Recent Advances on Positive-Unlabeled (PU) Learning

Gang Niu

Project Assistant Professor, UTokyo
Visiting Scientist, RIKEN AIP

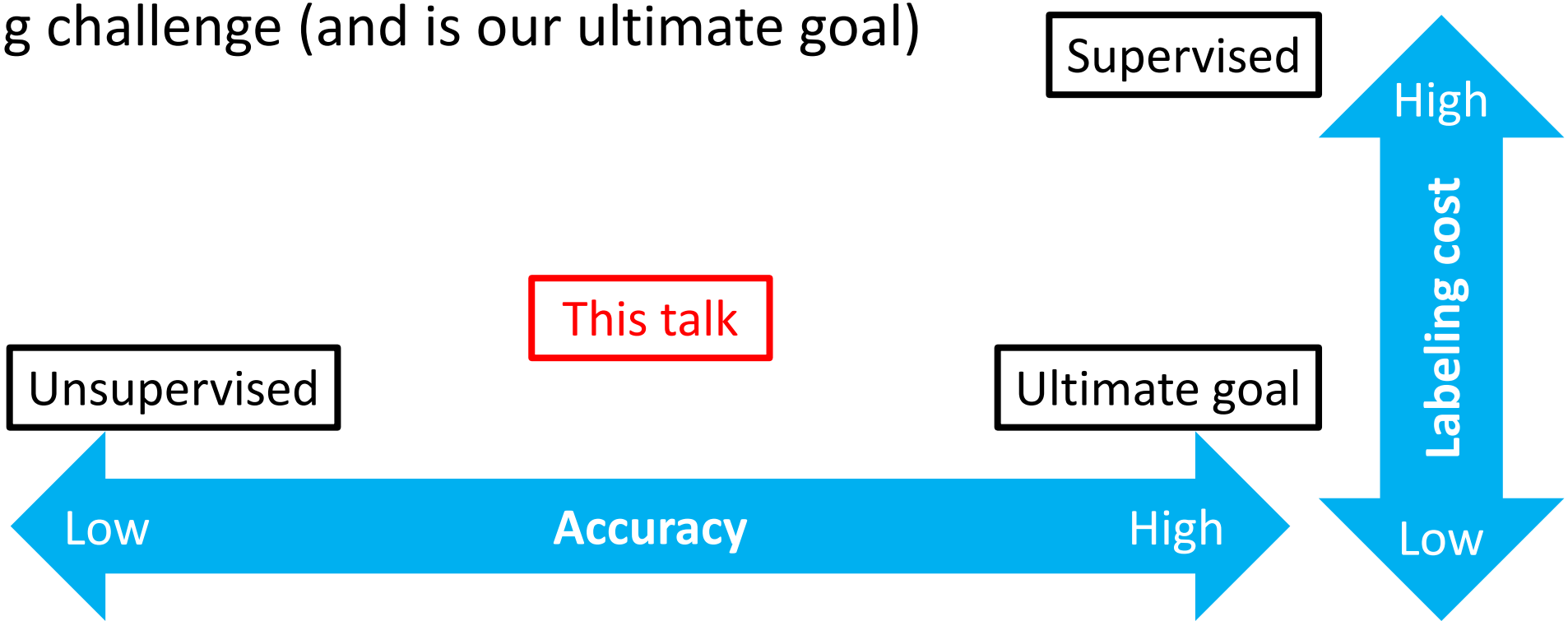
Collaborated with Masashi Sugiyama,
Marthinus C. du Plessis, Tomoya Sakai, Ryuichi Kiryo & Yao Ma

東京大学
THE UNIVERSITY OF TOKYO

RIKEN

What is this talk about?

- Fully supervised deep learning from big data is successful
- Nevertheless, massive labeled data is not always available
  - Medicine, manufacturing, disaster, infrastructure …
- Achieving high **accuracy** with low **labeling costs** is always a big challenge (and is our ultimate goal)



Overview

- Focus on binary classification – the most studied learning problem
- In PU learning, a binary classifier is trained from only P & U data
- Unbiased PU learning
  - State-of-the-art approach to training linear-in-parameter models
  - **Unbiased empirical risk estimators** are minimized
- Non-negative PU learning
  - State-of-the-art approach to training deep models
  - **Non-negative empirical risk estimators** are minimized
- PNU learning
  - **Convex combinations** of the objectives of PU & PN learning

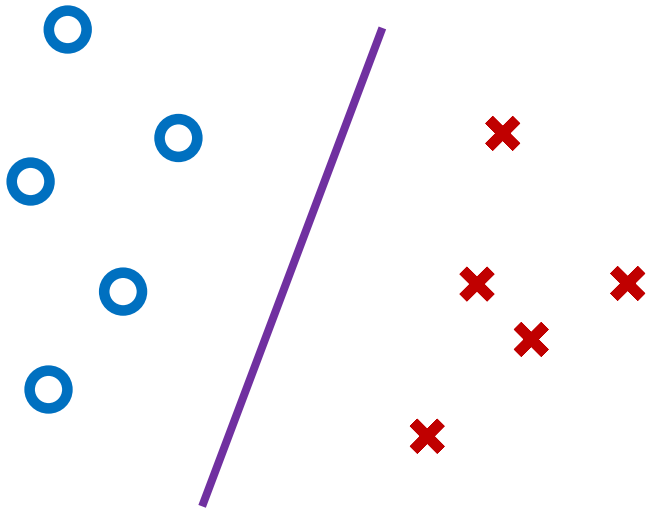
Outline

- Introduction
- Unbiased PU Learning (du Plessis, Niu & Sugiyama, *NIPS'14 & ICML'15*)
- Non-negative PU Learning (Kiryo, Niu, du Plessis & Sugiyama, *NIPS'17*)

  (for oral presentation; there are 40 orals/678 acceptance/3240 submissions)
- PNU Learning (Sakai, du Plessis, Niu & Sugiyama, *ICML'17*)
- Theoretical Analyses (Niu, du Plessis, Sakai, Ma & Sugiyama, *NIPS'16*)

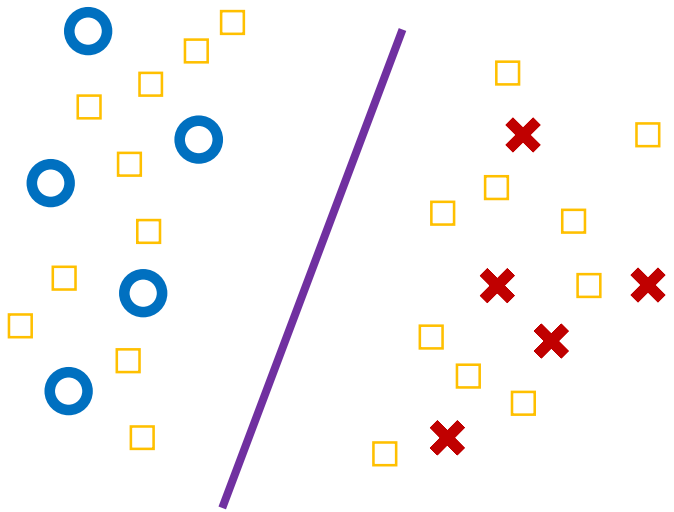
Problem settings in a nutshell

PN learning
(i.e., supervised learning)



