Statistically Optimal K-means Clustering via Nonnegative Low-rank Semidefinite Programming

Richard Y. Zhang
Univ. of Illinois at Urbana-Champaign
ryz@illinois.edu

Joint work with
Yubo Zhuang (UIUC), **Xiaohui Chen** (USC), **Yang Yun** (UIUC)

Why *statistically optimal* K-means clustering?

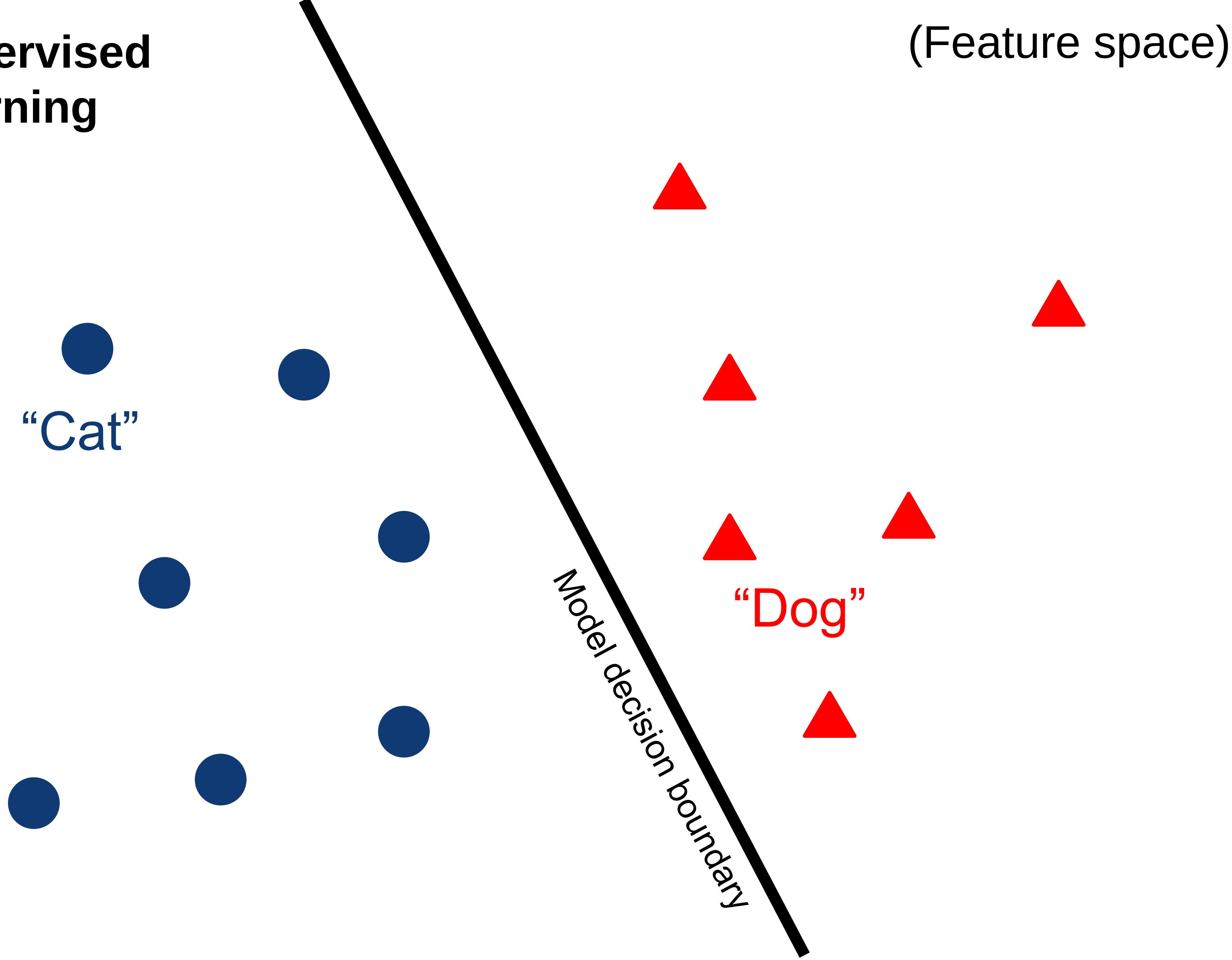
99% "Cat"

Secret sauce: Massive databases of *high-quality* labeled data

But need even more labeled data!

