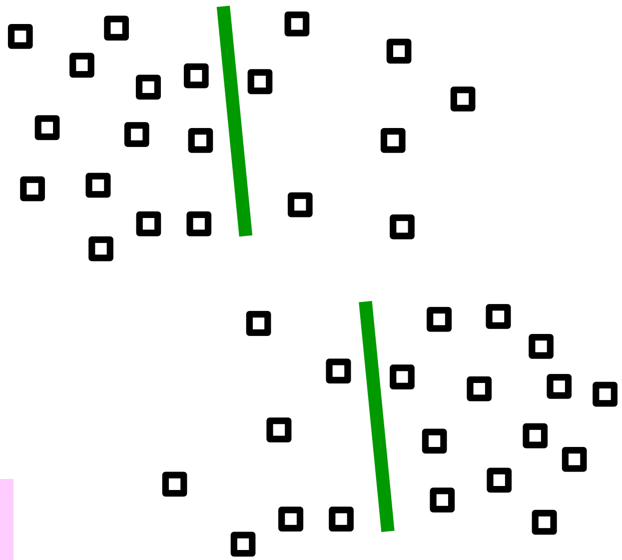
Positive-Unlabeled (PU)



du Plessis+ (NeurIPS2014, ICML2015, MLJ2017), Niu+ (NeurIPS2016), Kiryo+ (NeurIPS2017), Hsieh+ (ICML2019)

Click prediction

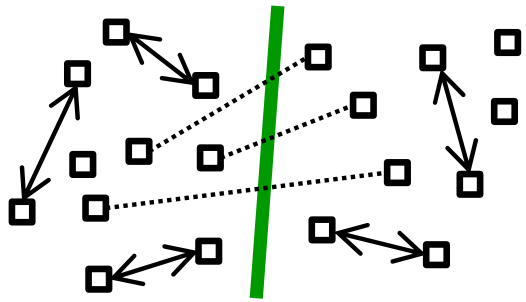
Unlabeled-Unlabeled (UU)



du Plessis+ (TAAI2013), Lu+ (ICLR2019, AISTATS2020), Charoenphakdee+ (ICML2019), Lei+ (ICML2021)

Different populations

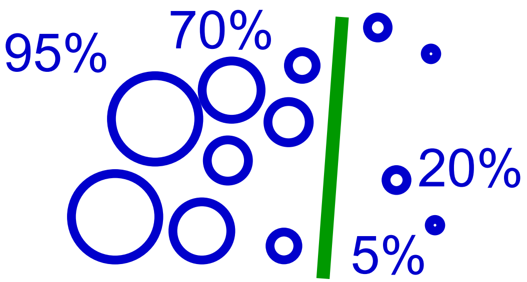
Similar-Dissimilar (SD)



Bao+ (ICML2018), Shimada+ (NeCo2021), Dan+ (ECMLPKDD2021), Cao+ (ICML2021), Feng+ (ICML2021)

Sensitive prediction

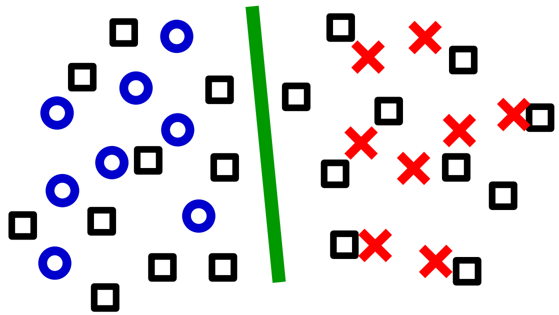
Positive-confidence (Pconf)



Ishida+ (NeurIPS2018), Shinoda+ (IJCAI2021)

Purchase prediction

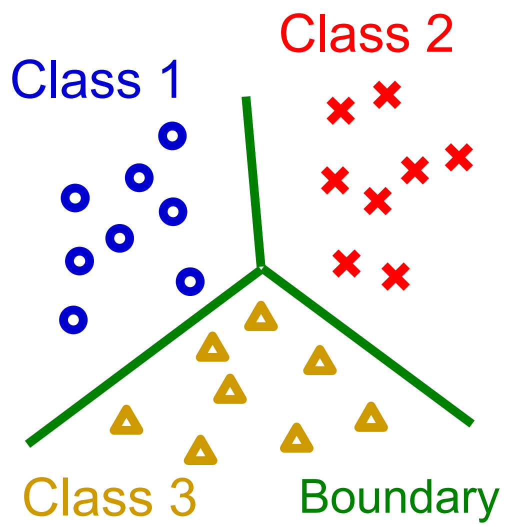
Positive-Negative-Unlabeled (PNU)



Sakai+ (ICML2017, ML2018)

Semi-supervised classification without manifold/clusters →

Various Extensions (Multiclass)



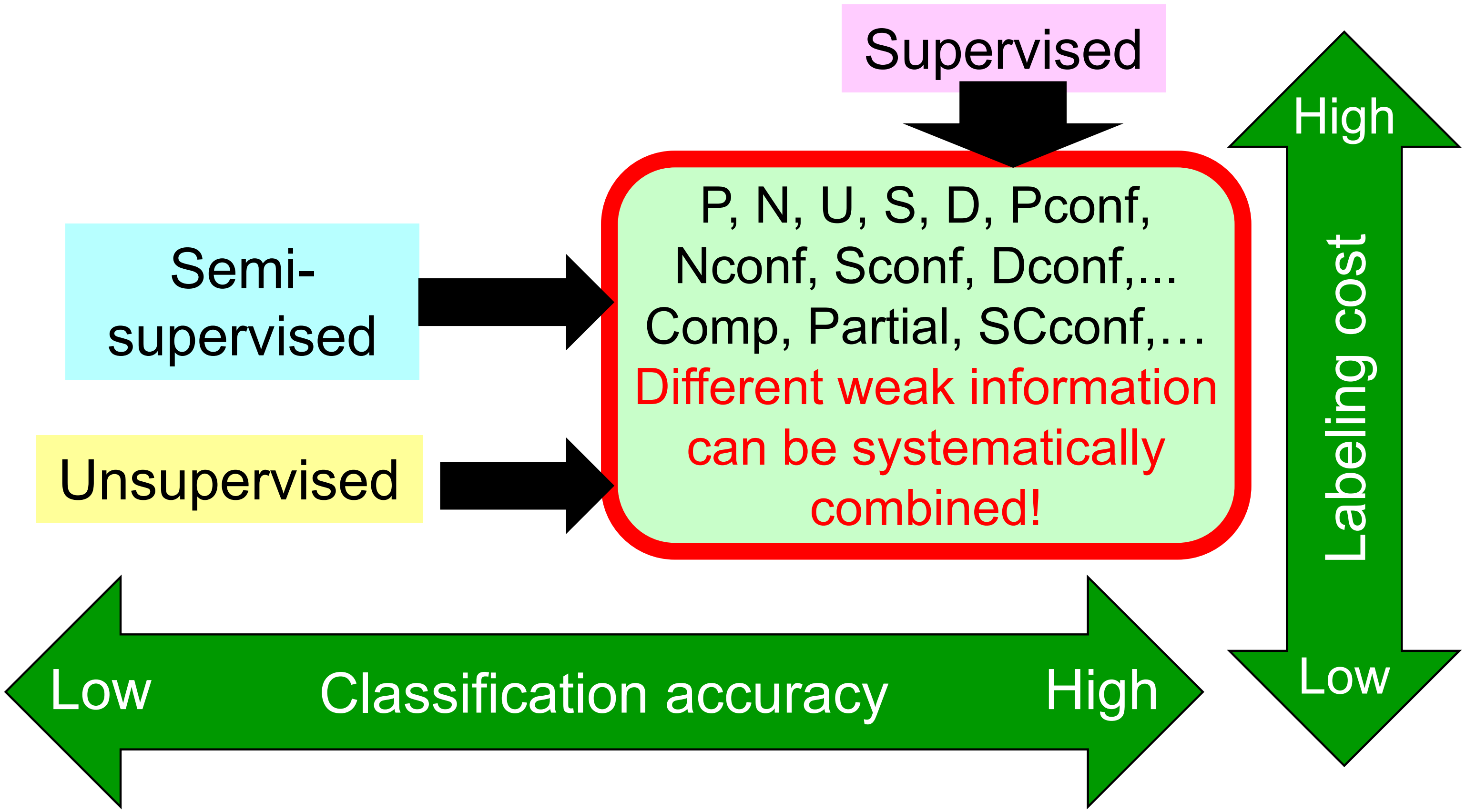
- Labeling patterns in multi-class problems is even more painful.

