Poisson Learning: Graph-based semi-supervised learning at very low label rates

Jeff Calder [1], Brendan Cook [1], Matthew Thorpe [2], and Dejan Slepčev [3]

[1]School of Mathematics, University of Minnesota
[2]Department of Mathematics, University of Manchester
[3]Department of Mathematical Sciences, Carnegie Mellon University

International Conference on Machine Learning (ICML)
July 12-18, 2020

Research supported by the National Science Foundation, European Research Council, and a University of Minnesota Grant in Aid award.

Outline

Graph-based semi-supervised learning

Graph: $G = (X, W)$

- $X = \{x_1, \dots, x_n\}$ are the vertices of the graph
- $W = (w_{ij})_{i,j=1}^n$ are nonnegative and symmetric ($w_{ij} = w_{ji}$) edge weights.
- $w_{ij} \approx 1$ if x_i, x_j similar, and $w_{ij} \approx 0$ when dissimilar.

Labels: We assume the first $m \ll n$ vertices are given labels

$$y_1, y_2, \dots, y_m \in \{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_k\} \in \mathbb{R}^k.$$

Task: Extend the labels to the rest of the vertices $x_{m+1}, \dots, x_n$.

Semi-supervised smoothness assumption

Similar points $x_i, x_j \in X$ in high density regions of the graph should have similar labels.

Laplace Learning/Label Propagation:

- Original work [Zhu et al., 2003]
- Learning [Zhou et al., 2005][Ando and Zhang, 2007]
- Manifold ranking [He et al., 2006] [Zhou et al., 2011] [Xu et al., 2011]

Laplace learning/Label propagation

Laplacian regularized semi-supervised learning solves the Laplace equation

$$\begin{cases} \mathcal{L}u(x_i) = 0, & \text{if } m+1 \leq i \leq n, \\ u(x_i) = y_i, & \text{if } 1 \leq i \leq m, \end{cases}$$

where $u : X \to \mathbb{R}^k$, and $\mathcal{L}$ is the graph Laplacian

$$\mathcal{L}u(x_i) = \sum_{j=1}^{n} w_{ij}(u(x_i) - u(x_j)).$$

The label decision for vertex x_i is determined by the largest component of $u(x_i)$

$$\ell(x_i) = \underset{j \in \{1,\ldots,k\}}{\text{argmax}} \{u_j(x)\}.$$

Label propagation

The solution of Laplace learning satisfies

$$\mathcal{L}u(x_i) = \sum_{j=1}^{n} w_{ij}(u(x_i) - u(x_j)) = 0. \qquad (m+1 \leq i \leq n)$$

Re-arranging, we see that u satisfies the mean-property

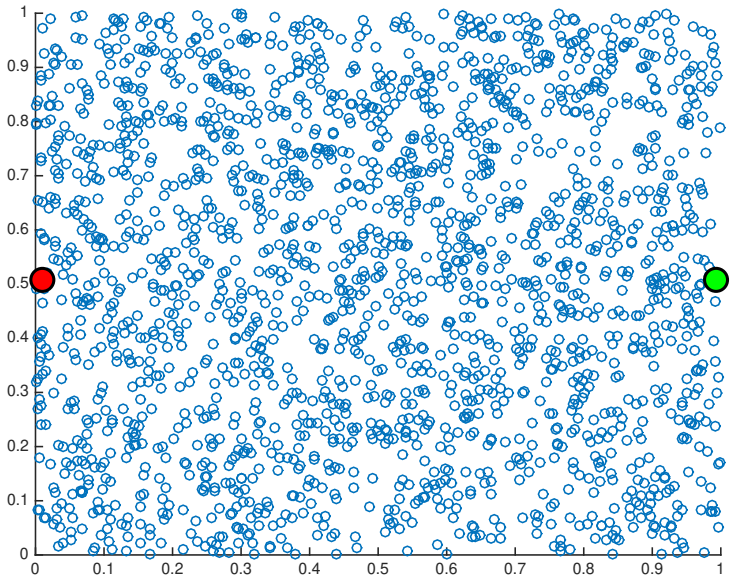
$$u(x_i) = \frac{\sum_{j=1}^{n} w_{ij}\, u(x_j)}{\sum_{j=1}^{n} w_{ij}}.$$

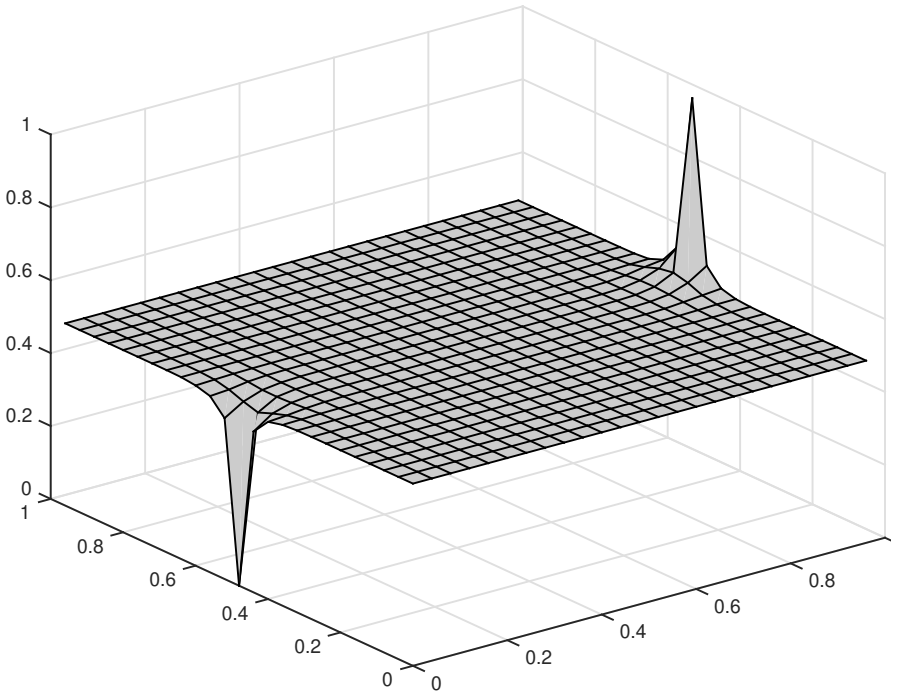
Label propagation [Zhu 2005] iterates

$$u^{k+1}(x_i) = \frac{\sum_{j=1}^{n} w_{ij}\, u^k(x_j)}{\sum_{j=1}^{n} w_{ij}},$$

and at convergence is equivalent to Laplace learning.

Ill-posed with small amount of labeled data

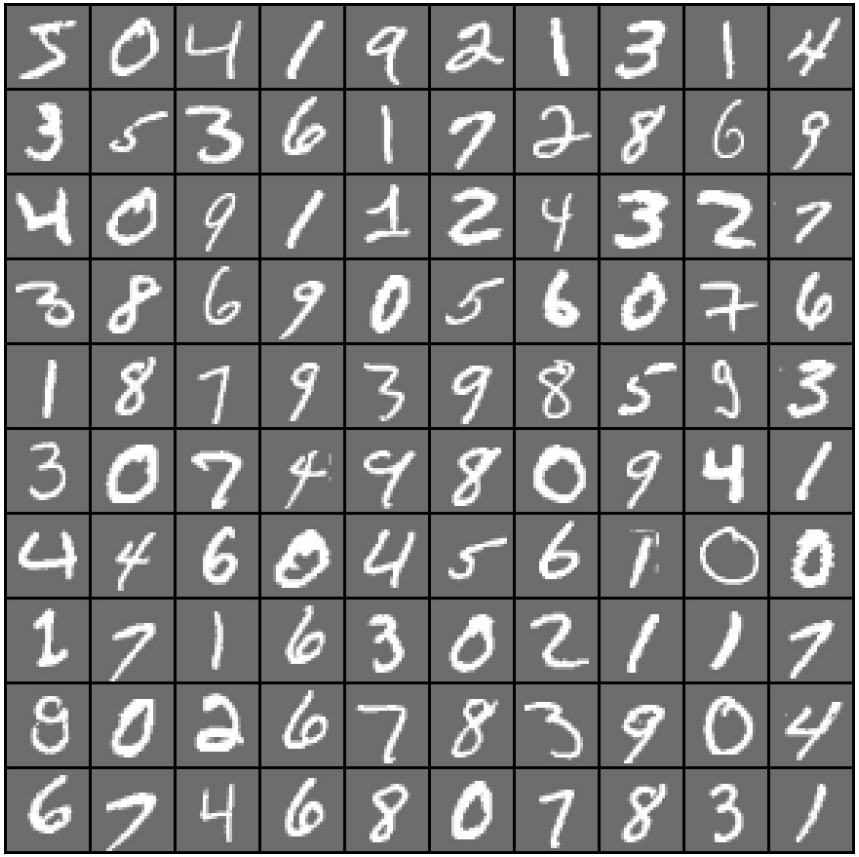




- Graph is $n = 10^5$ i.i.d. random variables uniformly drawn from $[0, 1]^2$.
- $w_{xy} = 1$ if $|x - y| < 0.01$ and $w_{xy} = 0$ otherwise.
- Two labels: $y_1 = 0$ at the Red point and $y_2 = 1$ at the Green point.
- Over 95% of labels in $[0.4975, 0.5025]$.

