A Quest for Structure: Joint Graph Structure & Semi-Supervised Inference

Leman Akoglu

Joint work with Xuan Wu and Lingxiao Zhao

NetSci 2018 Satellite on
Statistical Inference in Network Models

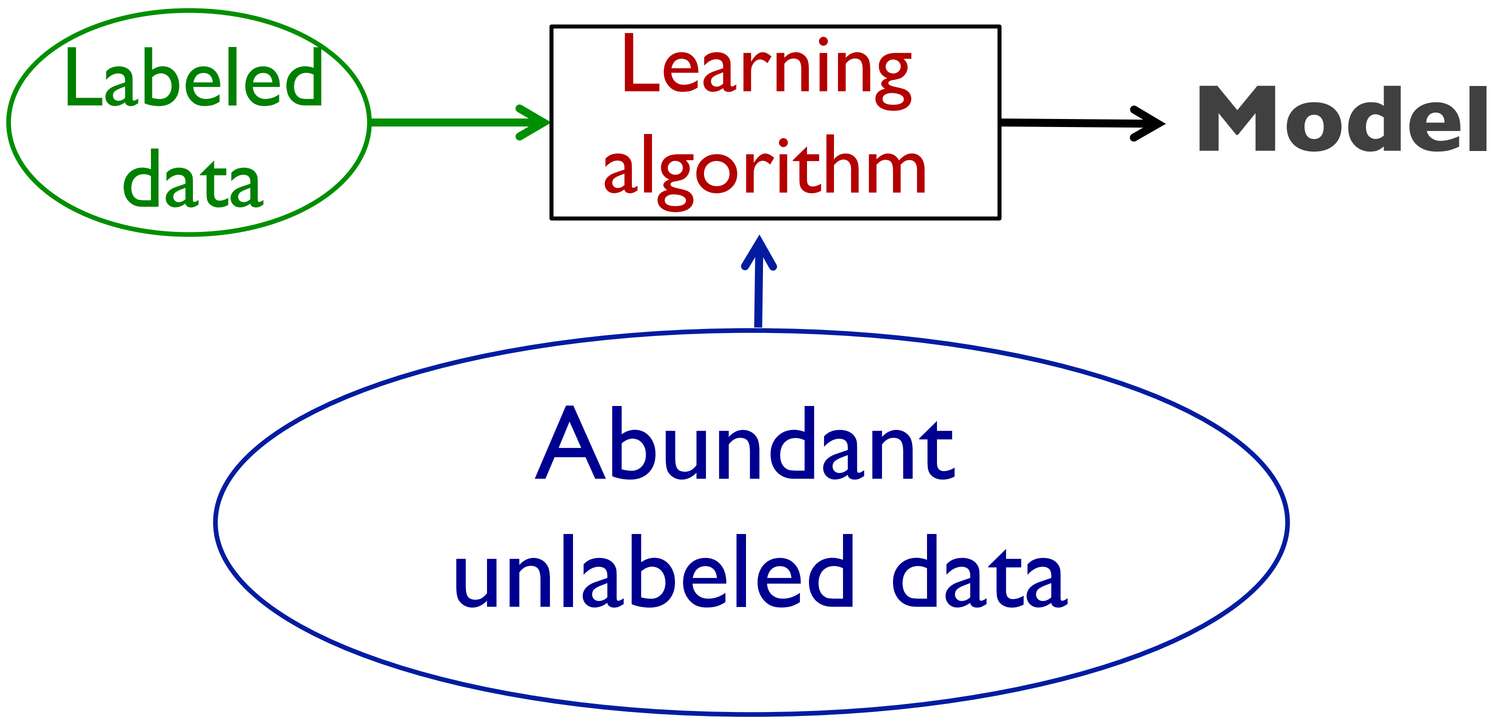
June 11, 2018

Carnegie Mellon University
Heinz College

This Talk

- Semi-Supervised Learning: Intro
- Graph-based SSL
  - Formulation & Solution
- The Quest for Structure

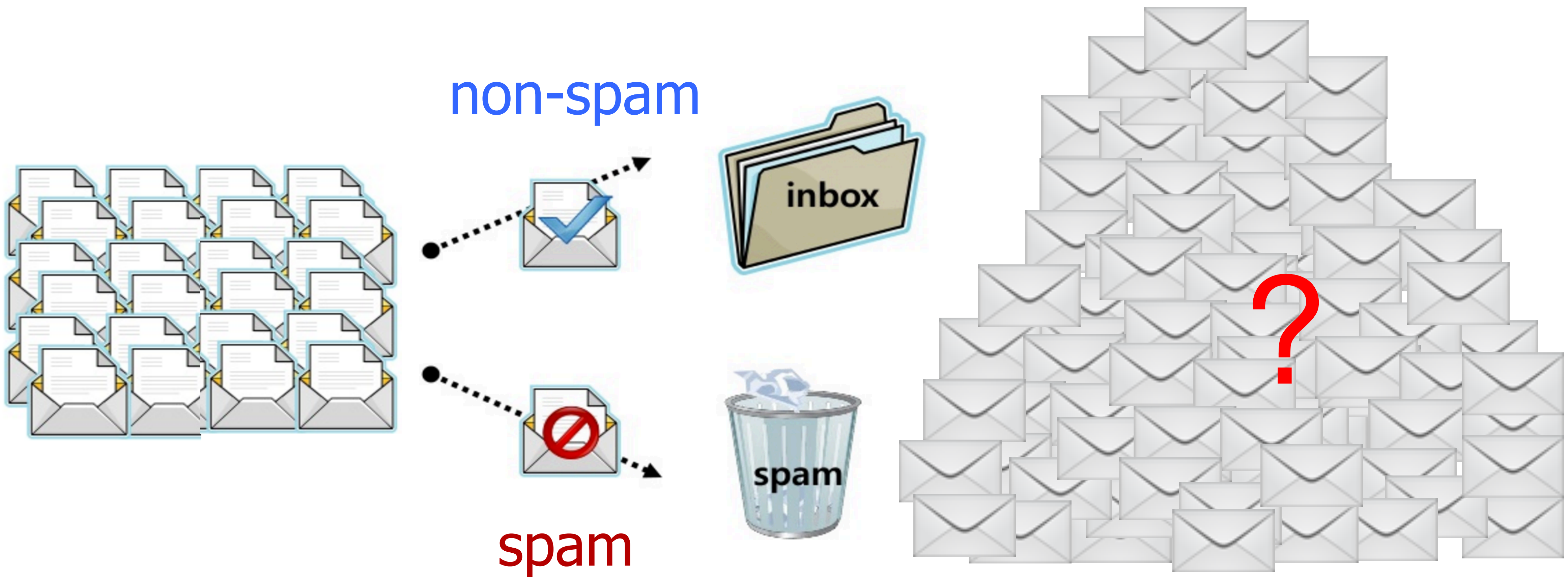
Semi-Supervised Learning: Motivation



- Labeling/annotation is expensive
  - Small amount of labeled data
  - Large amount of unlabeled data

