Trade-offs in Large-Scale Distributed Tuplewise Estimation and Learning

Published @ ECML-PKDD 2019

Robin Vogel[1,2]

Joint work with: Aurélien Bellet[3], Stephan Clémençon[1], Ons Jelassi[1] and Guillaume Papa[1].

[1] Telecom Paris, [2] IDEMIA, [3] Inria

Magnet seminar - 24/10/2019

My thesis

Started in 06/2017, CIFRE between **Telecom Paris** and **IDEMIA**.

Advisors: Stéphan Clémençon, Aurélien Bellet and Anne Sabourin.

IDEMIA stems from the merger in 2017 of Safran Identity and Security (formerly **Morpho**) and Oberthur Technologies.
It specializes in **biometrics** and computer security.

I have been part of the **facial recognition (FR) team** since 12/2016, and had some experience with deep face encoding models.

My thesis tackle **statistical machine learning** problems oriented by FR issues, with a focus on **ranking** and **similarity learning**.

Other publications:
· ICML 2018: guarantees for **ERM of TPR@FPR**$= \alpha$ with similarities,
· LOD 2019: learn **tree-based similarities** with good ROC curves.

Outline

Biometric verification (1/2)

$S(X, X') > t$

A biometric system uses:

- Two **measurements** X and X',
- A **similarity** S that quantifies the likeness of (X, X'),
- A **threshold** t that separates positive and negative pairs.

Aim: $S(X, X') > t$ is a good indicator of $Z = +1$ with:

$$Z = \begin{cases} +1 & \text{if } (X, X') \text{ from the same person,} \\ -1 & \text{otherwise.} \end{cases}$$

Two types of errors:

$$\mathrm{TPR}_S(t) := \mathbb{P}\{S(X, X') > t \mid Z = +1\},$$
$$\mathrm{FPR}_S(t) := \mathbb{P}\{S(X, X') > t \mid Z = -1\}.$$

The set $\{(FPR_S(t), TPR_S(t)) \mid t \in \mathbb{R}\}$ is known as the **ROC curve**.

Biometric verification (2/2)

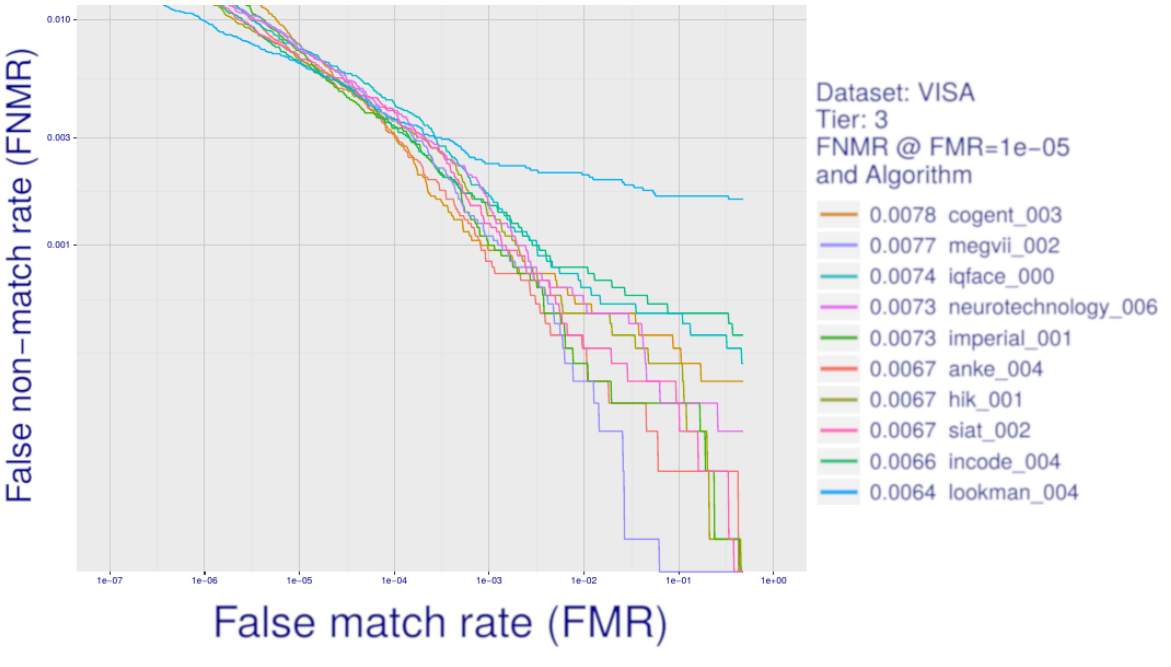


Figure: Extract of the NIST Face Recognition Vendor Test (FRVT) report.

Several criterions measure ROC accuracy:

- Area Under the ROC Curve (**AUC**),
- Pointwise ROC optimization (**pROC**) see [Vogel et al., 2018].

What if we estimate **AUC** or **pROC** in a **distributed environment** ?

Large scale distributed data processing

Very large datasets are common nowadays in ML.
$\rightarrow$ Distribute the data in **partitions** over several machines.

Cluster computing frameworks:

· Abstract network and communication aspects of distribution. ☺

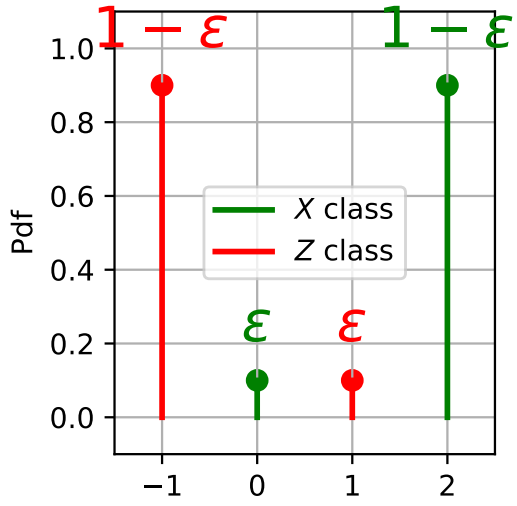
· Restrict the types of operations efficiently achieved. ☹

E.g. **Apache Spark** [Meng et al., 2016], **Petuum** [Xing et al., 2015], ...

Most ML techniques optimize **standard means** $\hat{L} = \sum_i \ell(x_i)/n$,
Those are **separable across partitions**.
$\rightarrow$ One can **efficiently estimate** those.