Rank	Model	Percentage correct	PARAMS	Accuracy	Extra Training Data	Paper	Code	Result	Year
1	**EffNet-L2** (SAM)	96.08			✓	Sharpness-Aware Minimization for Efficiently Improving Generalization			2020
2	**Swin-L + ML-Decoder**	95.1			✓	ML-Decoder: Scalable and Versatile Classification Head			2021
3	**µ2Net** (ViT-L/16)	94.95			✓	An Evolutionary Approach to Dynamic Introduction of Tasks in Large-scale Multitask Learning Systems			2022

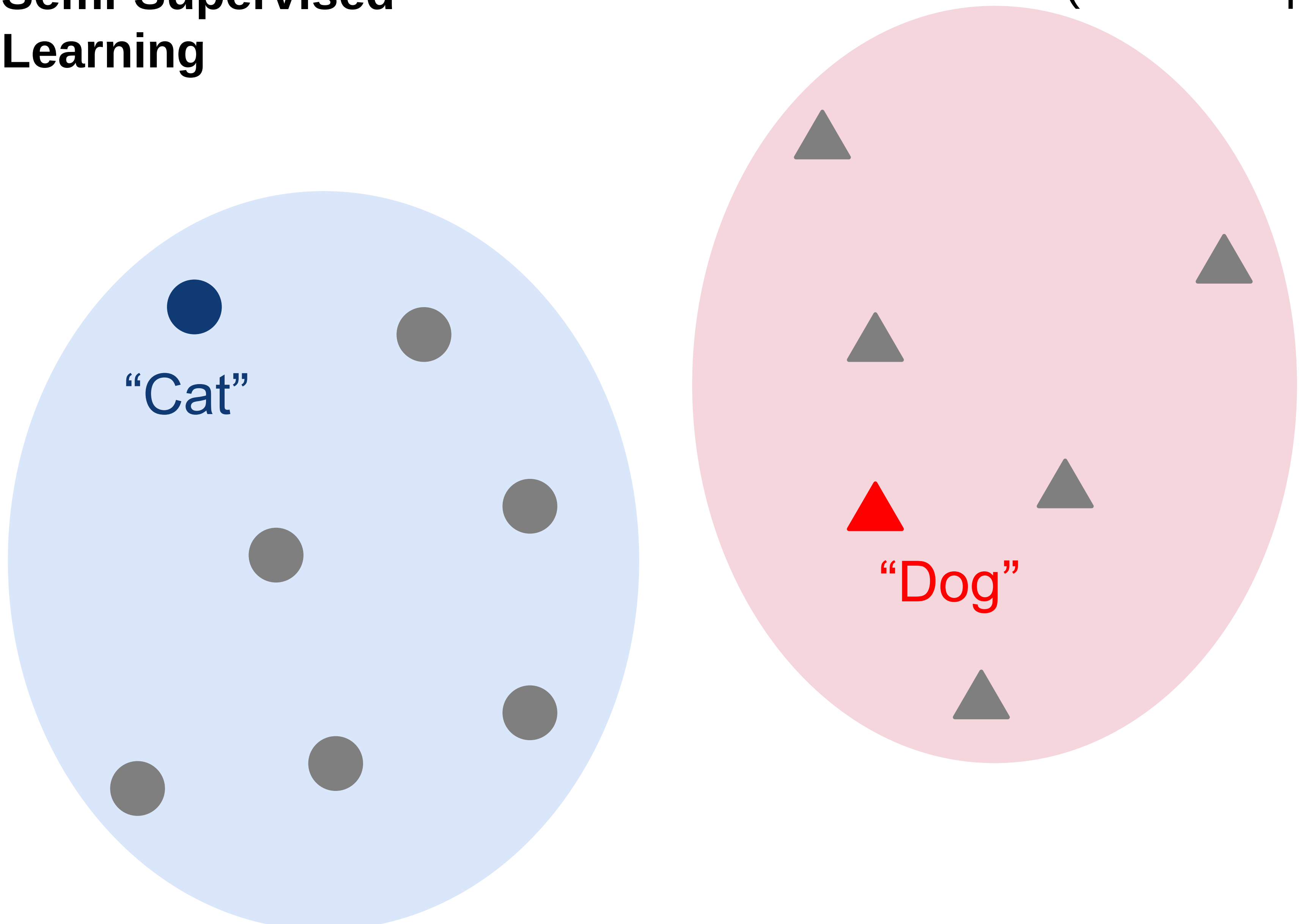
Supervised Learning



Example adapted from Carlini, "Poisoning the Unlabeled Dataset of Semi-Supervised Learning", USENIX Security '21