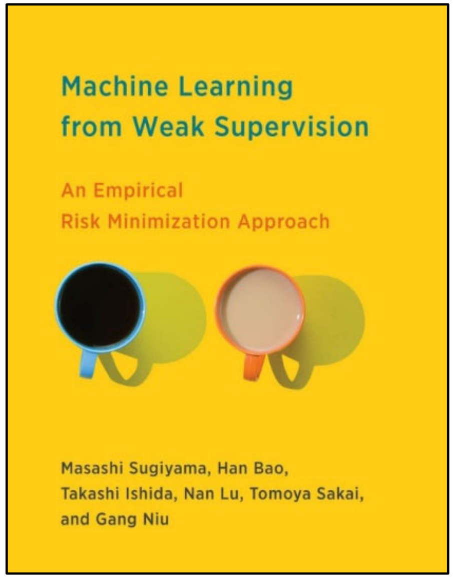
- Multi-class weak-labels:
  - Complementary label: Specifies a class that a pattern does not belong to ("not 1"). Ishida+ (NeurIPS2017, ICML2019), Chou+ (ICML2020)
  - Partial label: Specifies a subset of classes that contains the correct one ("1 or 2"). Feng+ (ICML2020, NeurIPS2020), Lv+ (ICML2020)
  - Single-class confidence: Cao+ (arXiv2021) One-class data with full confidence ("1 with 60%, 2 with 30%, and 3 with 10%")

- Similar unbiased risk estimation is possible!

Summary: Weakly Supervised Learning

- Empirical risk minimization framework for weakly supervised learning:
  - Any loss, classifier, and optimizer can be used.





Sugiyama, Bao, Ishida,
Lu, Sakai & Niu,
MIT Press, 2022.

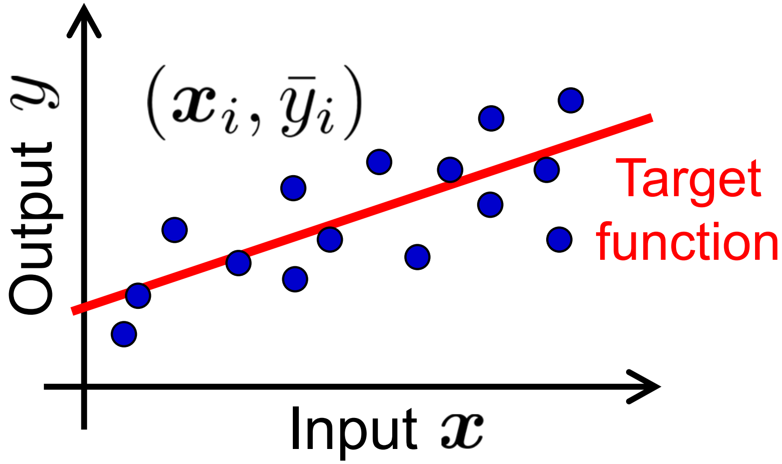
Contents

1. Weakly Supervised Learning
2. Noisy-Label Learning
   A) Noise Transition
   B) Algorithms
3. Transfer Learning
4. Towards More Reliable Learning

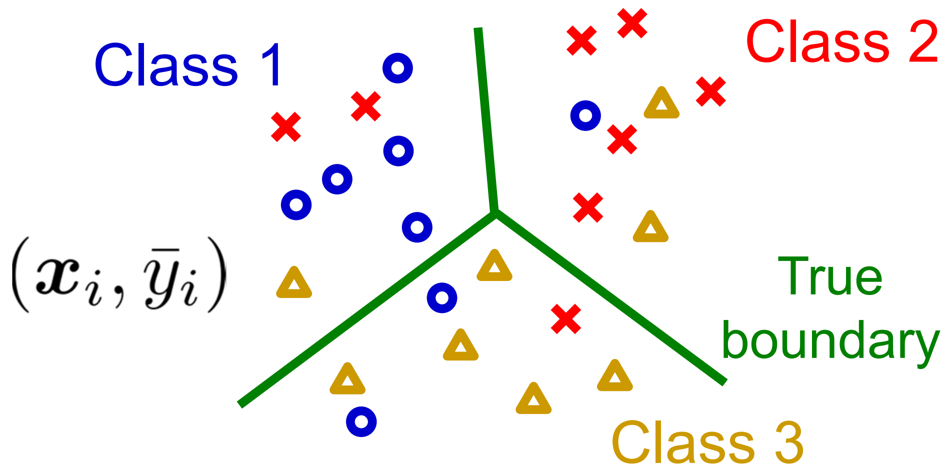
Slides→

Supervised Learning with Noisy Output

Regression (additive noise)



Classification (label flipping noise)



$$\min_{\boldsymbol{g}} \sum_{i=1}^{n} \ell\Big(\bar{y}_i, \boldsymbol{g}(\boldsymbol{x}_i)\Big)$$

ℓ: loss

$\bar{y}$: noisy output

$\boldsymbol{g}$: probabilistic classifier

- Hasn't such a classic problem been solved?
  - Regression: Yes, noisy big data yield consistency.
  - Classification: Specific noise reduction mechanism is needed to achieve consistency!

Contents

1. Weakly Supervised Learning
2. Noisy-Label Learning
   A) Noise Transition
   B) Algorithms
3. Transfer Learning
4. Towards More Reliable Learning

Slides→

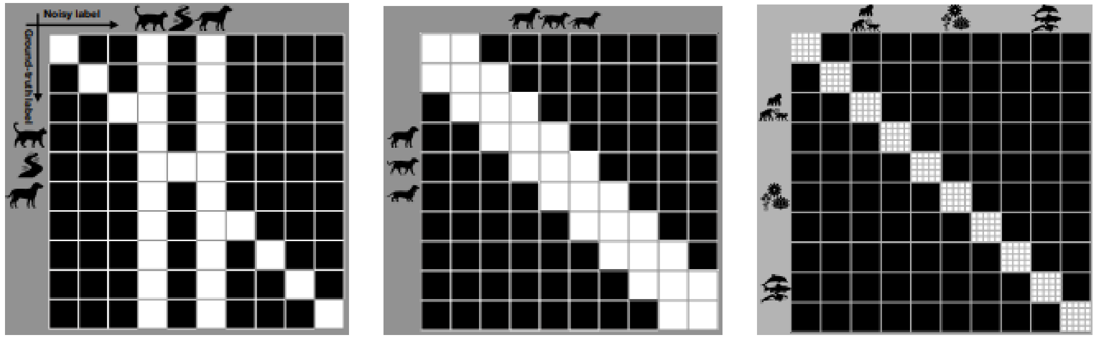
Modeling Input-Independent Noise

- Noise transition matrix: $T_{y,\bar{y}} = \bar{p}(\bar{y}|y)$
  - Probability of flipping y to $\bar{y}$.