Very common statistics - e.g. *U*-**statistics** - are not.
$\rightarrow$ Estimation can be **slow or inaccurate**.

Illustration: estimation of the AUC

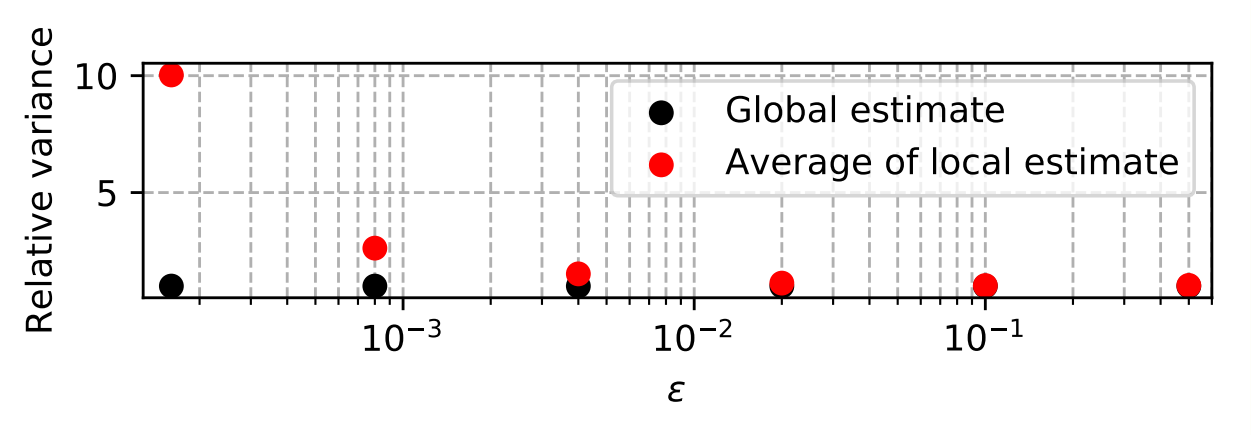


Let $X \in \{0, 2\}$ be the positive class input r.v.,
$Z \in \{-1, +1\}$ the negative class r.v., with:

$$\mathbb{P}\{X = 2\} = \mathbb{P}\{Z = 1\} = 1 - \epsilon.$$

We estimate the AUC $\mathbb{E}[h(X, Z)]$ with $h : x, z \mapsto \mathbb{I}\{x > z\}$, with $n = 5,000$ (resp. $m = 50$) positive (resp. negative) observations, with **global** or **distributed** estimators with $N = 10$ clusters.

As $\epsilon \to 0$, local estimators get **poorer**.



Our contribution

Problem:
When a statistic is **not separable across partitions**,
frameworks may be **unsuited to computing accurate estimators.**

Contribution: Quantified analysis for *U*-statistics,

1. **Efficient estimators** of *U*-statistics in a distributed setting.
2. The analysis of their **accuracy-vs-time tradeoff**.
3. Learning experiment with those as **gradient estimators**.

Plan:

- Properties of *U*-statistics: variance and bounds,
  see [Hoeffding, 1948], [Clémençon et al., 2016],
- Distributing the data,
- Contributions 1-3.

Outline

U-statistics: definition and examples

Introduce K independent i.i.d. samples $\mathcal{D}_k = \{X_1^{(k)}, \dots, X_{n_k}^{(k)}\} \subset \mathcal{X}_k$, and a **kernel** $h : \mathcal{X}_1^{d_1} \times \cdots \times \mathcal{X}_K^{d_K} \to \mathbb{R}$ with $h \in L^2$.

The **generalized U-statistic** of degrees $(d_1, \dots, d_K)$ is defined as:

$$U_{\mathbf{n}}(h) = \frac{1}{\prod_{k=1}^{K} \binom{n_k}{d_k}} \sum_{I_1} \dots \sum_{I_K} h(\mathbf{X}_{I_1}^{(1)}, \mathbf{X}_{I_2}^{(2)}, \dots, \mathbf{X}_{I_K}^{(K)}),$$

where $\sum_{I_k}$ is the sum over all $\binom{n_k}{d_k}$ subsets of d_k elements of $\mathcal{D}_k$.

Examples:

- a sample variance $h(x_1, x_2) = (x_1 - x_2)^2$,
- Kendall's tau $h((x_1, y_1), (x_2, y_2)) = \mathbb{I}\{(x_1 - x_2) \cdot (y_1 - y_2) > 0\}$,
- **clustering**, **metric learning** and **ranking** criterions.

Two-sample U-statistics of degree $(1, 1)$

We focus on a two-sample U-statistics for simplicity, but results can be extended.

Introduce, with $m \ll n$:

- the **abundant positive** i.i.d. sample $\mathcal{D}_n = \{X_1, \ldots, X_n\} \subset \mathcal{X}$,
- the **scarce negative** i.i.d. sample $\mathcal{Q}_m = \{Z_1, \ldots, Z_m\} \subset \mathcal{Z}$,
- a **kernel** $h : \mathcal{X} \times \mathcal{Z} \mapsto \mathbb{R}$.

Objective: Estimate $U(h) = \mathbb{E}[h(X_1, Z_1)]$.

With $\mathbf{n} = (n, m)$, the U**-statistic** $U_{\mathbf{n}}$, where:

$$U_{\mathbf{n}} := \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} h(X_i, Z_j),$$

is the **unbiased** estimator of $U(h)$ with **lowest variance**.

First Hoeffding decomposition

Introduce $\mathfrak{S}_n$ as the symmetric group over $\{1, \ldots, n\}$, and $N = n \wedge m$ the minimum of n and m.

The **first Hoeffding decomposition** writes U_n as an average of dependent standard empirical processes. Formally,

$$U_{\mathbf{n}} = \frac{1}{n!m!} \sum_{\sigma_1 \in \mathfrak{S}_n} \sum_{\sigma_2 \in \mathfrak{S}_m} \underbrace{\frac{1}{N} \sum_{i=1}^{N} h(X_{\sigma_1(i)}, Z_{\sigma_2(i)})}_{\bar{U}_n}.$$

Jensen's inequality applied to the Chernoff bound of $U_{\mathbf{n}} - \mathbb{E}[U]$, imply that $\mathbb{P}\{U_n - \mathbb{E}U > a\} \leq e^{-ta} \cdot \mathbb{E}\left[e^{t\left(\bar{U}_n - \mathbb{E}U\right)}\right]$.