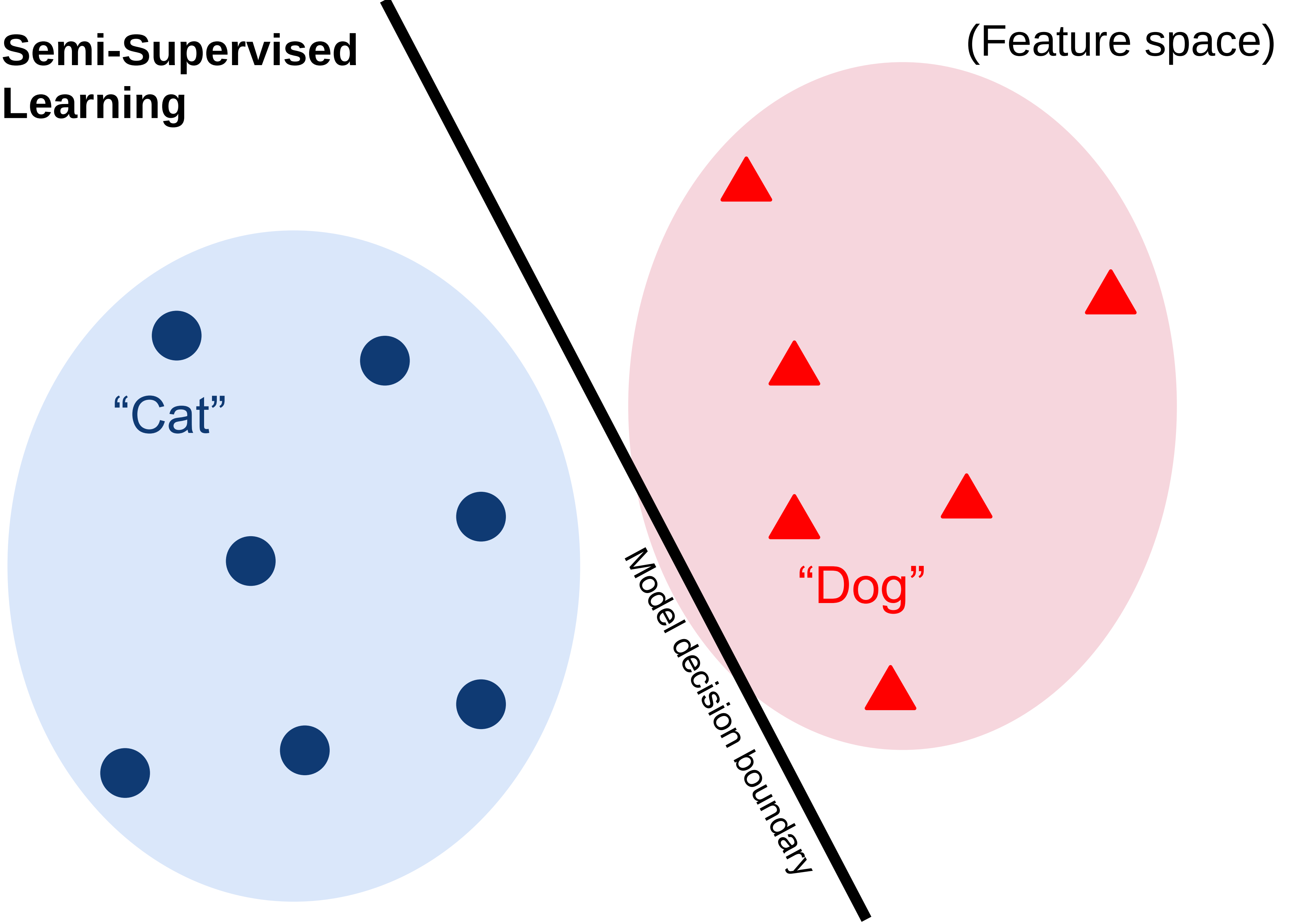
Semi-Supervised Learning

(Feature space)



Example adapted from Carlini, “Poisoning the Unlabeled Dataset of Semi-Supervised Learning”, USENIX Security '21

Semi-Supervised Learning

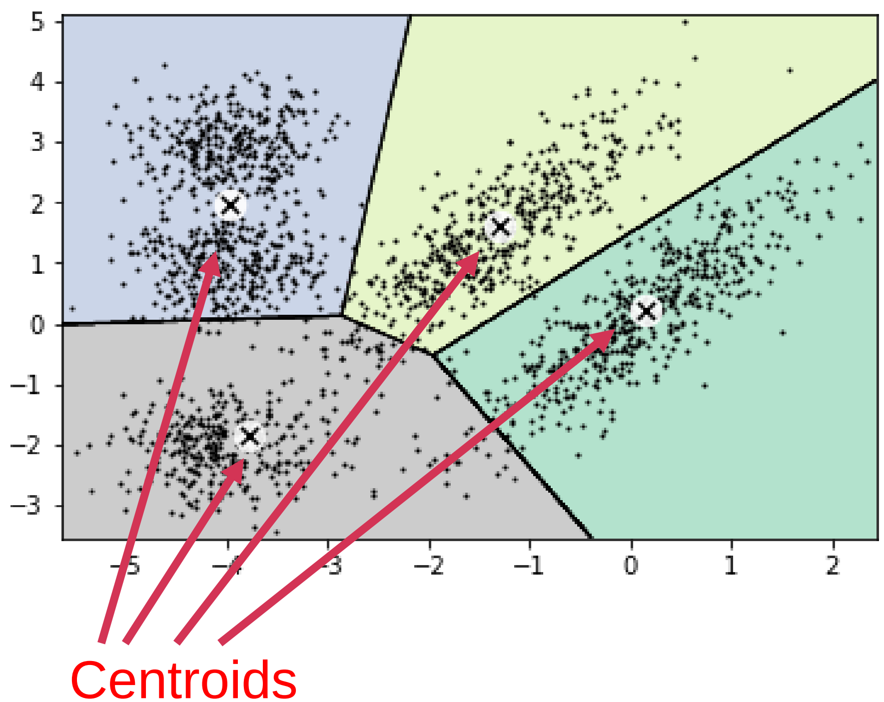


Example adapted from Carlini, "Poisoning the Unlabeled Dataset of Semi-Supervised Learning", USENIX Security '21