$$\boldsymbol{T} = \begin{array}{c} \bar{y} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0.1 & 0.8 & 0.1 \\ 0.5 & 0.5 & 0 \end{bmatrix} \end{array} \, y$$

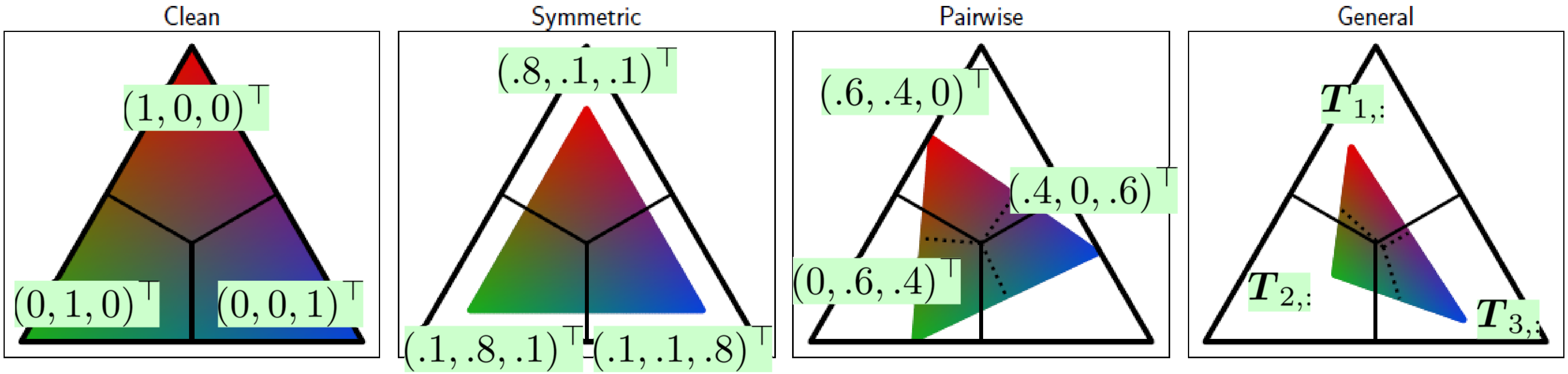
- Human-cognitive bias can be encoded in $\boldsymbol{T}$.

Han+ (NeurIPS2018)



(a) Column-diagonal (b) Tri-diagonal (c) Block-diagonal

- $\boldsymbol{T}$ can be visualized in a simplex.



Loss Correction with Noise Transition

Patrini+ (CVPR2017)

$\boldsymbol{\ell}$: vectorized loss

$\ell_y(\boldsymbol{g}(\boldsymbol{x})) = \ell(y, \boldsymbol{g}(\boldsymbol{x}))$

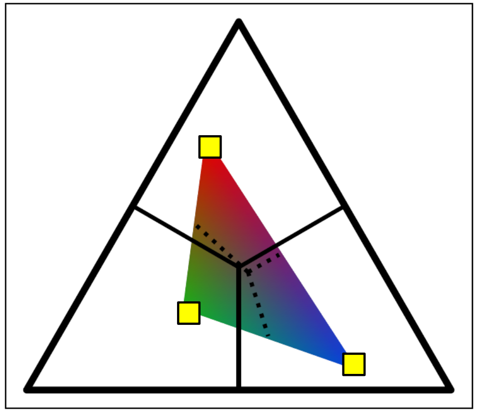
- Forward correction: $\boldsymbol{\ell}(\boldsymbol{T}^\top \boldsymbol{g}(\boldsymbol{x}))$
  - Add noise by $\boldsymbol{T}^\top$.
- Backward correction: $\boldsymbol{T}^{-1}\boldsymbol{\ell}(\boldsymbol{g}(\boldsymbol{x}))$
  - Remove noise by $\boldsymbol{T}^{-1}$.
- If $\boldsymbol{T}$ is given, consistency can be guaranteed!
- If $\boldsymbol{T}$ is unknown, how is it estimated?

Contents

1. Weakly Supervised Learning
2. Noisy-Label Learning
   A) Noise Transition
   B) Algorithms
3. Transfer Learning
4. Towards More Reliable Learning

Slides→

Basic Approach



- With noiseless labels, $\boldsymbol{T}$ can be obtained naively:
  - We know the vertices of the triangle.
- Can we estimate $\boldsymbol{T}$ from noisy labels?
  - Generally, $\boldsymbol{T}$ is non-identifiable:

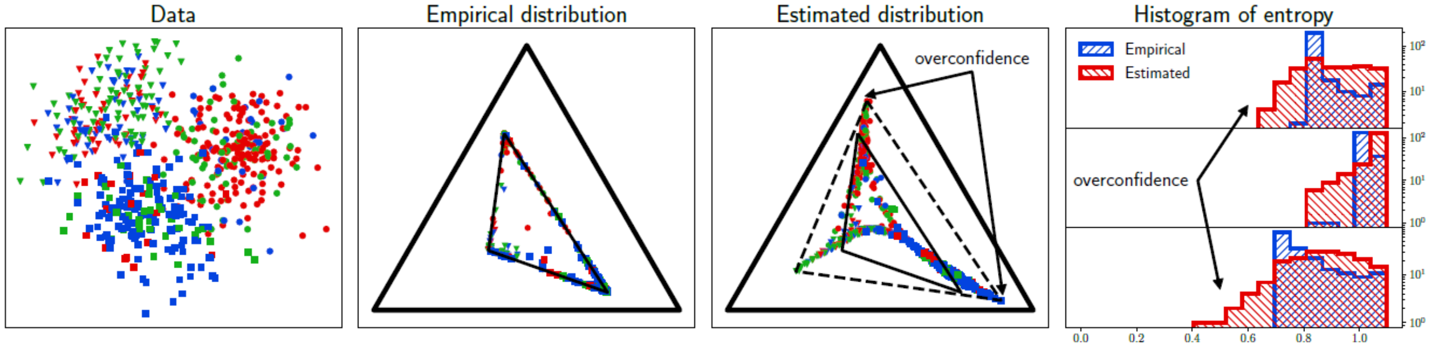
$$\boldsymbol{T}^\top \boldsymbol{p} = \boldsymbol{T}_2^\top (\boldsymbol{T}_1^\top \boldsymbol{p}) \qquad \boldsymbol{T} = \boldsymbol{T}_1 \boldsymbol{T}_2$$

- Assume noiseless labels exist in the training set:
  - Select the most confident data as noiseless ones.

$$\boldsymbol{x}^y \leftarrow \boldsymbol{x}_i \text{ s.t. } \widehat{g}_y(\boldsymbol{x}_i) \approx 1 \qquad \widehat{\boldsymbol{g}} = \operatorname*{argmin}_{\boldsymbol{g}} \sum_{i=1}^{n} \ell(\bar{y}_i, \boldsymbol{g}(\boldsymbol{x}_i))$$

Limitations

- Over-confidence of neural networks is harmful.