[Nadler et al., 2009][El Alaoui et al., 2016]

MNIST (70,000 28×28 pixel images of digits 0-9)



[Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE, 86(11):2278-2324, November 1998.]

Laplace learning on MNIST

# Labels/class	**1**	**2**	**3**	**4**	**5**
Laplace	16.1 (6.2)	28.2 (10)	42.0 (12)	57.8 (12)	69.5 (12)
Graph NN	58.8 (5.6)	66.6 (2.8)	70.2 (4)	71.3 (2.6)	73.4 (1.9)

# Labels/class	**10**	**50**	**100**	**500**	**1000**
Laplace	93.2 (2.3)	96.9 (0.1)	97.1 (0.1)	97.6 (0.1)	97.7 (0.0)
Graph NN	82.3 (1.0)	89.0 (0.5)	90.6 (0.4)	93.4 (0.1)	93.7 (0.1)

Average accuracy over 10 trials with standard deviation in brackets.

Graph NN: 1-nearest neighbor using graph geodesic distance.

Recent work

The low-label rate problem was originally identified in [Nadler 2009].

A lot of recent work has attempted to address this issue with new graph-based classification algorithms at low label rates.

- Higher-order regularization: [Zhou and Belkin, 2011], [Dunlop et al., 2019]
- p-Laplace regularization: [Alaoui et al., 2016], [Calder 2018,2019], [Slepcev & Thorpe 2019]
- Re-weighted Laplacians: [Shi et al., 2017], [Calder & Slepcev, 2019]
- Centered kernel method: [Mai & Couillet, 2018]

While we have lots of new models, the problem with Laplace learning at low label rates was still not well-understood.

In this talk:

1. We explain the degeneracy in terms of random walks.
2. We propose a new algorithm: Poisson learning.

Outline

Poisson learning

We propose to replace Laplace learning

$$(1) \qquad \text{(Laplace equation)} \quad \begin{cases} \mathcal{L}u(x_i) = 0, & \text{if } m+1 \leq i \leq n, \\ u(x_i) = y_i, & \text{if } 1 \leq i \leq m, \end{cases}$$

with Poisson learning

$$\text{(Poisson equation)} \quad \mathcal{L}u(x_i) = \sum_{j=1}^{m} (y_j - c)\delta_{ij} \quad \text{for } i = 1, \ldots, n$$

subject to $\sum_{i=1}^{n} d_i u(x_i) = 0$, where $c = \frac{1}{m} \sum_{i=1}^{m} y_i$.

In both cases, the label decision is the same:

$$\ell(x_i) = \underset{j \in \{1,\ldots,k\}}{\text{argmax}} \{u_j(x)\}.$$

Poisson learning

We propose to replace Laplace learning

$$(2) \qquad \text{(Laplace equation)} \quad \begin{cases} \mathcal{L}u(x_i) = 0, & \text{if } m+1 \le i \le n, \\ u(x_i) = y_i, & \text{if } 1 \le i \le m, \end{cases}$$

with Poisson learning

$$\text{(Poisson equation)} \quad \mathcal{L}u(x_i) = \sum_{j=1}^{m} (y_j - c)\delta_{ij} \quad \text{for } i = 1, \dots, n$$

subject to $\sum_{i=1}^{n} d_i u(x_i) = 0$, where $c = \frac{1}{m}\sum_{i=1}^{m} y_i$.

For Poisson learning, unbalanced class sizes can be incorporated:

$$\ell(x_i) = \operatorname*{argmax}_{j \in \{1,\dots,k\}} \left\{ \frac{p_j}{n_j} u_j(x) \right\},$$

p_j = Fraction of data in class j

n_j = Fraction of training data from class j.

Random Walk Perspective

Suppose u solves the Laplace learning equation

$$\begin{cases} \mathcal{L}u(x_i) = 0, & \text{if } m+1 \leq i \leq n, \\ u(x_i) = y_i, & \text{if } 1 \leq i \leq m. \end{cases}$$

Let $x \in X$ and let $X_0, X_1, X_2, \ldots$ be a random walk on X with transition probabilities

$$\mathbb{P}(X_k = x_j \mid X_{k-1} = x_i) = \frac{w_{ij}}{d_i} \qquad \text{where } d_i = \sum_{j=1}^{n} w_{ij}.$$

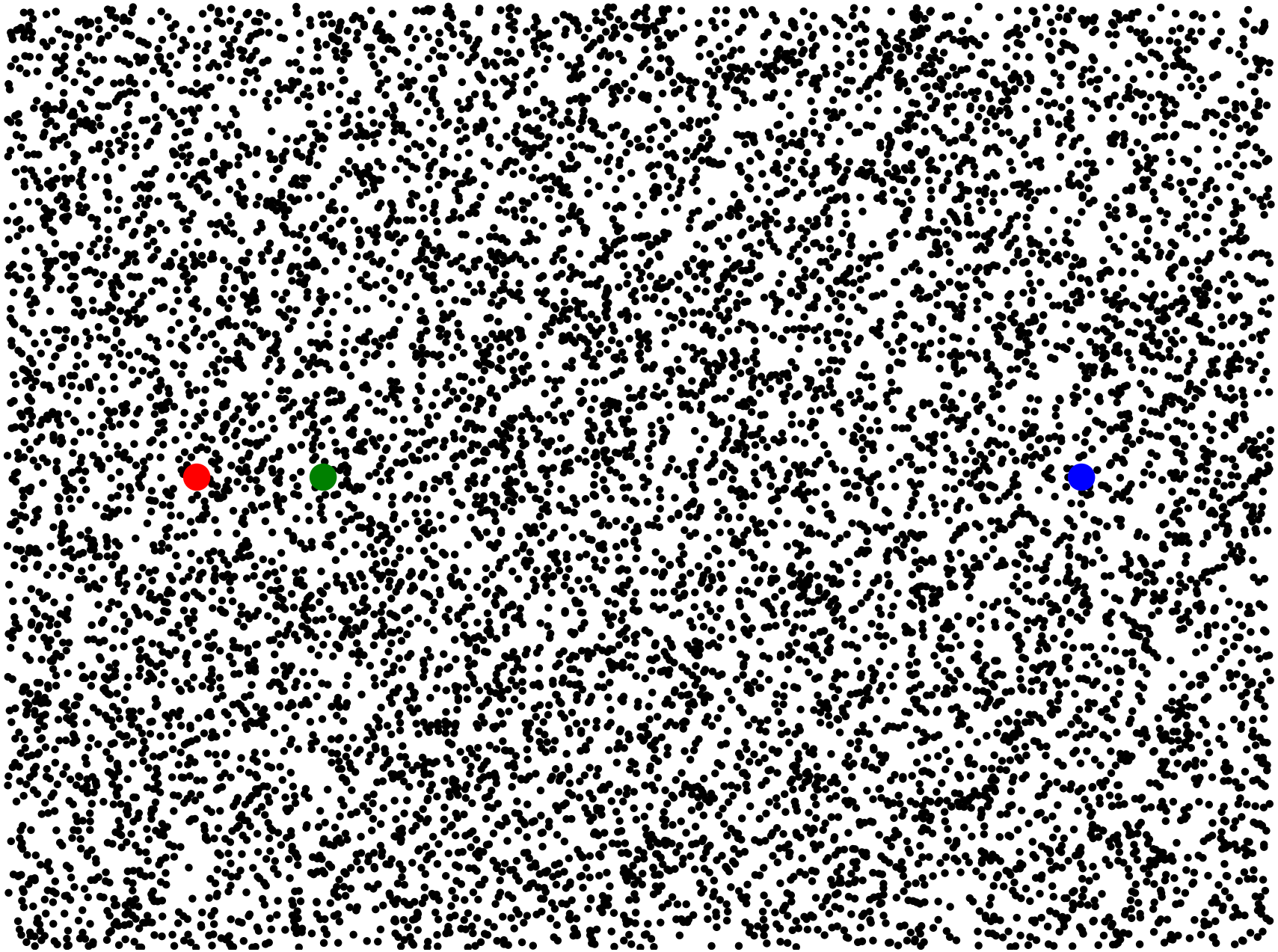
Define the stopping time to be the first time the walk hits a label, that is