Formulation: *K-Means* Clustering

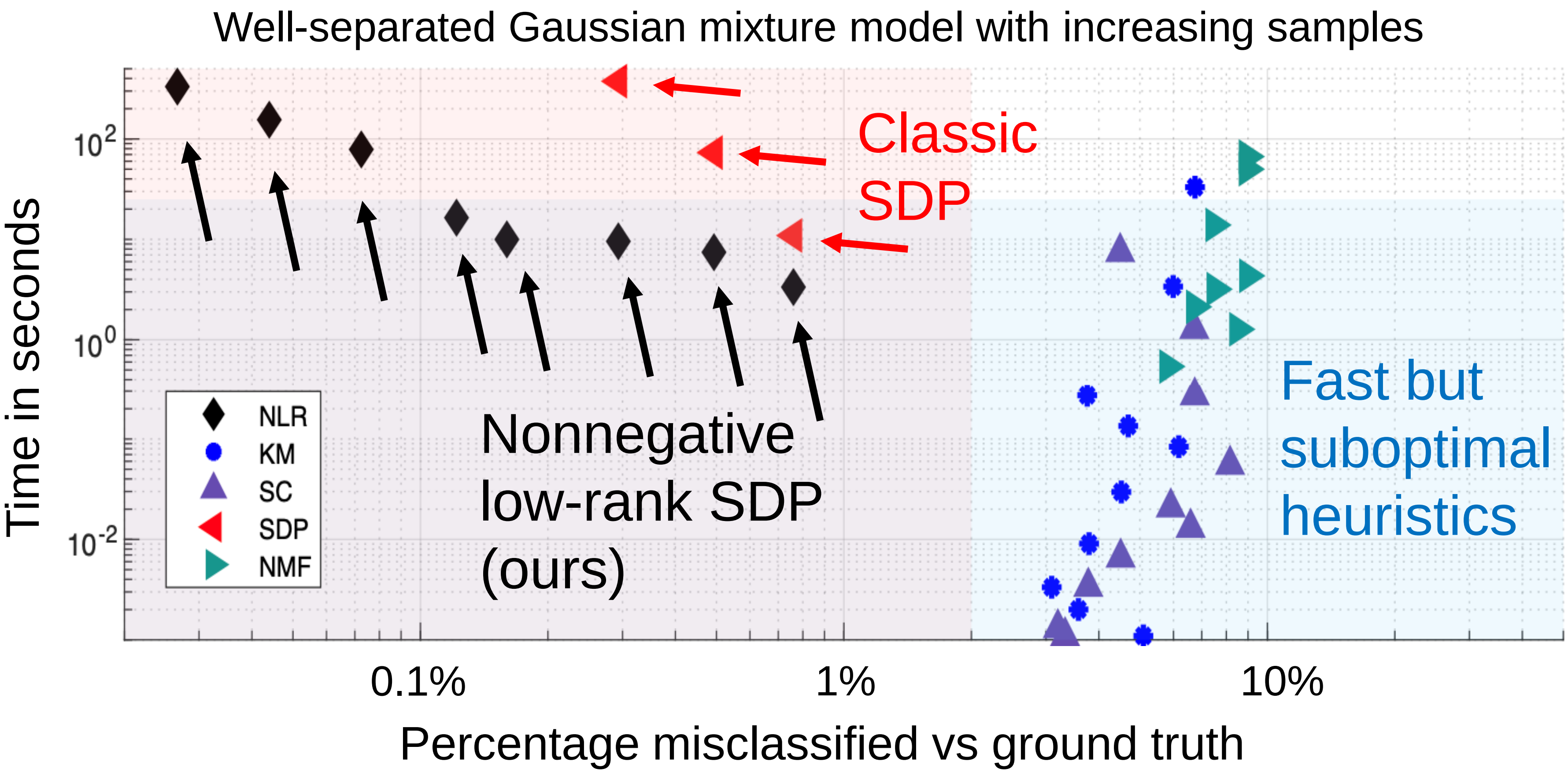
Given data $X_1, \dots, X_n \in \mathbb{R}^d$, divide into K disjoint clusters $G_1, \dots, G_K$, to **minimize distance** between **cluster points** and **cluster centroid**

$$\min_{G_1,\dots,G_K} \quad \sum_{k=1}^{K} \sum_{i \in G_k} \left\| X_i - \frac{1}{|G_k|} \sum_{j \in G_k} X_j \right\|^2 \quad \text{s.t.} \quad \bigsqcup_{k=1}^{K} G_k = [n]$$