Zhang+ (ICML2021)

- The two-step nature magnifies the estimation error:
  1. Noise transition estimation: $\widehat{\boldsymbol{T}}$
  2. Classifier training with estimated $\widehat{\boldsymbol{T}}$:

$$\min_{\boldsymbol{g}} \sum_{i=1}^{n} \ell(\bar{y}_i, \widehat{\boldsymbol{T}}^\top \boldsymbol{g}(\boldsymbol{x}_i))$$

  - Naïve simultaneous estimation suffers non-identifiability.

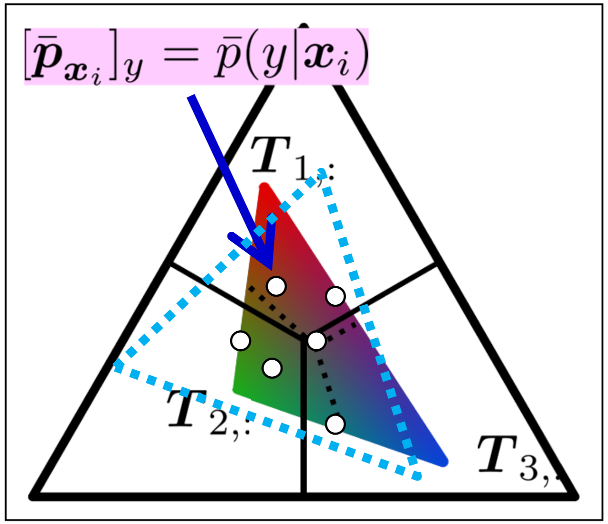
$$\min_{\boldsymbol{T}', \boldsymbol{g}} \sum_{i=1}^{n} \ell(\bar{y}_i, \boldsymbol{T}'^\top \boldsymbol{g}(\boldsymbol{x}_i))$$

- Assumption of having noiseless labels is too strong.

Volume Minimization

Li+ (ICML2021)

- Noisy training data $\{(\boldsymbol{x}_i, \bar{y}_i)\}_{i=1}^{n}$ can be mapped in the triangle formed by noise transition matrix $\boldsymbol{T}$.

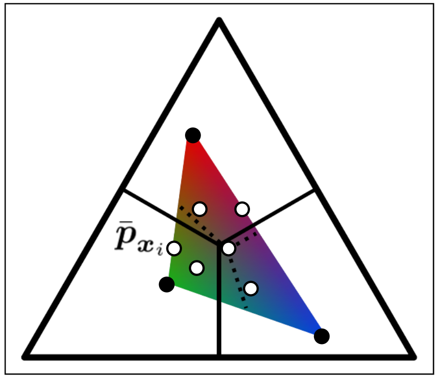


- Minimizing the volume of the triangle can give a solution:

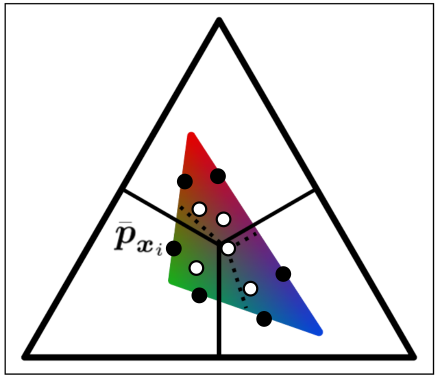
$$\min_{\boldsymbol{T}', \boldsymbol{g}} \sum_{i=1}^{n} \ell(\bar{y}_i, \boldsymbol{T}'^{\top} \boldsymbol{g}(\boldsymbol{x}_i)) + \lambda \log \det(\boldsymbol{T}')$$

$\lambda > 0$

  - With noiseless labels, we can find the true $\boldsymbol{T}$.

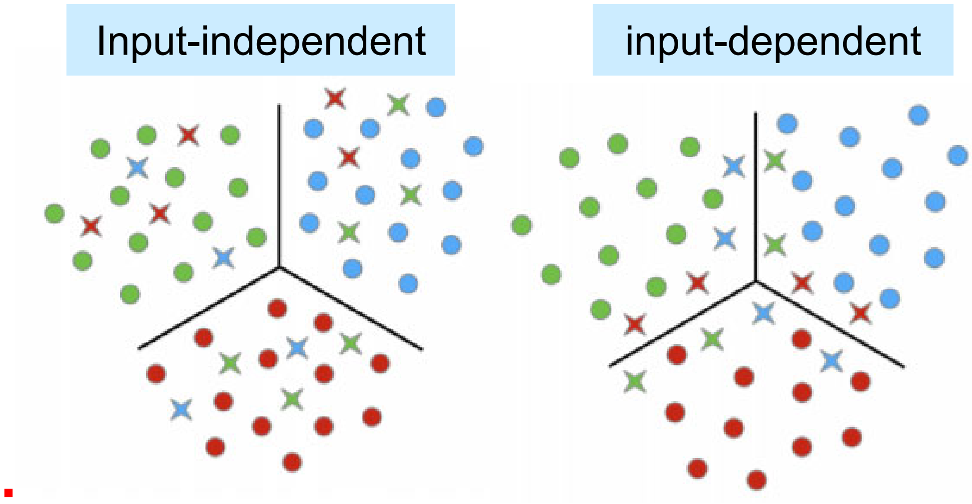


  - Even without noiseless labels, "sufficiently scattered" training data allow identification of the true $\boldsymbol{T}$!



Input-Dependent Noise

- Real-world noise is often input-dependent.
  - E.g., more noise near the boundary.



- Noise transition function:

$$T_{y,\bar{y}}(\boldsymbol{x}) = \bar{p}(\bar{y} | y, \boldsymbol{x})$$

  - Extremely challenging to estimate it!

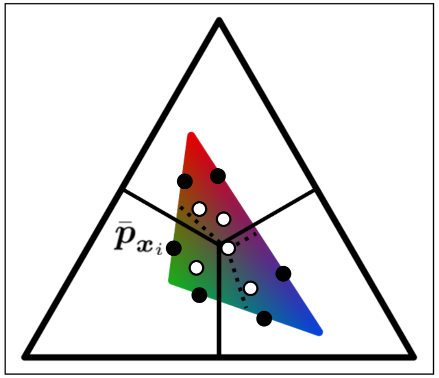
- Heuristics:
  - Parts-based estimation. Xia+ (NeurIPS2020)
  - Use of additional confidence scores. Berthon+ (ICML2021)
  - Manifold regularization. Cheng+ (CVPR2022)

Summary: Noisy-Label Learning

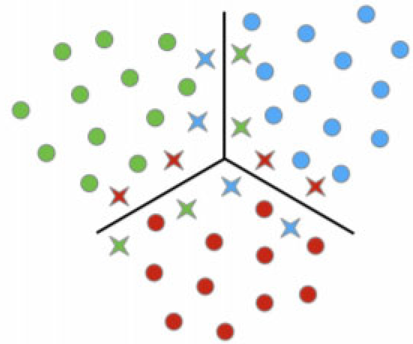
- Explicit treatment of label noise is necessary:
  - Loss correction by noise transition is promising.

$$T_{y,\bar{y}} = \bar{p}(\bar{y}|y)$$

- However, noise transition is generally non-identifiable:
  - Recent development allows consistent estimation under mild assumptions.



- Real-world noise is often input-dependent:
  - Heuristic solutions have been developed.
  - Further theoretical development is needed.