$$\tau = \inf\{k \geq 0 \,:\, X_k \in \{x_1, x_2, \ldots, x_m\}\}.$$

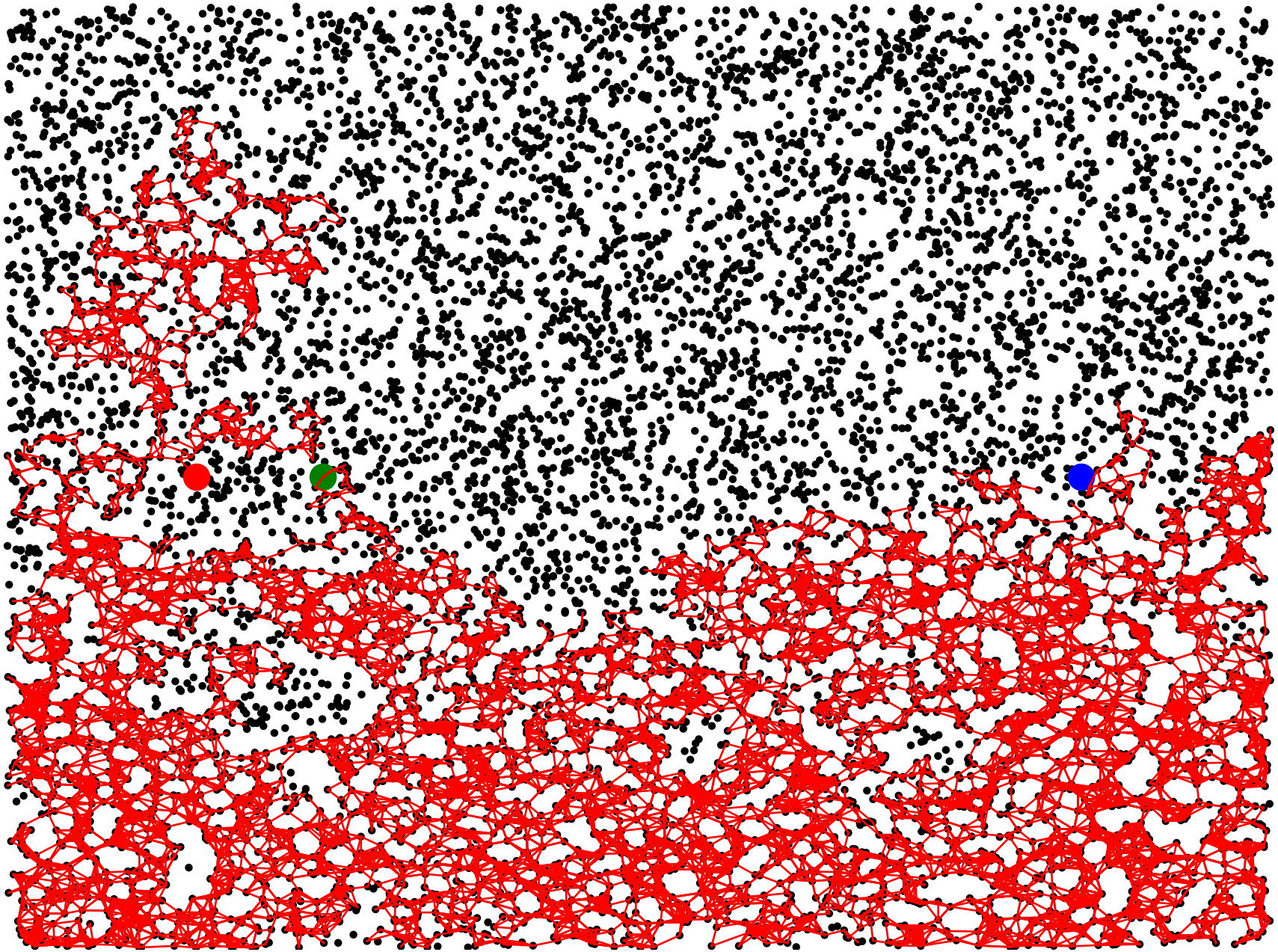
Let $i_\tau \leq m$ so that $X_\tau = x_{i_\tau}$. Then (by Doob's optimal stopping theorem)

(3) $$\boxed{u(x) = \mathbb{E}[y_{i_\tau} \mid X_0 = x].}$$

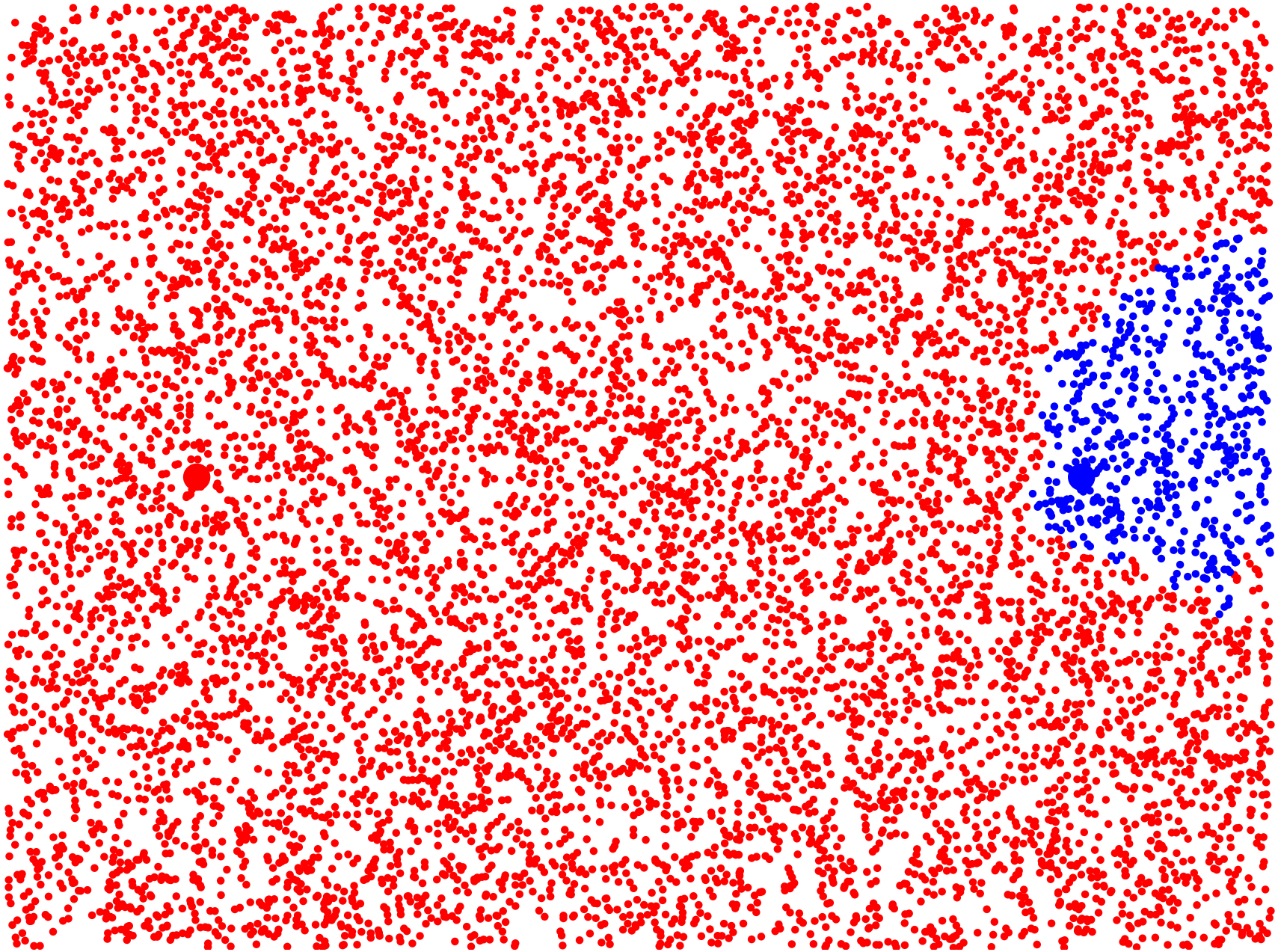
Classification experiment



Random walk experiment



Classification experiment



The Random walk perspective

At low label rates, the random walker reaches the mixing time before hitting a label.

- The label eventually hit is largely independent of where the walker starts.

After walking for a long time, the probability distribution of the walker approaches the invariant distribution π given by

$$\pi_i = \frac{d_i}{\sum_{j=1}^n d_j}.$$

Thus, the solution of Laplace learning is approximately

$$u(x_i) = \mathbb{E}[y_{i_\tau} \mid X_0 = x_i] \approx \frac{\sum_{j=1}^n d_j y_j}{\sum_{j=1}^n d_j} =: c \in \mathbb{R}^k.$$

Bottom line: Nearly everything is labeled by the one-hot vector closest to c!

The random walk perspective

Let $X_0^{x_j}, X_1^{x_j}, X_2^{x_j}$ be a random walk on the graph X starting from $x_j \in X$, and define

$$u_T(x_i) = \mathbb{E}\left[\sum_{k=0}^{T}\sum_{j=1}^{m} y_j \mathbb{1}_{\{X_k^{x_j} = x_i\}}\right].$$

Idea: We release random walkers from the labeled nodes, and record how often each label's walker visits x_i.