Consequence: One can derive concentration inequalities in N of the same type as those for empirical processes.

Empirical Risk Minimization (ERM) of U-statistics

Proposition 1

Proposition 2 of [Clémençon et al., 2016].
Let $\mathcal{H}$ be a collection of symmetric kernels bounded by 1.
Suppose that $\mathcal{H}$ is a VC-major class of functions with VC dimension V.
For all $\delta \in (0,1)$, we have with probability at least $1-\delta$,

$$\sup_{h \in \mathcal{H}} |U_{\mathbf{n}} - U| \leq 2\sqrt{\frac{2V \log(1+N)}{N}} + \sqrt{\frac{\log(1/\delta)}{N}}.$$

Consequence: Finite-time bounds for ERM of U-statistics,
since with $h_{\mathbf{n}}$ minimizer of $U_{\mathbf{n}}$:

$$U(h_{\mathbf{n}}) - U(h^*) \leq 2 \sup_{h \in \mathcal{H}} |U_{\mathbf{n}} - U|.$$

Second Hoeffding decomposition

Set the **Hajèk projections** $h_1(x) = \mathbb{E}[h(x, Z_1)], h_2(z) = \mathbb{E}[h(X_1, z)]$,

$$\text{and} \quad h_0(x, z) = h(x, z) - h_1(x) - h_2(z) + U(h).$$

A U-statistic $U_{\mathbf{n}}$ is called **degenerate** when $h_1 = U$ and $h_2 = U$ a. s.

The **second Hoeffding decomposition** decomposes U_n as a sum of: two empirical processes and a degenerate U-statistic.

Formally, $U_{\mathbf{n}} = T_1 + T_2 + W_0 - U$ with

$$T_1 = \frac{1}{n}\sum_{i=1}^{n} h_1(X_i) \text{ and } T_2 = \frac{1}{m}\sum_{j=1}^{m} h_2(Z_j) \text{ and } W_0 = \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m} h_0(X_i, Z_j),$$

Consequences:

· Easy derivation of the variance of $U_{\mathbf{n}}$,

· Sharper learning bounds under noise assumptions.

Variance of a U-statistic

Introducing $\sigma_1^2 = \mathrm{Var}(h_1(X))$, $\sigma_2^2 = \mathrm{Var}(h_2(Z))$, $\sigma_0^2 = \mathrm{Var}(h_0(X_1, Z_1))$:

$$\mathrm{Var}(U_{\mathbf{n}}) = \frac{\sigma_1^2}{n} + \frac{\sigma_2^2}{m} + \frac{\sigma_0^2}{nm}.$$

The variance of $U_{\mathbf{n}}$ depends on:
· the distribution of X and Z,
· the kernel h.

Their contribution is summarized by the coefficients σ_0^2, σ_1^2 and σ_2^2, as well as that of all the estimators in this talk !

Examples: with X_1, Z_1 centered random variables:

$h(x, z) = x + z$ gives $\sigma_0^2 = 0$. | $h(x, z) = x \cdot z$ gives $\sigma_1^2, \sigma_2^2 = 0$.

Faster bounds for ERM of U-statistics

Proposition 2

See [Arcones and Giné, 1994].
Under similar assumptions as Proposition 1.
For all $\delta \in (0,1)$, we have with probability at least $1-\delta$,

$$\sup_{h \in \mathcal{H}} |W_0| \leq C \frac{V \log(N/\delta)}{N}.$$

Introduce a variance condition for T_1 and T_2, of the form, for T_1:

$$\mathrm{Var}(h_1(X)) \leq c \left(\mathbb{E}[h_1(X)]\right)^a,$$

with $c \in \mathbb{R}_+$, $a \in [0,1]$.

Using **Bernstein's or Talagrand's inequality**, solving a fixed point inequality gives a bound in $O(N^{-1/(2-a)})$ on T_1, without log terms.

See [Clémençon et al., 2008] for more details.

Incomplete U-statistics

$U_{\mathbf{n}}$ is an average of nm elements $\rightarrow$ **What if** $n = 10^9$**,** $m = 10^3$**?**:

One answer is **incomplete** U**-statistics**,

$$\widetilde{U}_B := \frac{1}{B} \sum_{(i,j) \in \mathcal{D}_B} h(X_i, Z_j),$$

where $\mathcal{D}_B$ is a set of B elements selected with sampling with replacement (SWR) in $\Lambda = \{(i,j)\}_{i \in [\![n]\!], j \in [\![m]\!]}$.

Equation (21) in [Clémençon et al., 2016]:

$$\mathrm{Var}(\tilde{U}_B) = \left(1 - \frac{1}{B}\right) \mathrm{Var}(U_{\mathbf{n}}) + \frac{1}{B} \mathrm{Var}(h(X, Z)).$$

Learning with incomplete U-statistics

Theorem 3

Theorem 6 of [Clémençon et al., 2016].
Under the same assumptions as Proposition 1:
For all $\delta \in (0,1)$, we have with probability at least $1-\delta$,

$$\sup_{h \in \mathcal{H}} |\tilde{U}_B - U_{\mathbf{n}}| \leq \sqrt{2 \frac{V \log(1+\#\Lambda) + \log(2/\delta)}{B}}.$$

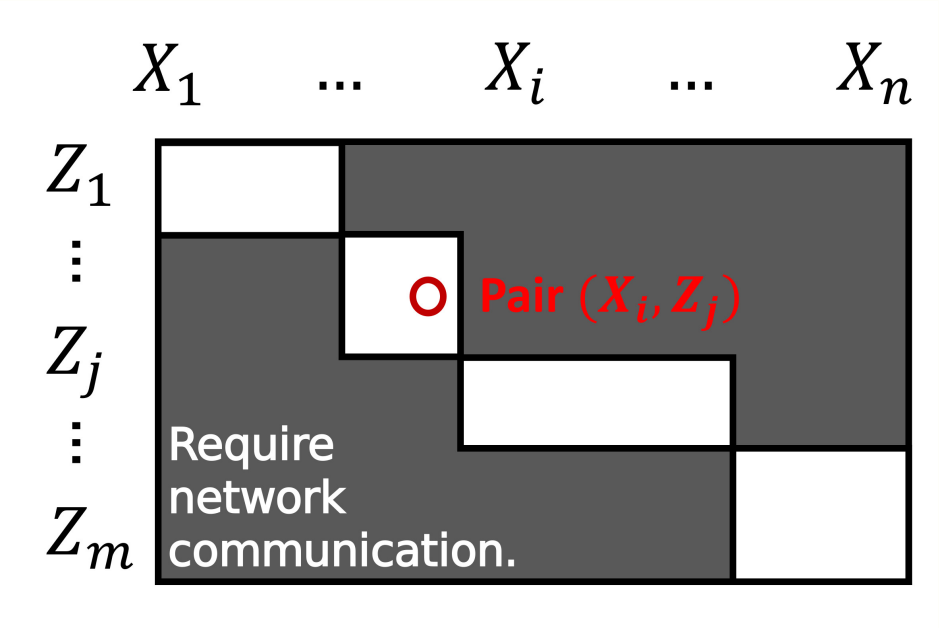
Consequences:

- Choosing $B \sim N$, yields an ERM bound in $O(\sqrt{\log(N)/N})$ for the minimization of $\tilde{U}_B$.
- Minimizing complete U-statistics of a small subsample that forms $B \sim N$ pairs, yields an ERM bound in $O(\sqrt{\log(N)}/N^{1/4})$.

Outline

Distributed environment

We **distribute the data** on *N* workers, see figure below.



One is the **master node**, its role is to **aggregate local estimates**.

In this context,
computing $U_{\mathbf{n}}$ or $\tilde{U}_B$ require too much **network communication**.

Ways of distributing data

One can distribute the instances following:

· sampling without replacement (**SWOR**),
  $\rightarrow$ e.g. when splitting large datasets on several servers.

· sampling with replacement (**SWR**),
  $\rightarrow$ e.g. for parallel calculus on lots of batches from a large memory.

⚠ Using **SWOR** or **SWR**, maybe no elements of $\mathcal{Q}_m$ are in a worker.
One can pass a **default value** for the local estimates,
but it makes the analysis complicated.

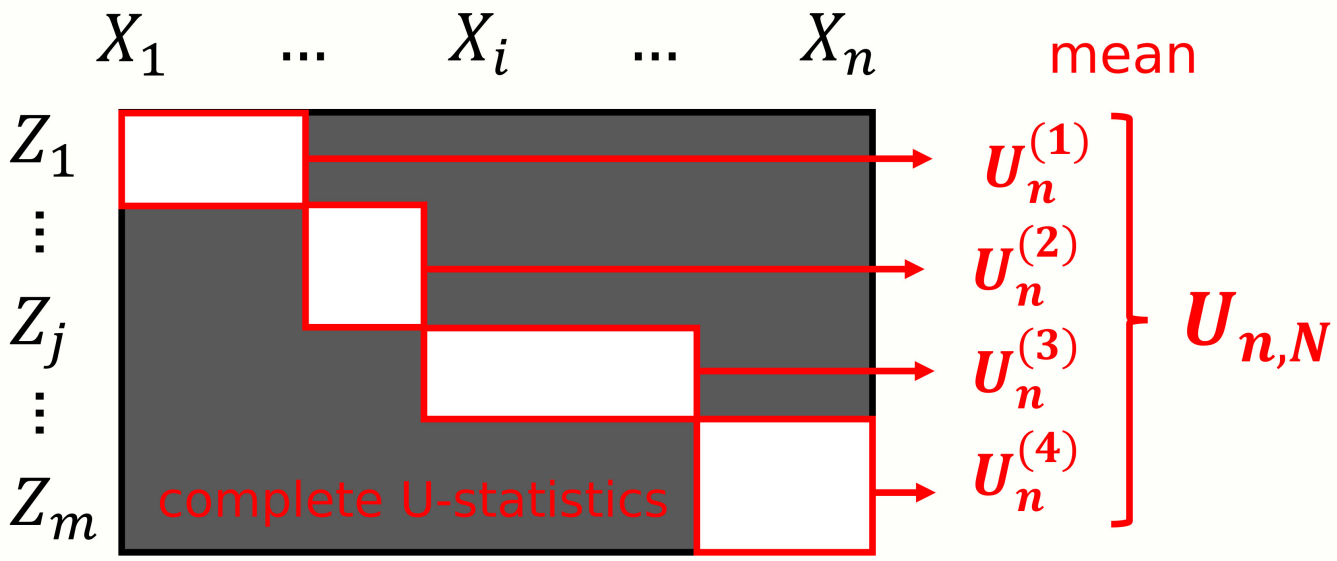
We can distribute data **proportionally**:
· each cluster contains n/N instances from $\mathcal{D}_n$, and m/M from $\mathcal{Q}_m$.
· For **SWOR**, achieved by **sharing a random seed** between workers.
These settings are named **prop-SWOR** and **prop-SWR**.

Here: prop-SWOR unless otherwise specified (see paper for others).

Outline

Naive estimators (1/2)

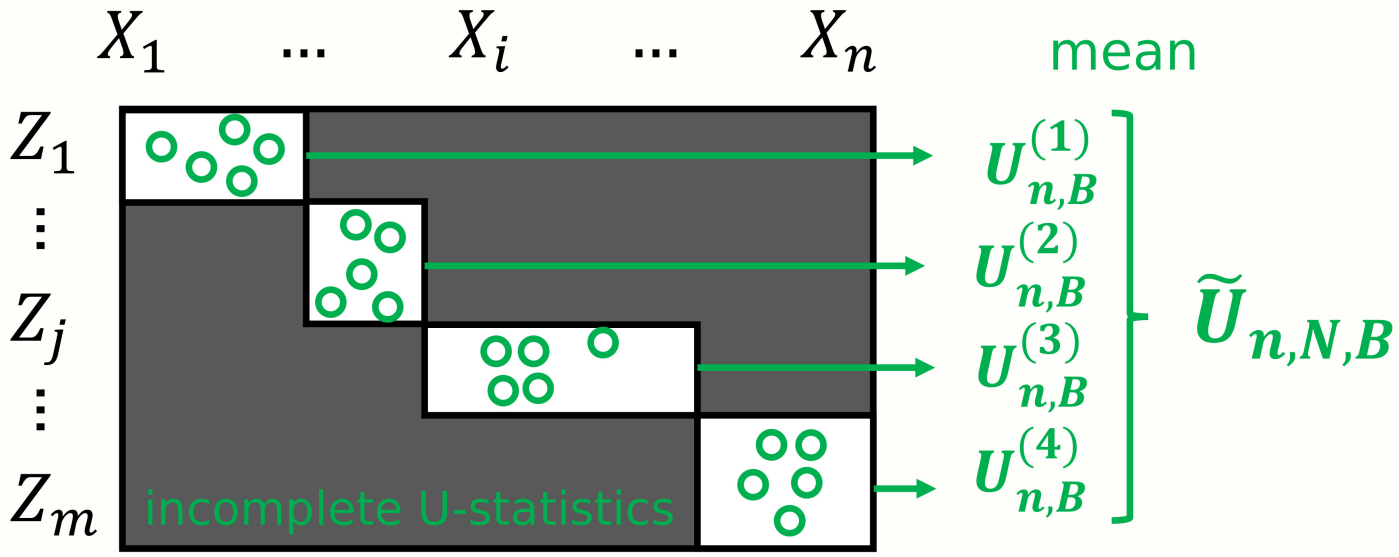
Averaging full *U*-statistics from each cluster gives $U_{\mathbf{n},N}$.