Contents

1. Weakly Supervised Learning
2. Noisy-Label Learning
3. Transfer Learning
   - A) Importance Weighting
   - B) Continuous Distribution Shift
4. Towards More Reliable Learning

Slides→

Learning under Distribution Shift

Given:

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

$\boldsymbol{x}$: Input

y : Output

Goal:

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

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

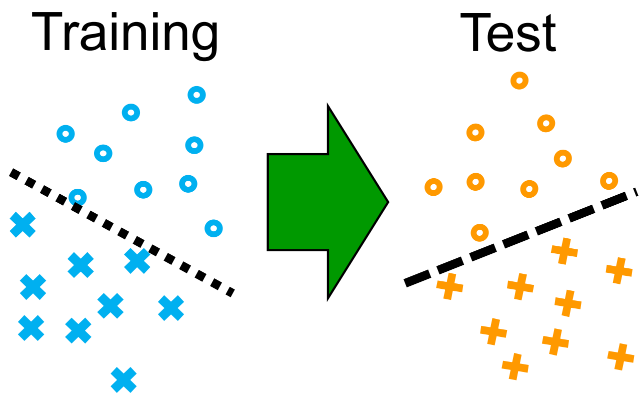
ℓ : loss

Challenge:

- Overcome changing distributions!

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

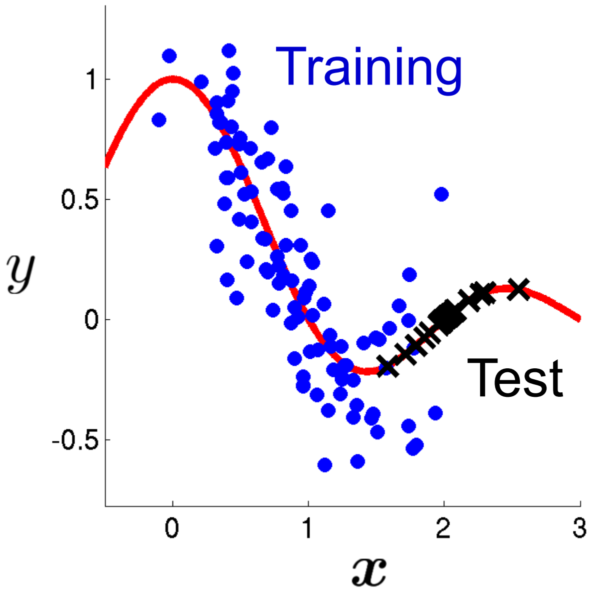
  - Non-stationary of the environments.
  - Sample selection bias due to privacy concerns.

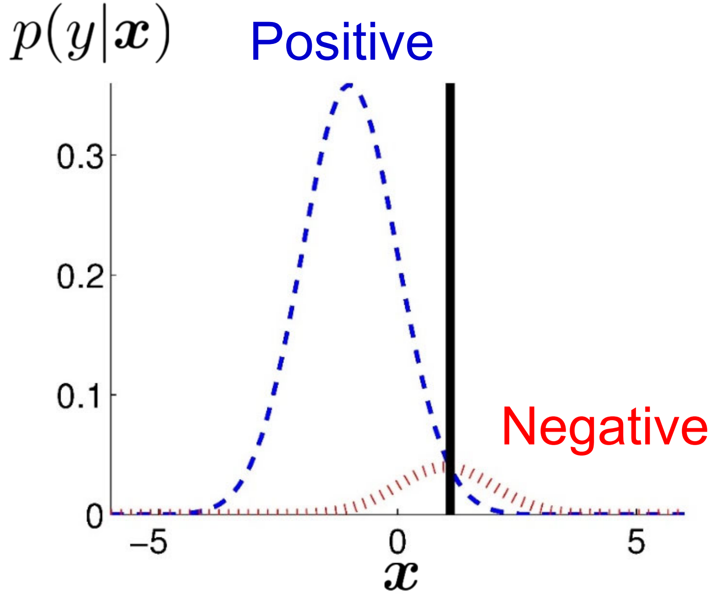


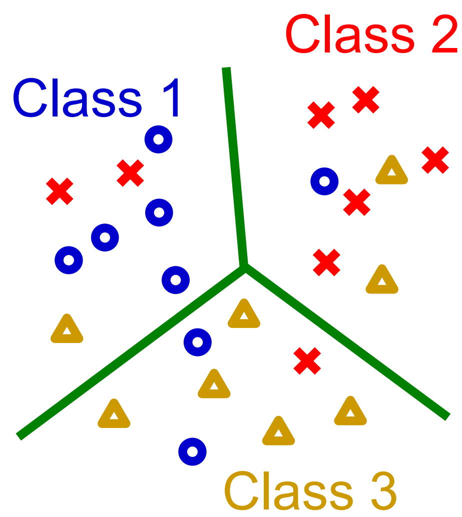
Types of Distribution Shift

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

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







Contents

1. Weakly Supervised Learning
2. Noisy-Label Learning
3. Transfer Learning
    - A) Importance Weighting
    - B) Continuous Distribution Shift
4. Towards More Reliable Learning

Slides→

Covariate Shift

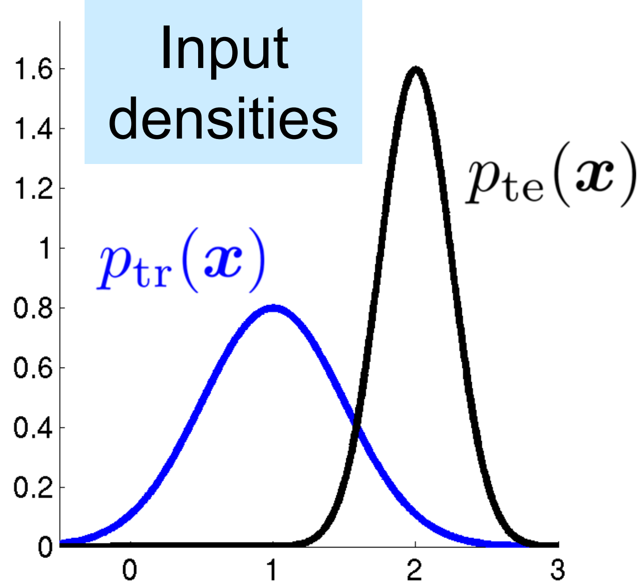
Shimodaira (JSPI2000)

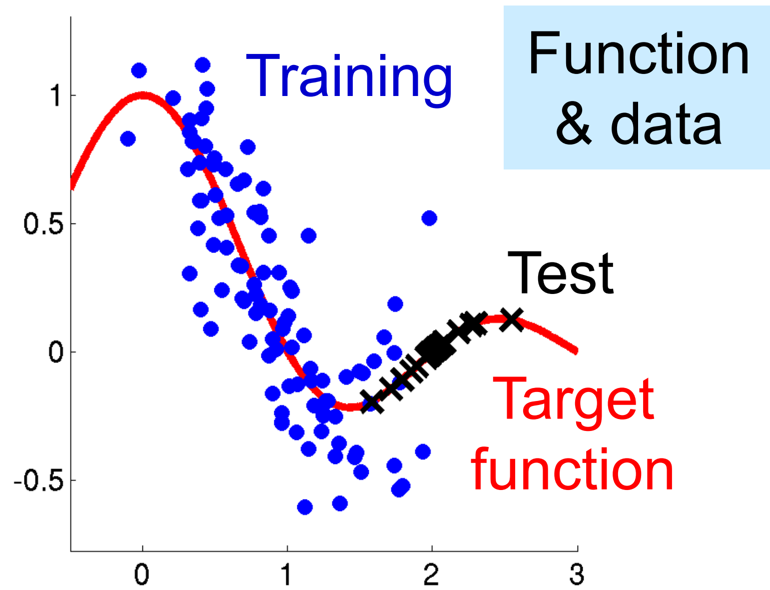
- Training and test input distributions are different,