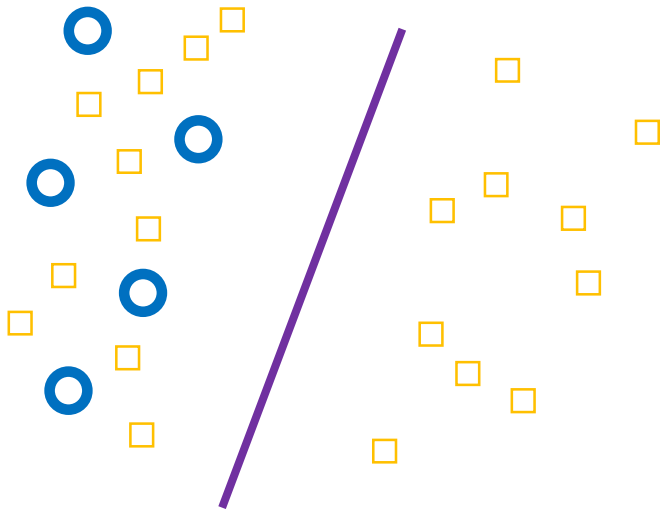
P & N data are
available for training

PNU learning
(i.e., semi-supervised learning)



P, N & U data are
available for training

PU learning



P & U data are
available for training

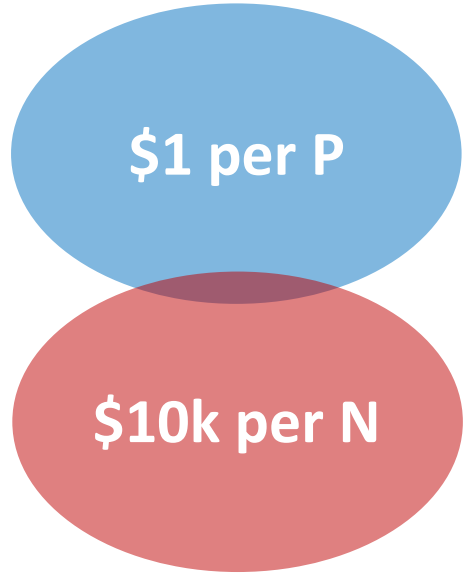
: positive data : negative data : unlabeled data

Motivation of PU learning

- PN learning (whether deep or not) is data demanding
- Then, is PNU learning the most natural choice?
  - Certainly, if the two classes are **symmetric**
    ➔ Which is P & which is N does not matter
  - Not really, if there is **intrinsic difference** in them
    ➔ Which is P & which is N matters
- PU learning is preferred due to the following reasons
  I. N data are too expensive
  II. N data are too diverse
  III. "N data" are impure

Case I: N data are too expensive

- Some data-collecting activity is prohibited by law
  - Suppose I am a market researcher at **Apple**
  - Let **Samsung** be the imaginary enemy
  - Plan A: Hack the data center of Samsung
  - Plan B: Send corporate spies to Samsung



- Some activity is costly & risky
  - Some clinical trials involve **healthy subjects** with no pre-existing medical conditions ➔ Extreme financial incentives
  - While others pertain to **patients** with specific health conditions who are willing to try an experimental treatment (https://en.wikipedia.org/wiki/Clinical_trial)

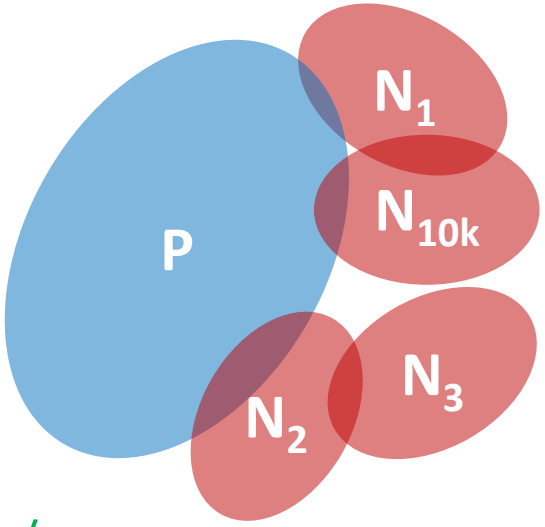
Case II: N data are too diverse

- In the first running example
  - **Android** is neither the only competitor
  - Nor is Samsung the only Android vendor

Period	Samsung	Apple	Huawei	OPPO	vivo	Others
2016Q1	23.8%	15.4%	8.4%	5.9%	4.4%	42.1%
2016Q2	22.7%	11.7%	9.3%	6.6%	4.8%	45.0%
2016Q3	20.9%	12.5%	9.3%	7.1%	5.9%	44.3%
2016Q4	18.0%	18.2%	10.5%	7.3%	5.7%	40.2%
2017Q1	23.3%	14.7%	10.0%	7.5%	5.5%	39.0%

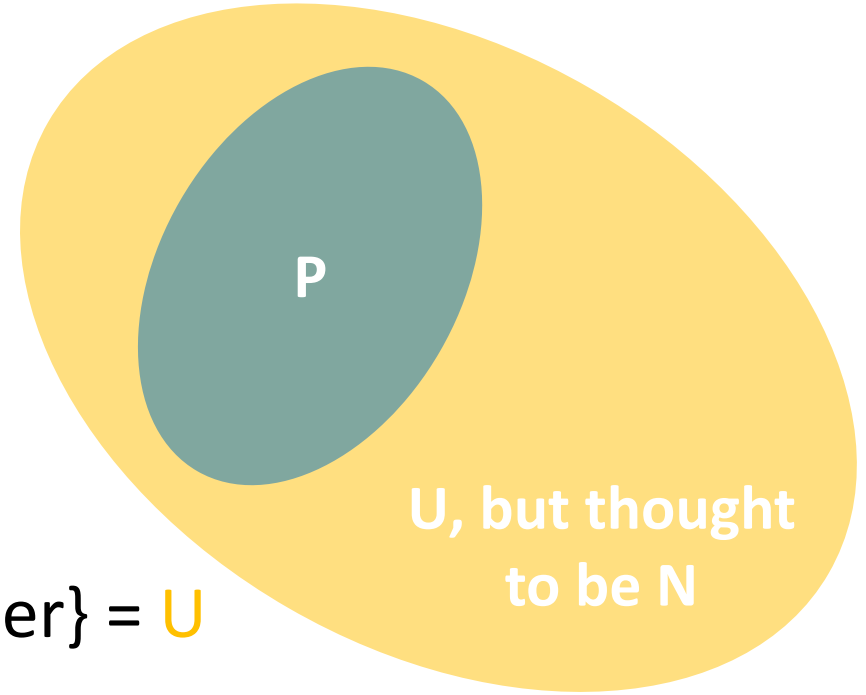
Source: IDC, May 2017

(http://www.idc.com/promo/smartphone-market-share/vendor)



- In the second running example
  - Too difficult to sample healthy subjects without **selection bias**

Case III: “N data” are impure



- Wait, and rethink the definition of N
- For the sake of smartphone advertising
  - A customer is N = S/he hates **iPhone** and would never buy it in the whole life
  - N = {non-potential user} ∈ {not existing user} = U
- For the purpose of drug testing
  - Many diseases are due to **chromosomal abnormality**
  - P = {specific **genetic disorder**} ∋ {specific **medical condition**}
  - N = {not possess this disorder} ∈ {not observe this condition} = U