	Scalable	Optimal
Lloyd	✓	✗
Spectral	✓	✗
NMF	✓	✗
SDP	✗	✓
NLR (ours)	✓	✓

Contribution: State-of-the-art trade-off between scability and optimality



Oral presentation at ICLR 2024

(one of 85 out of 7262 submissions)

Prelim: Exact reform as low-rank optim

Lemma. Let $X = [X_1, X_2, \dots, X_n]^T$. Then, $i \in G_k^\star \Leftrightarrow U_{i,k}^\star \neq 0$

$$\min_{G_1,\dots,G_K} \sum_{k=1}^{K} \sum_{i \in G_k} \left\| X_i - \frac{1}{|G_k|} \sum_{j \in G_k} X_j \right\|^2 \quad \text{s.t.} \quad \bigsqcup_{k=1}^{K} G_k = [n]$$

$$\max_{U \in \mathbb{R}^{n \times K}} \langle XX^T, UU^T \rangle \quad \text{s.t.} \quad UU^T \mathbf{1}_n = \mathbf{1}_n, \; U^T U = I_K, \; U \geq 0.$$

See Carlson, Mixon, Villar, Ward (2017) or Prasad & Hanasusanto (2018)

Proof.

$$\min_Z \frac{1}{2} \langle D, Z \rangle \quad \text{s.t.} \quad Z = \sum_{k=1}^{K} \frac{1}{|G_k|} \mathbf{1}_{G_k} \mathbf{1}_{G_k}^T \text{ where } D_{i,j} = \|X_i - X_j\|^2$$

$$UU^T = \sum_{k=1}^{K} \frac{1}{|G_k|} \mathbf{1}_{G_k} \mathbf{1}_{G_k}^T \iff UU^T \mathbf{1}_n = \mathbf{1}_n, \; U^T U = I_K, \; U \geq 0$$

$$D = \mathbf{1}^T d + d \mathbf{1}^T - XX^T \text{ where } d = \mathrm{diag}(XX^T) \qquad \square$$

Prior work: Semidefinite Programming

Exact reformulation ($U^{\star}_{i,k} \neq 0$ if and only if $i \in G^{\star}_{k}$)

$$\max_{U \in \mathbb{R}^{n \times K}} \langle XX^T, UU^T \rangle \quad \text{s.t.} \quad UU^T \mathbf{1}_n = \mathbf{1}_n,\ U^T U = I_K,\ U \geq 0.$$

SDP relaxation of Peng & Wei 2007 ($Z = UU^T, U \geq 0$ implies $Z \succcurlyeq 0, Z \geq 0$)

$$\leq \quad \max_{Z \succeq 0} \quad \langle XX^T, Z \rangle \quad \text{s.t.} \quad Z\mathbf{1}_n = \mathbf{1}_n, \quad \mathrm{tr}(Z) = K, \quad Z \geq 0$$

Theorem (Chen & Yang 2021).
Gaussian mixture: $X_i = \mu_k + \epsilon_i$ where $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$.
Centroid separation: $\Theta = \min_{k \neq k'} \|\mu_k - \mu_{k'}\|$.
If $\Theta < \overline{\Theta}$, impossible to exactly recover $G^{\star}_1, \dots, G^{\star}_K$
If $\Theta > \overline{\Theta}$, SDP is tight, perfectly recovers $G^{\star}_1, \dots, G^{\star}_K$

Not scalable: Optimize $n \times n$ matrix over n^2 inequalities

Prior work: Nonneg matrix factorization

Exact reformulation ($U^\star_{i,k} \neq 0$ if and only if $i \in G^\star_k$)

$$\max_{U \in \mathbb{R}^{n\times K}} \langle XX^T, UU^T\rangle \quad \text{s.t.} \quad UU^T\mathbf{1}_n = \mathbf{1}_n,\; U^TU = I_K,\; U \geq 0.$$

Substitute $\|XX^T - UU^T\|_F^2 = \|XX^T\|_F^2 + \|UU^T\|_F^2 - 2\langle XX^T, UU^T\rangle$ and relax

$$\leq \quad \frac{1}{2}\left(\|XX^T\|_F^2 + K\right) - \boxed{\min_{U \in \mathbb{R}_+^{n\times r}} \quad \frac{1}{2}\|XX^T - UU^T\|_F^2}$$

Scalable: Proj gradient descent easily scales to $n = 10^6$.

Rank overparameterization: Empirically, fewer spur loc min as search rank r increases; compare with Zhang (2022).

Not optimal: Relaxed constraints are critical for exact recovery.

Critical question: How to design algorithm that is as scalable as NMF, but as optimal as SDP?