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

  but the output-given-input distribution is unchanged:

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





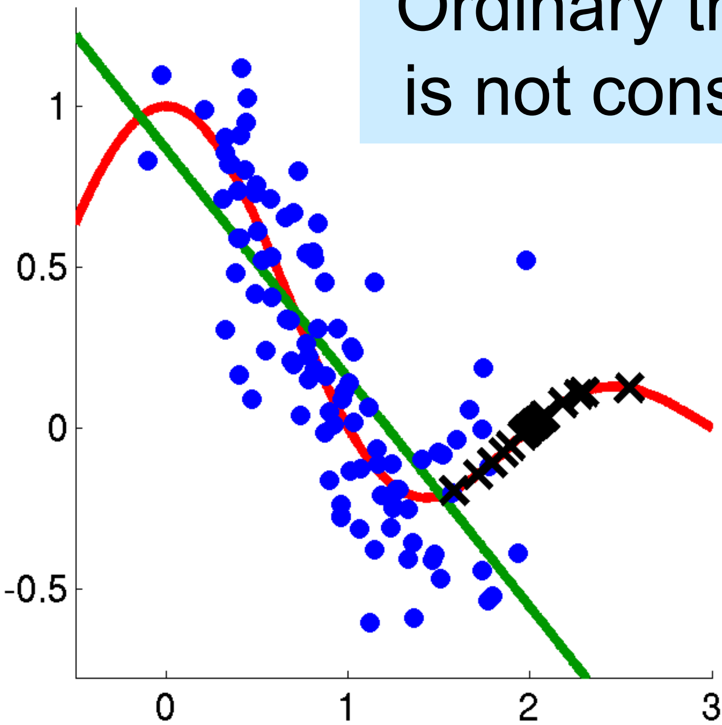
- Given:
  - Labeled training data: $\{(\boldsymbol{x}_i^{\mathrm{tr}}, y_i^{\mathrm{tr}})\}_{i=1}^{n_{\mathrm{tr}}} \overset{\mathrm{i.i.d.}}{\sim} p_{\mathrm{tr}}(\boldsymbol{x}, y)$
  - Unlabeled test data: $\{\boldsymbol{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}} \overset{\mathrm{i.i.d.}}{\sim}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

Importance-Weighted Training

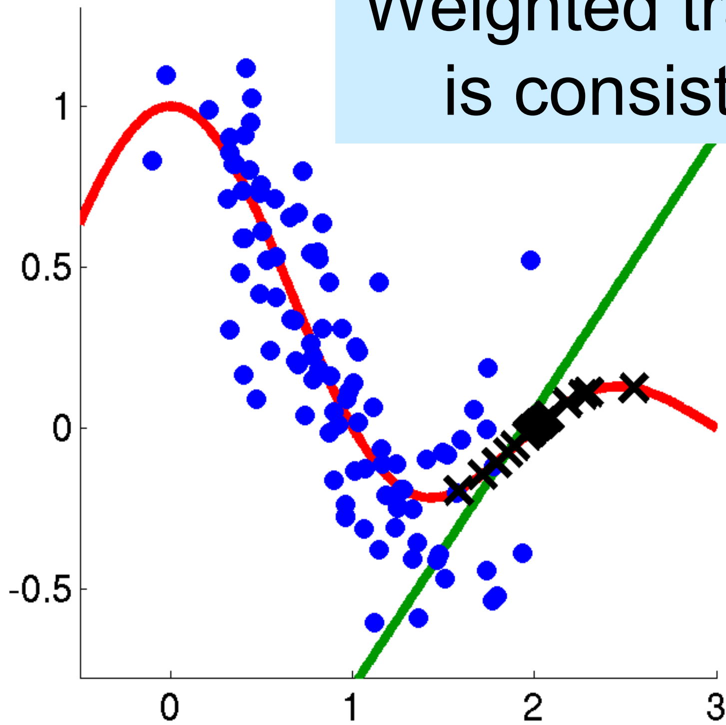
$$\underset{f}{\operatorname{argmin}}\left[\sum_{i=1}^{n_{\mathrm{tr}}}\ell(f(\boldsymbol{x}_i^{\mathrm{tr}}),y_i^{\mathrm{tr}})\right]$$

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

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



Ordinary training is not consistent



Weighted training is consistent

- How do we estimate the importance?

Direct Importance Estimation

- Given: training and test input data

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

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

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

$K(\boldsymbol{x},\cdot)$: kernel

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

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

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

- They do not estimate $p_{\mathrm{tr}}(\boldsymbol{x}), p_{\mathrm{te}}(\boldsymbol{x})$, but $\frac{p_{\mathrm{te}}(\boldsymbol{x})}{p_{\mathrm{tr}}(\boldsymbol{x})}$ directly!

Joint Importance-Predictor Estimation

Zhang+ (ACML2020, SNCS2021)

- The classical approaches are two steps:
  1. Importance weight estimation (e.g., LSIF):

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

  2. Importance-weighted predictor training:

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

- For $\ell_{\mathrm{te}} \leq 1, \ell_{\mathrm{tr}} \geq \ell_{\mathrm{te}}, r \geq 0$, the test risk $R_\ell(f) = \mathbb{E}_{p_{\mathrm{te}}(\boldsymbol{x}, y)}[\ell(f(\boldsymbol{x}), y)]$ can be bounded as

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

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

Contents

1. Weakly Supervised Learning
2. Noisy-Label Learning
3. Transfer Learning
   - A) Importance Weighting
   - B) Continuous Distribution Shift
4. Towards More Reliable Learning

Slides→

Continuous Class-Prior Shift

Bai+ (NeurIPS2022)

- Class-priors $p_t(y)$ change arbitrarily over time, but class-conditional is unchanged: $p_{\mathrm{tr}}(\boldsymbol{x}|y) = p_t(\boldsymbol{x}|y)$

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

- We use online convex optimization: Hazan (2016)
  - convex loss ℓ (e.g., logistic),
  - linear model $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x}, \quad \boldsymbol{w} \in \mathcal{W}$.
  - $p_{t-1}(y)$ is estimated by black box shift estimation. Lipton+ (ICML2018)

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

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

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

$\eta > 0$: step size

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:

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

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

Freund+ (JCSS1997)

$$\boldsymbol{w}_t = \sum_{m=1}^{M} p_t^{(m)}\boldsymbol{w}_t^{(m)}$$

$$p_t^{(m)} \propto \exp\left(-\varepsilon\sum_{s=1}^{t-1}\widehat{R}_s(\boldsymbol{w}_s^{(m)})\right)$$

$$\varepsilon = \Theta\left(\sqrt{\frac{\ln M}{T}}\right)$$

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:
  - Online learning with step size $\eta = \Theta(V_T^{\frac{1}{3}} T^{-\frac{1}{3}})$ achieves the optimal dynamic regret:

$$\mathbb{E}\left[\underbrace{\sum_{t=1}^{T} R_t(\boldsymbol{w}_t)}_{\text{Risk of our model}} - \underbrace{\sum_{t=1}^{T} \min_{\boldsymbol{w} \in \mathcal{W}} R_t(\boldsymbol{w})}_{\text{Risk of the best model at each iteration}}\right] = \mathcal{O}(V_T^{\frac{1}{3}} T^{\frac{2}{3}})$$