We can write

$$u_T(x_i) = \sum_{j=1}^{m} y_j \sum_{k=0}^{T} \mathbb{P}(X_k^{x_j} = x_i).$$

The inner term is a Green's function for a random walk. As $T \to \infty$, $u_T \to \infty$.

We center u_T by its mean value:

$$\sum_{i=1}^{n} u_T(x_i) = \sum_{k=0}^{T}\sum_{j=1}^{m} y_j = \sum_{k=0}^{T} mc, \qquad \text{where } c = \frac{1}{m}\sum_{j=1}^{m} y_j.$$

The random walk perspective

Subtracting off the mean of u_T, and normalizing by d_i, we arrive at

$$u_T(x_i) := \mathbb{E}\left[\sum_{k=0}^{T} \frac{1}{d_i} \sum_{j=1}^{m} (y_j - c)\mathbb{1}_{\{X_k^{x_j} = x_i\}}\right], \quad \text{where } c = \frac{1}{m}\sum_{j=1}^{m} y_j.$$

Theorem

For every $T \geq 0$ we have

$$u_{T+1}(x_i) = u_T(x_i) + \frac{1}{d_i}\left(\sum_{j=1}^{m}(y_j - c)\delta_{ij} - \mathcal{L}u_T(x_i)\right).$$

If the graph G is connected and the Markov chain induced by the random walk is aperiodic, then $u_T \to u$ as $T \to \infty$, where $u : X \to \mathbb{R}$ is the solution of

$$\mathcal{L}u(x_i) = \sum_{j=1}^{m}(y_j - c)\delta_{ij} \quad \text{for } i = 1, \ldots, n$$

satisfying $\sum_{i=1}^{n} d_i u(x_i) = 0$.

The variational interpretation

We define the space of weighted mean-zero functions

$$\ell_0^2(X) = \left\{ u : X \to \mathbb{R} \, : \, \sum_{i=1}^{n} d_i \, u(x_i) = 0 \right\}.$$

Consider the variational problem

$$\min_{u \in \ell_0^2(X)} \left\{ \sum_{i,j=1}^{n} w_{ij} |u(x_i) - u(x_j)|^2 - \sum_{j=1}^{m} (y_j - c) \cdot u(x_j) \right\}, \tag{4}$$

where $c = \frac{1}{m} \sum_{i=1}^{m} y_i$.

Theorem

Assume G is connected. Then there exists a unique solution $u \in \ell_0^2(X)$ of (4), and furthermore, u satisfies the Poisson equation

$$\mathcal{L} u(x_i) = \sum_{j=1}^{m} (y_j - c) \delta_{ij}.$$

Poisson vs Laplace

The variational interpretation of Poisson learning is

$$\min_{u \in \ell_0^2(X)} \left\{ \sum_{i,j=1}^{n} w_{ij} |u(x_i) - u(x_j)|^2 - \sum_{j=1}^{m} (y_j - c) \cdot u(x_j) \right\}.$$

We compare this with the variational interpretation for Laplace learning, which is

$$\min_{u \in \ell^2(X)} \left\{ \sum_{i,j=1}^{n} w_{ij} |u(x_i) - u(x_j)|^2 \ : \ u(x_i) = y_i \text{ for } i = 1, \dots, m \right\}.$$

Takeaway: Instead of hard constraints, Poisson equations use soft constraints that are affine functions of the label values.

Outline

Code Online

All code is on GitHub as part of the GraphLearning package:

https://github.com/jwcalder/GraphLearning

Algorithmic details

Algorithm 1 Poisson Learning

Input: $\mathbf{W}, \mathbf{F}, \mathbf{b}, T$ $\{\mathbf{F} \in \mathbb{R}^{k \times m}$ are label vectors, $\mathbf{b} \in \mathbb{R}^k$ are class sizes.$\}$
Output: $\mathbf{U} \in \mathbb{R}^{n \times k}$
$\mathbf{D} \leftarrow \text{diag}(\mathbf{W}\mathbb{1})$
$\mathbf{L} \leftarrow \mathbf{D} - \mathbf{W}$
$\mathbf{c} \leftarrow \frac{1}{m}\mathbf{F}\mathbb{1}$
$\mathbf{B} \leftarrow [\mathbf{F} - \mathbf{c}, \textbf{zeros}(k, n-m)]$
$\mathbf{U} \leftarrow \textbf{zeros}(n, k)$
for $i = 1$ **to** T **do**
    $\mathbf{U} \leftarrow \mathbf{U} + \mathbf{D}^{-1}(\mathbf{B}^T - \mathbf{L}\mathbf{U})$
end for
$\mathbf{U} \leftarrow \mathbf{U} \cdot \text{diag}(\mathbf{b}/\mathbf{c})$ $\{$Accounts for unbalanced class sizes.$\}$

1. We only need about $T = 100$ iterations on MNIST, FashionMNIST, CIFAR-10, to get good results. CPU Time: 8 seconds on CPU, 1 second on GPU.

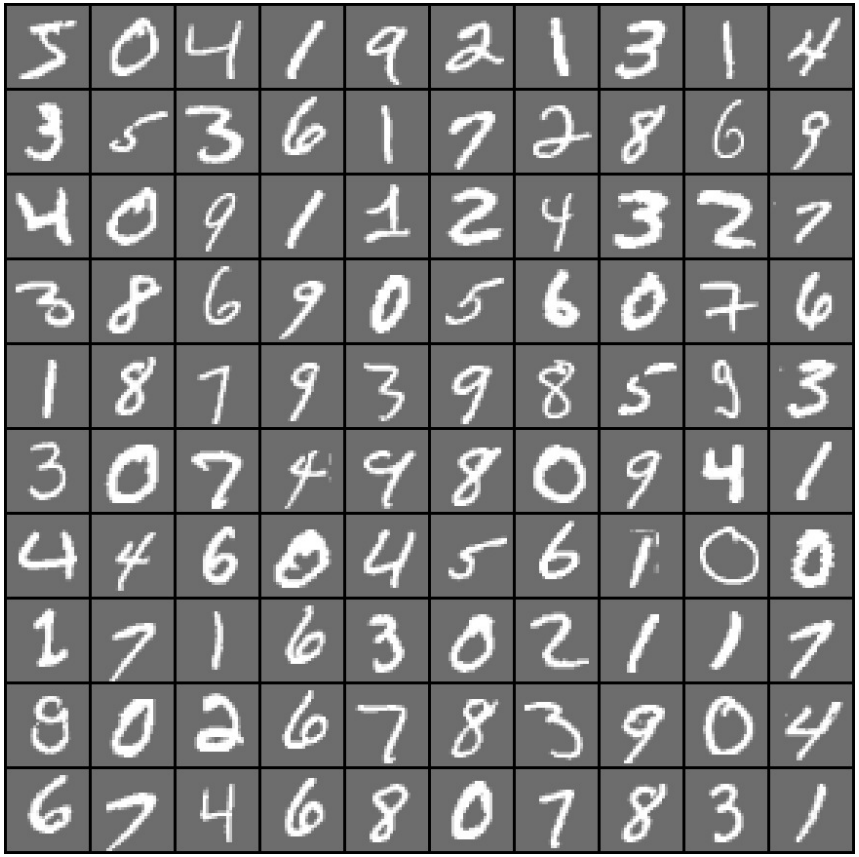
Easy to add volume constraints

Algorithm 2 Poisson MBO

1: **Input:** $\mathbf{W}, \mathbf{F}, N_{\text{inner}}, N_{\text{outer}}, \mathbf{b}, \mu, T > 0$
2: **Output:** $\mathbf{U} \in \mathbb{R}^{n \times k}$
3: $\mathbf{U} \leftarrow \text{PoissonLearning}(\mathbf{W}, \mathbf{F}, \mathbf{b}, T)$
4: $\mathrm{d}t \leftarrow 1/\max_{1 \le i \le n} \mathbf{D}_{ii}$
5: **for** $i = 1$ **to** N_{outer} **do**
6: **for** $j = 1$ **to** N_{inner} **do**
7: $\mathbf{U} \leftarrow \mathbf{U} - \mathrm{d}t(\mathbf{L}\mathbf{U} - \mu\mathbf{B}^T)$
8: **end for**
9: $\mathbf{U} \leftarrow \text{VolumeConstrainedLabelProjection}(\mathbf{U}, \mathbf{b})$
10: **end for**