Proposed formulation

Exact reformulation of K-means clustering

$$\max_{U \in \mathbb{R}^{n \times K}} \quad \langle XX^T, UU^T \rangle \quad \text{s.t.} \quad UU^T \mathbf{1}_n = \mathbf{1}_n, \quad U^T U = I_K, \quad U \geq 0.$$

Proposed nonnegative low-rank SDP relaxation

$$\leq \quad \max_{U \in \mathbb{R}^{n \times r}} \quad \langle XX^T, UU^T \rangle \quad \text{s.t.} \quad UU^T \mathbf{1}_n = \mathbf{1}_n, \quad \underline{\operatorname{tr}(UU^T) = K}, \quad U \geq 0$$

Classical SDP relaxation ($Z = UU^T, U \geq 0$ implies $Z \succcurlyeq 0, Z \geq 0$)

$$\leq \quad \max_{Z \succeq 0} \quad \langle XX^T, Z \rangle \quad \text{s.t.} \quad Z\mathbf{1}_n = \mathbf{1}_n, \quad \operatorname{tr}(Z) = K, \quad Z \geq 0$$

Optimal: At least as tight as SDP, which is already provably tight.

Scalable: Proj gradient descent easily scale to $n = 10^6$.

Proposed Algorithm

$$\mathrm{proj}_{\Omega}(U) = \frac{\sqrt{K}\cdot\max\{U,0\}}{\|\max\{U,0\}\|_F}$$

$$\max_{U\in\mathbb{R}^{n\times r}} \quad \langle XX^T, UU^T\rangle \quad \text{s.t.} \quad \underbrace{UU^T\mathbf{1}_n = \mathbf{1}_n,}_{\text{Hard to enforce}} \quad \underbrace{\mathrm{tr}(UU^T) = K, \quad U \geq 0}_{\text{Closed-form projection}}$$

First try, relax difficult constraint into quadratic penalty

$$\min_{U\in\Omega} \quad \langle -XX^T, UU^T\rangle + \frac{\beta}{2}\|UU^T\mathbf{1}_n - \mathbf{1}_n\|^2$$

$$\text{where } \Omega \overset{\text{def}}{=} \{\mathbb{R}^{n\times r} : U \geq 0,\ \|U\|_F = \sqrt{K}\}$$

Easily solved using proj grad desc, but no perfect recovery.

Better idea, enforce using **augmented Lagrangian method**

$$U \leftarrow \arg\min_{U\in\Omega} \quad \langle -XX^T, UU^T\rangle + \langle y, UU^T\mathbf{1}_n - \mathbf{1}_n\rangle + \frac{\beta}{2}\|UU^T\mathbf{1}_n - \mathbf{1}_n\|^2$$

$$y \leftarrow y + \beta(UU^T\mathbf{1}_n - \mathbf{1}_n)$$

Solve primal using proj grad desc, update dual, repeat.
Combined algorithm is like NMF; five lines of code.

Main theoretical result:
NLR + ProjGD → Primal-dual local linear convergence, even with rank overparam

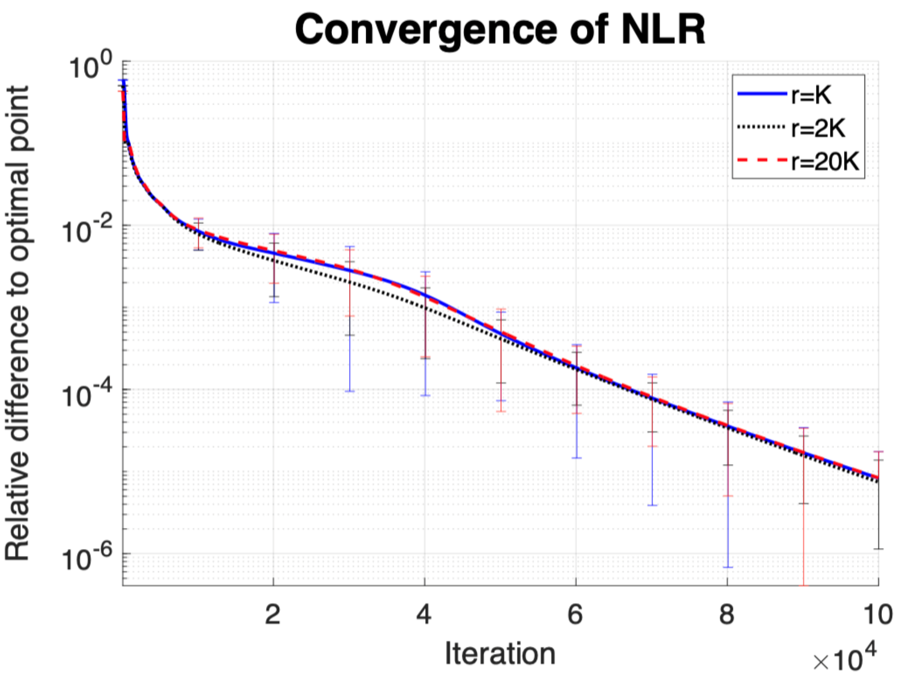
Theorem [Zhuang, Chen, Yang, Zhang, 2024]

Assume Gaussian Mixtures.