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

Continuous Covariate Shift

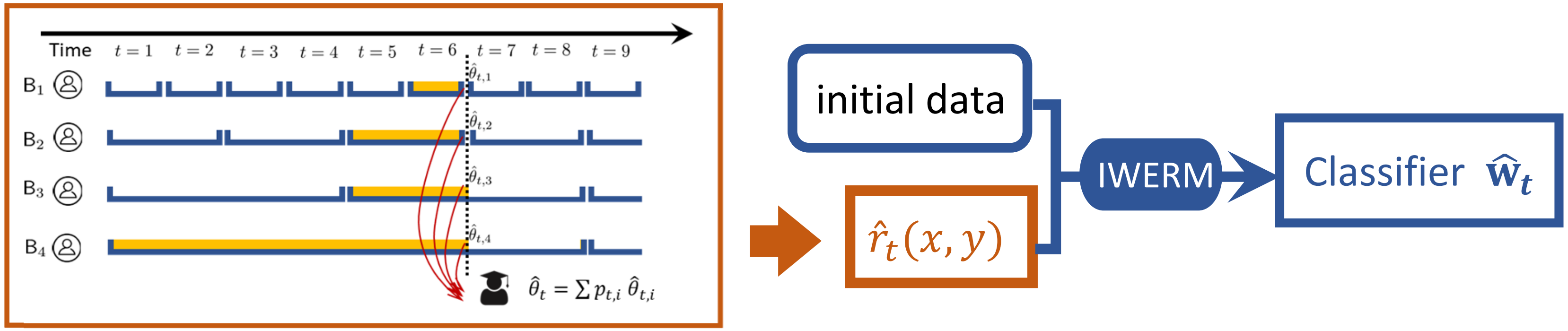
Zhang+ (arXiv2023)

- **Input density** $p_t(\boldsymbol{x})$ **change** arbitrarily over time, but **output-given-input** is unchanged: $p_{\mathrm{tr}}(y|\boldsymbol{x}) = p_t(y|\boldsymbol{x})$

$t = 1, \ldots, T$

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

- We use **online density ratio estimation**:



Stay tuned!

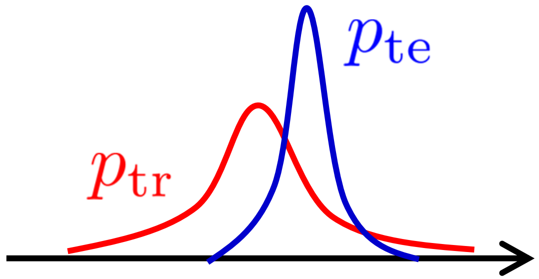
Contents

1. Weakly Supervised Learning
2. Noisy-Label Learning
3. Transfer Learning
4. Towards More Reliable Learning

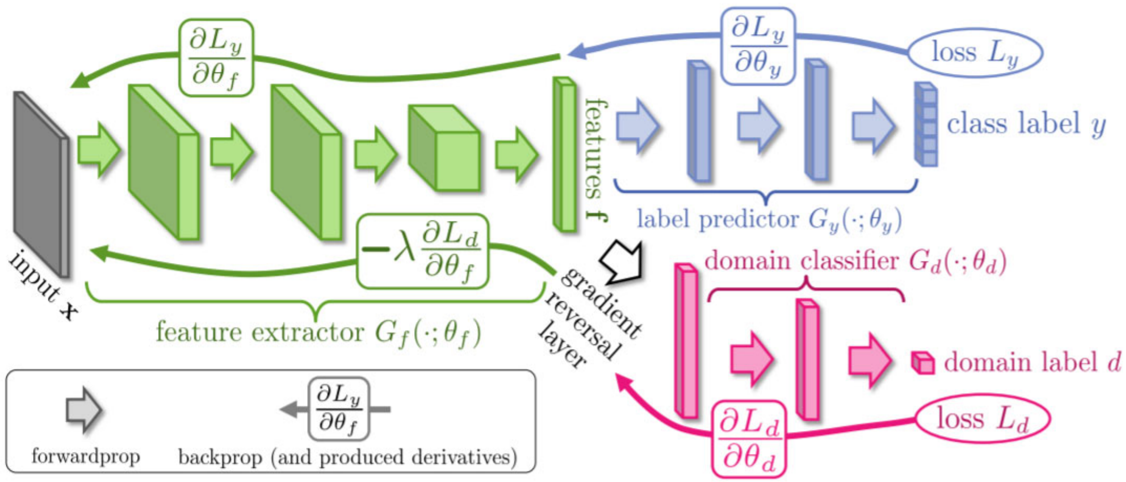
Slides→

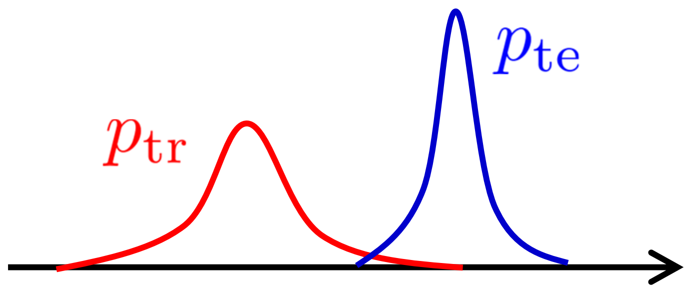
Beyond Importance Weighting?

- Limitation of importance weighting:
  - The training domain must cover the test domain. $\dfrac{p_{\mathrm{te}}(\boldsymbol{x}, y)}{p_{\mathrm{tr}}(\boldsymbol{x}, y)} < \infty$



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





Ben-David+ (NeurIPS2006), Ganin+ (ICML2015)

- However, considering covariate shift is still essential.

Joint Shift

- Many distribution shift works focus on a particular shift type (e.g., covariate shift):

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

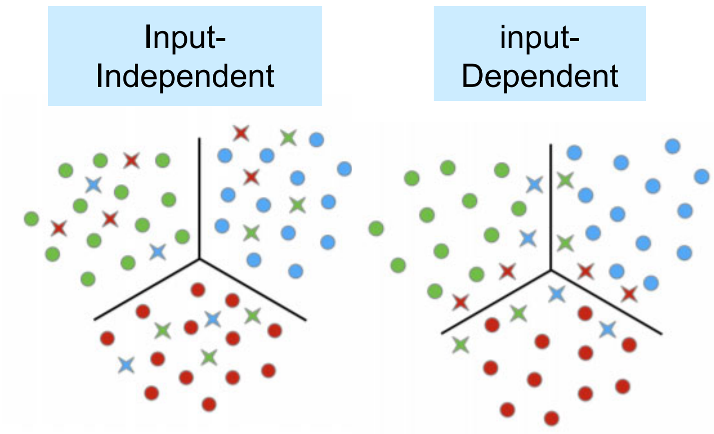
  - However, identification of the shift type is challenging.

- 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.
  - But input-dependent noise is hard.