Semi-Supervised Learning: Examples

Spam filtering



labeled data

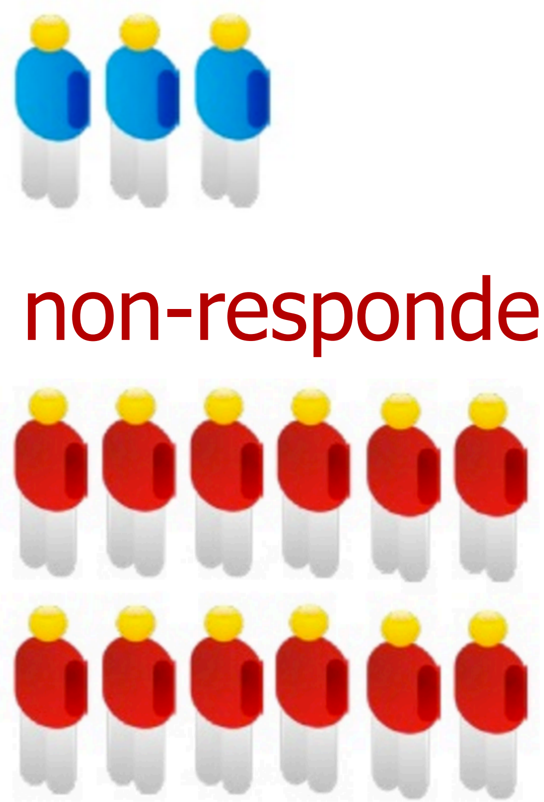
unlabeled data

Semi-Supervised Learning: Examples

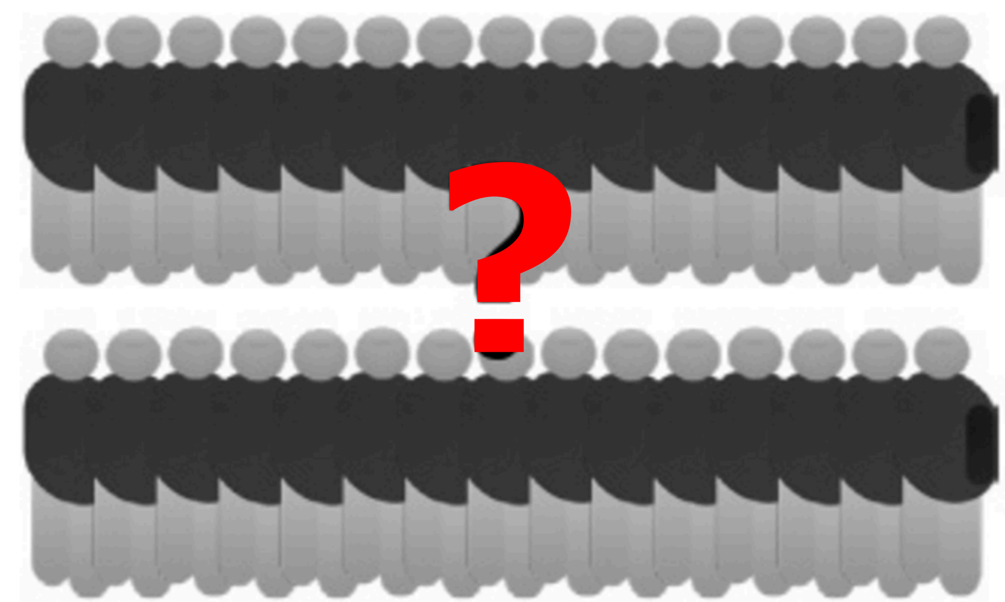
Response modeling

respondents

non-respondents



labeled data



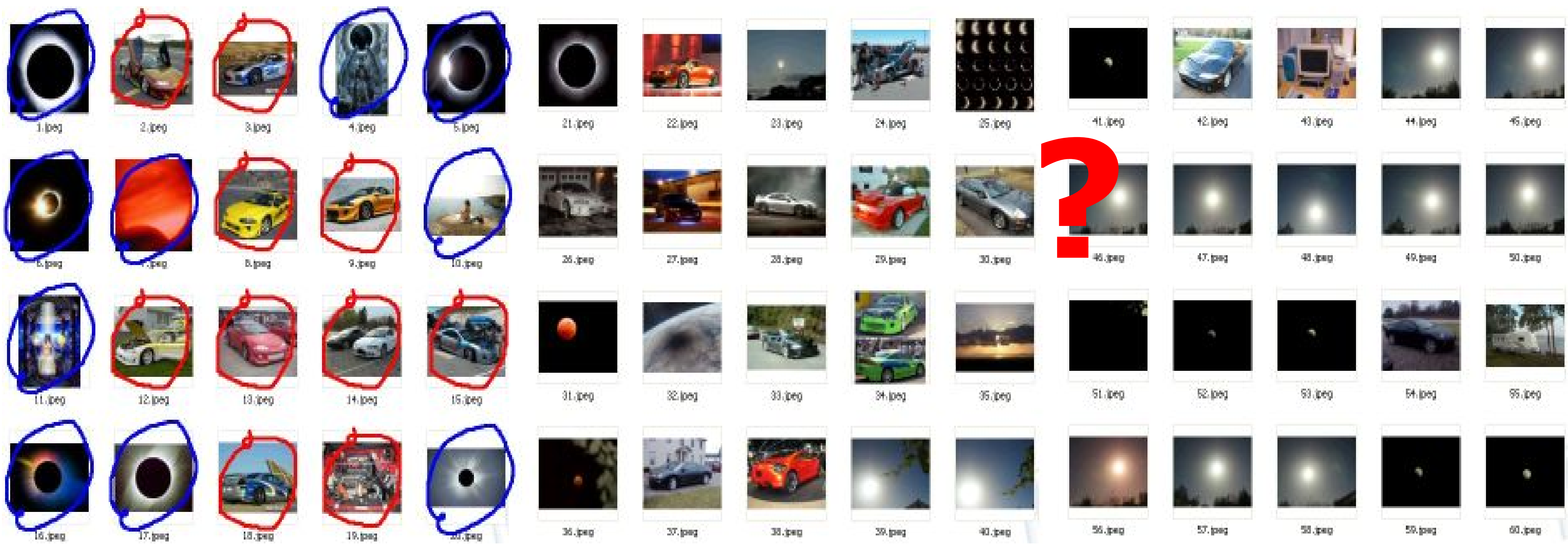
unlabeled data

Semi-Supervised Learning: Examples

Image classification

Semi-Supervised Learning: Examples

Image classification

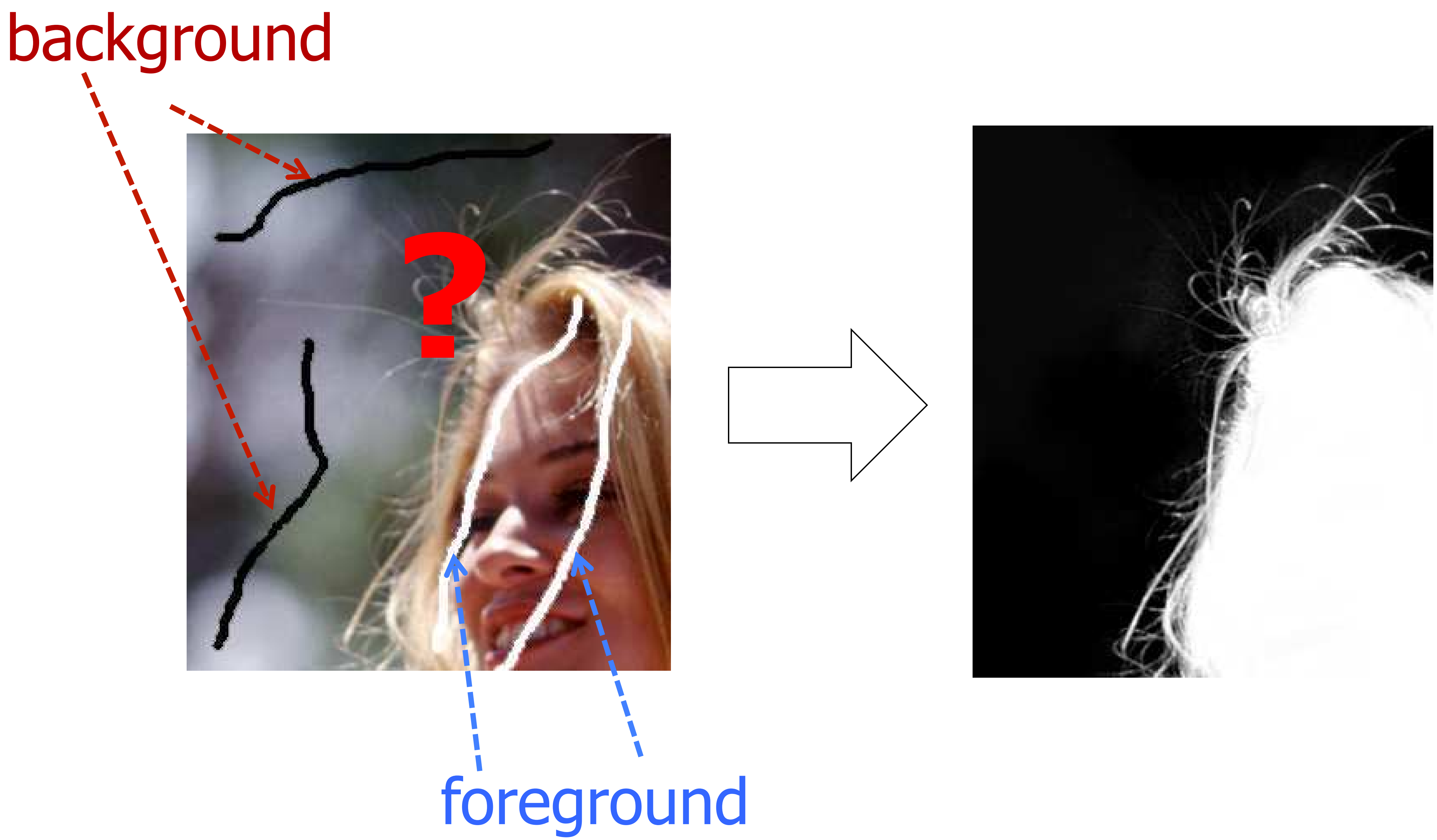


labeled data

unlabeled data

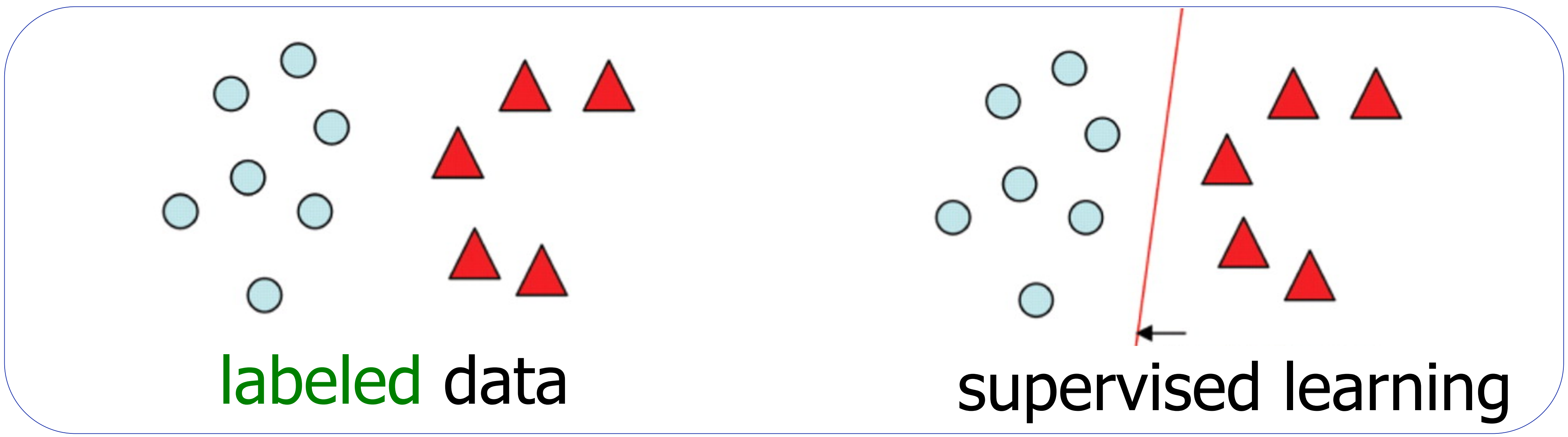
Semi-Supervised Learning: Examples

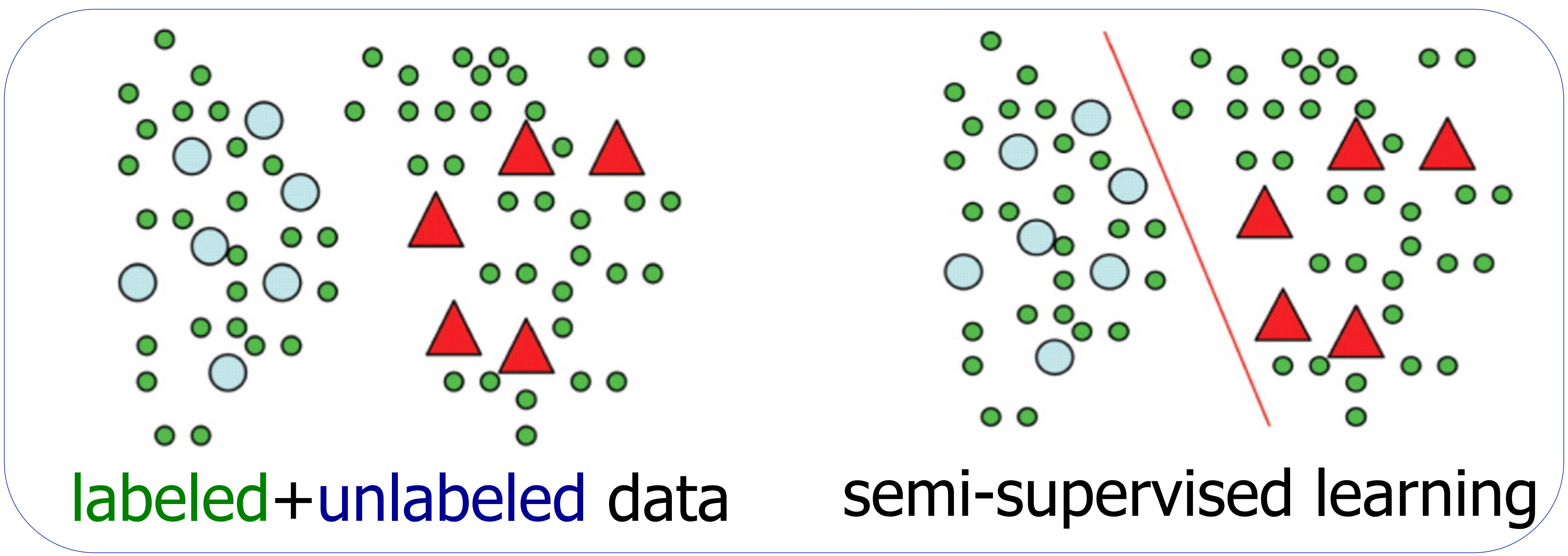
Image segmentation



Classification with Unlabeled Data

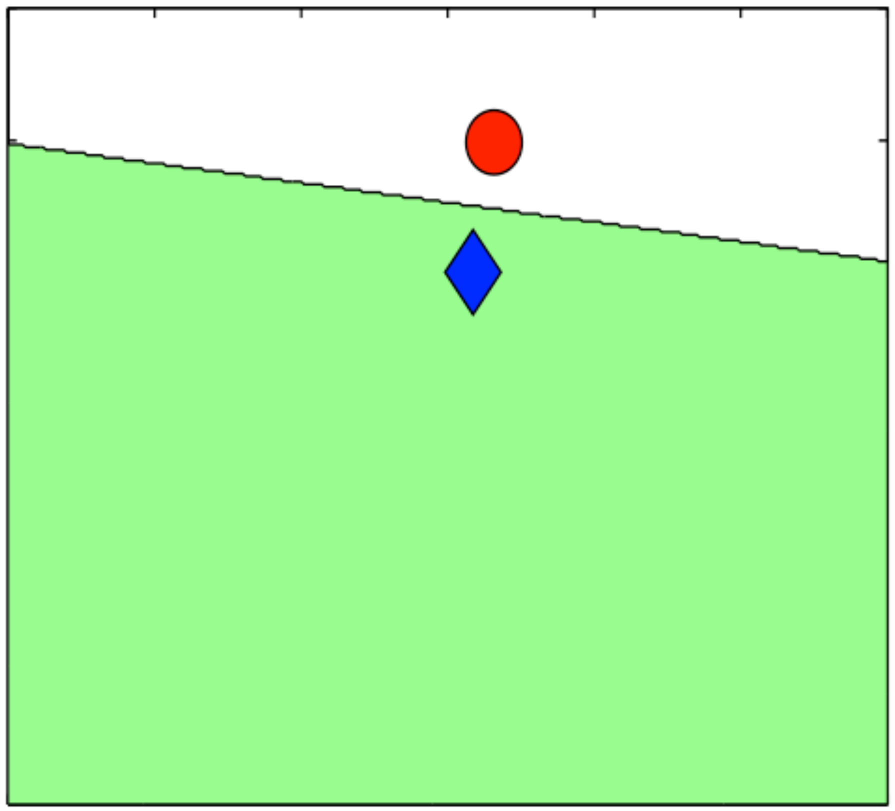
- Why should unlabeled data be helpful?



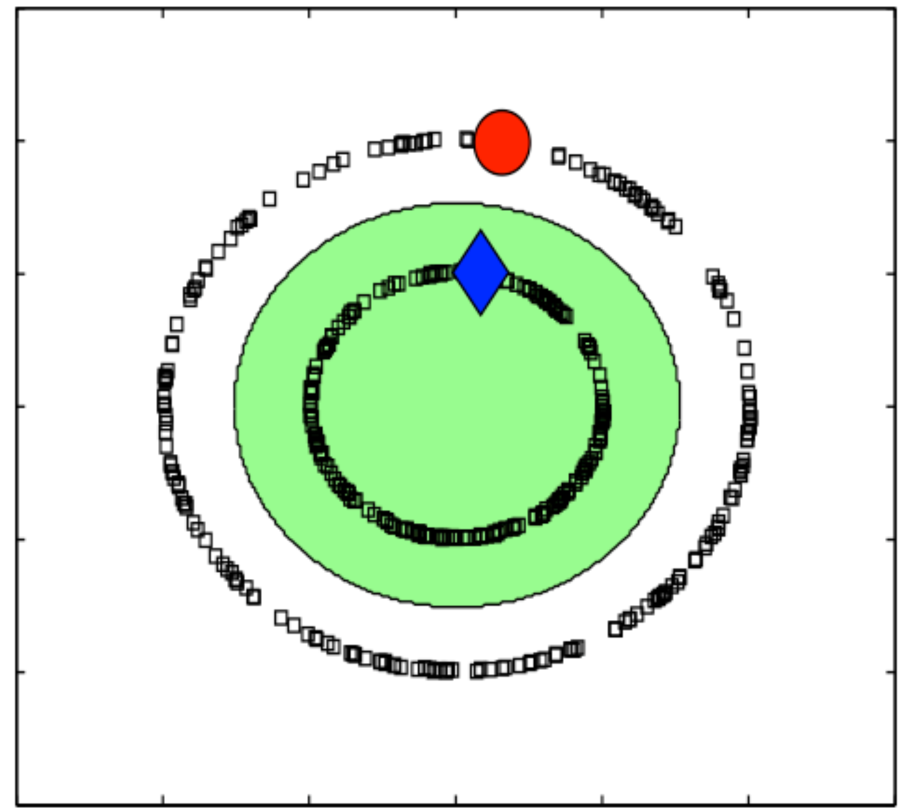


Classification with Unlabeled Data

- Why should unlabeled data be helpful?



supervised learning
w/ labeled data

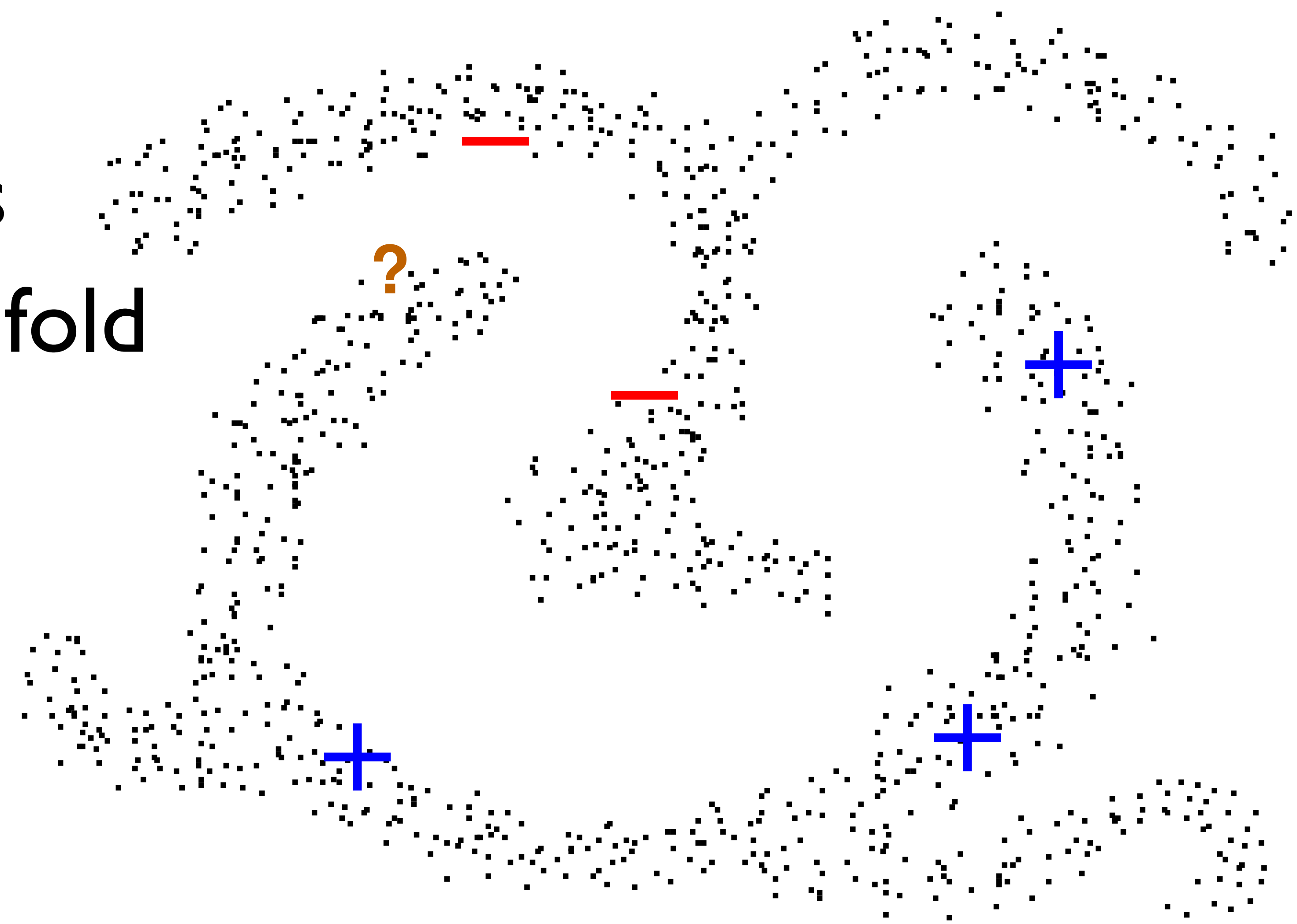


semi-supervised learning
w/ labeled+unlabeled data

[Belkin+ JMLR 2006]