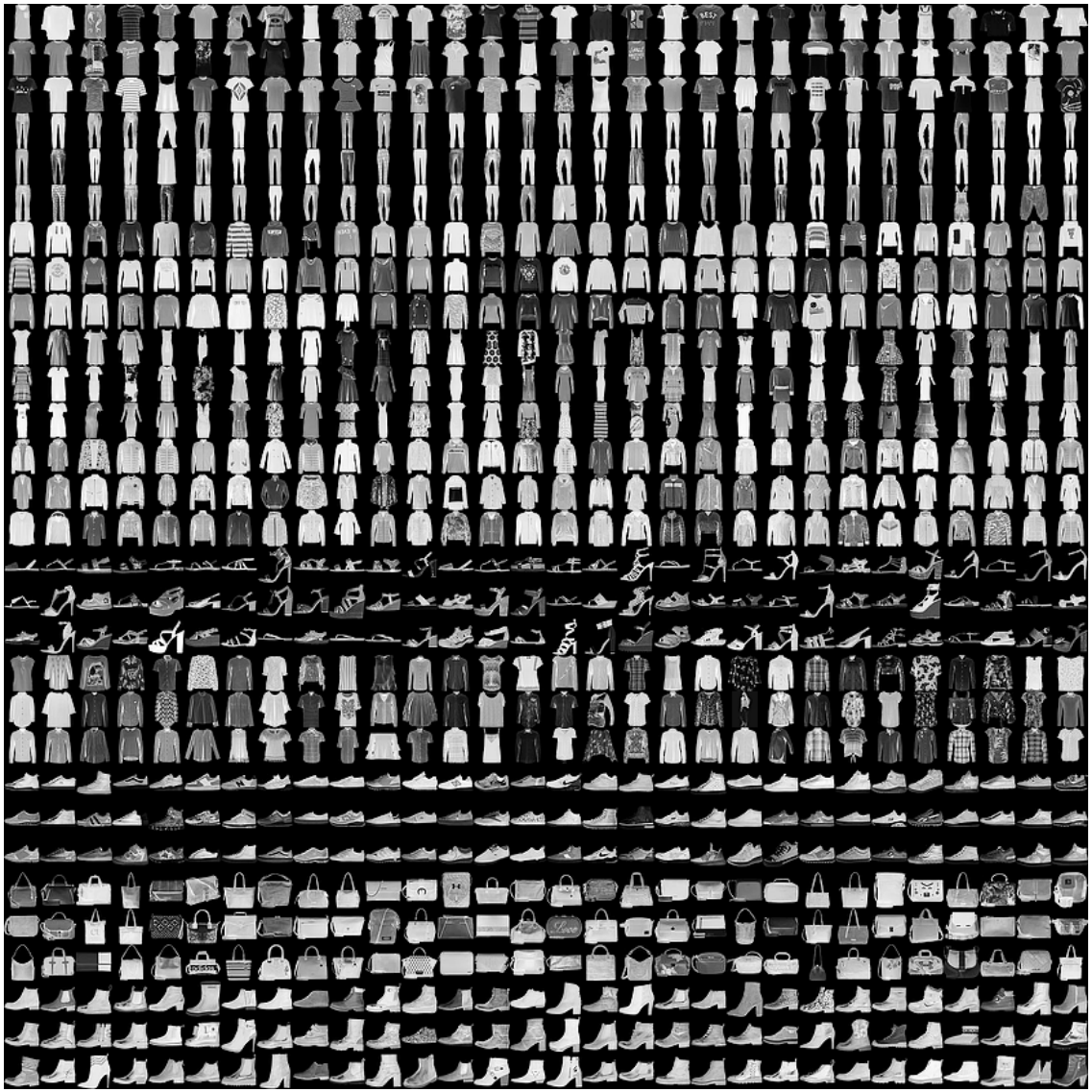
1. The iterations in Steps 7-9 are volume preserving.

MNIST (70,000 28×28 pixel images of digits 0-9)



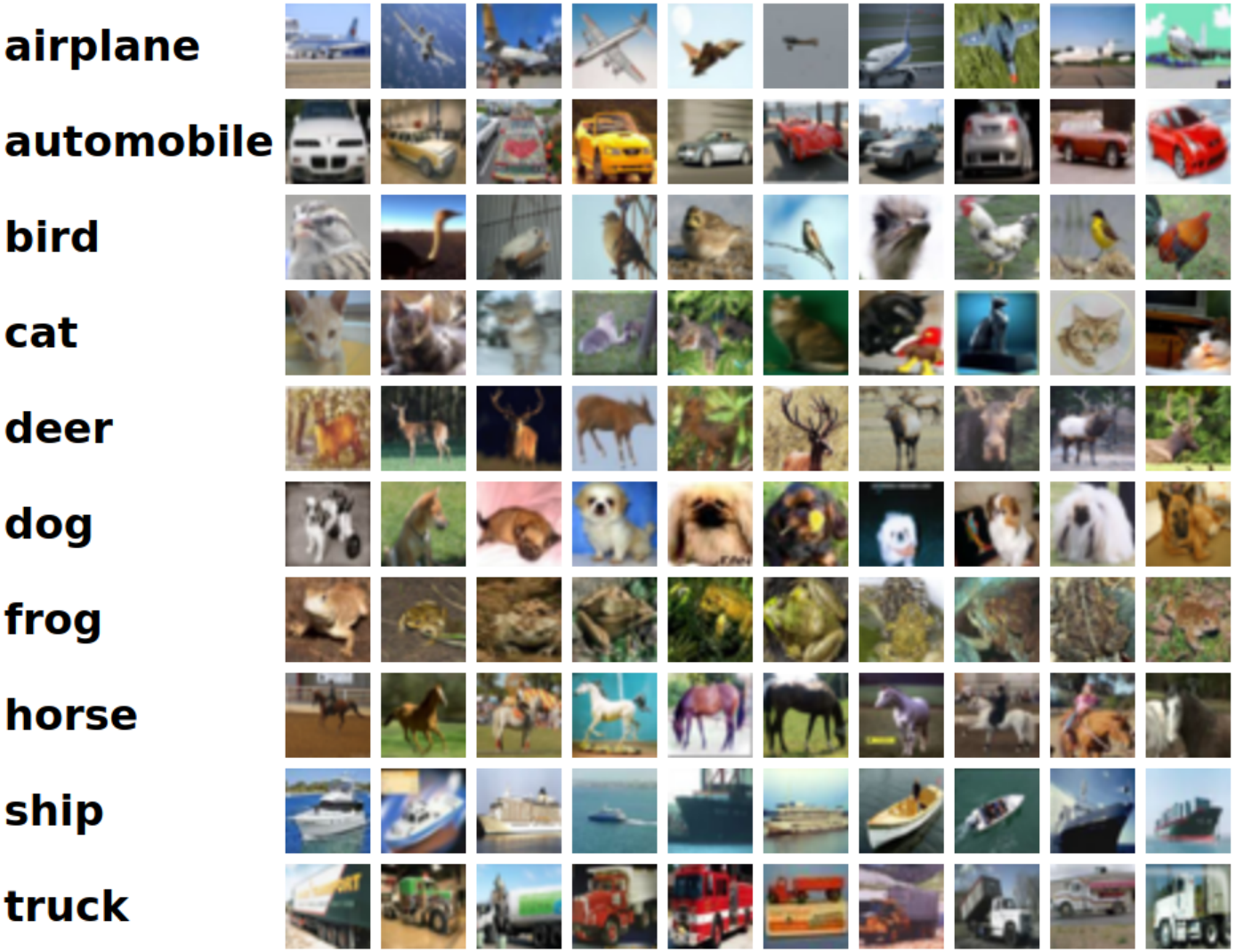
[Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE, 86(11):2278-2324, November 1998.]

FashionMNIST (70,000 28×28 images of fashion items)



[Xiao, Han, Kashif Rasul, and Roland Vollgraf. "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms." arXiv preprint arXiv:1708.07747 (2017).]

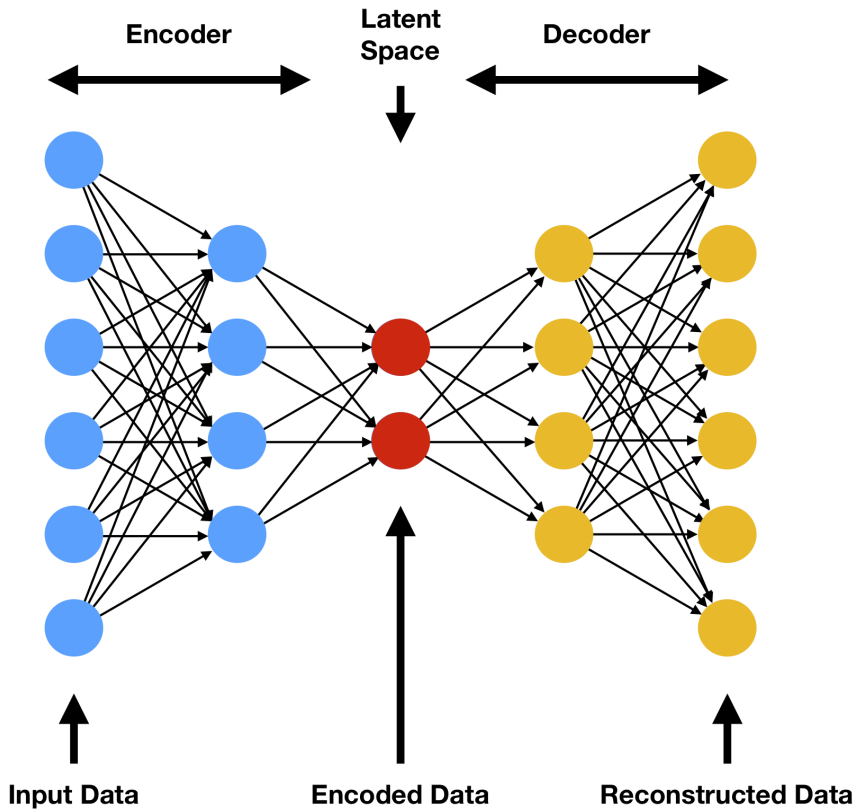
CIFAR-10



[Krizhevsky, Alex, and Geoffrey Hinton. "Learning multiple layers of features from tiny images." (2009): 7.]