Proposed statistics: $U_{\mathbf{n},N}$

Naive estimators (2/2)

Averaging B pairs SWR from each cluster gives $\widetilde{U}_{\mathbf{n},N,B}$.

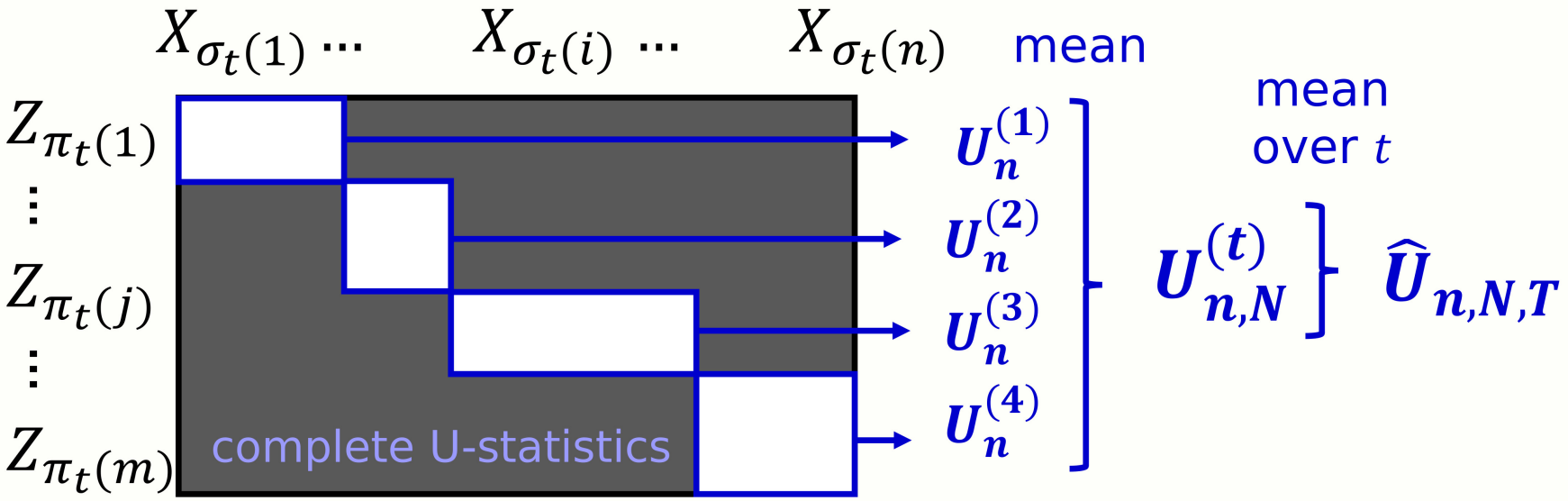


Proposed statistics: $U_{\mathbf{n},N}$, $\widetilde{U}_{\mathbf{n},N,B}$,

Idea: Improve the precision by **averaging unseen pairs**.

Estimators with redistribution (1/2)

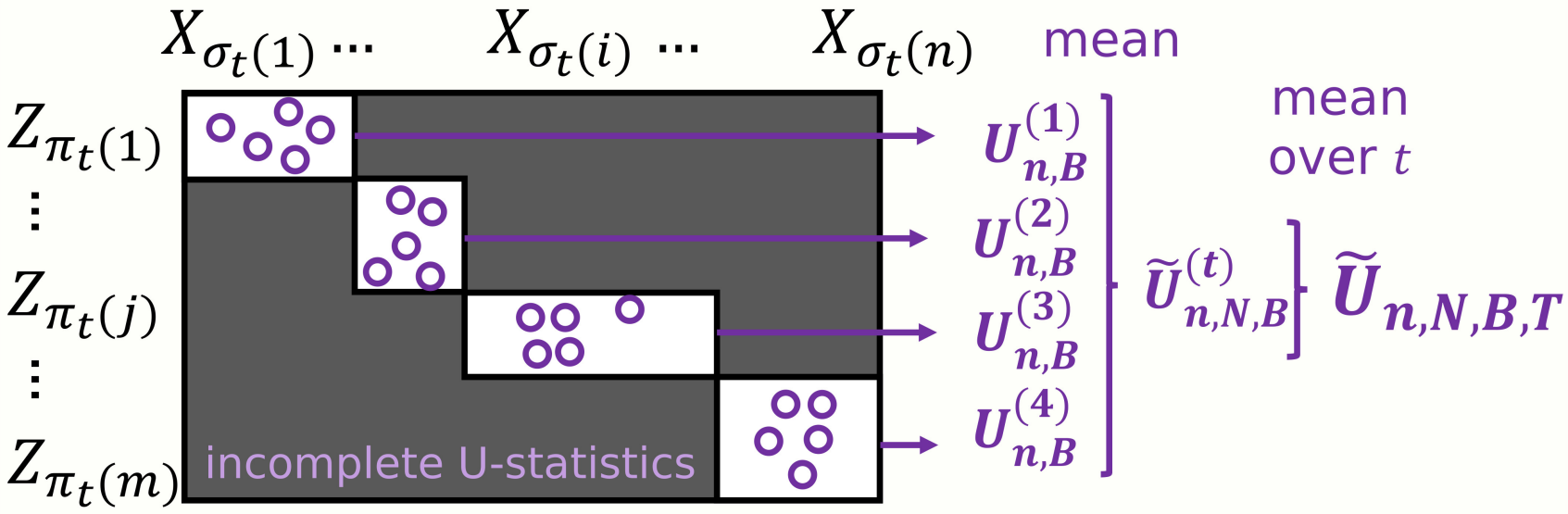
Averaging $U_{\mathbf{n},N}$ on T redistributions of the data gives $\widehat{U}_{\mathbf{n},N,T}$.