- Let's consider joint shift:

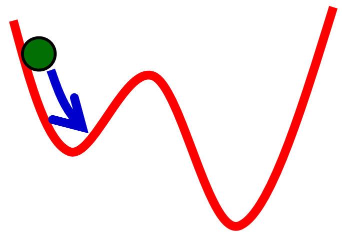
$$p_{\mathrm{tr}}(\boldsymbol{x}, y) \neq p_{\mathrm{te}}(\boldsymbol{x}, y)$$

Mini-Batch-Wise Loss Matching

Fang+ (NeurIPS2020)

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

- 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:

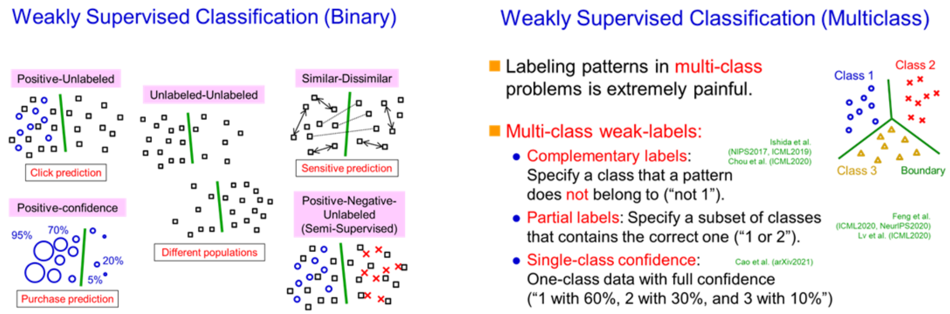
Huang+ (NeurIPS2006)

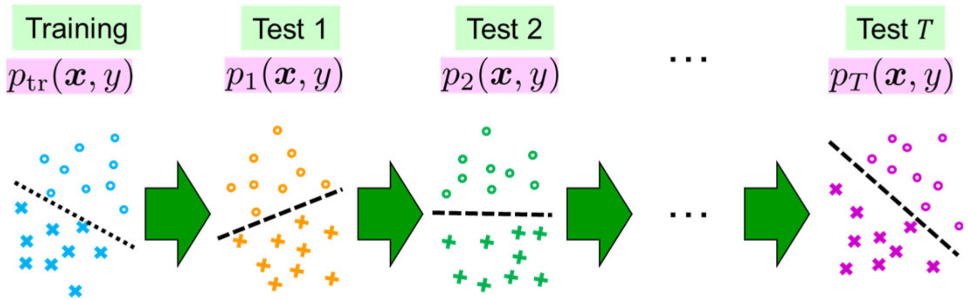
$$\frac{1}{\tilde{n}_{\mathrm{tr}}} \sum_{i=1}^{\tilde{n}_{\mathrm{tr}}} r_i \ell(f(\tilde{\boldsymbol{x}}_i^{\mathrm{tr}}), \tilde{y}_i^{\mathrm{tr}}) \approx \frac{1}{\tilde{n}_{\mathrm{te}}} \sum_{j=1}^{\tilde{n}_{\mathrm{te}}} \ell(f(\tilde{\boldsymbol{x}}_j^{\mathrm{te}}), \tilde{y}_j^{\mathrm{te}})$$

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

Future Challenges

- For joint shift, requiring labeled test data is too strong.
  - Can we perform joint shift adaptation from weak supervision?
  - Can we extend it to continuous joint shift?
  - Can we extend it to a limited-memory setting?





- In real-world application, updating the system online is dangerous because new data can be malicious:
  - Updating the system periodically (daily, etc.) is practical.
  - But we want the system to reflect the latest data.
  - Can we systematically use a buffer for temporary update?

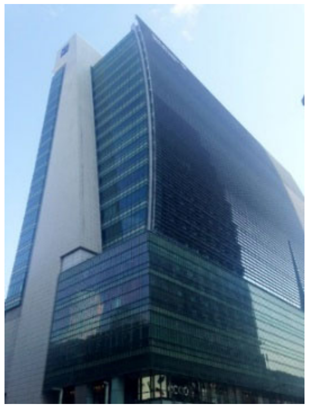
Grateful to Collaborators!

RIKEN AIP

Team leader Masashi Sugiyama	Research scientist Gang Niu
Postdoctoral researcher Jingfeng Zhang	Postdoctoral researcher Jiaqi Lyu
Postdoctoral researcher Shuo Chen	Senior visiting scientist Shinichi Nakajima
Visiting scientist Futoshi Futami	Visiting scientist Florian Yger
Visiting scientist Takashi Ishida	Visiting scientist Miao Xu
Visiting scientist Takayuki Osa	Visiting scientist Bo Han
Visiting scientist Takahiro Mimori	Visiting scientist Feng Liu
Visiting scientist Lei Feng	Visiting scientist Tongliang Liu
Part-time worker I Masahiro Fujisawa	Part-time worker I Yifan Zhang

and many great interns!

- Professor
  - Masashi Sugiyama (Complexity, Computer, Information, RIKEN)
- Associate Professor
  - Naoto Yokoya (Complexity, Computer, Information, RIKEN)
- Lecturer
  - Takashi Ishida (Complexity, Computer, Information)
- Project Lecturer
  - Nobutaka Ito (Complexity)
- Professor (to Sato Lab from April 202
  - Issei Sato (Computer, Informati
- Project Assistant Professor
  - Chao-Kai Chiang (Complexity)
- Project Researcher (Postdoctoral Rese
  - Dongxian Wu (Complexity)
- Project Specialist
  - Yuko Kawashima (Complexity)
  - Soma Yokoi (Complexity)
  - Fumi Sato (Complexity)



- Doctoral Student
  - Shinji Nakadai (Computer)
  - Ryuichi Kiryo (Computer)
  - Jongyeong Lee (Computer)
  - Tianyi Zhang (Complexity)
  - Yivan Zhang (Computer)
  - Riou Charles (Computer)
  - Valliappa Chockalingam (Comput
  - Tongtong Fang (Complexity)
  - Boyo Chen (Complexity)
  - Xiaoyu Dong (Complexity)
  - Yujie Zhang (Complexity)
  - Xinqiang Cai (Complexity)
  - Jian Song (Complexity)
  - Wanshui Gan (Complexity)
  - Yuting Tang (Complexity)
  - Shintaro Nakamura (Complexity)
  - Or Raveh (Complexity)
  - Johannes Ackermann (Computer
  - Wei Wang (Complexity)
  - Hongruixuan Chen (Complexity)
  - Huanjian Zhou (Complexity)
  - Zhiyuan Zhan (Complexity)
  - Zhihao Liu (Complexity)

- Master Student
  - Hyunggyu Park (Complexity)* Sato lab.
  - Jiahuan Li (Computer)
  - Kun Yang (Complexity)
  - Xiaomou Hou (Complexity)
  - Anan Methasate (Computer)
  - Cemal Erat (Computer)
  - Kento Yamamoto (Computer)
  - Kazuki Ota (Computer)
  - Iu Yahiro (Computer)
  - Hikaru Fujita (Computer)
  - Yu Yao (Complexity)
  - Yoshifumi Nakano (Complexity)
  - Soichiro Nishimori (Complexity)
  - Ryota Ushio (Complexity)
  - Tiankui Xian (Complexity)
  - Thanawat Lodkaew (Computer)
  - Masahiro Negishi (Computer)
  - Yuto Nozaki (Computer)
  - Kanta Shimizu (Computer)
  - Zhongle Zhu (Computer)
  - Fang Liu (Computer)
  - Ming Li (Complexity)
  - Luheng Wang (Complexity)
  - Liuzhuozheng Li (Complexity)

東京大学 THE UNIVERSITY OF TOKYO

- Research Student
  - Meike Tütken (Computer)
  - Serhii Khomenko (Information Science)
  - Artem Lubkivskyi (Information Science)