Autoencoders

For each dataset, we build the graph by training autoencoders.



`www.compthree.com`

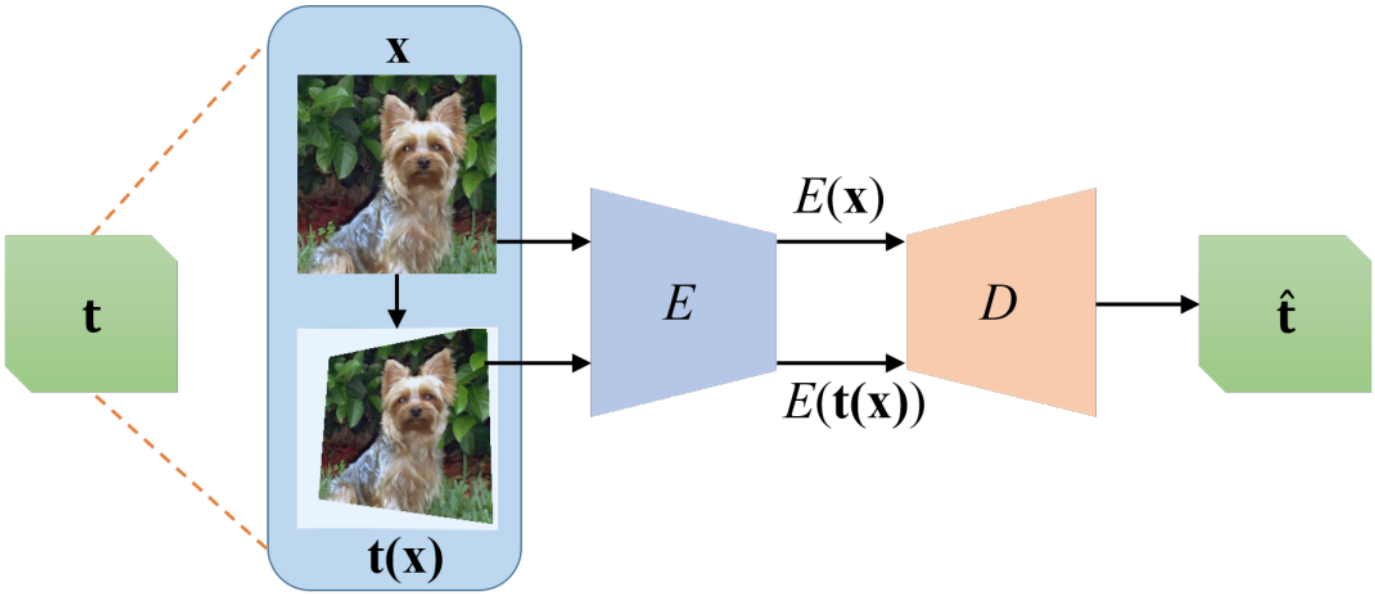
Autoencoders are "Nonlinear versions of PCA"

Building graphs from autoencoders

For MNIST and FashionMNIST, we use a 4-layer variational autoencoder with 20 (MNIST) and 30 (FashionMNIST) latent variables:

[Kingma and Welling. Auto-encoding variational Bayes. ICML 2014]

For CIFAR-10, we use the autoencoding framework from [Zhang et al. AutoEncoding Transformations (AET), CVPR 2019] with thousands of latent features.



Building graphs from autoencoders

After training autoencoders, we build a $k = 10$ nearest neighbor graphs in the latent space with Gaussian weights

$$w_{ij} = \exp\left(-\frac{4|x_i - x_j|^2}{d_k(x_i)^2}\right),$$

where $d_k(x_i)$ is the distance in the latent space between x_i and its k^{th} nearest neighbor. The weight matrix was then symmetrized by replacing W with $W + W^T$.

For CIFAR-10, the latent feature vectors were normalized to unit norm (equivalent to using an angular similarity).

Other algorithms

We compared against many algorithms:

- Graph nearest neighbor for a baseline
- Laplace/Label propagation: [Zhu et al., 2003]
- Lazy random walks: [Zhou et al., 2004]
- Mutli-class MBO: [Garcia-Cardona et al., 2014]
- Sparse Label Propagation: [Jung et al., 2016]
- Volume constrained MBO: [Jacobs et al., 2017]
- Weighted Nonlocal Laplacian (WNLL): [Shi et al., 2017]
- Centered kernel method: [Mai & Couillet, 2018]
- p-Laplace regularization: [Flores et al. 2019]

MNIST results

Table: Average (standard deviation) classification accuracy over 100 trials.

# Labels per class	**1**	**2**	**3**	**4**	**5**
Laplace/LP	16.1 (6.2)	28.2 (10.3)	42.0 (12.4)	57.8 (12.3)	69.5 (12.2)
Nearest Neighbor	55.8 (5.1)	65.0 (3.2)	68.9 (3.2)	72.1 (2.8)	74.1 (2.4)
Random Walk	66.4 (5.3)	76.2 (3.3)	80.0 (2.7)	82.8 (2.3)	84.5 (2.0)
MBO	19.4 (6.2)	29.3 (6.9)	40.2 (7.4)	50.7 (6.0)	59.2 (6.0)
VolumeMBO	89.9 (7.3)	95.6 (1.9)	96.2 (1.2)	96.6 (0.6)	96.7 (0.6)
WNLL	55.8 (15.2)	82.8 (7.6)	90.5 (3.3)	93.6 (1.5)	94.6 (1.1)
Centered Kernel	19.1 (1.9)	24.2 (2.3)	28.8 (3.4)	32.6 (4.1)	35.6 (4.6)
Sparse LP	14.0 (5.5)	14.0 (4.0)	14.5 (4.0)	18.0 (5.9)	16.2 (4.2)
p-Laplace	72.3 (9.1)	86.5 (3.9)	89.7 (1.6)	90.3 (1.6)	91.9 (1.0)
Poisson	90.2 (4.0)	93.6 (1.6)	94.5 (1.1)	94.9 (0.8)	95.3 (0.7)
PoissonMBO	**96.5 (2.6)**	**97.2 (0.1)**	**97.2 (0.1)**	**97.2 (0.1)**	**97.2 (0.1)**

FashionMNIST results

Table: Average (standard deviation) classification accuracy over 100 trials.

# Labels per class	**1**	**2**	**3**	**4**	**5**
Laplace/LP	18.4 (7.3)	32.5 (8.2)	44.0 (8.6)	52.2 (6.2)	57.9 (6.7)
Nearest Neighbor	44.5 (4.2)	50.8 (3.5)	54.6 (3.0)	56.6 (2.5)	58.3 (2.4)
Random Walk	49.0 (4.4)	55.6 (3.8)	59.4 (3.0)	61.6 (2.5)	63.4 (2.5)
MBO	15.7 (4.1)	20.1 (4.6)	25.7 (4.9)	30.7 (4.9)	34.8 (4.3)
VolumeMBO	54.7 (5.2)	61.7 (4.4)	66.1 (3.3)	68.5 (2.8)	70.1 (2.8)
WNLL	44.6 (7.1)	59.1 (4.7)	64.7 (3.5)	67.4 (3.3)	70.0 (2.8)
Centered Kernel	11.8 (0.4)	13.1 (0.7)	14.3 (0.8)	15.2 (0.9)	16.3 (1.1)
Sparse LP	14.1 (3.8)	16.5 (2.0)	13.7 (3.3)	13.8 (3.3)	16.1 (2.5)
p-Laplace	54.6 (4.0)	57.4 (3.8)	65.4 (2.8)	68.0 (2.9)	68.4 (0.5)
Poisson	60.8 (4.6)	66.1 (3.9)	69.6 (2.6)	71.2 (2.2)	72.4 (2.3)
PoissonMBO	**62.0 (5.7)**	**67.2 (4.8)**	**70.4 (2.9)**	**72.1 (2.5)**	**73.1 (2.7)**

Note: Compare to clustering result of 67.2% [McConville et al., 2019]

CIFAR-10 results

Table: Average (standard deviation) classification accuracy over 100 trials.