Proposed estimators: $U_{\mathbf{n},N}$, $\widetilde{U}_{\mathbf{n},N,B}$, $\widehat{U}_{\mathbf{n},N,T}$.
With σ_t, π_t random permutations at time t.

Estimators with redistribution (2/2)

Averaging $\widetilde{U}_{\mathbf{n},N,B}$ on T redistributions of the data gives $\widetilde{U}_{\mathbf{n},N,B,T}$.



Proposed estimators: $U_{\mathbf{n},N}$, $\widetilde{U}_{\mathbf{n},N,B}$, $\widehat{U}_{\mathbf{n},N,T}$, and $\widetilde{U}_{\mathbf{n},N,B,T}$.
With σ_t, π_t random permutations at time t.

All of the estimators are **unbiased**.

Variance expressions

We have: Variances in **closed form**.

For the **naive estimators**:

$$\mathrm{Var}(U_{\mathbf{n},N}) = \mathrm{Var}(U_{\mathbf{n}}) + (N-1)\frac{\sigma_0^2}{nm},$$
$$\mathrm{Var}(\widetilde{U}_{\mathbf{n},N,B}) = \left(1 - \frac{1}{B}\right)\mathrm{Var}(U_{\mathbf{n},N}) + \frac{\sigma^2}{NB}.$$

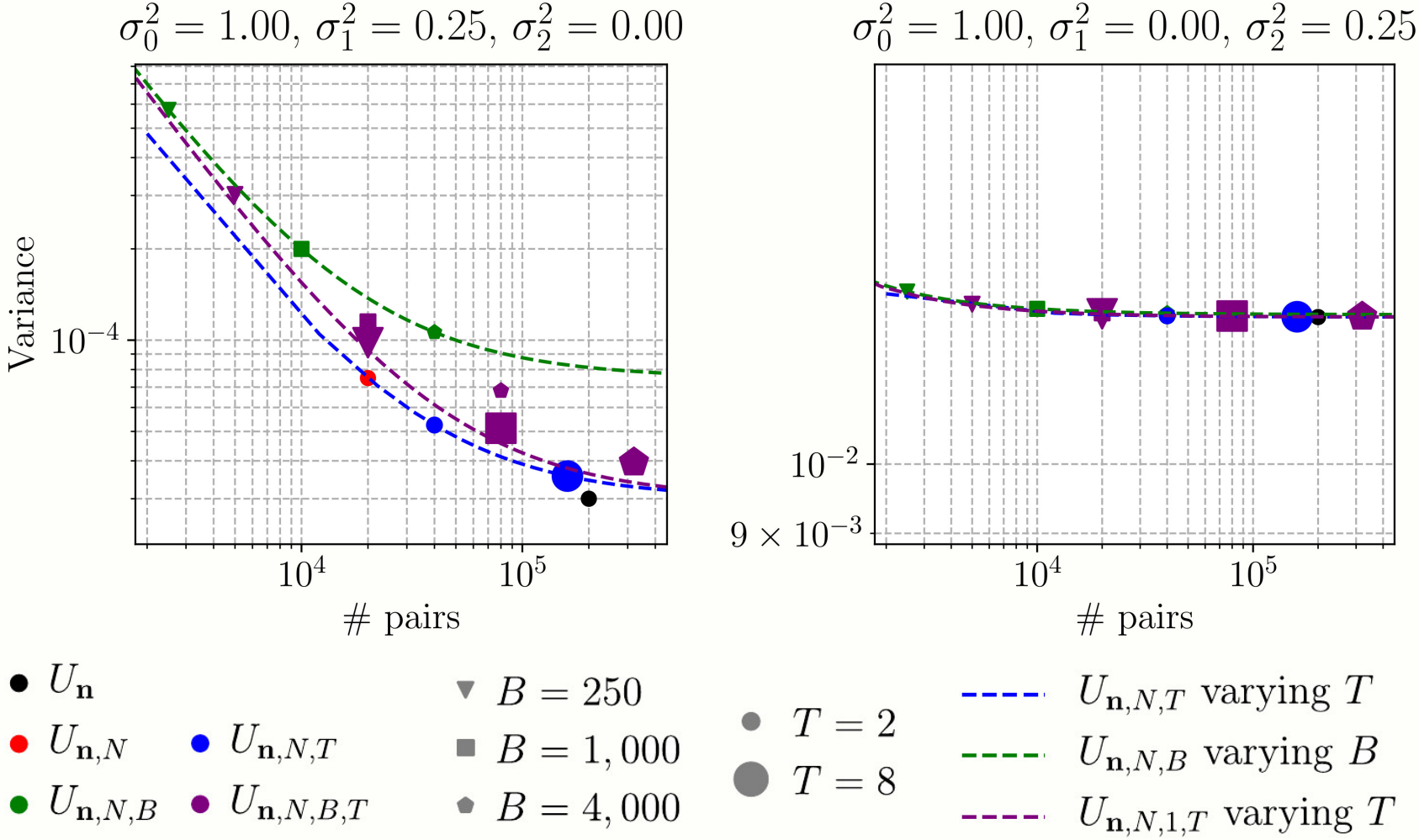
$\rightarrow$ Term in $N\sigma_0^2/nm$.

For the **estimators with redistribution**:

$$\mathrm{Var}(\widehat{U}_{\mathbf{n},N,T}) = \mathrm{Var}(U_{\mathbf{n}}) + (N-1)\frac{\sigma_0^2}{nmT},$$
$$\mathrm{Var}(\widetilde{U}_{\mathbf{n},N,B,T}) = \mathrm{Var}(\widehat{U}_{\mathbf{n},N,T}) - \frac{1}{TB}\mathrm{Var}(U_{\mathbf{n},N}) + \frac{\sigma^2}{NTB}.$$

$\rightarrow$ Term in $N\sigma_0^2/Tnm$.