Previous work in PU learning

- Binary classification (applied to retrieval & novelty/outlier detection)
  - Statistical query model Denis (*ALT'98*); De Comité+ (*ALT'99*); Letouzey+ (*ALT'00*)
  - Linear(-in-parameter) model
    Liu+ (*ICML'02*); Li & Liu (*IJCAI'03*); Lee & Liu (*ICML'03*); Liu+ (*ICDM'03*);
    Elkan & Noto (*KDD'08*); ← The first that might be unbiased
    Ward+ (*Biometrics 2009*); Scott & Blanchard (*AISTATS'09*); Blanchard+ (*JMLR 2010*);
    du Plessis+ (*NIPS'14*) ← The first that must be unbiased
- Other applications
  - Matrix completion Hsieh+ (*ICML'15*)
  - Sequential data Li+ (*SDM'09*); Nguyen+ (*IJCAI'11*)

Statistical learning theory

Outline

- Introduction
- Unbiased PU Learning
- Non-negative PU Learning
- PNU Learning
- Theoretical Analyses

Empirical risk minimization (Vapnik, *Statistical Learning Theory*, 1998)

- A "cookbook" procedure of ERM
  1. Choose a loss, so that the (expected) **learning objective** (which is known as the **risk**) can be defined
  2. Choose a **model**, so that the risk can be minimized over this model family (rather than all measurable functions)
  3. Approximate the risk by an **empirical risk estimator**
  4. Minimize the empirical risk by an **optimization algorithm**
- Nice to have independent learning obj., model & opt. alg. Arguably a key reason for the great success of deep learning
- 3. – Straightforward for PN; non-trivial for PU

Notation

- $X \in \mathbb{R}^d, Y \in \{\pm 1\}$: Input & output RVs
- $p(x, y)$: Underlying joint density
- $p_{\mathrm{p}}(x) = p(x|Y = +1)$: P marginal $p(x)$: U marginal
  $p_{\mathrm{n}}(x) = p(x|Y = -1)$: N marginal
- $\pi_{\mathrm{p}} = p(Y = +1)$: Class-prior probability – Assumed known;
  can be estimated Ramaswamy+ (*ICML'16*); Jain+ (*NIPS'16*); du Plessis+ (*MLJ 2017*)
- $\mathbb{E}_{\mathrm{p}}[\cdot] = \mathbb{E}_{X \sim p_{\mathrm{p}}}[\cdot], \mathbb{E}_{\mathrm{n}}[\cdot] = \mathbb{E}_{X \sim p_{\mathrm{n}}}[\cdot]$: Expectation over P/N marginal
- $\mathcal{X}_{\mathrm{p}} = \{x_i^{\mathrm{p}}\}_{i=1}^{n_{\mathrm{p}}} \overset{\text{i.i.d.}}{\sim} p_{\mathrm{p}}(x)$: P data $\mathcal{X}_{\mathrm{u}} = \{x_i^{\mathrm{u}}\}_{i=1}^{n_{\mathrm{u}}} \overset{\text{i.i.d.}}{\sim} p(x)$: U data
  $\mathcal{X}_{\mathrm{n}} = \{x_i^{\mathrm{n}}\}_{i=1}^{n_{\mathrm{n}}} \overset{\text{i.i.d.}}{\sim} p_{\mathrm{n}}(x)$: N data

Empirical risk estimator in PN learning

- Let g be a **decision function** & ℓ be a **loss function**
- The **risk** of g is

$$R(g) = \mathbb{E}_{(X,Y)\sim p(x,y)}\left[\ell\left(Yg(X)\right)\right]$$
$$= \pi_{\mathrm{p}}\mathbb{E}_{\mathrm{p}}\left[\ell\left(g(X)\right)\right] + \pi_{\mathrm{n}}\mathbb{E}_{\mathrm{n}}\left[\ell\left(-g(X)\right)\right]$$

  where $\pi_{\mathrm{n}} = 1 - \pi_{\mathrm{p}}$

- The risk can be approximated directly by

$$\hat{R}_{\mathrm{pn}}(g) = \frac{\pi_{\mathrm{p}}}{n_{\mathrm{p}}}\sum_{x\in\mathcal{X}_{\mathrm{p}}}\ell\left(g(x)\right) + \frac{\pi_{\mathrm{n}}}{n_{\mathrm{n}}}\sum_{x\in\mathcal{X}_{\mathrm{n}}}\ell\left(-g(x)\right)$$

- This doesn't work for PU learning!

Empirical risk estimator in PU learning (du Plessis+, *ICML'15*)

- Key observations
  - $\pi_{\mathrm{n}} p_{\mathrm{n}}(x) = p(x) - \pi_{\mathrm{p}} p_{\mathrm{p}}(x)$
  - $\pi_{\mathrm{n}} \mathbb{E}_{\mathrm{n}}\left[\ell\left(-g(X)\right)\right] = \mathbb{E}_X\left[\ell\left(-g(X)\right)\right] - \pi_{\mathrm{p}} \mathbb{E}_{\mathrm{p}}\left[\ell\left(-g(X)\right)\right]$
- Thus the risk can be expressed as

$$R(g) = \pi_{\mathrm{p}} \mathbb{E}_{\mathrm{p}}\left[\ell\left(g(X)\right) - \ell\left(-g(X)\right)\right] + \mathbb{E}_X\left[\ell\left(-g(X)\right)\right]$$

- This can be approximated indirectly by

$$\hat{R}_{\mathrm{pu}}(g) = \frac{\pi_{\mathrm{p}}}{n_{\mathrm{p}}} \sum_{x \in \mathcal{X}_{\mathrm{p}}} \left[\ell\left(g(x)\right) - \ell\left(-g(x)\right)\right] + \frac{1}{n_{\mathrm{u}}} \sum_{x \in \mathcal{X}_{\mathrm{u}}} \ell\left(-g(x)\right)$$

- Simple in retrospect!

Non-convex special case (du Plessis+, *NIPS'14*)

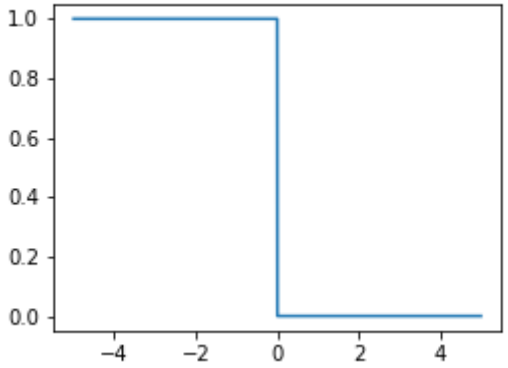
- If $\ell(t) + \ell(-t) = 1$

$$R(g) = 2\pi_{\mathrm{p}}\mathbb{E}_{\mathrm{p}}\left[\ell\left(g(X)\right)\right] + \mathbb{E}_X\left[\ell\left(-g(X)\right)\right] - \pi_{\mathrm{p}}$$