# Labels per class	**1**	**2**	**3**	**4**	**5**
Laplace/LP	10.4 (1.3)	11.0 (2.1)	11.6 (2.7)	12.9 (3.9)	14.1 (5.0)
Nearest Neighbor	31.4 (4.2)	35.3 (3.9)	37.3 (2.8)	39.0 (2.6)	40.3 (2.3)
Random Walk	36.4 (4.9)	42.0 (4.4)	45.1 (3.3)	47.5 (2.9)	49.0 (2.6)
MBO	14.2 (4.1)	19.3 (5.2)	24.3 (5.6)	28.5 (5.6)	33.5 (5.7)
VolumeMBO	38.0 (7.2)	46.4 (7.2)	50.1 (5.7)	53.3 (4.4)	55.3 (3.8)
WNLL	16.6 (5.2)	26.2 (6.8)	33.2 (7.0)	39.0 (6.2)	44.0 (5.5)
Centered Kernel	15.4 (1.6)	16.9 (2.0)	18.8 (2.1)	19.9 (2.0)	21.7 (2.2)
Sparse LP	11.8 (2.4)	12.3 (2.4)	11.1 (3.3)	14.4 (3.5)	11.0 (2.9)
p-Laplace	26.0 (6.7)	35.0 (5.4)	42.1 (3.1)	48.1 (2.6)	49.7 (3.8)
Poisson	40.7 (5.5)	46.5 (5.1)	49.9 (3.4)	52.3 (3.1)	53.8 (2.6)
PoissonMBO	**41.8 (6.5)**	**50.2 (6.0)**	**53.5 (4.4)**	**56.5 (3.5)**	**57.9 (3.2)**

Note: Compare to clustering result of 41.2% [Mukherjee et al., ClusterGAN, CVPR 2019].

FashionMNIST at moderate label rates

Table: Average (standard deviation) classification accuracy over 100 trials.

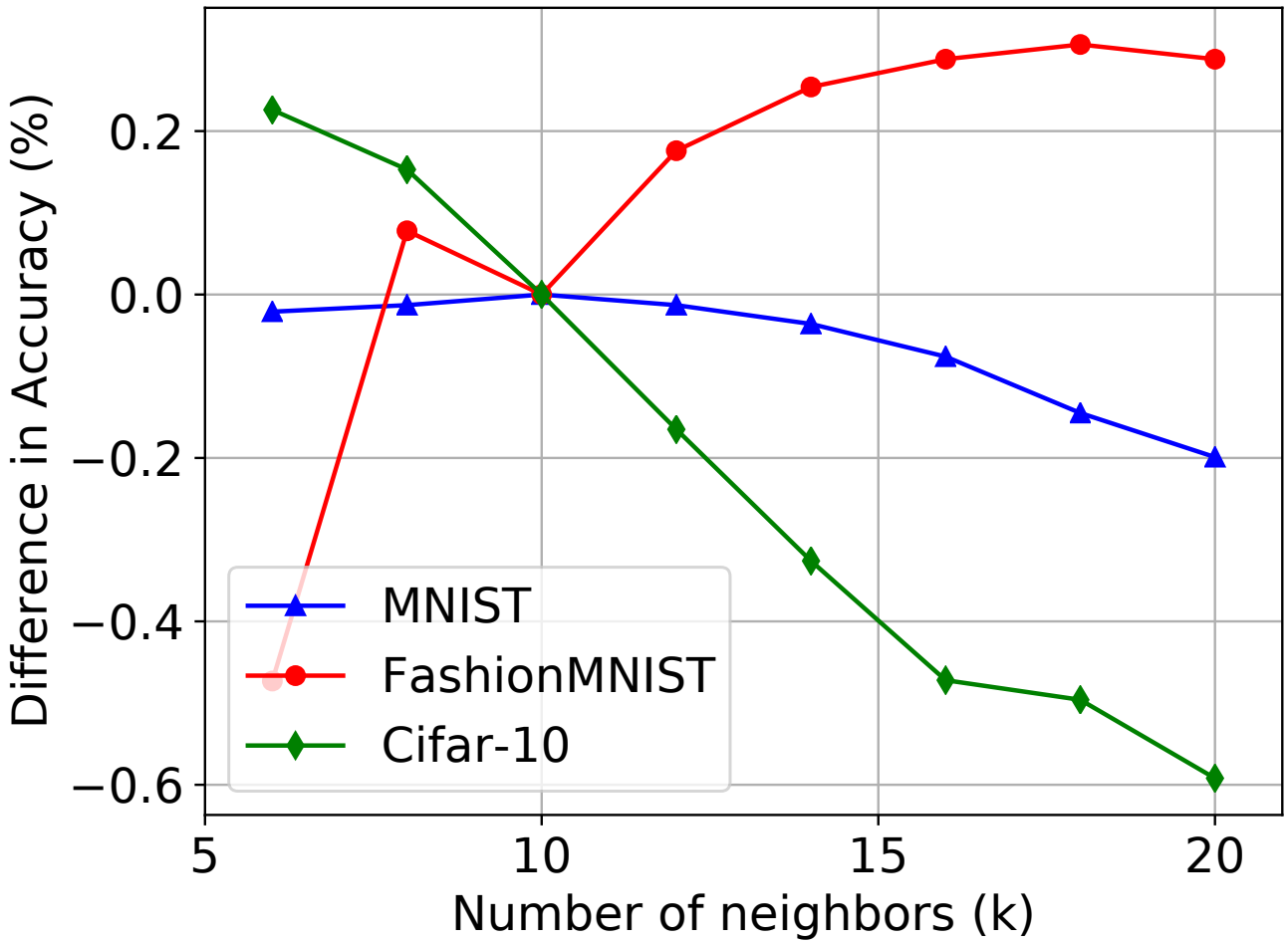
# Labels per class	**10**	**20**	**40**	**80**	**160**
Laplace/LP	70.6 (3.1)	76.5 (1.4)	79.2 (0.7)	80.9 (0.5)	**82.3 (0.3)**
Nearest Neighbor	62.9 (1.7)	66.9 (1.1)	70.0 (0.8)	72.5 (0.6)	74.7 (0.4)
Random Walk	68.2 (1.6)	72.0 (1.0)	75.0 (0.7)	77.4 (0.5)	79.5 (0.3)
MBO	52.7 (4.1)	67.3 (2.0)	75.7 (1.1)	79.6 (0.7)	81.6 (0.4)
VolumeMBO	74.4 (1.5)	77.4 (1.0)	79.5 (0.7)	**81.0 (0.5)**	82.1 (0.3)
WNLL	74.4 (1.6)	77.6 (1.1)	79.4 (0.6)	80.6 (0.4)	81.5 (0.3)
Centered Kernel	20.6 (1.5)	27.8 (2.3)	37.9 (2.6)	51.3 (3.3)	64.3 (2.6)
Sparse LP	15.2 (2.5)	15.9 (2.0)	14.5 (1.5)	13.8 (1.4)	51.9 (2.1)
p-Laplace	73.0 (0.9)	76.2 (0.8)	78.0 (0.3)	79.7 (0.5)	80.9 (0.3)
Poisson	75.2 (1.5)	77.3 (1.1)	78.8 (0.7)	79.9 (0.6)	80.7 (0.5)
PoissonMBO	**76.1 (1.4)**	**78.2 (1.1)**	**79.5 (0.7)**	80.7 (0.6)	81.6 (0.5)

Cifar-10 at moderate label rates

Table: Average (standard deviation) classification accuracy over 100 trials.