(Initialization) If U^0 within an $O(1)$ neighborhood of the optimal solution U^*

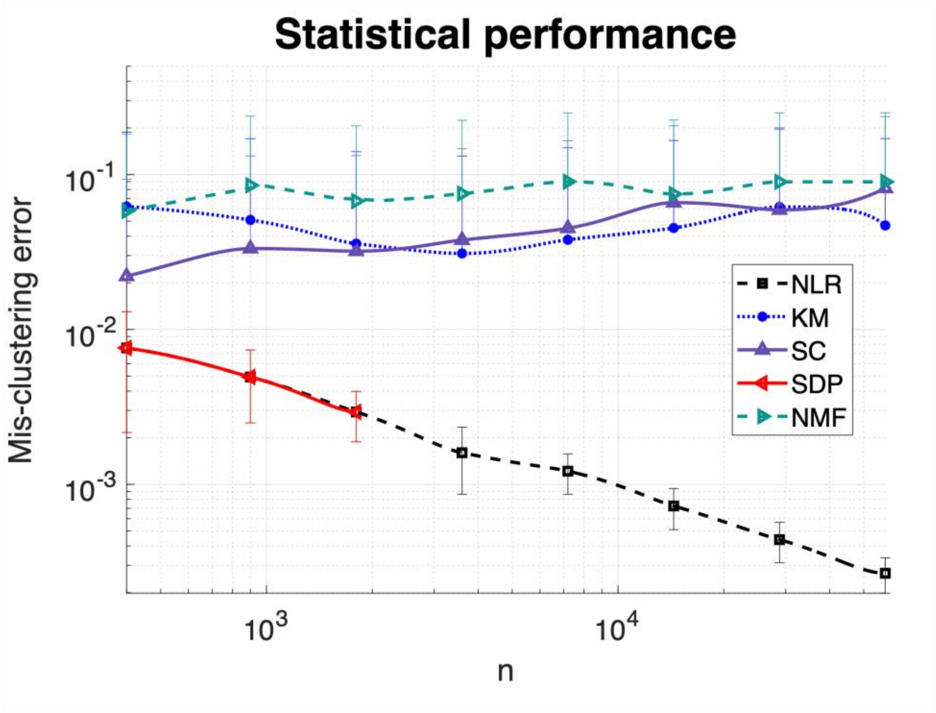
(Search rank) and $r \geq K$,

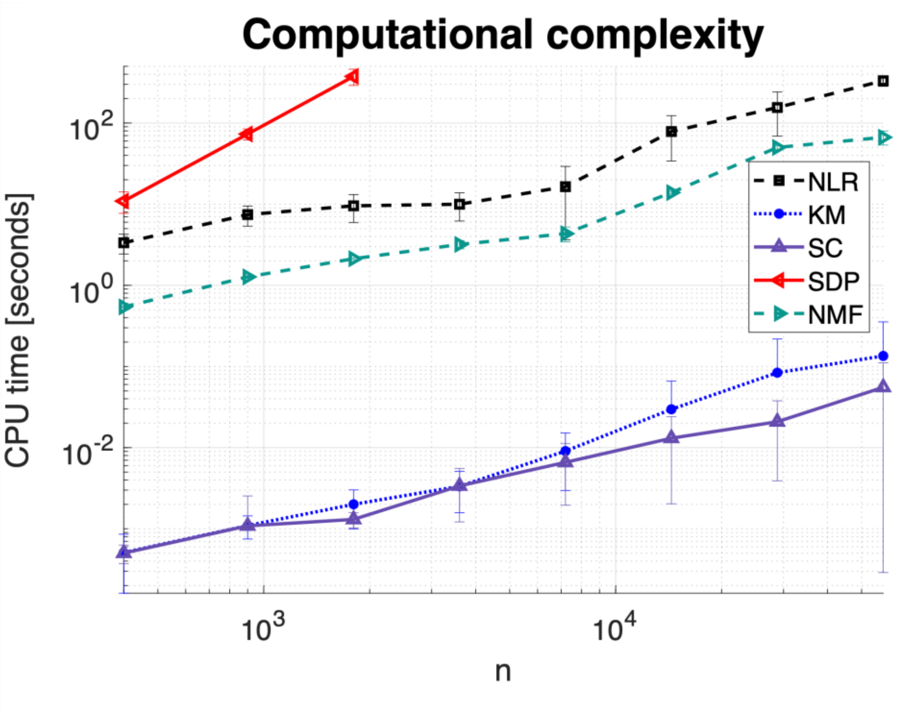
Then U^t converges to U^* at a **linear rate**.