Classification with Unlabeled Data

- Working assumption: there is information in **data distribution**
  - data form clusters
  - data fall on a manifold



- Intuition: locally **similar points** have **similar labels**

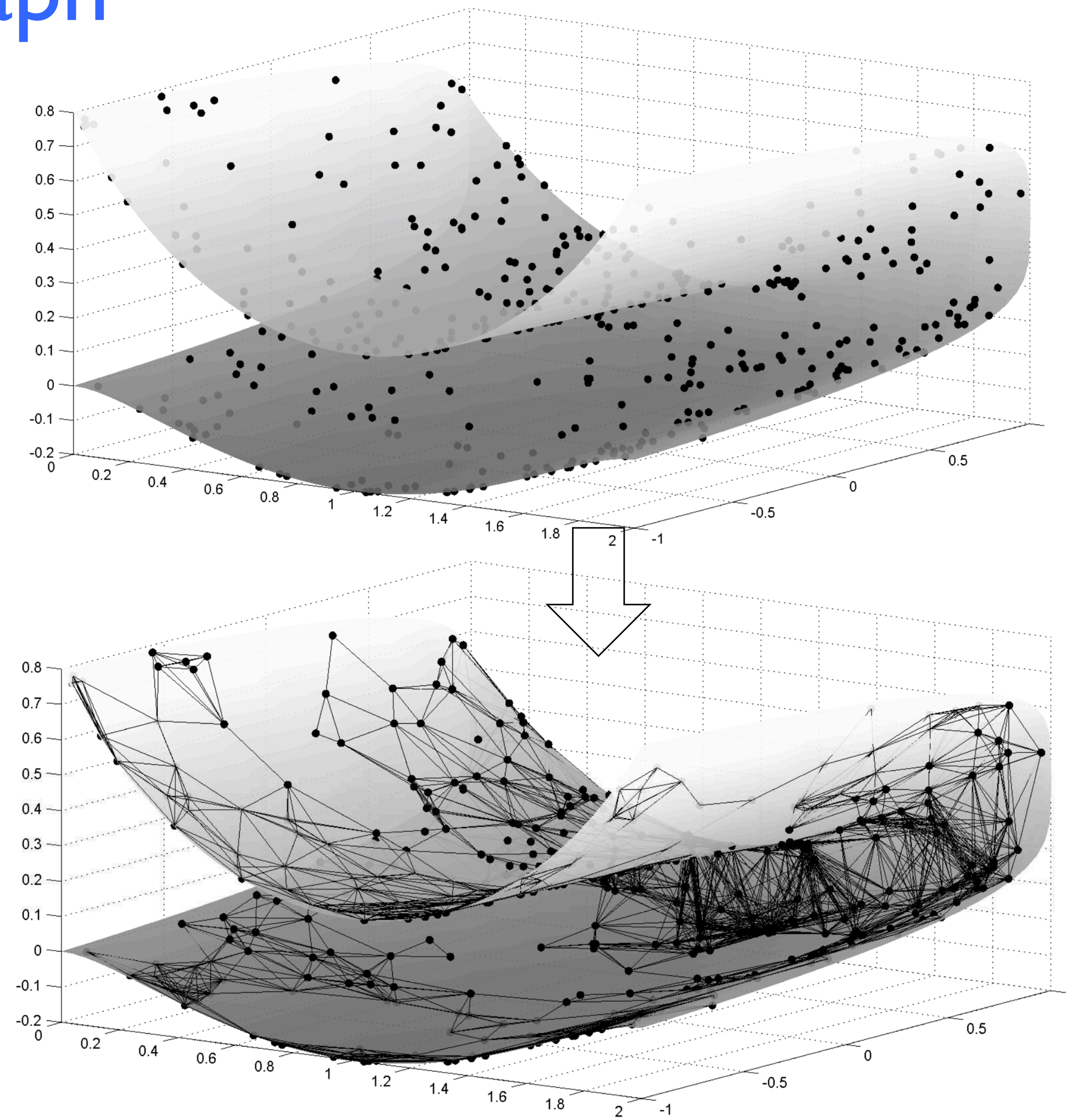
  → **homophily (autocorrelation)**

This Talk

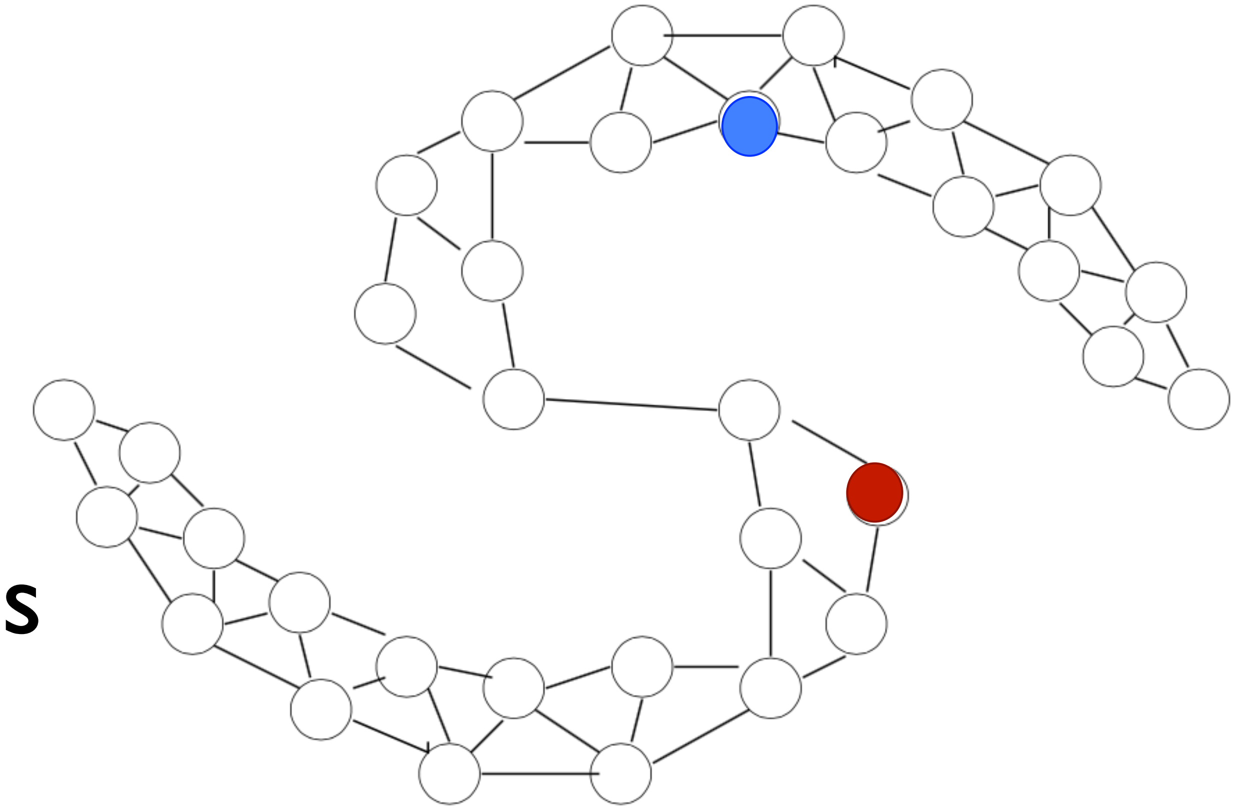
- Semi-supervised learning: Intro
- Graph-based SSL
  - Formulation & Solution
- The Quest for Structure

Graph-based SSL

- Approach: use a graph
  - to approximate the data manifold
  - by connecting similar points



Graph-based SSL: The Problem



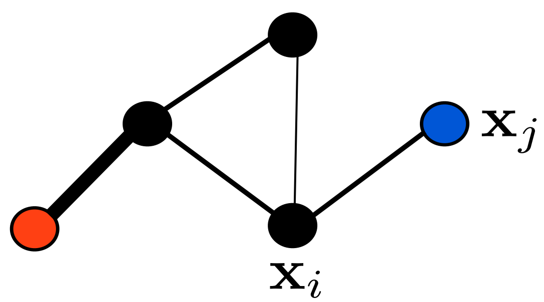
- **Given**
  - a graph with adjacency **W**
  - set *L* of labeled nodes
  - set *U* of **unlabeled** nodes

$$T = L \cup U$$

- **Assign** binary labels to $u \in U$

$$y_u \in \{-1, 1\}$$

Graph-based SSL: Formulations



[Zhu, Ghahramani, Lafferty 2002]

$$\boxed{\arg\min_{f \in \mathbb{R}^n,\ f_L = Y_L} \sum_{i,j \in T}^{n} w_{ij}(f_i - f_j)^2}.$$

[Belkin and Niyogi 2003]

$$\boxed{\arg\min_{f \in \mathbb{R}^n} \sum_{i \in L} (y_i - f_i)^2 + \lambda \sum_{i,j \in T} w_{ij}(f_i - f_j)^2}.$$

[Zhou, Bousquet, Lal, Weston and Schoelkopf 2003]

$$\boxed{\arg\min_{f \in \mathbb{R}^n} \sum_{i \in T} (y_i - f_i)^2 + \lambda \sum_{i,j \in T} w_{ij} \left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2},$$

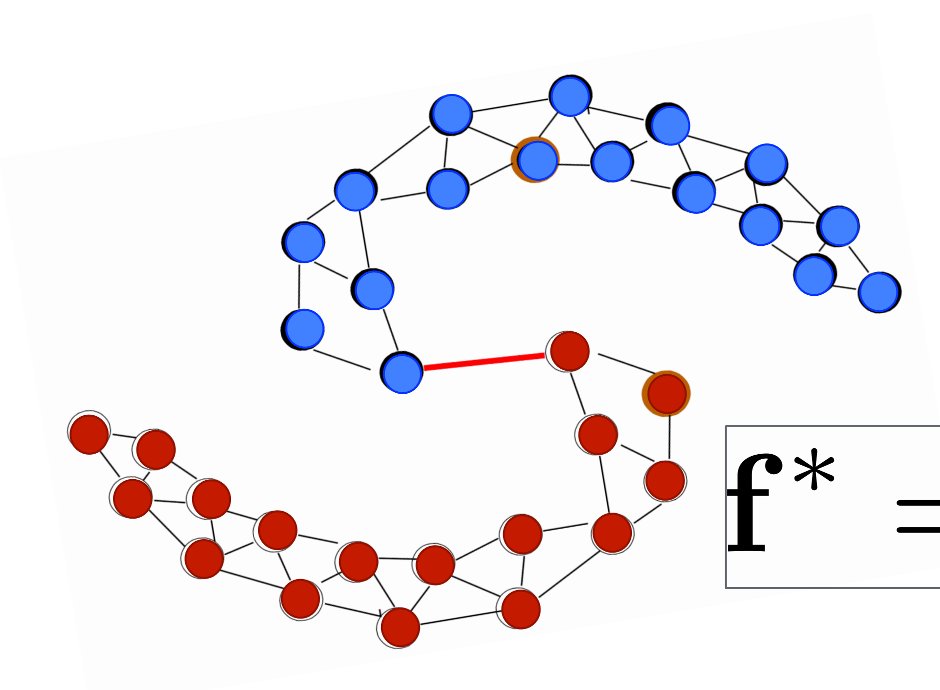
where $y_i = 0$ if $i \in U$.

Graph-based SSL: Solution

$$\boxed{\arg\min_{f\in\mathbb{R}^n} \sum_{i\in T}(y_i - f_i)^2 + \lambda \sum_{i,j\in T} w_{ij}\left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}}\right)^2},$$

$$\Downarrow$$

$$\boxed{\arg\min_{\mathbf{f}} \|\mathbf{f}-\mathbf{y}\|_2^2 + \alpha \mathbf{f}^T\mathbf{L}\mathbf{f}}$$



$$\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$$

is the normalized graph Laplacian

$$D := diag(W\mathbf{1}_n)$$

$$\Downarrow$$

$$\boxed{\mathbf{f}^* = (\mathbf{I} + \alpha\mathbf{L})^{-1}\mathbf{y}}$$

Graph-based Multi-class SSL:

$$\boxed{\arg\min_{F \in \mathbb{R}^{n\times c}} tr((F-Y)^T(F-Y)+\alpha F^T L F)}$$

Objective function

$$\boxed{\mathbf{F}^* = (\mathbf{I}+\alpha\mathbf{L})^{-1}\mathbf{Y}}$$

Closed-form solution

$$\boxed{F^{(t+1)} \leftarrow \mu P F^{(t)} + (1-\mu)Y}$$

Iterative solution

$$P = D^{-1/2} W D^{-1/2}$$

$$\mu = \frac{\alpha}{1+\alpha}$$

This Talk

- Semi-supervised learning: Intro
- Graph-based SSL
  - Formulation & Solution
  - **What graph should one use?**
- The Quest for Structure

Graph Construction Matters

- Choice of the similarity measure has considerable effect on clustering and outlier detection.