Illustrations of the variances - **prop-SWOR**

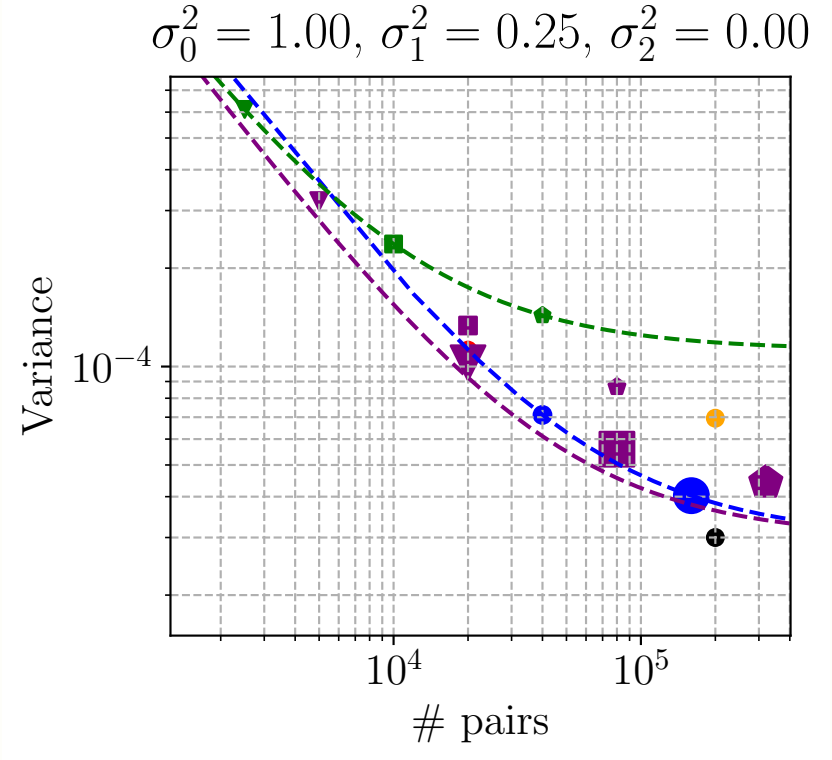


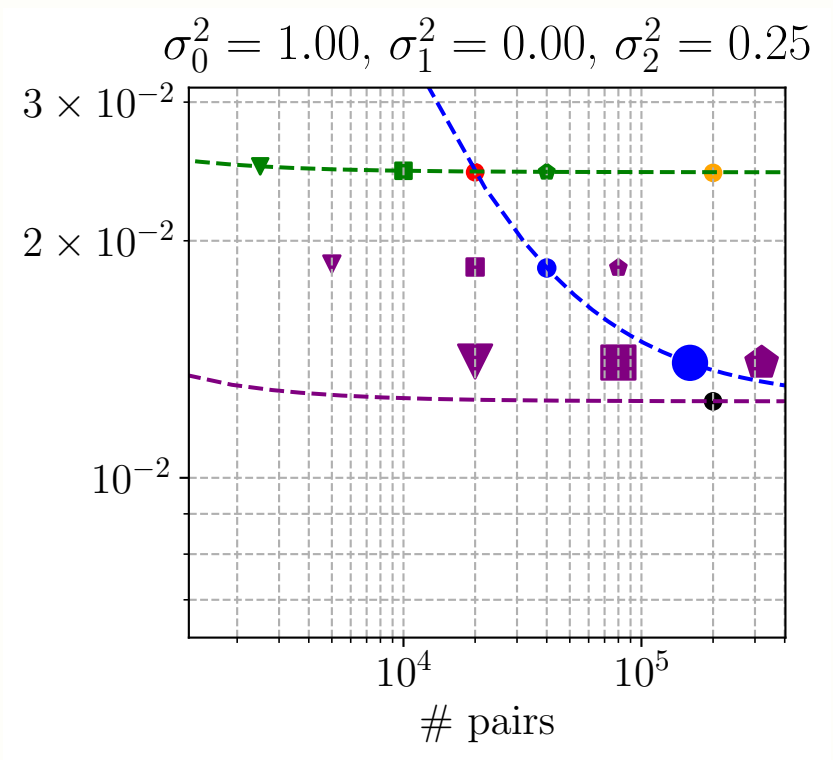
With $n = 100,000$, $m = 200$ and $N = 100$.

· Redistribution is useful when $\sigma_2^2 \ll \sigma_0^2$,

· For any same # of pairs, $\widetilde{U}_{\mathbf{n},N,B,T}$ is worse than $\widehat{U}_{\mathbf{n},N,T}$.

Illustrations of the variances - **prop-SWR**





- $U_{\mathbf{n}}$
- $U_{\mathbf{n},N}$
- $U_{\mathbf{n},N,B}$
- $U_{\mathbf{n},N,T}$
- $U_{\mathbf{n},N,B,T}$

- $B = 250$
- $B = 1,000$
- $B = 4,000$

- $T = 2$
- $T = 8$

- $U_{\mathbf{n},N,T}$ varying T
- $U_{\mathbf{n},N,B}$ varying B
- $U_{\mathbf{n},N,1,T}$ varying T

With $n = 100,000$, $m = 200$ and $N = 100$.

· Similar results,

· $\widetilde{U}_{\mathbf{n},N,B,T}$ corrects the redundancy induced by **SWOR**.

Outline

Repartitioning for stochastic gradient descent

Objective:
Assume that $\mathcal{H}$ is indexed by a parameter $\theta \in \mathbb{R}^q$, minimize $U_{\mathbf{n}}(h_\theta)$, the most accurate approximation of $U(h_\theta)$, by gradient descent.

$U_{\mathbf{n}}$ and $\tilde{U}_B$ are not efficient in a distributed environment.
See [Papa et al., 2015] for the analysis of SGD for incomplete U-stats.

Idea: Use **gradient estimators with redistribution**.
The gradient estimators are $\nabla_\theta \tilde{U}_{\mathbf{n},N,B}(h_\theta)$,
but we redistribute the data every n_r gradient steps,
$n_r = 1$ optimizes for $U_{\mathbf{n}}$, while $n_r = +\infty$ optimizes for $U_{\mathbf{n},N}$.

Challenge: Studying such an optimization process is hard,
since the **objective changes every n_r iterations**.
$\rightarrow$ We demonstrate its effectiveness empirically.