  - Non-convex in g

- Examples

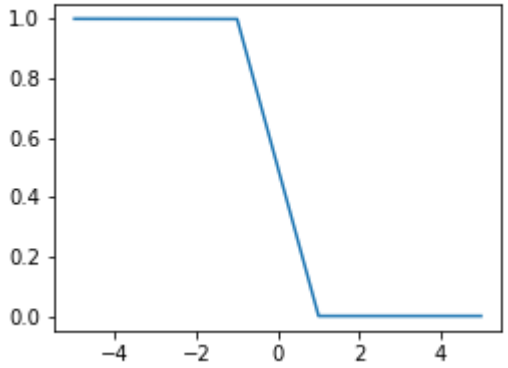
Zero-one loss



$(1 - \mathrm{sign}(t))/2$

Fit evaluation & validation

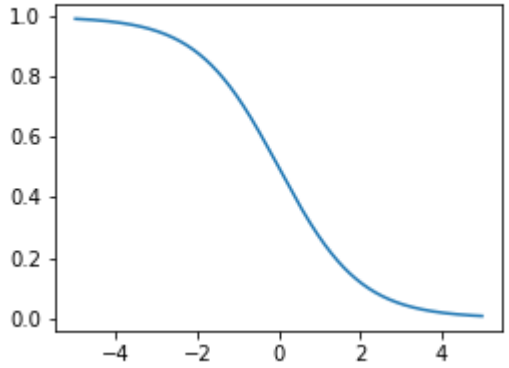
(scaled) **Ramp** loss



$\max\{0, \min\{1, (1-t)/2\}\}$

Fit CCCP solver

Sigmoid loss



$1/(1 + \exp(t))$

Fit SGD solver

Convex special case (du Plessis+, *ICML'15*)

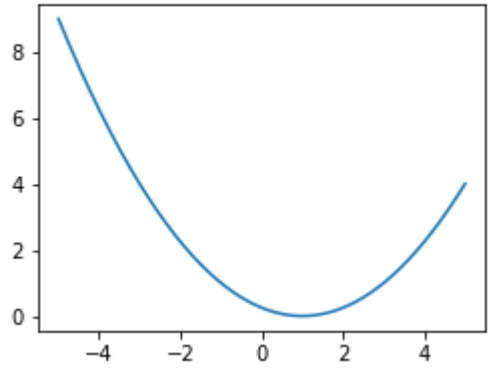
- If $\ell(t) - \ell(-t) = -t$ (Natarajan+, *NIPS'13*; Patrini+, *ICML'16*)

$$R(g) = \pi_{\mathrm{p}} \mathbb{E}_{\mathrm{p}}[-g(X)] + \mathbb{E}_{X}[\ell(-g(X))]$$

  - Convex in g, and convex in θ if $g(x;\theta)$ is linear in θ

- Examples

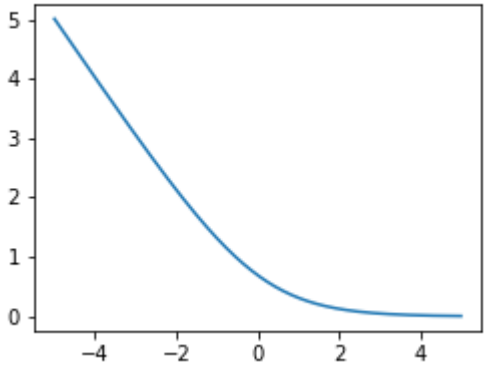
(scaled) **Squared** loss



$(t-1)^2/4$

Analytic solution

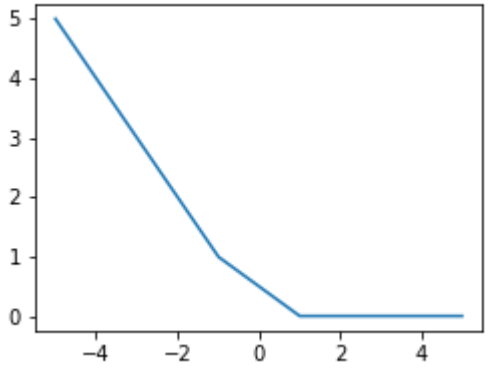
Logistic loss



$\ln(1+\exp(-t))$

Fit SGD solver

Double hinge loss



$\max\{0, (1-t)/2, -t\}$

Fit QP solver

Remarks

- $\hat{R}_{\mathrm{pu}}(g)$ is unbiased & consistent, similarly to $\hat{R}_{\mathrm{pn}}(g)$
  - Why is unbiasedness important? Easy to be consistent
- **Biased SVM** (Liu+, *ICDM'03*) – best method prior to Elkan & Noto (*KDD'08*)

$$\frac{1}{2}\|w\|^2 + C_{\mathrm{p}} \sum_{x \in \mathcal{X}_{\mathrm{p}}} \ell_{\mathrm{H}}(w \cdot x + b) + C_{\mathrm{u}} \sum_{x \in \mathcal{X}_{\mathrm{u}}} \ell_{\mathrm{H}}(-w \cdot x - b)$$

  - Guess/search C_{p} & C_{u}
  - No learning guarantee
- **SMO** solver speeding up double hinge loss (Sansone+, arXiv 2016)

Note that theoretical & experimental results omitted for simplicity

Outline

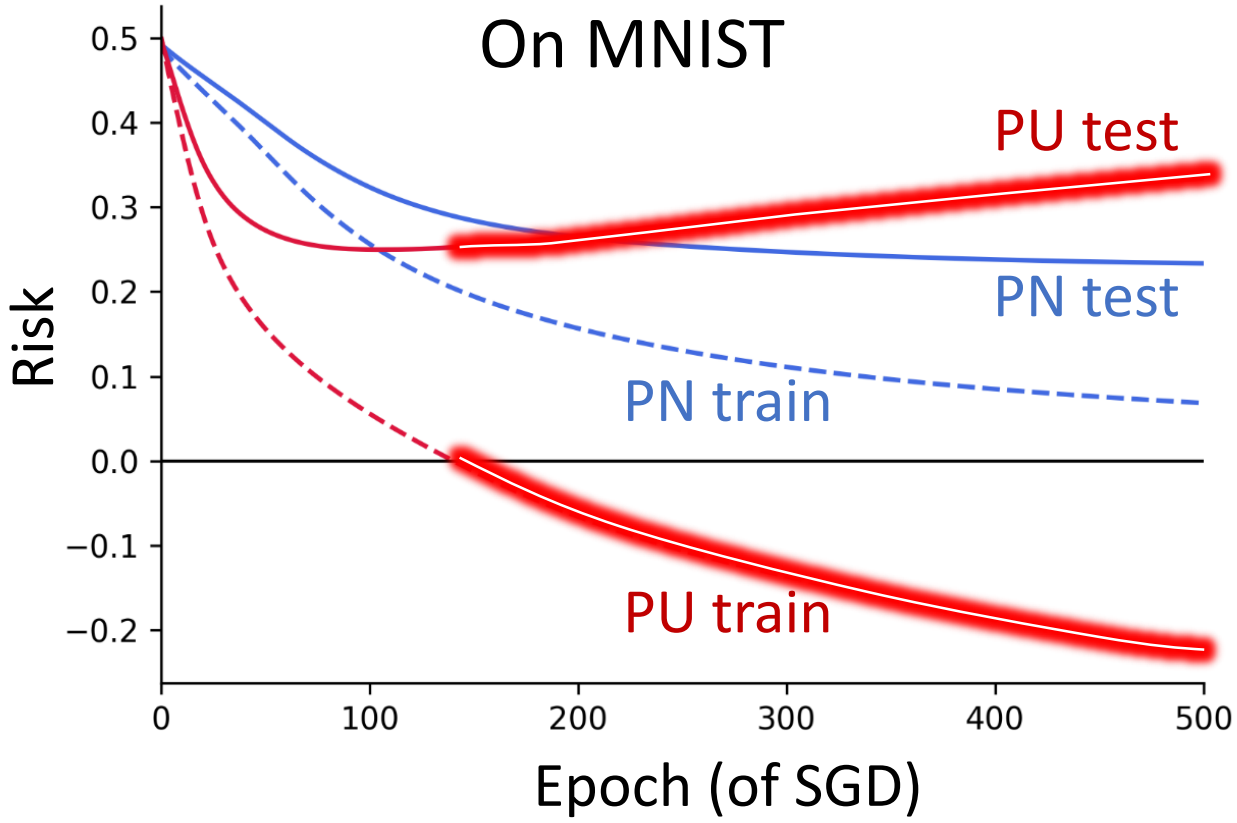
- Introduction
- Unbiased PU Learning
- Non-negative PU Learning
- PNU Learning
- Theoretical Analyses