  Influence of Graph Construction on Graph-based Clustering Measures. Markus Maier, Ulrike von Luxburg, and Matthias Hein. NIPS 2008.

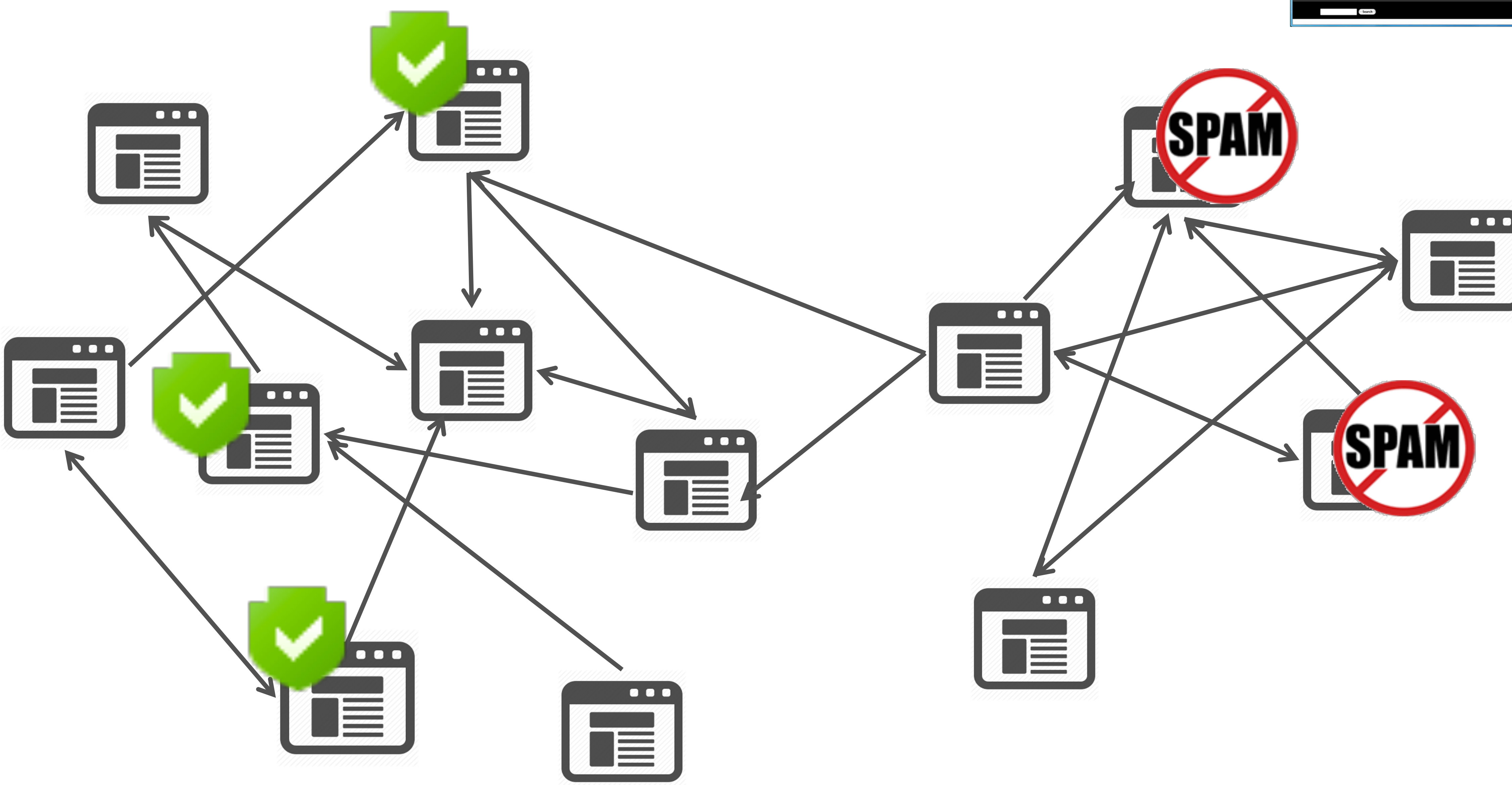
- SSL is no exception!

  "SSL algorithms are strongly affected by the graph sparsification parameter value and the choice of the adjacency graph construction and weighted matrix generation methods."

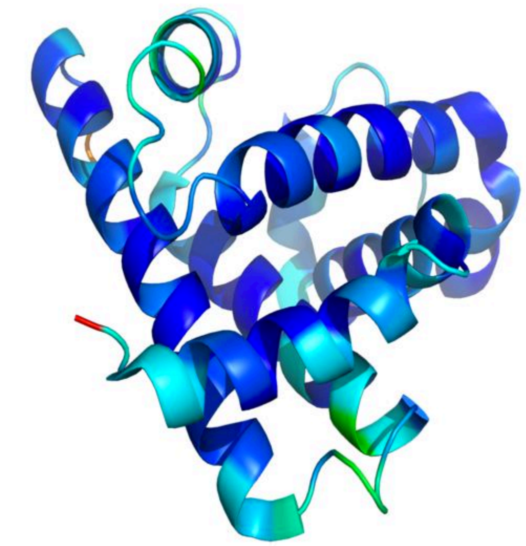
  Influence of Graph Construction on Semi-supervised Learning. Celso Andre R. de Sousa, Solange O. Rezende, Gustavo E. A. P. A. Batista. ECML/PKDD 2013.

Graph-based SSL: examples

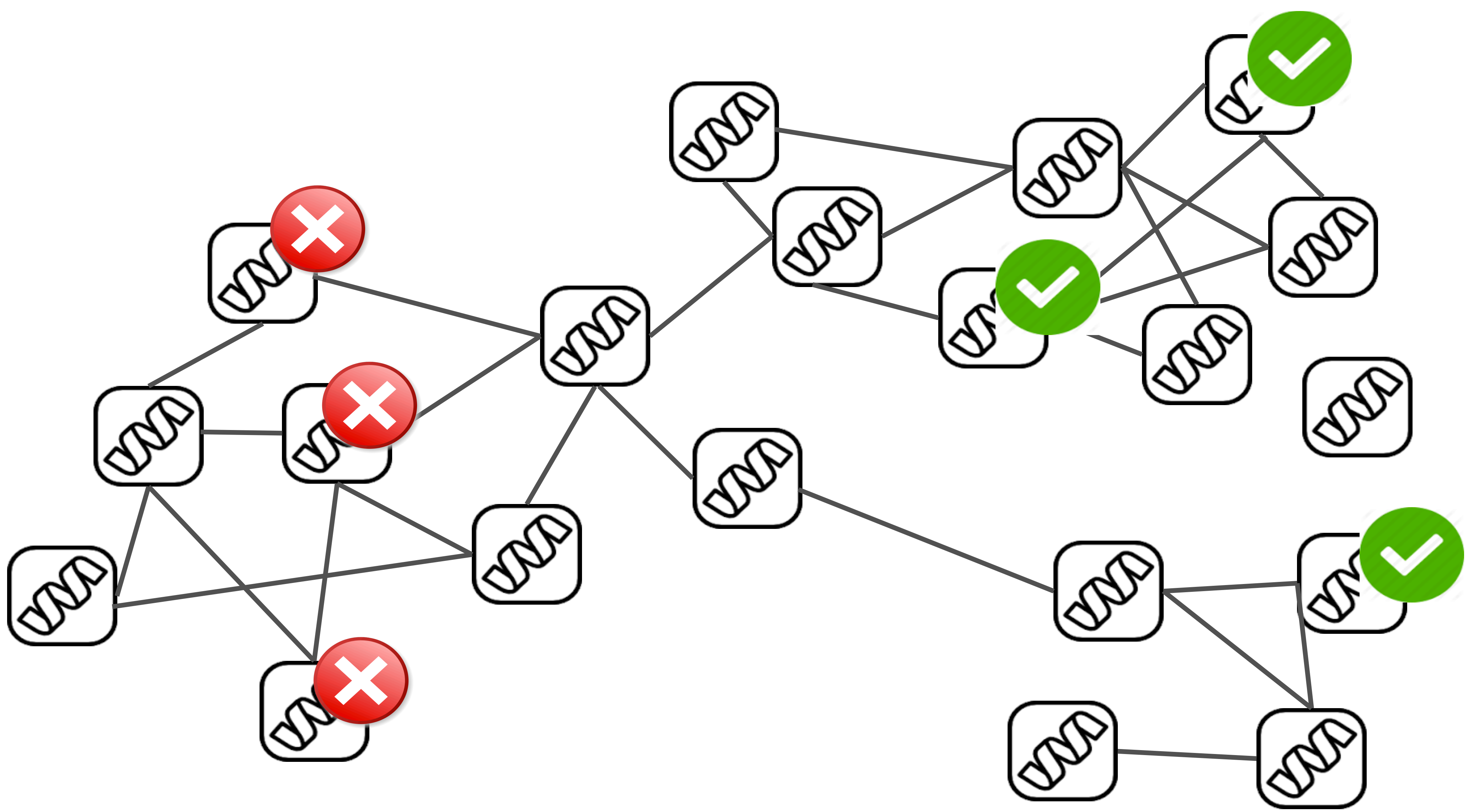
- Sometimes data is naturally a graph …
  - Graph: Web hyperlinks
  - Task: Spam page detection



Graph-based SSL: examples

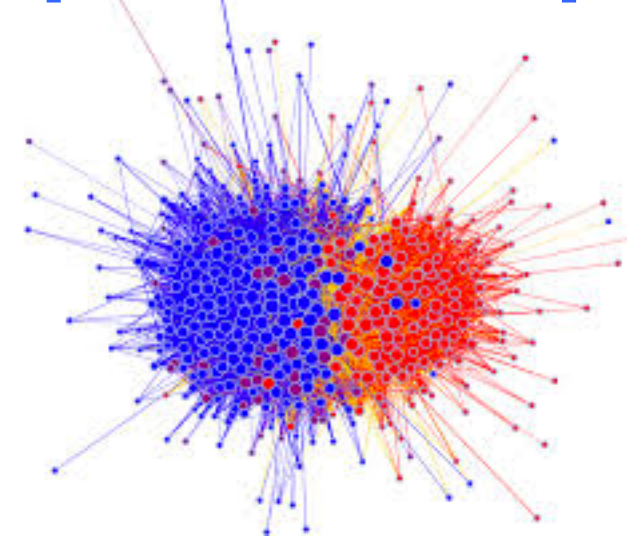


- Sometimes data is naturally a graph …
  - Graph: Protein interactions
  - Task: Protein function prediction

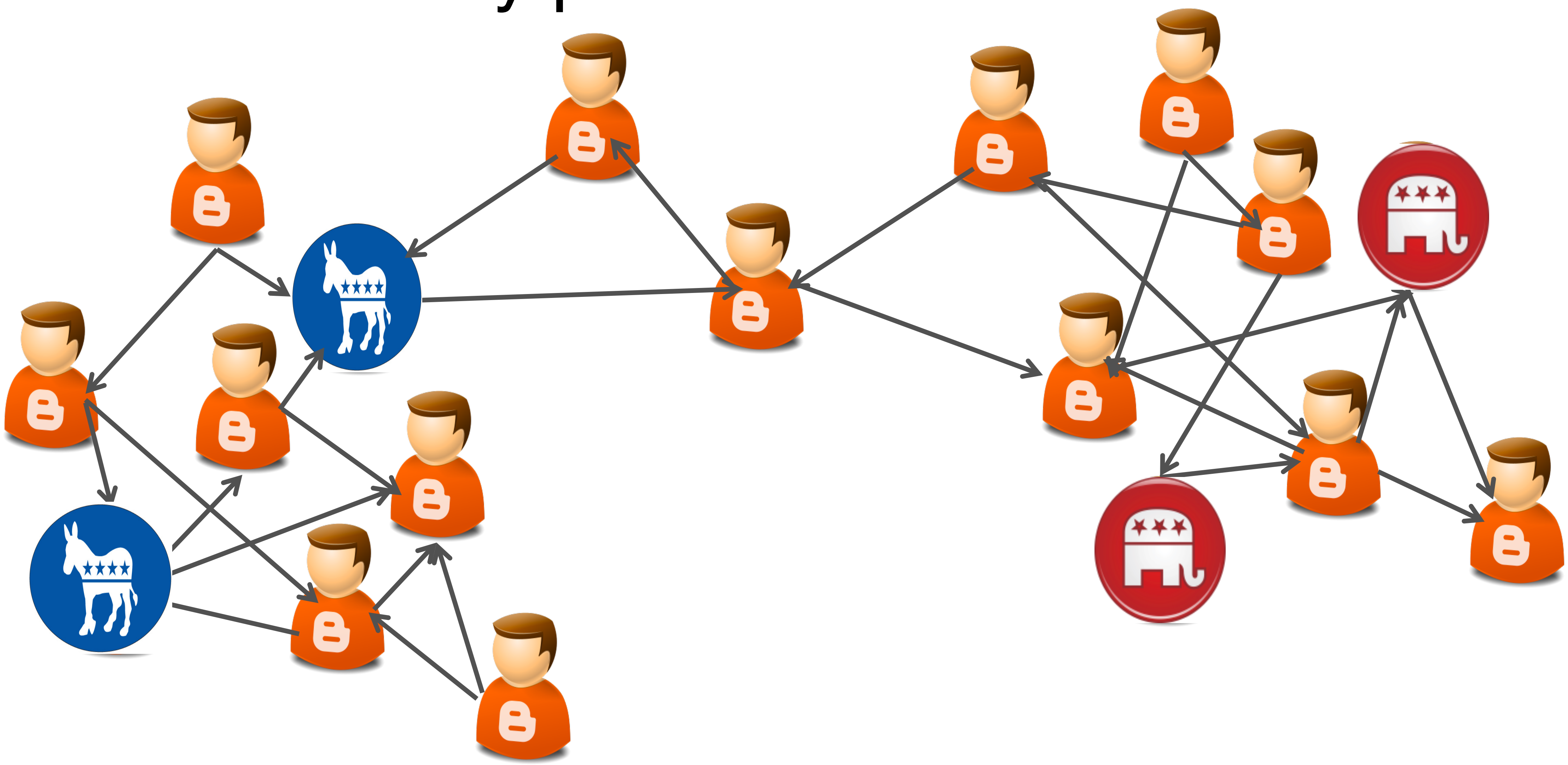


Graph-based SSL: examples

[Adamic+ 2005]