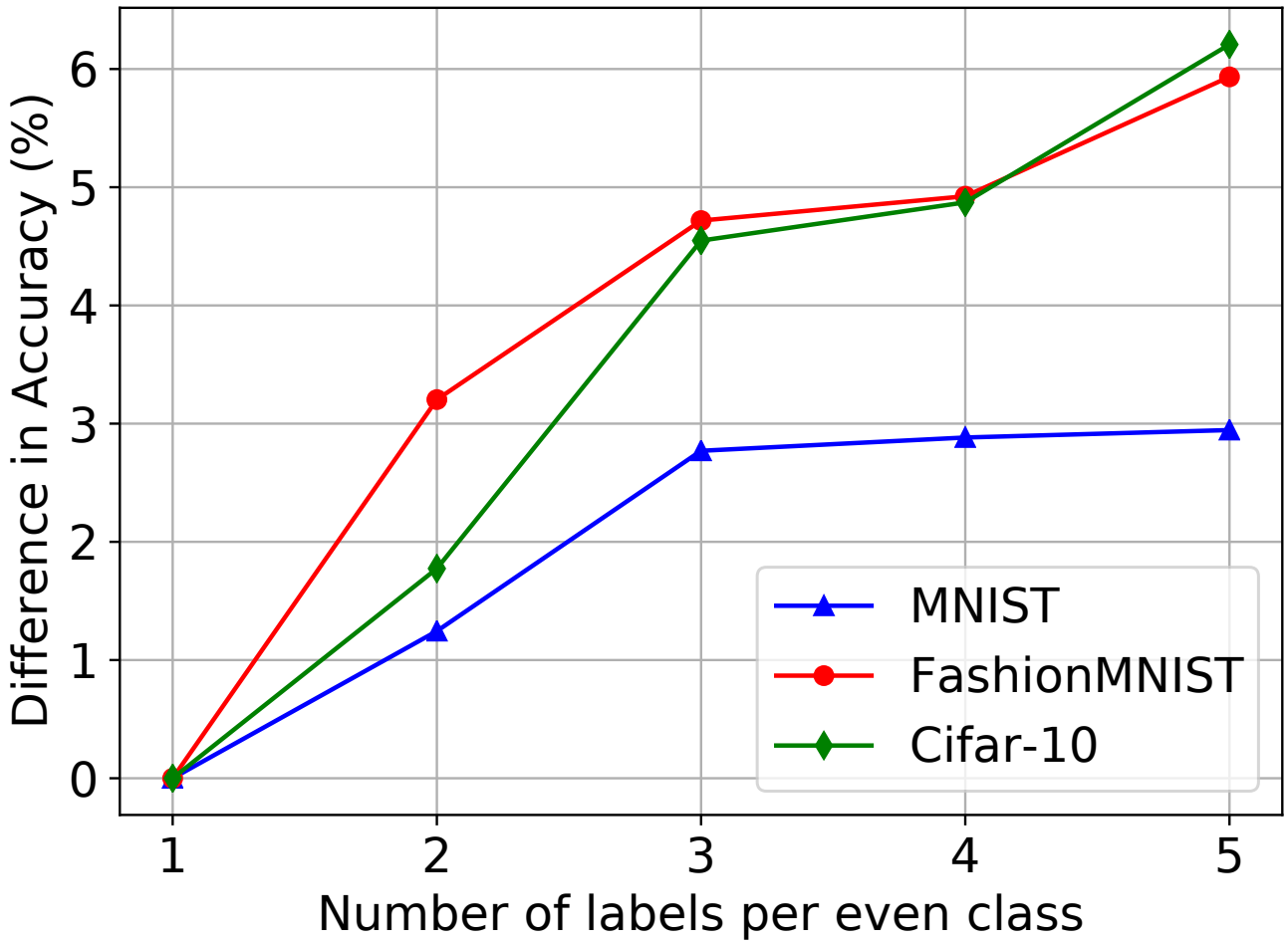
# Labels per class	**10**	**20**	**40**	**80**	**160**
Laplace/LP	21.8 (7.4)	38.6 (8.2)	54.8 (4.4)	62.7 (1.4)	66.6 (0.7)
Nearest Neighbor	43.3 (1.7)	46.7 (1.2)	49.9 (0.8)	52.9 (0.6)	55.5 (0.5)
Random Walk	53.9 (1.6)	57.9 (1.1)	61.7 (0.6)	65.4 (0.5)	68.0 (0.4)
MBO	46.0 (4.0)	56.7 (1.9)	62.4 (1.0)	65.5 (0.8)	68.2 (0.5)
VolumeMBO	59.2 (3.2)	61.8 (2.0)	63.6 (1.4)	64.5 (1.3)	65.8 (0.9)
WNLL	54.0 (2.8)	60.3 (1.6)	64.2 (0.7)	66.6 (0.6)	68.2 (0.4)
Centered Kernel	27.3 (2.1)	35.4 (1.8)	44.9 (1.8)	53.7 (1.9)	60.1 (1.5)
Sparse LP	15.6 (3.1)	17.4 (3.9)	20.0 (1.9)	21.7 (1.3)	15.0 (1.1)
p-Laplace	56.4 (1.8)	60.4 (1.2)	63.8 (0.6)	66.3 (0.6)	68.7 (0.3)
Poisson	58.3 (1.7)	61.5 (1.3)	63.8 (0.8)	65.6 (0.6)	67.3 (0.4)
PoissonMBO	**61.8 (2.2)**	**64.5 (1.6)**	**66.9 (0.8)**	**68.7 (0.6)**	**70.3 (0.4)**

Varying number of neighbors k



5 labels per class for all classes.

Unbalanced training data



Odd numbered classes got 1 label per class.

Outline


Alamgir, M. and Luxburg, U. V. (2011).
Phase transition in the family of p-resistances.
In *Advances in Neural Information Processing Systems*, pages 379–387.


Amghibech, S. (2003).
Eigenvalues of the discrete p-Laplacian for graphs.
Ars Combinatoria, 67:283–302.


Ando, R. K. and Zhang, T. (2007).
Learning on graph with Laplacian regularization.
In *Advances in Neural Information Processing Systems*, pages 25–32.


Bridle, N. and Zhu, X. (2013).
p-voltages: Laplacian regularization for semi-supervised learning on high-dimensional data.
In *Eleventh Workshop on Mining and Learning with Graphs (MLG2013)*.


Bruna, J. and Mallat, S. (2013).
Invariant scattering convolution networks.
IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(8):1872–1886.


Bühler, T. and Hein, M. (2009).
Spectral clustering based on the graph p-Laplacian.
In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 81–88. ACM.


Calder, J. (2019).
Consistency of Lipschitz learning with infinite unlabeled data and finite labeled data.
To appear in SIAM Journal on Mathematics of Data Science.


Calder, J. (2018).
The game theoretic p-Laplacian and semi-supervised learning with few labels.
Nonlinearity, 32(1).


Calder, J. and Slepčev, D. (2018).
Properly-weighted graph Laplacian for semi-supervised learning.
arXiv:1810.04351.
`https://arxiv.org/abs/1810.04351`.


Chapelle, O., Scholkopf, B., and Zien, A. (2006).
Semi-supervised learning.
MIT.


Chaudhari, P. and Soatto, S. (2018).
Stochastic gradient descent peforms variational inference, converges to limit cycles for deep networks.
arXiv:1710.11029.

El Alaoui, A., Cheng, X., Ramdas, A., Wainwright, M. J., and Jordan, M. I. (2016).
Asymptotic behavior of ℓ_p-based Laplacian regularization in semi-supervised learning.
In *Conference on Learning Theory*, pages 879–906.

Finlay, C., Abbasi, B., Calder, J., and Oberman, A. M. (2018).
Lipschitz regularized Deep Neural Networks generalize and are adversarially robust.
arXiv:1808.09540.

Flores, M., Calder, J., and Lerman, G. (2019).
Algorithms for Lp-based semi-supervised learning on graphs.
arXiv:1901.05031.

He, J., Li, M., Zhang, H.-J., Tong, H., and Zhang, C. (2006).
Generalized manifold-ranking-based image retrieval.
IEEE Transactions on Image Processing, 15(10):3170–3177.

Kyng, R., Rao, A., Sachdeva, S., and Spielman, D. A. (2015).
Algorithms for Lipschitz learning on graphs.
In *Conference on Learning Theory*, pages 1190–1223.

Luo, D., Huang, H., Ding, C., and Nie, F. (2010).
On the eigenvectors of p-Laplacian.
Machine Learning, 81(1):37–51.


Luxburg, U. v. and Bousquet, O. (2004).
Distance-based classification with lipschitz functions.
Journal of Machine Learning Research, 5(Jun):669–695.


Nadler, B., Srebro, N., and Zhou, X. (2009).
Semi-supervised learning with the graph Laplacian: The limit of infinite unlabelled data.
Advances in Neural Information Processing Systems, 22:1330–1338.


Shi, Z., Osher, S., and Zhu, W. (2017).
Weighted nonlocal Laplacian on interpolation from sparse data.
Journal of Scientific Computing, 73(2-3):1164–1177.


Slepčev, D. and Thorpe, M. (2019).
Analysis of p-Laplacian regularization in semisupervised learning.
SIAM Journal on Mathematical Analysis, 51(3):2085–2120.


Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014).
Dropout: a simple way to prevent neural networks from overfitting.
The journal of machine learning research, 15(1):1929–1958.


Wang, Y., Cheema, M. A., Lin, X., and Zhang, Q. (2013).
Multi-manifold ranking: Using multiple features for better image retrieval.
In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 449–460. Springer.


Xu, B., Bu, J., Chen, C., Cai, D., He, X., Liu, W., and Luo, J. (2011).
Efficient manifold ranking for image retrieval.
In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 525–534. ACM.


Yang, C., Zhang, L., Lu, H., Ruan, X., and Yang, M.-H. (2013).
Saliency detection via graph-based manifold ranking.
In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3166–3173.


Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2017).
Understanding deep learning requires rethinking generalization.
ICLR.


Zhou, D., Huang, J., and Schölkopf, B. (2005).
Learning from labeled and unlabeled data on a directed graph.
In *Proceedings of the 22nd International Conference on Machine Learning*, pages 1036–1043. ACM.


Zhou, D. and Schölkopf, B. (2005).
Regularization on discrete spaces.
In *Joint Pattern Recognition Symposium*, pages 361–368. Springer.


Zhou, X., Belkin, M., and Srebro, N. (2011).
An iterated graph Laplacian approach for ranking on manifolds.
In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 877–885. ACM.


Zhu, X., Ghahramani, Z., and Lafferty, J. D. (2003).
Semi-supervised learning using Gaussian fields and harmonic functions.
In *Proceedings of the 20th International Conference on Machine learning (ICML-03)*, pages 912–919.