Revisit ERM

2. Choose a model $\mathcal{G}$
   - **Approximation error** $\inf_{g \in \mathcal{G}} R(g) - \inf_g R(g)$
     Smaller for more flexible model
3. Approximate $R(g)$ by $\hat{R}(g)$
   - **Estimation error** $R(\hat{g}) - \inf_{g \in \mathcal{G}} R(g)$ ← $\hat{g} = \arg\inf_{g \in \mathcal{G}} \hat{R}(g)$
     Smaller for less flexible model, or bigger training data
     Converge to zero, if learning is consistent
4. Minimize $\hat{R}(g)$ by an optimization algorithm which returns $\hat{g}'$
   - **Optimization error** $\hat{R}(\hat{g}') - \hat{R}(\hat{g})$

Motivation

- $\hat{R}_{\mathrm{pu}}(g)$ is nice for training linear-in-parameter models
- However, it cannot be used for training deep networks



Overfitting

Emp. risk < 0

- P = {even digits}
  N = {odd digits}
- $n_{\mathrm{p}} = 100$
  $n_{\mathrm{n}} = 50$
  $n_{\mathrm{u}} = 59{,}900$
- P too limited
  U cannot help

Non-negative risk estimator (Kiryo+, *NIPS'17*)

- Rescue with neither changing model nor labeling more data
- Recall $\pi_{\mathrm{n}}\mathbb{E}_{\mathrm{n}}\left[\ell\left(-g(X)\right)\right] = \mathbb{E}_X\left[\ell\left(-g(X)\right)\right] - \pi_{\mathrm{p}}\mathbb{E}_{\mathrm{p}}\left[\ell\left(-g(X)\right)\right]$
  - Approximate left-hand-side ➔ $\hat{R}_{\mathrm{pn}}(g) \geq 0$
  - Approximate right-hand-side ➔ $\hat{R}_{\mathrm{pu}}(g) \not\geq 0$
- Force it to be non-negative!

$$\tilde{R}_{\mathrm{pu}}(g) = \frac{\pi_{\mathrm{p}}}{n_{\mathrm{p}}}\sum_{x\in\mathcal{X}_{\mathrm{p}}}\left[\ell\left(g(x)\right)\right] + \max\left\{0, \frac{1}{n_{\mathrm{u}}}\sum_{x\in\mathcal{X}_{\mathrm{u}}}\ell\left(-g(x)\right) - \frac{\pi_{\mathrm{p}}}{n_{\mathrm{p}}}\sum_{x\in\mathcal{X}_{\mathrm{p}}}\left[\ell\left(-g(x)\right)\right]\right\}$$

- Minimizing $\tilde{R}_{\mathrm{pu}}(g)$ is no longer **embarrassingly parallel**

Large-scale learning algorithm (Kiryo+, *NIPS'17*)

- Let $(\mathcal{X}_{\mathrm{p}}^1, \mathcal{X}_{\mathrm{u}}^1), \ldots, (\mathcal{X}_{\mathrm{p}}^N, \mathcal{X}_{\mathrm{u}}^N)$ be mini-batches

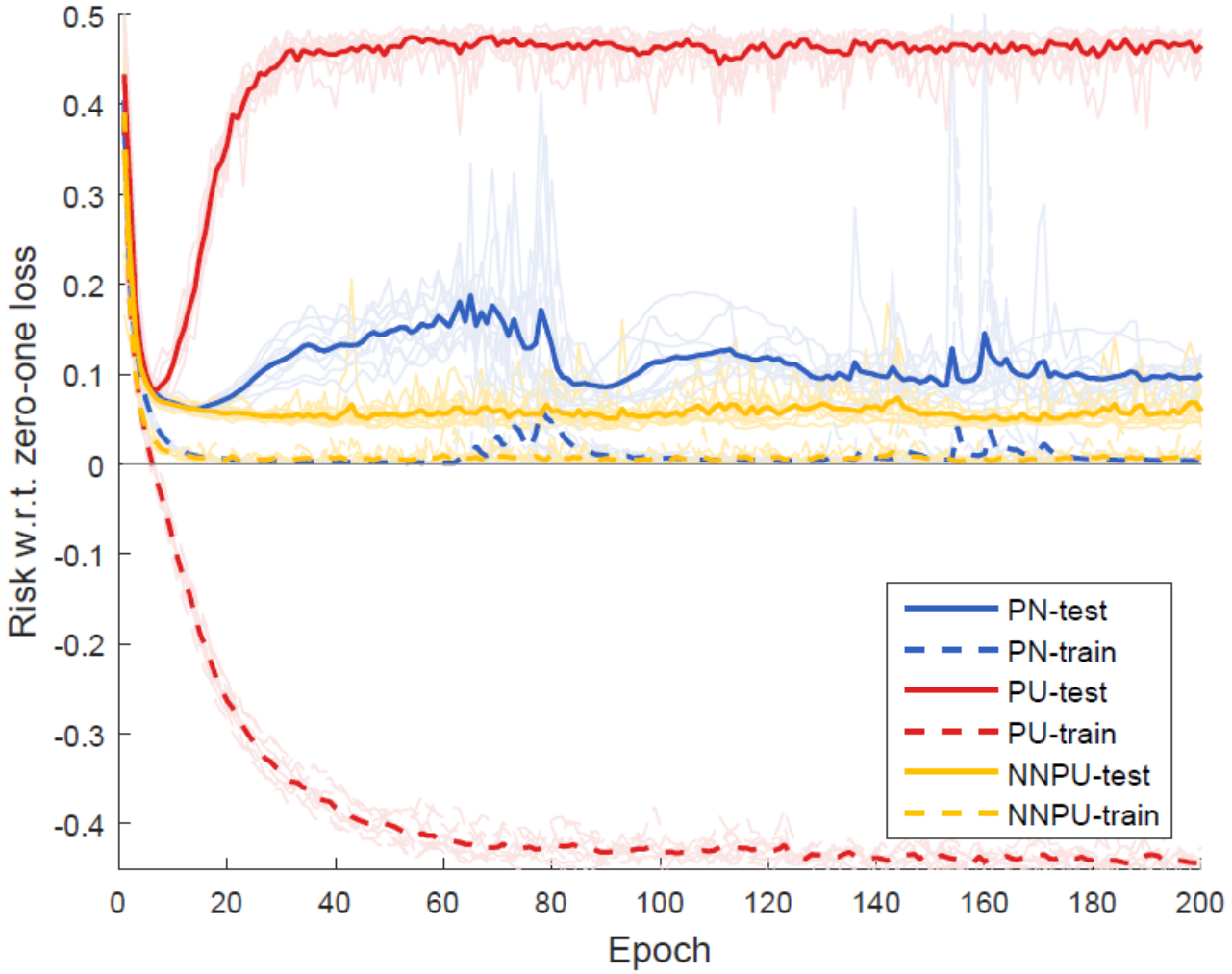
$$\max\left\{0, \frac{1}{n_{\mathrm{u}}}\sum_{x\in\mathcal{X}_{\mathrm{u}}}\ell(-g(x)) - \frac{\pi_{\mathrm{p}}}{n_{\mathrm{p}}}\sum_{x\in\mathcal{X}_{\mathrm{p}}}\left[\ell(-g(x))\right]\right\}$$