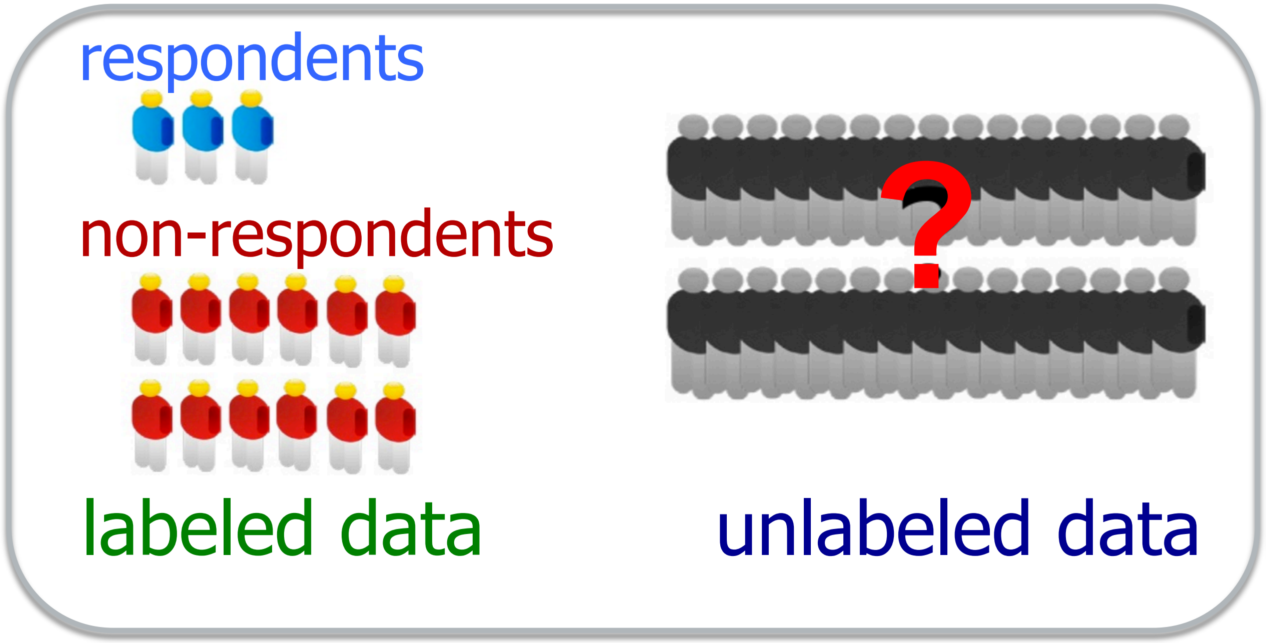


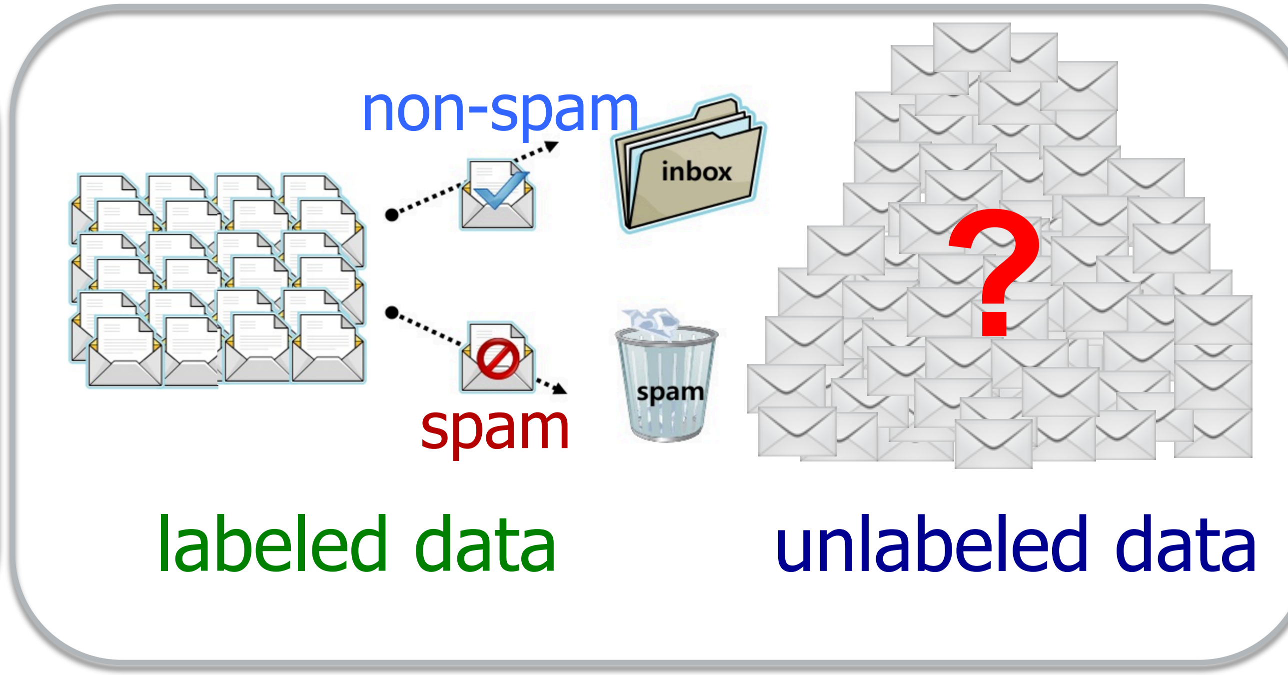
- Sometimes data is naturally a graph …
  - Graph: Political blog citations
  - Task: Polarity prediction



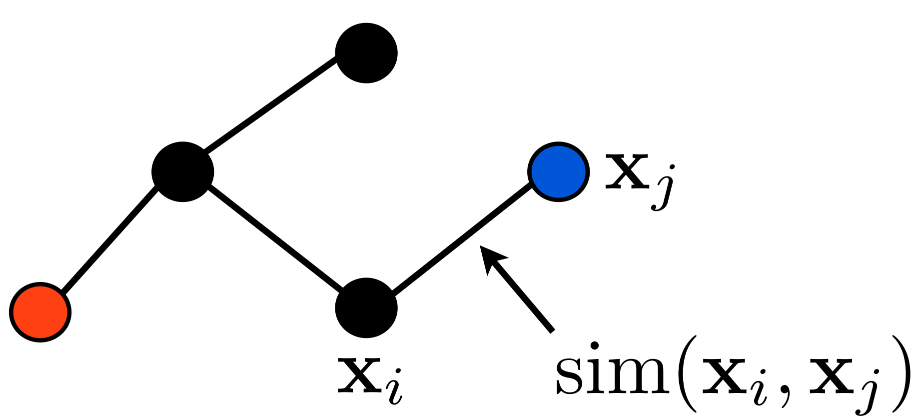
Graph-based SSL

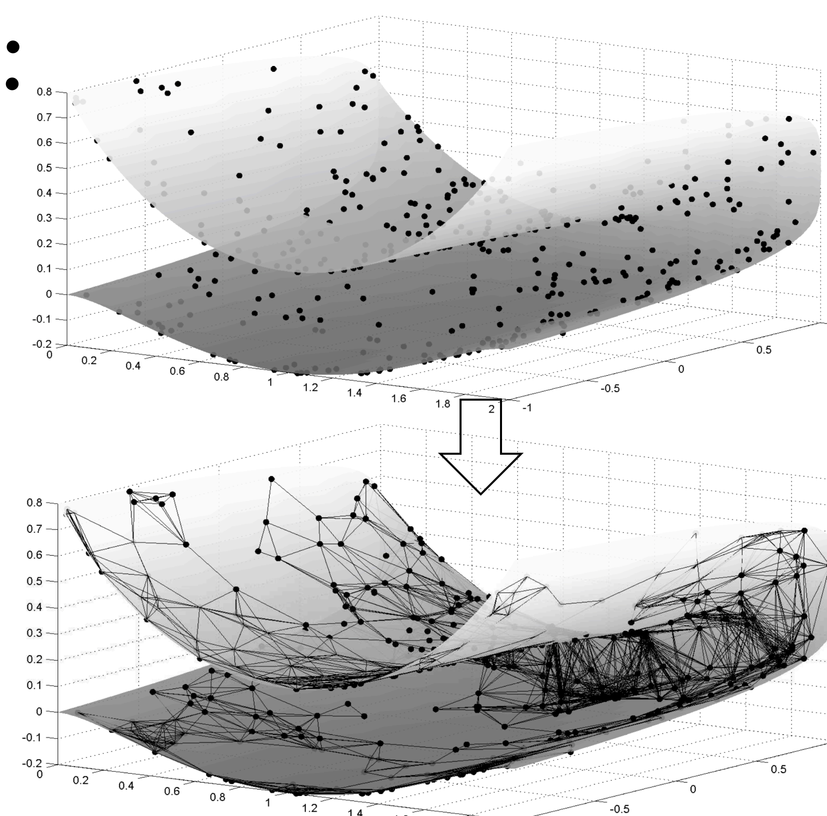
- In others we get vector (point-cloud) data …



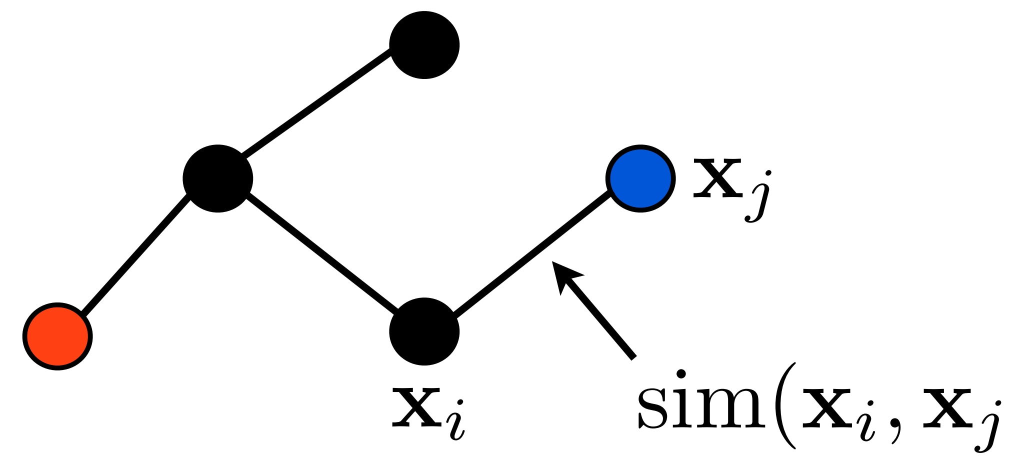


- … from which we construct a graph:
  - by connecting "similar" points





Graph Construction for SSL



Most typically:

- Connecting "similar" points by e.g. RBF (Gaussian) kernel

$$\mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|/(2\sigma^2))$$

- Sparsification
  - ε-neighborhood: node pairs within distance ε connected,
  - kNN: each node is connected to its k nearest neighbors

- Hyperparameters (σ, ε) or (σ, k) chosen by grid search based on cross validation error

Graph Construction for SSL

- Unsupervised
  - Locally Linear Embedding [Roweis&Soul *Science* 2000]
  - b-matching [Jebara+ *ICML* 2009]
  - Low-Rank Representation [Liu+ *ICML* 2010]
  - Anchor Graph Regularization [Wang+ *TKDE* 2016]

→ no use of labels, not graph *learning*

- Supervised
  - Distance metric learning [Dhillon+ *ACL* 2010]
  - Multiple kernel learning [Li+ *IJCAI* 2016]
  - Constrained self-representation [Zhuang+, Image Proc. 2017]
  - …

→ not task-driven and/or scalable

This Talk

- Semi-supervised learning: Intro
- Graph-based SSL
  - Formulation & Solution
  - **What graph should one use?**
- The Quest for Structure

Graph Construction for SSL

- A more flexible graph family:

$$\mathcal{K} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}\,, \quad \boldsymbol{W}_{ij} = \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

- **dimension-specific** kernel bandwidth

$$\boxed{\mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(- \sum_{m=1}^{d} \frac{(\boldsymbol{x}_{im} - \boldsymbol{x}_{jm})^2}{\sigma_m^2} \right)}$$

$$\boldsymbol{W}_{ij} = \exp\Big(- (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{A} \, (\boldsymbol{x}_i - \boldsymbol{x}_j)\Big)$$

$$A := diag(\boldsymbol{a})$$

$$A_{mm} = a_m = 1/\sigma_m^2$$

Joint Graph Structure & SSL Inference: Problem Statement

- **Given**

$$\mathcal{D} := \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_l, y_l), \boldsymbol{x}_{l+1}, \ldots, \boldsymbol{x}_{l+u}\}, y_i \in \mathbb{N}_c$$

- **Infer**
  - $A := diag(\boldsymbol{a})$: bandwidths per dimension
  - k : for sparse kNN graph construction

  to align the graph structure with the underlying (hidden) data manifold and the given SSL task.
  - labels for unlabeled points

This Talk

- Semi-supervised learning: Intro
- Graph-based SSL
  - Formulation & Solution
  - What graph should one use?
- The Quest for Structure
  - Problem Statement
  - Gradient-based sequential search

    \+ Adaptive parallel search

Joint Graph Structure & SSL Inference:

Gradient-based iterative hyperparameter search:

Initialize k and $\boldsymbol{a}$ (vector containing a_m's); $\quad t := 0$

repeat

Compute $\boldsymbol{F}^{(t)}$ using kNN graph on current a_m's

Compute gradient $\frac{\partial g}{\partial a_m}$ based on $\boldsymbol{F}^{(t)}$ for each a_m

Update a_m's by $\boldsymbol{a}^{(t+1)} := \boldsymbol{a}^{(t)} - \gamma \frac{\mathrm{d}g}{\mathrm{d}\boldsymbol{a}}$; $\quad t := t + 1$

until a_m's have converged

Validation Loss $g(\cdot)$ & Gradient Updates

- Subset of labels designated as validation set

$$\mathcal{V} \subset \mathcal{L}$$

- One could use validation error:

$$g_A(\mathcal{V}) = \sum_{v \in \mathcal{V}} (1 - F_{vc_v})$$

  - and others: $-\log F_{vc_v}$, $(1 - F_{vc_v})^x$, $x^{-F_{vc_v}}$

- To make the most of (small) validation set, a pairwise learning-to-rank objective:

$$\boxed{g_A(\mathcal{V}) = \sum_{c'=1}^{c} \sum_{\substack{(v,v'):\, v \in \mathcal{V}_{c'},\\ v' \in \mathcal{V} \setminus \mathcal{V}_{c'}}} -\log \sigma(F_{vc'} - F_{v'c'})}$$