Application: Optimize a relaxation of the AUC

Objective: optimize the AUC using a its convex upper-bound:

$$U_n(h_{\text{AUC}}) := \frac{1}{nm} \sum_{i,j} \mathbb{I}\{w^\top (X_i - Z_j) > 0\},$$
$$\leq \frac{1}{nm} \sum_{i,j} \left[1 + w^\top (X_i - Z_j)\right]_+ =: U_n(h_{\text{conv}}).$$

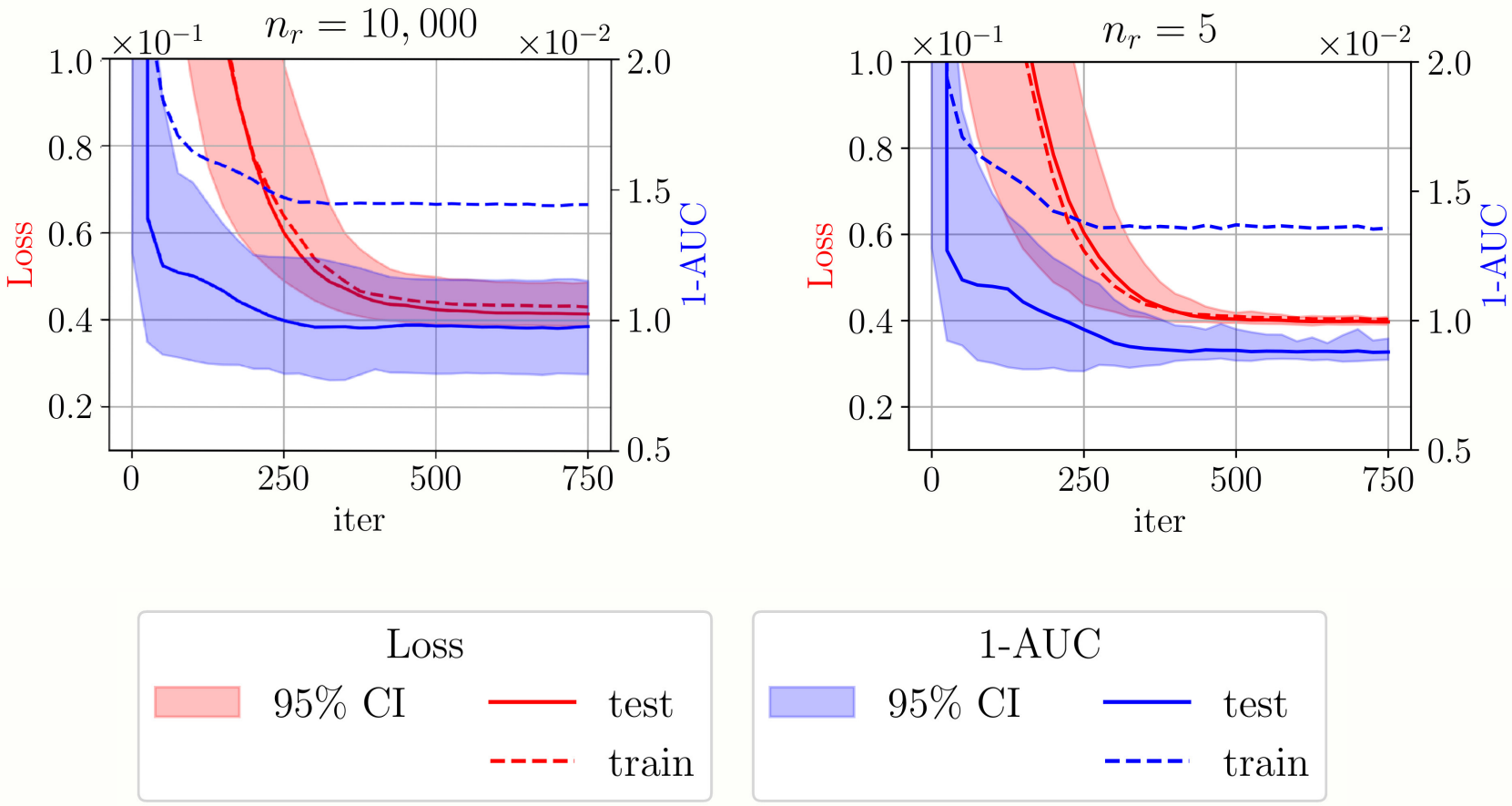
Optimizer: Momentum BGD with LR 10^{-2}, mom. 0.9 for 5×10^3 iter.

Loss: $U_{\mathbf{n},N,B}(h_{\text{conv}}) + \lambda \|w\|_2^2$, with redistribution each n_r iter.

Parameters: $N = 100, B = 500, \lambda = 0.01, n_r \in \{1, 5, 25, +\infty\}$.

Dataset: Shuttle (outlier dataset), $n = 45,000$, $m = 3,500$,
20% as test set, train monitored on a fixed set of $450K$ pairs,
see [Rayana, 2016].

Application: Results



- Not redistributing $\rightarrow$ end performance very noisy ☹
- Redistributing $\rightarrow$ better performance (with high. prob.) ☺

Thank you for your attention !

References I

Arcones, M. and Giné, E. (1994).

U-processes indexed by Vapnik-Chervonenkis classes of functions with applications to asymptotics and bootstrap of U-statistics with estimated parameters.

Stochastic Processes and their Applications, 52:17–38.

Clémençon, S., Colin, I., and Bellet, A. (2016).

Scaling-up Empirical Risk Minimization: Optimization of Incomplete *U*-statistics.

Journal of Machine Learning Research, 17(76):1–36.

Clémençon, S., Lugosi, G., and Vayatis, N. (2008).

Ranking and Empirical Minimization of U-Statistics.

The Annals of Statistics, 36(2):844–874.

Hoeffding, W. (1948).

A class of statistics with asymptotically normal distribution.

The Annals of Mathematical Statistics, 19:293–325.

References II

Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., Freeman, J., Tsai, D., Amde, M., Owen, S., Xin, D., Xin, R., Franklin, M. J., Zadeh, R., Zaharia, M., and Talwalkar, A. (2016).

MLlib: Machine Learning in Apache Spark.

Journal of Machine Learning Research, 17(34):1–7.

Papa, G., Clémençon, S., and Bellet, A. (2015).

Sgd algorithms based on incomplete u-statistics: Large-scale minimization of empirical risk.

In *NIPS 2015*.

Rayana, S. (2016).

Odds library.

Vogel, R., Bellet, A., and Clémençon, S. (2018).

A probabilistic theory of supervised similarity learning for pointwise ROC curve optimization.

In *ICML 2018*.

References III

Xing, E. P., Ho, Q., Dai, W., Kim, J. K., Wei, J., Lee, S., Zheng, X., Xie, P., Kumar, A., and Yu, Y. (2015).

Petuum: A New Platform for Distributed Machine Learning on Big Data.

IEEE Transactions on Big Data, 1(2):49–67.