$$= \max\left\{0, \sum_{i=1}^{N}\frac{1}{n_{\mathrm{u}}}\sum_{x\in\mathcal{X}_{\mathrm{u}}^i}\ell(-g(x)) - \sum_{i=1}^{N}\frac{\pi_{\mathrm{p}}}{n_{\mathrm{p}}}\sum_{x\in\mathcal{X}_{\mathrm{p}}^i}\left[\ell(-g(x))\right]\right\}$$

$$\leq \frac{1}{N}\sum_{i=1}^{N}\max\left\{0, \frac{1}{n_{\mathrm{u}}/N}\sum_{x\in\mathcal{X}_{\mathrm{u}}^i}\ell(-g(x)) - \frac{\pi_{\mathrm{p}}}{n_{\mathrm{p}}/N}\sum_{x\in\mathcal{X}_{\mathrm{p}}^i}\left[\ell(-g(x))\right]\right\}$$

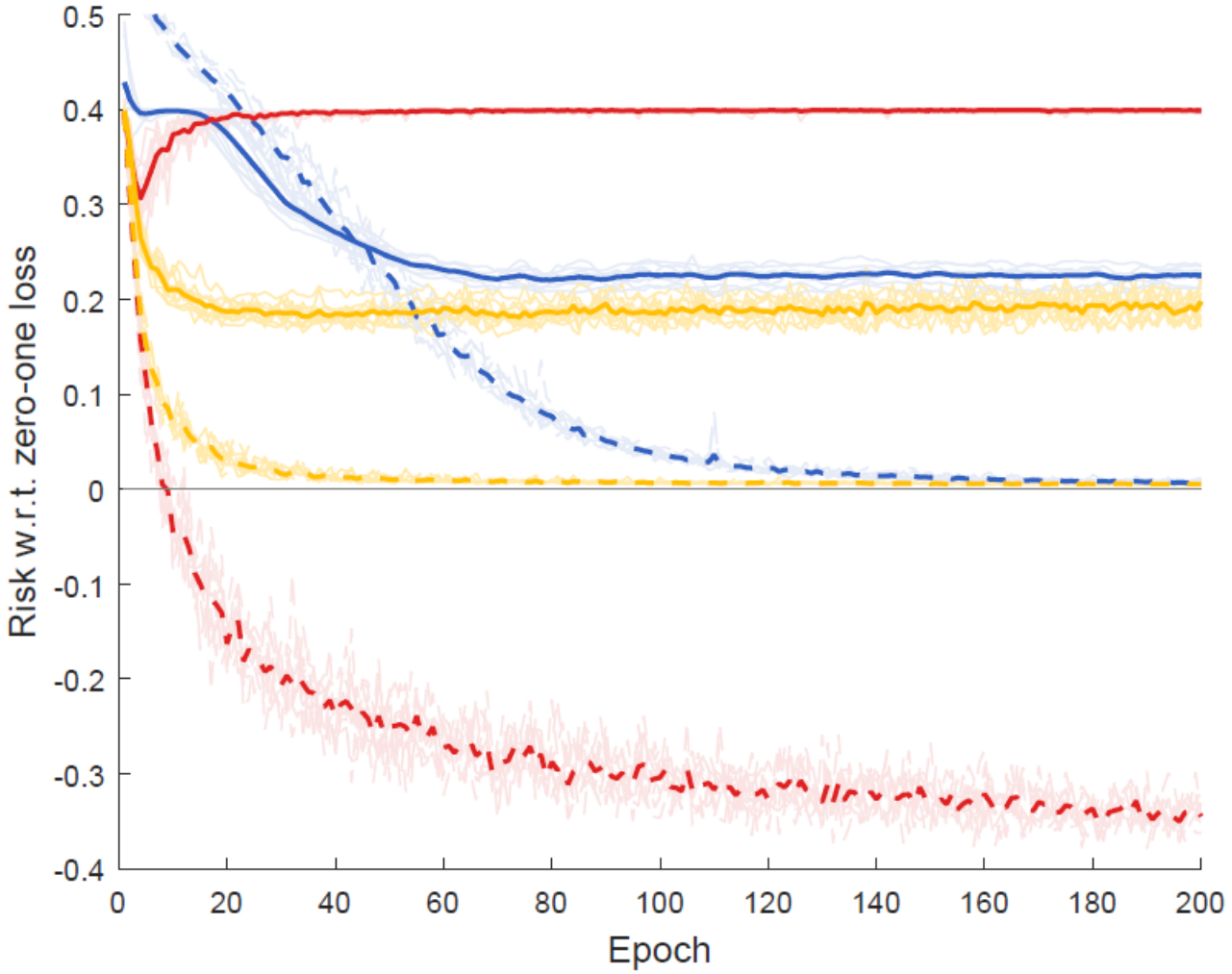
- Safe to minimize $\tilde{R}_{\mathrm{pu}}(g)$ averaged over mini-batches
- You can choose whatever stochastic opt. alg. you like
  - **Adam** (Kingma+, *ICLR'15*) or **AdaGrad** (Duchi+, *JMLR 2011*)

Experiments on MNIST



- P = {even digits, i.e., 0, 2, 4, 6 & 8}
  N = {odd digits, i.e., 1, 3, 5, 7 & 9}
- $\pi_{\mathrm{p}} = 0.49$
  $n_{\mathrm{p}} = 1{,}000$
  $n_{\mathrm{n}} = \left(\pi_{\mathrm{n}}/2\pi_{\mathrm{p}}\right)^2 n_{\mathrm{p}}$
  $n_{\mathrm{u}} = 60{,}000$
- Model: 6-layer MLP with **ReLU** (Nair & Hinton, *ICML'10*) & **BN** (Ioffe & Szegedy, *ICML'15*)

Experiments on CIFAR10



- P = {airplane, automobile, ship & truck}
  N = {bird, cat, deer, dog, frog & horse}
- $\pi_{\mathrm{p}} = 0.40$
  $n_{\mathrm{p}} = 1{,}000$
  $n_{\mathrm{n}} = \left(\pi_{\mathrm{n}}/2\pi_{\mathrm{p}}\right)^2 n_{\mathrm{p}}$
  $n_{\mathrm{u}} = 50{,}000$
- Model: 13-layer CNN – **All Convolutional Net** (Springenberg+, *ICLR'15*)