Validation Loss $g(\cdot)$ & Gradient Updates

- Pairwise learning-to-rank objective:

$$g_A(\mathcal{V}) = \sum_{c'=1}^{c} \sum_{\substack{(v,v'):\, v\in\mathcal{V}_{c'},\\ v'\in\mathcal{V}\setminus\mathcal{V}_{c'}}} -\log\sigma(F_{vc'} - F_{v'c'})$$

- Gradient formulas omitted for brevity, we show
  - Computational complexity

    $$\boxed{O(n[kctd + dk^2 + \log n])}$$

  - Memory complexity

    $$\boxed{O(knd)}$$

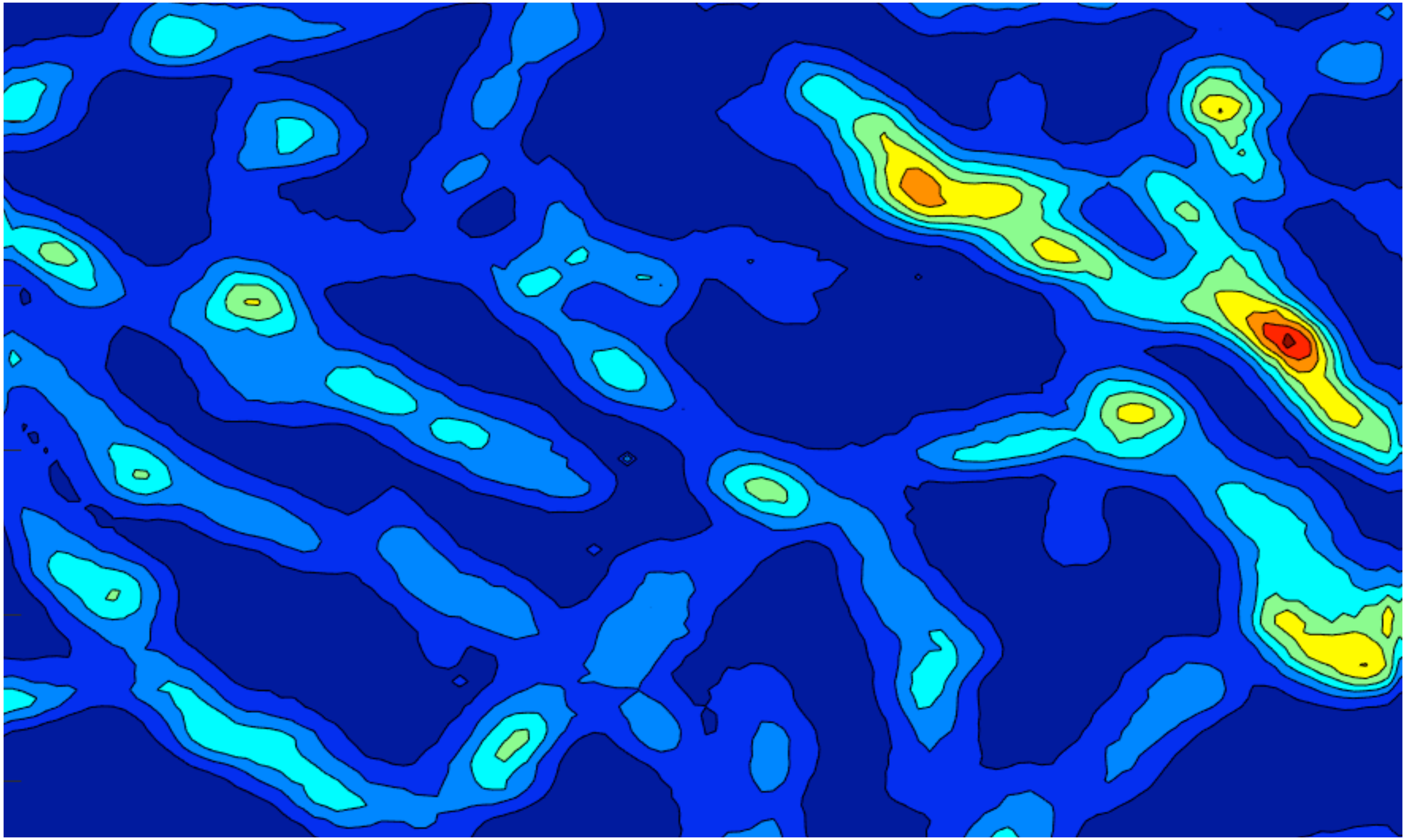
  k: #NNs, c: #classes, t: #power method iterations,

  - linear in dimensionality, log-linear in sample size
  - linear in both dimensionality & size

Large Search Space

- Flexible family of graphs to choose from

→ **Numerous** hyperparameters → **Huge** search space

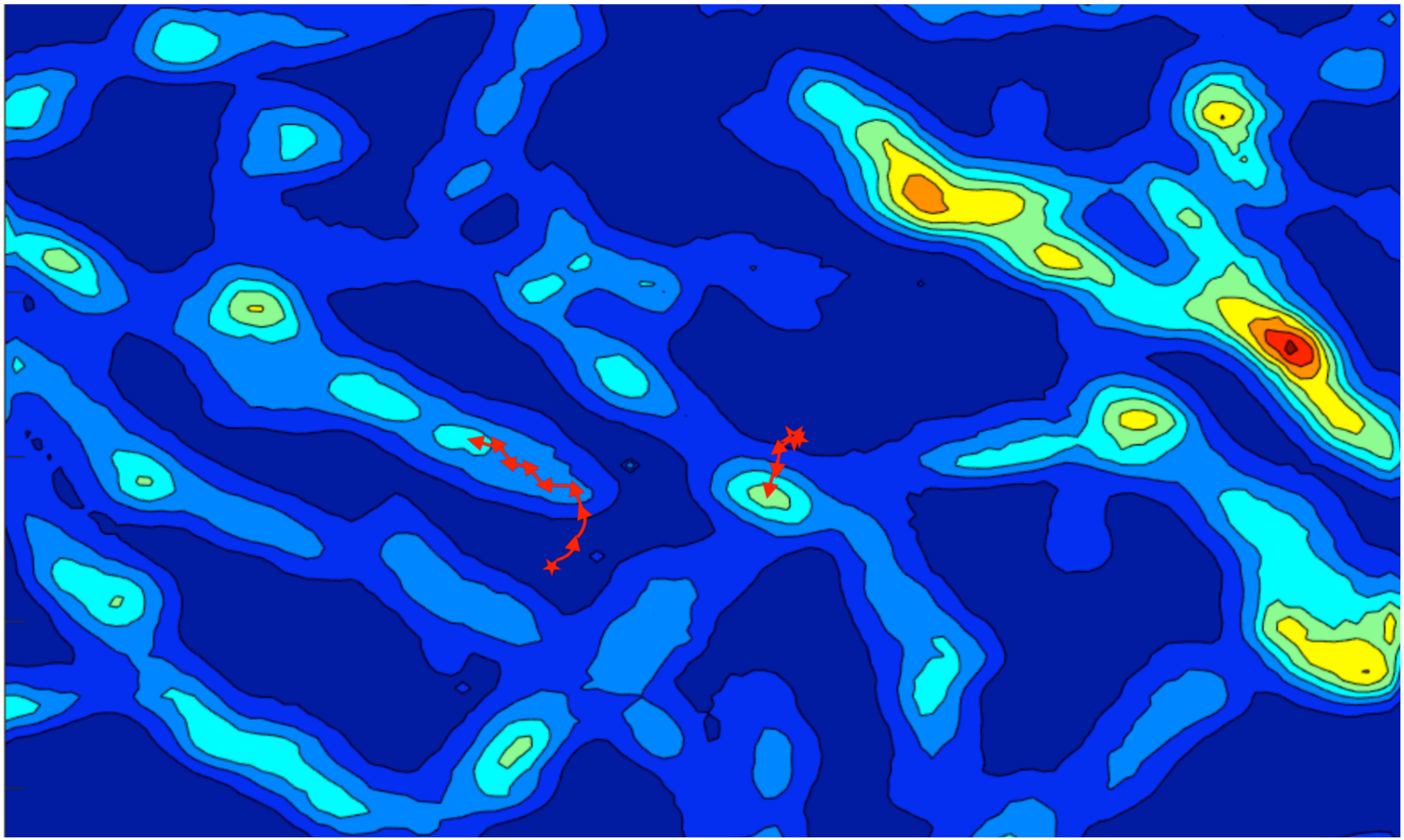


validation error in 2-d search space, **red: lower error**

Large Search Space

→ **Numerous** hyperparameters → **Huge** search space

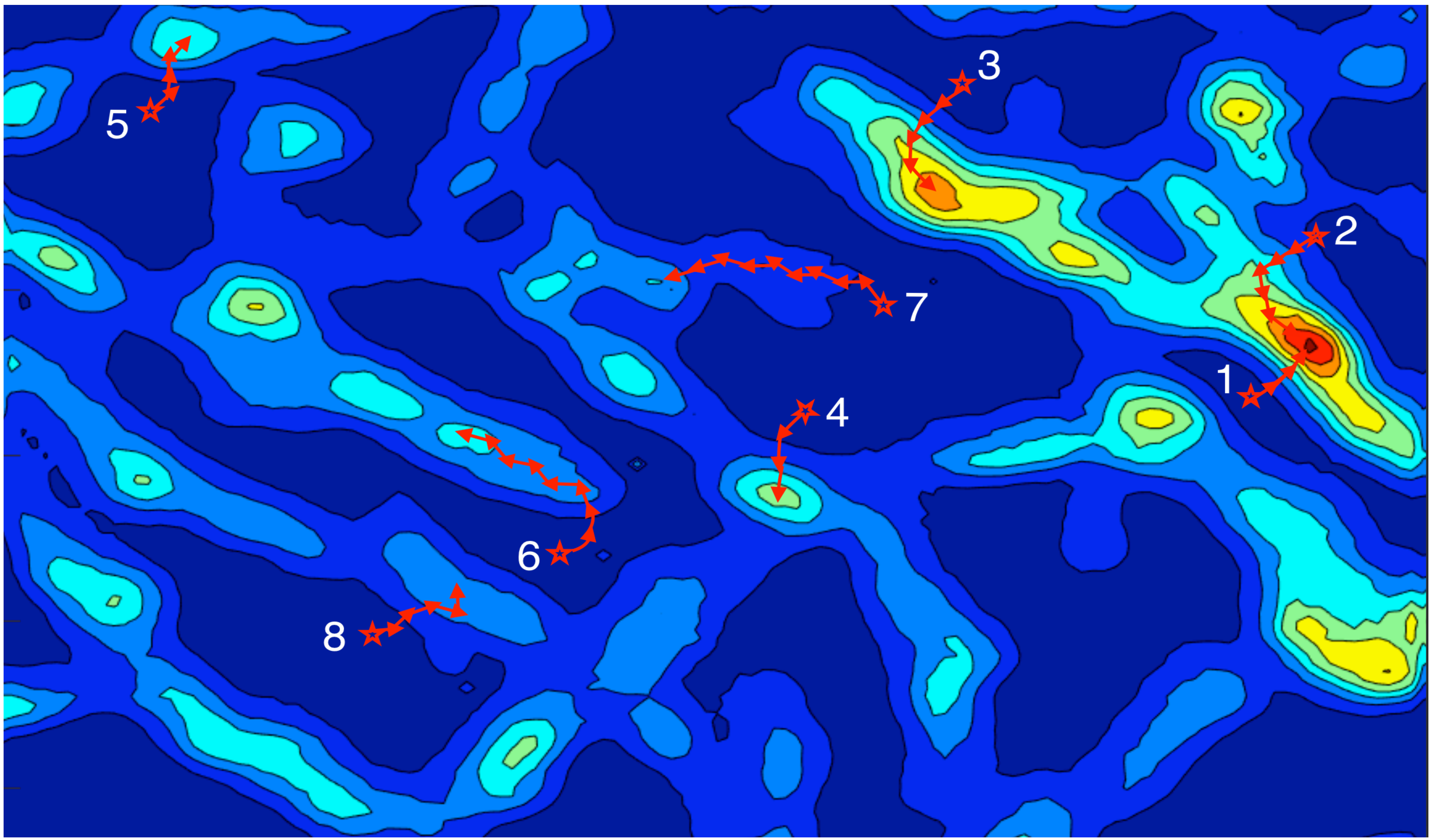
→ Most often the search is **not satisfactory**



validation error in 2-d search space, red: lower error

Search Space: Research Questions

- Can we perform more #searches in given time?
- Can we quit "unpromising" configurations early?



validation error in 2-d search space, red: lower error

This Talk

- Semi-supervised learning: Intro
- Graph-based SSL
  - Formulation & Solution
  - What graph should one use?
- The Quest for Structure
  - Problem Statement
  - Gradient-based sequential search
  - + Adaptive parallel search

Resource-adaptive search

- A simple & effective idea – Successive Halving:

  [Jamieson & Talwalkar, AISTATS 2016]

  (for hyperparameter tuning for iterative machine learning algorithms)