Overall time complexity of NLR: $O(nrK^6)$

Validation on synthetic data



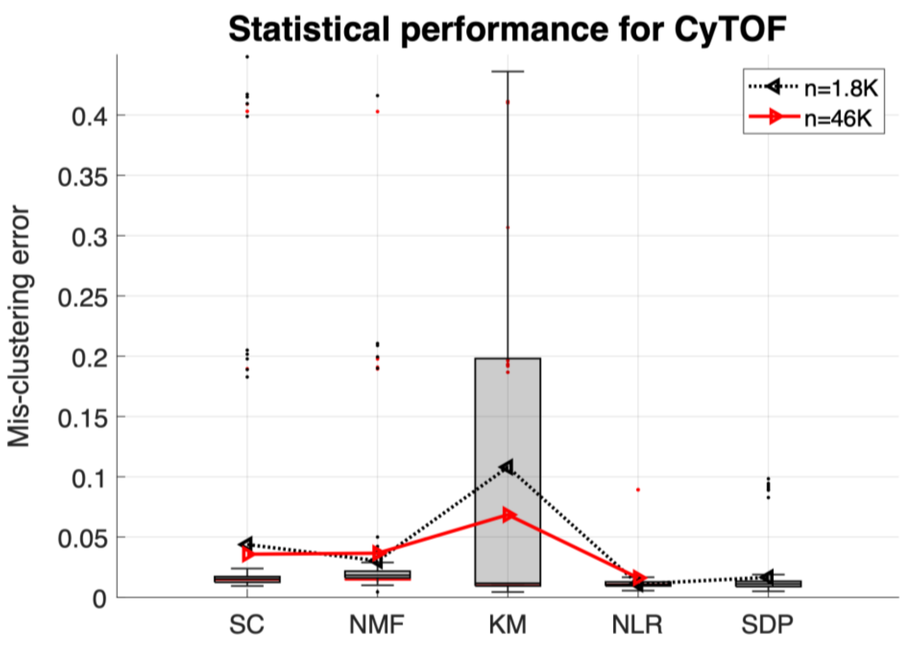


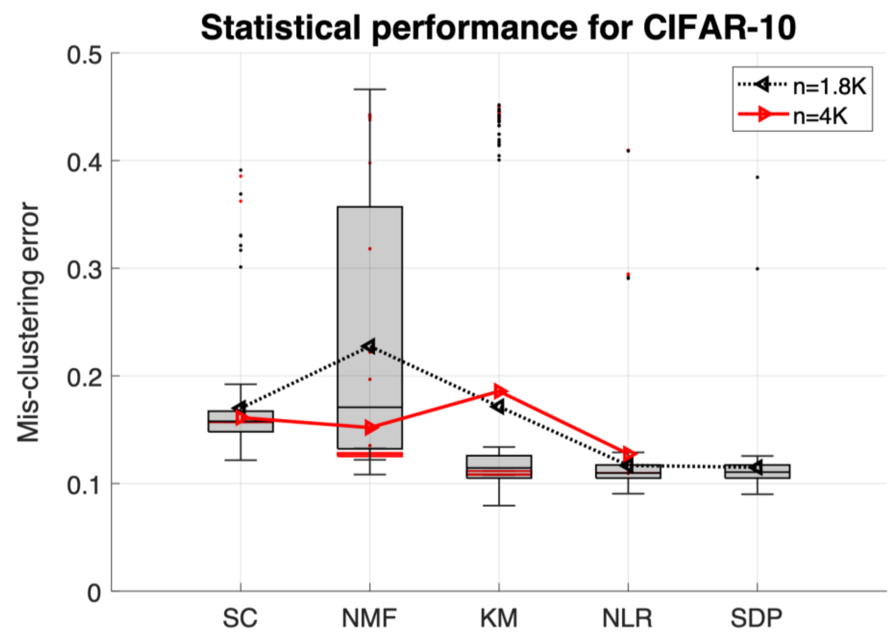
- Best performance with SDP and NLR, error goes to zero as n increases.
- SDP and NLR have similar performance, but SDP cannot scale past n=2000.
- Compute time of NLR and KM/SC/NMF scale linearly to sample size n.

Validation on real-world data

Mass Cytometry (CyTOF) dataset
Sample size n=1800 and n=46258

CIFAR-10 dataset
(color images of size 32x32x3)
Sample size n=1800 and n=4000





- SDP and NLR are similarly optimal and consistent, but only NLR scales.
- KM and NMF can be optimal, but inconsistent between datasets and trials.
- Spectral clustering works well, but SDP and NLR are provably tighter.

Conclusions - Thank you!

- Various approximations and relaxations for K-means clustering: Lloyd, spectral, nonnegative matrix factorization (NMF), semidefinite programming (SDP).
- SDP achieves sharp information-theoretical threshold for exact recovery.
- **Goal: computational scalability and statistical optimality.**
- **This paper: an algorithm simultaneously achieving O(n) per iteration complexity + local linear convergence + same SDP recovery guarantee.**
- Future work: Partial recovery? Optimization landscape?

C3.ai Digital Transformation Institute