Remarks

- $\tilde{R}_{\mathrm{pu}}(g)$ is not a **shrinkage estimator**
  - $\tilde{R}_{\mathrm{pu}}(g) = \hat{R}_{\mathrm{pu}}(g)$ if $\hat{R}_{\mathrm{pu}}(g) \geq 0$ ← Identical for simple models
  - $\tilde{R}_{\mathrm{pu}}(g) > \hat{R}_{\mathrm{pu}}(g)$ if $\hat{R}_{\mathrm{pu}}(g) < 0$ ← Different for complex models
- $\tilde{R}_{\mathrm{pu}}(g)$ is biased but still consistent; bias is in $\mathcal{O}\left(\exp\left(-\frac{1}{1/n_{\mathrm{p}}+1/n_{\mathrm{u}}}\right)\right)$
- **Mean squared error** is reduced for certain losses
- Learning is consistent; **estimation error bound** is in $\mathcal{O}_p\left(\frac{1}{\sqrt{n_p}}+\frac{1}{\sqrt{n_{\mathrm{u}}}}\right)$ for linear-in-parameter models

Outline

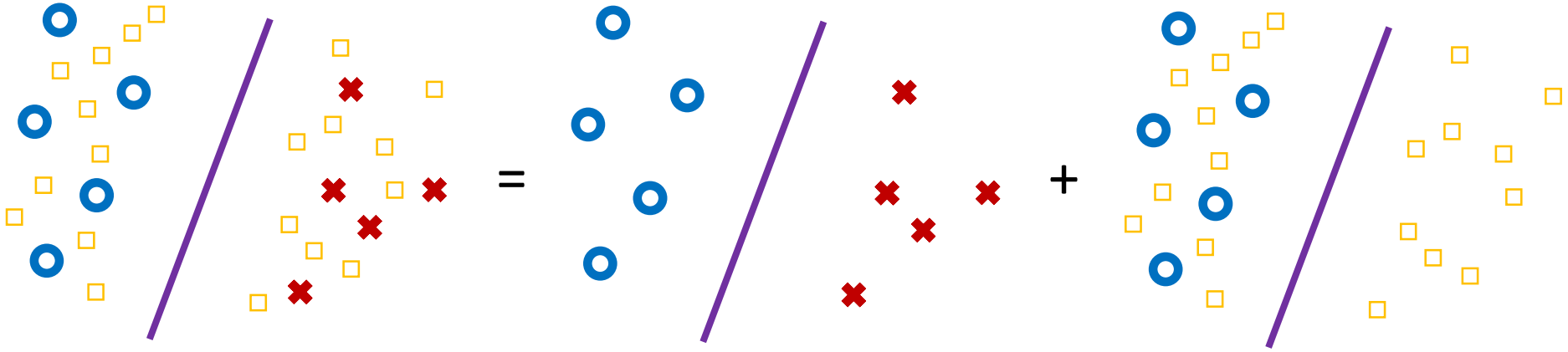
- Introduction
- Unbiased PU Learning
- Non-negative PU Learning
- PNU Learning
- Theoretical Analyses

Motivation

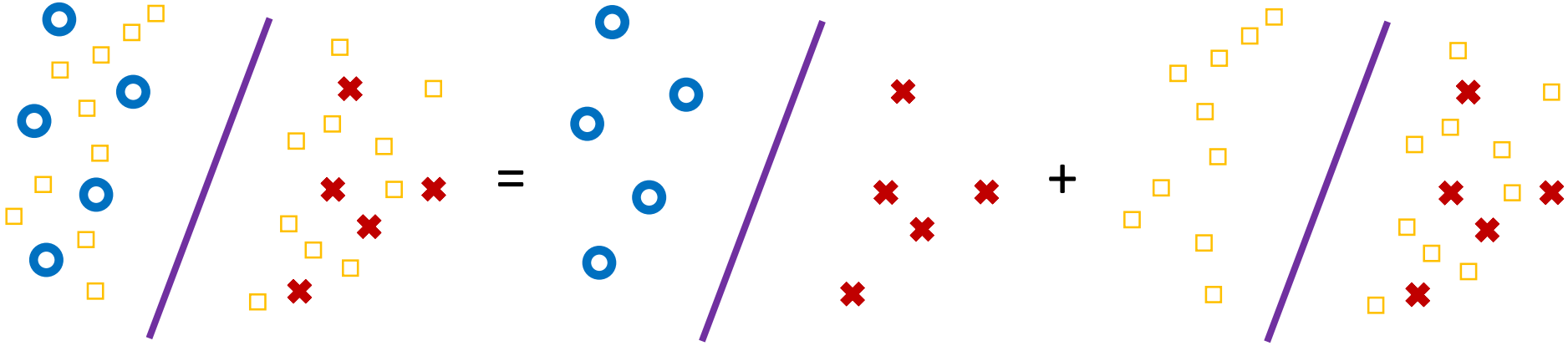
- SSL usually needs additional **distributional assumptions**
  - Cluster (Chapelle+, *NIPS'02*), manifold (Belkin+, *JMLR 2006*) …
  - U data are used in **regularization**
    ➔ They will bias the classifier (learned following ERM)
- Violated assumption ➔ U data become harmful & hurt
  (Cozman+, *ICML'03*; Sokolovska+, *ICML'08*; Li & Zhou, *TPAMI 2015*; Krijthe & Loog, *PR 2017*)
- PU learning has no additional distributional assumption
  - U data are used in **risk evaluation**
    ➔ They will not bias the classifier

Illustration

PNU learning = PN learning + PU learning



PNU learning = PN learning + NU learning



Unbiased risk estimators (Sakai+, *ICML'17*)

- Given $\gamma \in [0,1]$, PN+PU learning is to minimize

$$\hat{R}^{\gamma}_{\mathrm{pn,pu}}(g) = (1-\gamma)\hat{R}_{\mathrm{pn}}(g) + \gamma\hat{R}_{\mathrm{pu}}(g)$$

- Analogously, PN+NU learning is to minimize

$$\hat{R}^{\gamma}_{\mathrm{pn,nu}}(g) = (1-\gamma)\hat{R}_{\mathrm{pn}}(g) + \gamma\hat{R}_{\mathrm{nu}}(g)$$

where $\hat{R}_{\mathrm{nu}}(g) = \frac{\pi_{\mathrm{n}}}{n_{\mathrm{n}}}\sum_{x\in\mathcal{X}_{\mathrm{n}}}\left[\ell\left(-g(x)\right) - \ell\left(g(x)\right)\right] + \frac{1}{n_{\mathrm{u}}}\sum_{x\in\mathcal{X}_{\mathrm{u}}}\ell\left(g(x)\right)$

- Given $\eta \in [-1,1]$, we define PNU learning as to minimize

$$\hat{R}^{\eta}_{\mathrm{pnu}}(g) = \begin{cases} \hat{R}^{\eta}_{\mathrm{pn,pu}}(g), & \text{if } \eta \geq 0 \\ \hat{R}^{-\eta}_{\mathrm{pn,nu}}(g), & \text{if } \eta < 0 \end{cases}$$

➔ NU learning, if $\eta = -1$
PN learning, if $\eta = 0$
PU learning, if $\eta = +1$

Why not PU+NU?