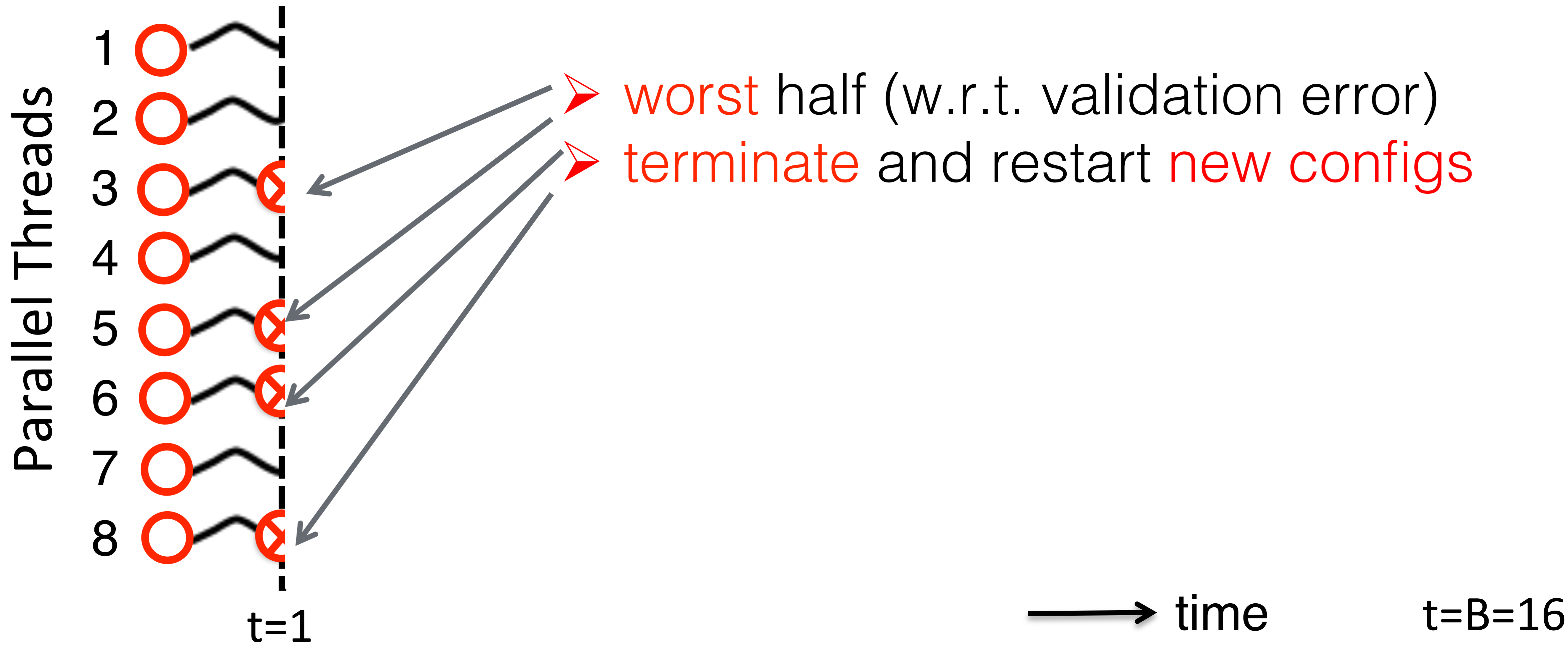
  1. pick **a set** of (hyperparameter) configurations
  2. run for a fixed amount of time (i.e. iterations)
  3. evaluate configurations (metric of interest)
  4. keep the **best half** (terminate the worst half)
  5. repeat 2. – 4. until **one** configuration remains

Parallel resource-adaptive search

- A simple & effective idea – Successive Halving:
  [Jamieson & Talwalkar, AISTATS 2016]
  (for hyperparameter tuning for iterative ML algo.s)
  - SH is originally proposed for 0th-order optimization (i.e. can be used for derivative-free functions);
    → we use 1st-order optimization via gradient,
    not only SSL inference but also gradient is iterative
  - SH runs rounds in succession
    → we parallelize the configurations and
    fully utilize idle (terminated) threads
    by restarting new configurations

Parallel resource-adaptive search: Pictorially

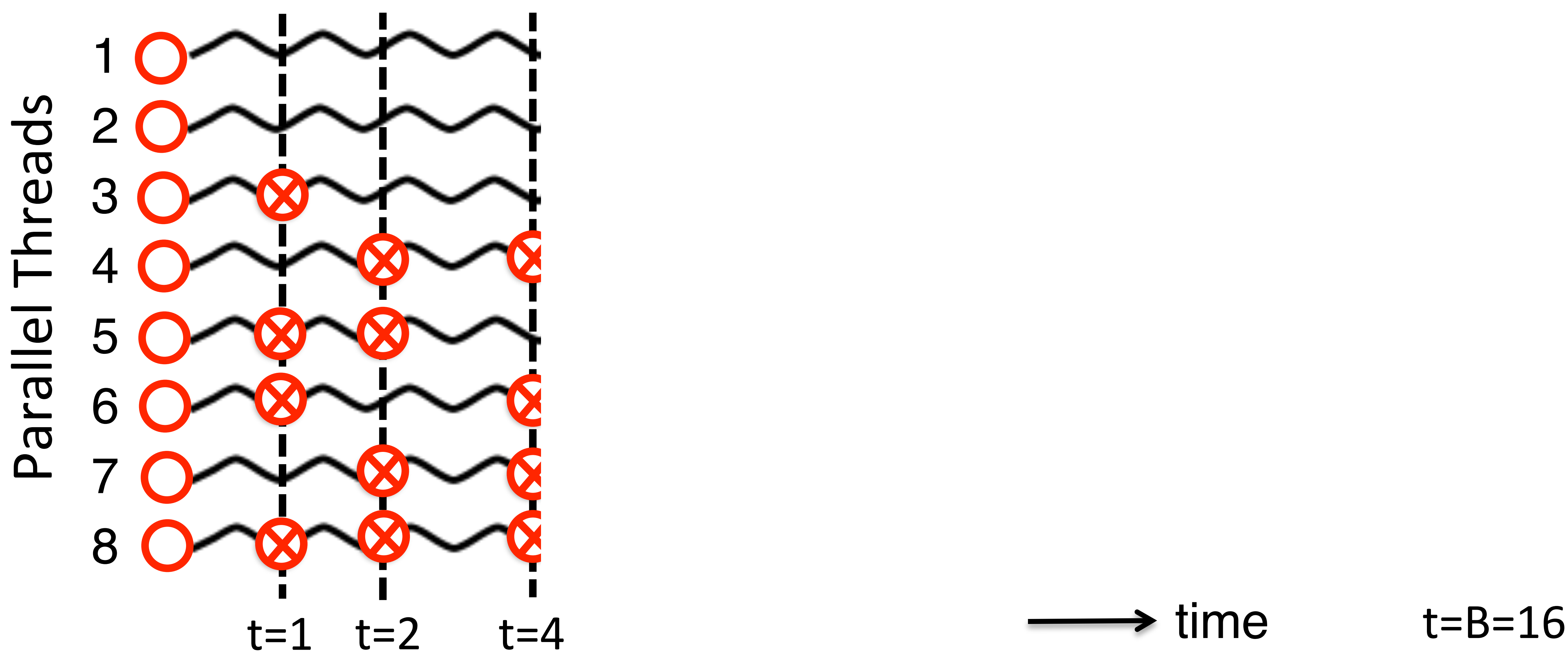
Parallel resource-adaptive search: Pictorially



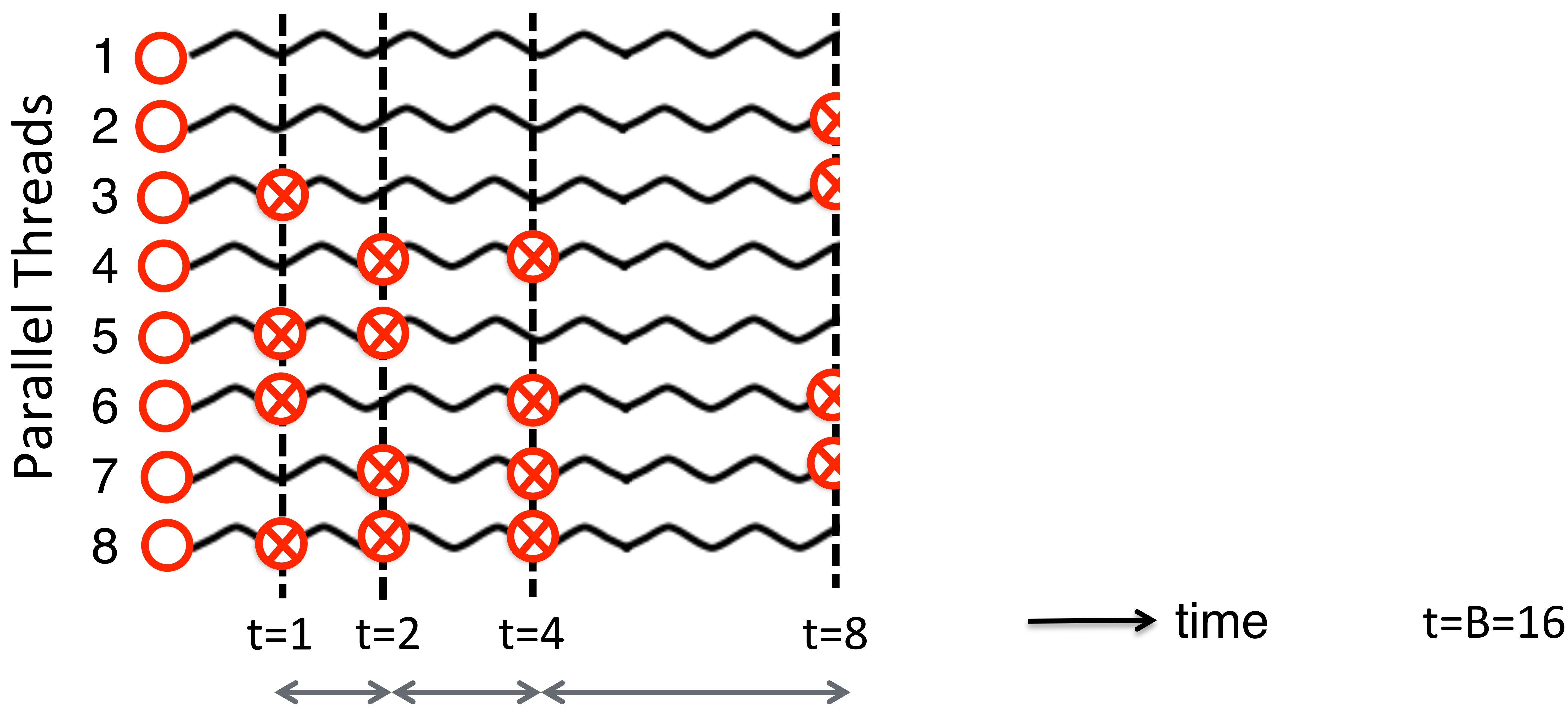
- worst half (w.r.t. validation error)
- terminate and restart new configs

Parallel resource-adaptive search: Pictorially

Parallel resource-adaptive search: Pictorially



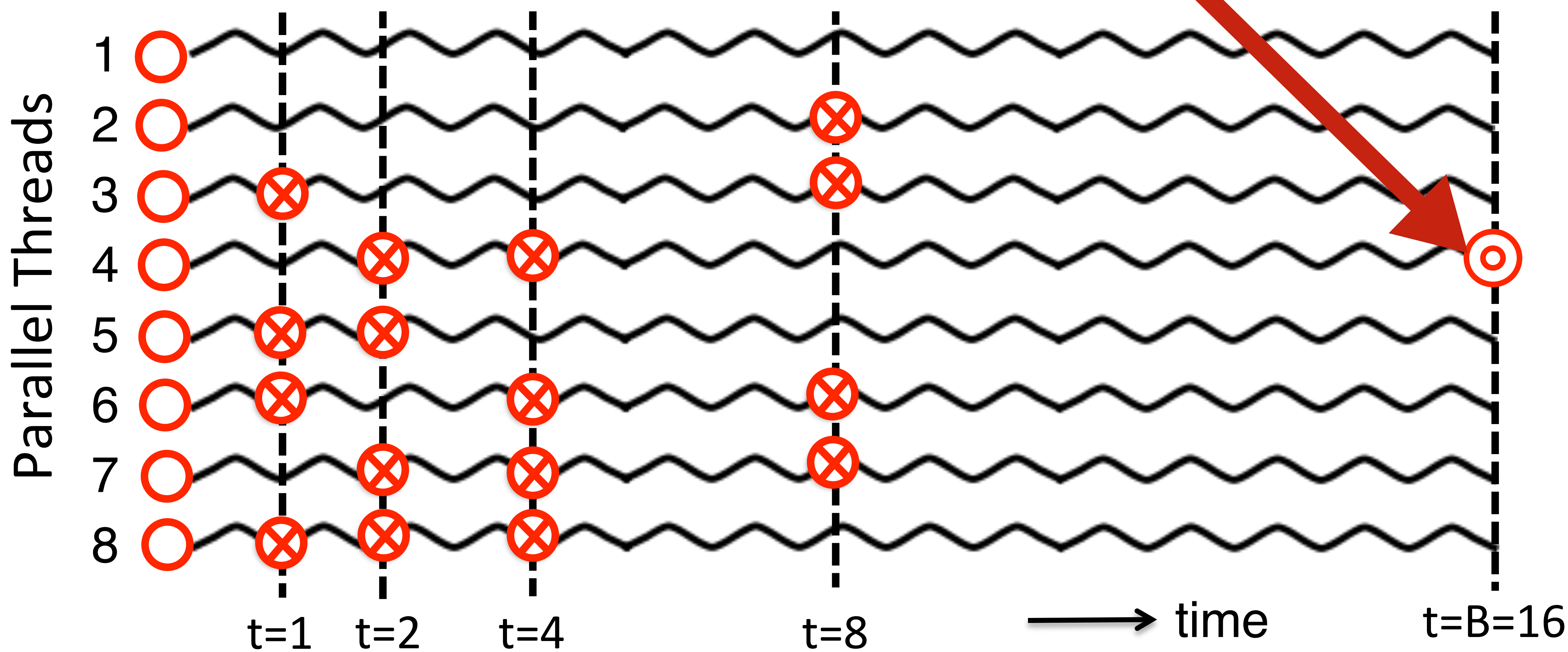
Parallel resource-adaptive search: Pictorially



➢ increase thread-time as gradient decays

Parallel resource-adaptive search: Pictorially

➢ return best result at budget time



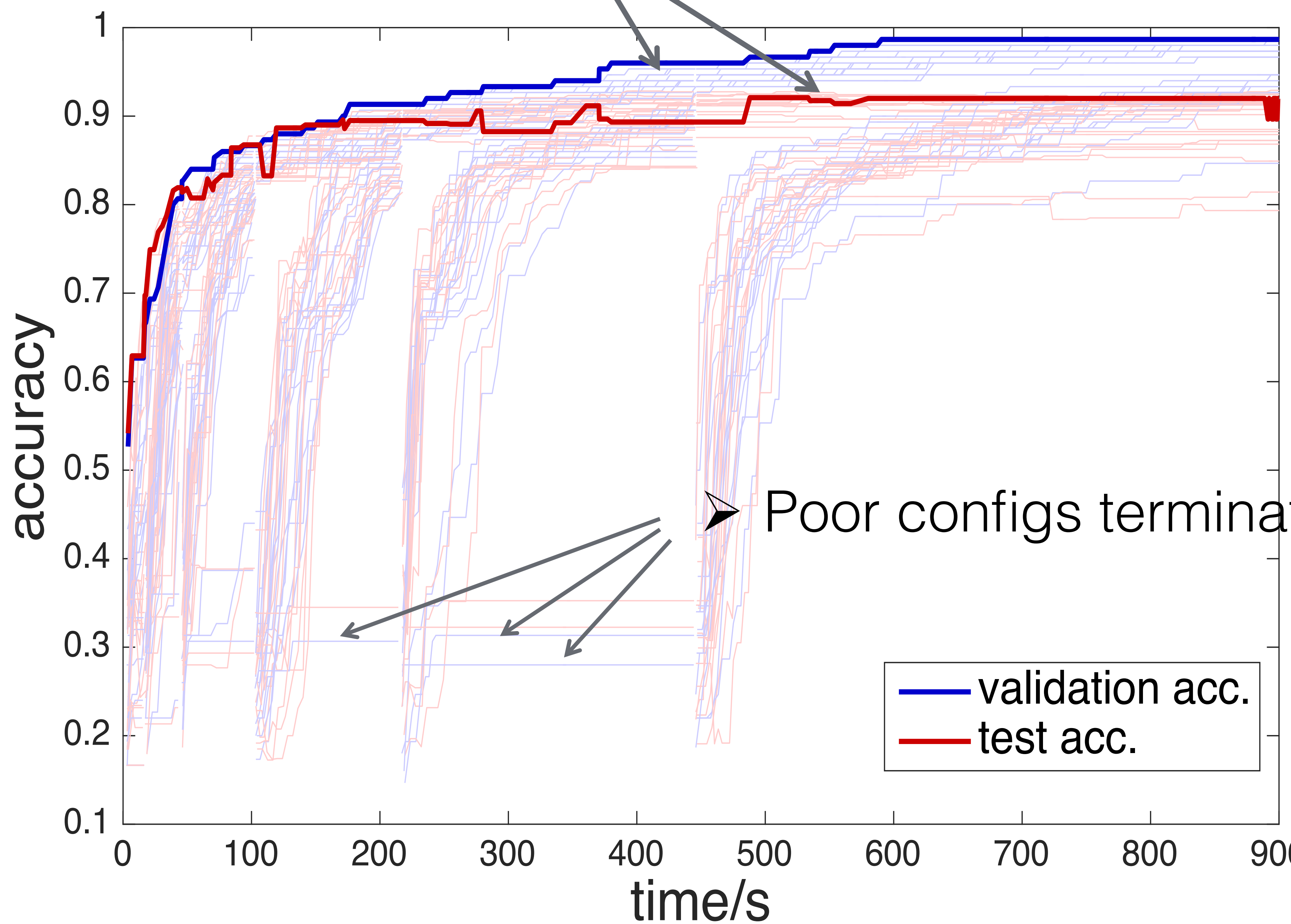
$$\boxed{T + (1 - 1/r)T\lfloor \log_r B \rfloor} = 8 + 4\lfloor \log_2 16 \rfloor = 24 \text{ configs}$$

quit-rate #threads budget

vs. 8 configs

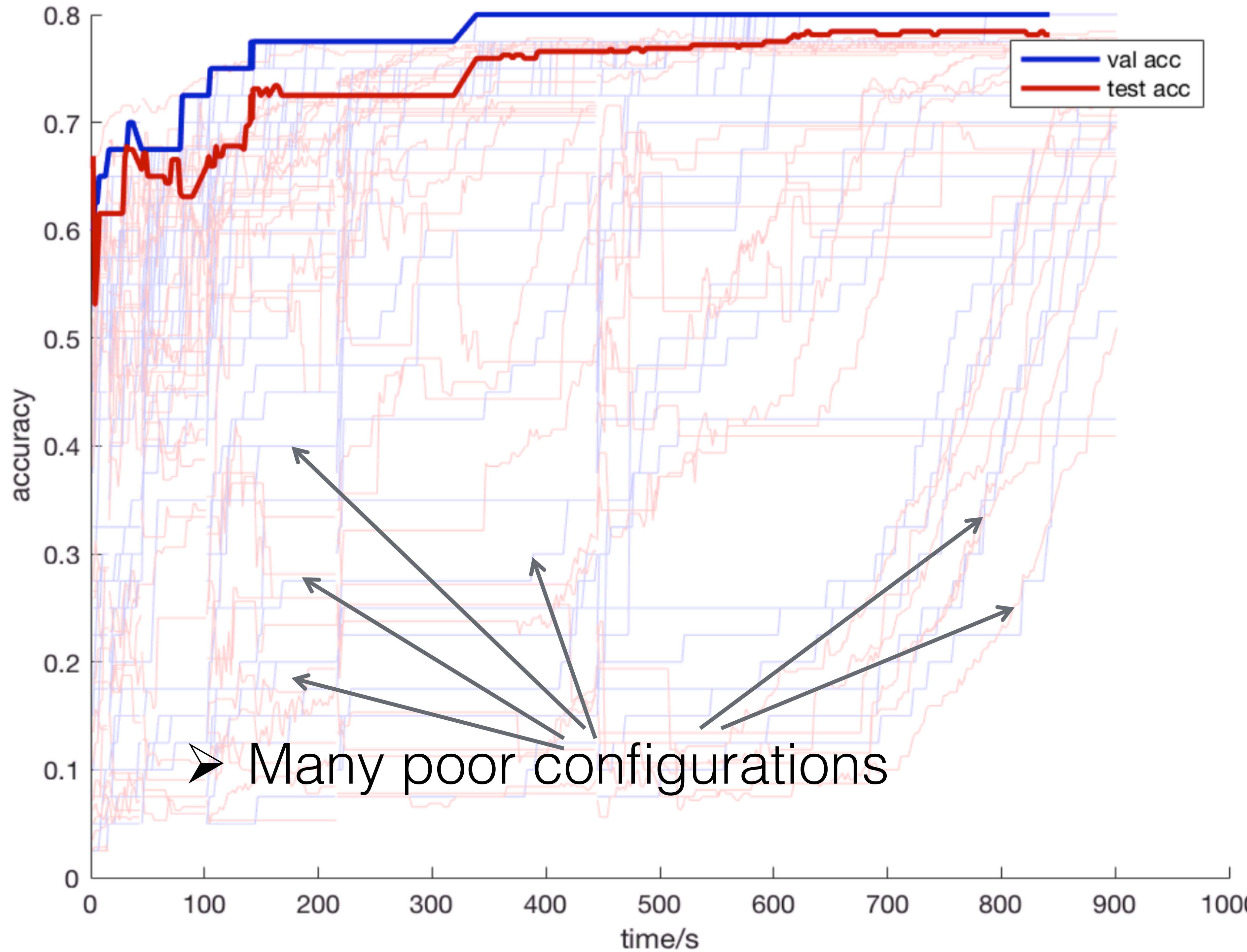
Parallel resource-adaptive search: Example

➢ Test follows validation accuracy



Parallel resource-adaptive search: Example

➢ Test accuracy improves by time



➢ Many poor configurations

This Talk

- Semi-supervised learning: Intro
- Graph-based SSL
  - Formulation & Solution
  - What graph should one use?
- The Quest for Structure
  - PG-Learn: parallel graph search with adaptive resource allocation
- Experiments

Multi-class classification datasets

Name	#pts n	#dim d	#cls c	description
COIL	1500	241	6	objects with various shapes
USPS	1000	256	10	handwritten digits
MNIST	1000	784	10	handwritten digits
UMIST	575	644	20	faces (diff. race/gender/etc.)
YALE	320	1024	5	faces (diff. illuminations)

Graph Construction Baselines

(1) *Grid* search (GS): k-NN graph with RBF kernel where k and strawmen are chosen via grid search,

(2) $Rand_d$ search (RS): k-NN with RBF kernel where k and different bandwidths $\boldsymbol{a}_{1:d}$ are randomly chosen,

(3) *MinEnt*: gradient-based based tuning of $\boldsymbol{a}_{1:d}$'s as proposed by Zhu et al. (generalized to multi-class),

(4) *AEW*: self-representation ing by Karasuyama et al. that estimates $\boldsymbol{a}_{1:d}$'s through local linear reconstruction, and

(5) *IDML*: metric learning ing scheme combined with distance metric learning by Dhillon et al.

Single-thread results

Dataset	PG-LRN	***MinEnt***	***IDML***	***AEW***	***Grid***	***Rand$_d$***
COIL	**0.9232**	0.9116▲	0.7508▲	0.9100▲	0.8929▲	0.8764▲
USPS	**0.9066**	**0.9088**	0.8565▲	0.8951▲	0.8732▲	0.8169▲
MNIST	**0.8241**	**0.8163**	0.7801△	0.7828▲	0.7550▲	0.7324▲
UMIST	**0.9321**	0.8954▲	0.8973△	0.8975▲	0.8859▲	0.8704▲
YALE	**0.8234**	0.7648△	0.7331▲	0.7386▲	0.6576▲	0.6797▲

avg'ed across 10 random samples

Symbols ▲ (p<0.005) and △ (p<0.01)
w.r.t. the paired Wilcoxon signed rank test.

10% labeled data

Single-thread results
increasing labeling %

Labeled	PG-L	***MinEnt***	***IDML***	***AEW***	***Grid***	***Rand$_d$***
10% acc.	**0.8819**	0.8594▲	0.8036▲	0.8448▲	0.8129▲	0.7952▲
rank	**1.20**	2.20	4.40	2.80	4.80	5.60
20% acc.	**0.8900**	0.8504▲	0.8118▲	0.8462▲	0.8099▲	0.8088▲
rank	**1.42**	2.83	4.17	2.92	4.83	4.83
30% acc.	**0.9085**	0.8636▲	0.8551▲	0.8613▲	0.8454▲	0.8386▲
rank	**1.33**	3.67	3.83	3.17	4.00	5.00
40% acc.	**0.9153**	0.8617▲	0.8323▲	0.8552▲	0.8381▲	0.8303▲
rank	**1.67**	3.67	3.50	3.67	4.00	4.50
50% acc.	**0.9251**	0.8700△	0.8647▲	0.8635▲	0.8556▲	0.8459▲
rank	**1.50**	3.17	3.83	3.67	4.00	4.83

Symbols ▲ (p<0.005) and △ (p<0.01)
w.r.t. the paired Wilcoxon signed rank test.

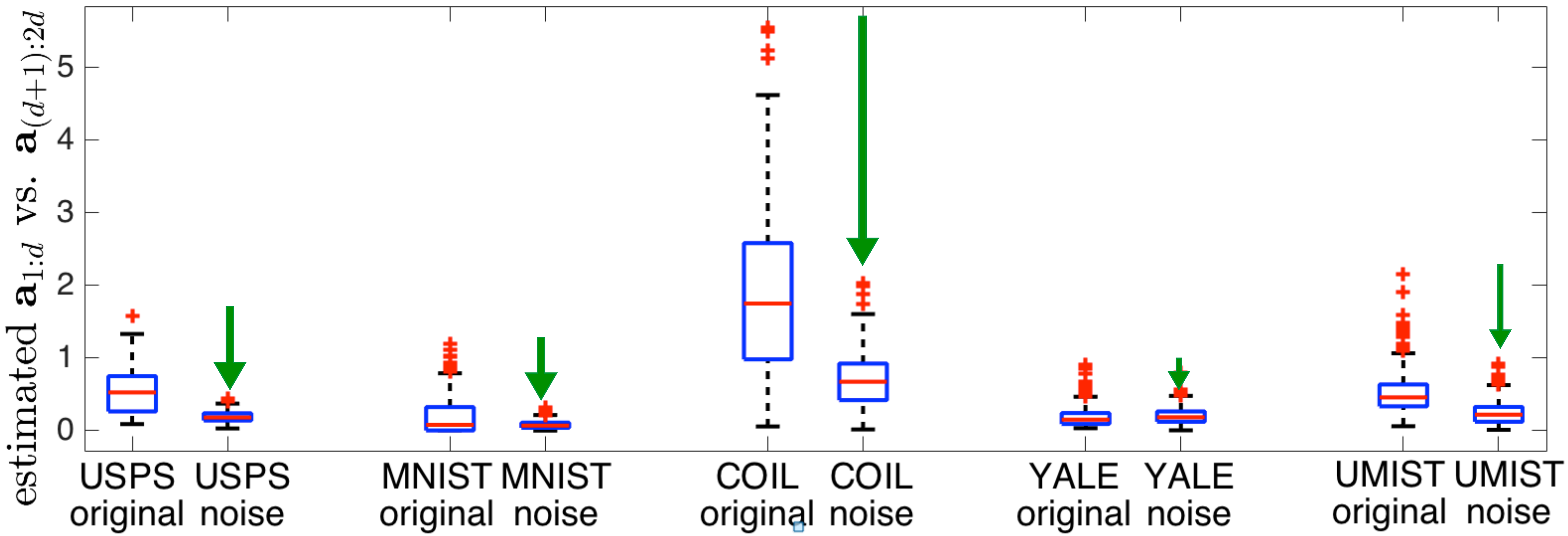
Parallel results with noisy features

- Double the feature space by adding 100% new columns with Normal(0,1) noise

Dataset	PG-LRN	***MinEnt***	***Grid***	***Rand$_d$***
COIL	**0.9044**	0.8197▲	0.6311▲	0.6954▲
USPS	**0.9154**	0.8779△	0.8746▲	0.7619▲
MNIST	**0.8634**	0.8006▲	0.7932▲	0.6668▲
UMIST	**0.8789**	0.7756▲	0.7124▲	0.6405▲
YALE	**0.6859**	0.5671▲	0.5925▲	0.5298▲

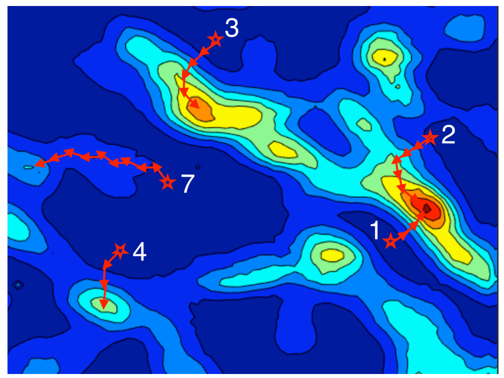
- ➢ IDML failed to learn metric due to degeneracy
- ➢ AEW authors' implementation threw out-of-memory errors

Parallel results with noisy features
investigating learned feature weights



➢ PG-Learn estimates lower weights for noisy columns

Code, Data, Slides



PG-Learn

https://bit.ly/2IZmPCs

lakoglu@andrew.cmu.edu

Thanks!

NSF CAREER 1452425,
DARPA FA8650-15-C-7561