- Motivated by **theoretical comparisons** (Niu+, *NIPS'16*)
- PN learning can never be the worst among PN, PU & NU learning
  - If n_{u} is sufficiently large (compared with n_{p} & n_{n})
    ➔ PN learning is the second best
  - Otherwise ➔ PN learning is the best
- PU+NU learning is not the best possible combination
  PN+PU & PN+NU learning are the best combinations

Note that theoretical & experimental results omitted for simplicity

Outline

- Introduction
- Unbiased PU Learning
- Non-negative PU Learning
- PNU Learning
- Theoretical Analyses

To be short and intuitive

- Based on upper bounds on **estimation errors**
- Find simple conditions, such that
  - PU learning is likely to outperform PN learning
    if $\pi_{\mathrm{p}}/\sqrt{n_{\mathrm{p}}} + 1/\sqrt{n_{\mathrm{u}}} < \pi_{\mathrm{n}}/\sqrt{n_{\mathrm{n}}}$
  - NU learning is likely to outperform PN learning
    if $\pi_{\mathrm{n}}/\sqrt{n_{\mathrm{n}}} + 1/\sqrt{n_{\mathrm{u}}} < \pi_{\mathrm{p}}/\sqrt{n_{\mathrm{p}}}$
- Either PU or NU learning (depending on π_{p}, π_{n}, n_{p} & n_{n})
  given infinite U data will improve on PN learning

Simplified estimation error bounds (Niu+, *NIPS'16*)

- Let $g^*/\hat{g}_{\mathrm{pn}}$, $\hat{g}_{\mathrm{pu}}$ & $\hat{g}_{\mathrm{nu}}$ be **true/empirical risk minimizers**
- Assume **Rademacher complexities** (Mohri+, *Foundations of ML*, 2012) of the **function class** $\mathcal{G}$ (a.k.a. **model family**) are in $\mathcal{O}(1/\sqrt{n})$
- Bounds below hold separately with probability at least $1-\delta$

$$R(\hat{g}_{\mathrm{pn}}) - R(g^*) \leq C(\delta) \cdot \left(\pi_{\mathrm{p}}/\sqrt{n_{\mathrm{p}}} + \pi_{\mathrm{n}}/\sqrt{n_{\mathrm{n}}}\right)$$
$$R(\hat{g}_{\mathrm{pu}}) - R(g^*) \leq C(\delta) \cdot \left(2\pi_{\mathrm{p}}/\sqrt{n_{\mathrm{p}}} + 1/\sqrt{n_{\mathrm{u}}}\right)$$
$$R(\hat{g}_{\mathrm{nu}}) - R(g^*) \leq C(\delta) \cdot \left(1/\sqrt{n_{\mathrm{u}}} + 2\pi_{\mathrm{n}}/\sqrt{n_{\mathrm{n}}}\right)$$

  where $C(\delta)$ is a function of δ

- Originally for du Plessis+ (*NIPS'14*) but can be extended to du Plessis+ (*ICML'15*)

Finite-sample comparisons (Niu+, *NIPS'16*)

- Tighter PU bound than PN bound, if $\alpha_{\mathrm{pu,pn}} = \frac{\pi_{\mathrm{p}}/\sqrt{n_{\mathrm{p}}}+1/\sqrt{n_{\mathrm{u}}}}{\pi_{\mathrm{n}}/\sqrt{n_{\mathrm{n}}}} < 1$
- Tighter NU bound than PN bound, if $\alpha_{\mathrm{nu,pn}} = \frac{\pi_{\mathrm{n}}/\sqrt{n_{\mathrm{n}}}+1/\sqrt{n_{\mathrm{u}}}}{\pi_{\mathrm{p}}/\sqrt{n_{\mathrm{p}}}} < 1$
- $\alpha_{\mathrm{pu,pn}}$ is monotonically decreasing in n_{p} & n_{u}
  - More U data ➔ PU improves
  - More P data ➔ PN improves too, but PU improves more!
- Many other implications discussed as well
  - When n_{p}, n_{n} & n_{u} are proportional
  - When $n_{\mathrm{p}}/n_{\mathrm{n}} \approx \pi_{\mathrm{p}}/\pi_{\mathrm{n}}$ is further imposed

Asymptotic comparisons (Niu+, *NIPS'16*)

- In practice, we may find PU worse than PN and $\alpha_{\mathrm{pu,pn}} > 1$
  Give up PU? Collect more U data (in order to improve PU)?
- Assume $n_{\mathrm{p}}, n_{\mathrm{n}} < \infty$ & $n_{\mathrm{u}} \to \infty$, or $\mathcal{O}(n_{\mathrm{p}}) = \mathcal{O}(n_{\mathrm{n}}) < \mathcal{O}(n_{\mathrm{u}})$
  - $\alpha^*_{\mathrm{pu,pn}}$ & $\alpha^*_{\mathrm{nu,pn}}$ exist as the limits of $\alpha_{\mathrm{pu,pn}}$ & $\alpha_{\mathrm{nu,pn}}$
- Then, $\alpha^*_{\mathrm{pu,pn}} \cdot \alpha^*_{\mathrm{nu,pn}} = 1$
  - Either $\alpha^*_{\mathrm{pu,pn}} < 1$ ← Limit of PU will improve on that of PN
  - Or $\alpha^*_{\mathrm{nu,pn}} < 1$ ← Limit of NU will improve on that of PN
  - Exception: $n_{\mathrm{p}}/n_{\mathrm{n}} \to \pi_{\mathrm{p}}^2/\pi_{\mathrm{n}}^2$ ← All 3 limits are equally good

Take-home message

- Unbiased PU learning and simple models given small data
  - **Logistic** loss or **double hinge** loss for convexity
- Non-negative PU learning and complex models given big data
  - **Sigmoid** loss for a similar shape to **zero-one** loss
- When PU doesn't work ➔ Label some N to see whether PN works
  - PN works and $\pi_{\mathrm{p}}/\sqrt{n_{\mathrm{p}}} > \pi_{\mathrm{n}}/\sqrt{n_{\mathrm{n}}}$ ➔ Label more P (expensive) or switch to NU/PNU if you want
  - PN works and $\pi_{\mathrm{p}}/\sqrt{n_{\mathrm{p}}} < \pi_{\mathrm{n}}/\sqrt{n_{\mathrm{n}}}$ ➔ Collect more U (cheap)
  - PN doesn't work as well ➔ Find a more suitable model

Beyond this talk …

- We are **two-sample** PU (Ward, Hastie, Barry, Elith & Leathwick, *Biometrics 2009*)
  - Fairly different from **one-sample** PU (Elkan & Noto, *KDD'08*)
  - According to Menon, Van Rooyen, Ong, Williamson (*ICML'15*)
    $\in$ **mutually contaminated** learning (Scott, *AISTATS'15*)
    $\notin$ **class-conditional noise** learning (Natarajan, Dhillon, Ravikumar & Tewari, *NIPS'13*)
- Following work in PU learning
  - Multi-label ranking (Kanehira & Harada, *CVPR'16*) ← Fancy application
  - AUC maximization (PU & SSL) (Sakai, Niu & Sugiyama, *MLJ*, to appear)
  - Multi-instance binary classification (Bao, Sakai, Sato & Sugiyama, *arXiv 2017*)
  - Multi-class classification (Konno, *UTokyo bachelor thesis 2017*)