Improved Large-Scale Graph Learning through Ridge Spectral Sparsification

Daniele Calandriello, Ioannis Koutis, Alessandro Lazaric, Michal Valko
SequeL INRIA Lille, LCSL-IIT/MIT, NJIT, FAIR

ICML 2018

Graph Learning

Graph are ubiquitous in machine learning:

constructed graphs	discretized PDE, similarity function
natural graphs	social networks, gene interaction, co-purchase

Machine learning is ubiquitous on graphs:

De-noising

Semi-Supervised Learning (SSL)/Label propagation

Spectral clustering

Pagerank

Facebook's trillion edge graph: $n = 10^9$ and $m = 10^{12}$ [Ching et al. 2015]

Large graphs do not fit in a single machine memory

Scalable Graph Learning

$\mathcal{G}$ with n nodes and m edges
↳ naively $\mathcal{O}(m)$ space and $\mathcal{O}(mt)$ time for $t \leq n$ iterations

Hard to solve with engineering:
↳ multiple passes slow, distribution has communication costs

Black-box acceleration methods:

Reduce iterations t: fast graph solvers $\mathcal{O}(m \cdot \log(n))$ time
[Koutis et al. 2011; Kyng and Sachdeva 2016]

Reduce the number of edges m

Hard to do in natural graphs where sparsity level cannot be chosen
↳ removing edges impacts structure/accuracy

Make the graph sparse, while preserving its structure for learning

Graph Spectral Sparsification

Definition (Spielman and Srivastava 2011)

An ε-sparsifier of $\mathcal{G}$ is a reweighted subgraph $\mathcal{H}$ whose Laplacian $\mathbf{L}_{\mathcal{H}}$ satisfies

$$\boxed{(1-\varepsilon)\mathbf{L}_{\mathcal{G}} \preceq \mathbf{L}_{\mathcal{H}} \preceq (1+\varepsilon)\mathbf{L}_{\mathcal{G}}} \tag{1}$$

Proposition (Spielman and Srivastava 2011; Kyng, Pachocki, et al. 2016)

There exists an algorithm that can construct an ε-sparsifier

with only $\mathcal{O}(n\log(n)/\varepsilon^2)$ edges

in $\mathcal{O}(m\log^2(n))$ time and $\mathcal{O}(n\log(n)/\varepsilon^2)$ space

a single pass over the data

Graph Spectral Sparsification in Machine Learning

Laplacian smoothing (denoising): given $\mathbf{y} \triangleq \mathbf{f}^\star + \xi$ and $\mathcal{G}$ compute

$$\min_{\mathbf{f}\in\mathbb{R}^n} (\mathbf{f}-\mathbf{y})^\mathsf{T}(\mathbf{f}-\mathbf{y}) + \lambda \mathbf{f}^\mathsf{T}\mathbf{L}_\mathcal{G}\mathbf{f} \tag{2}$$

	Preproc	Time	Space
$\widehat{\mathbf{f}} = (\lambda \mathbf{L}_\mathcal{G} + \mathbf{I})^{-1}\mathbf{y}$	0	$\mathcal{O}(m\log(n))$	$\mathcal{O}(m)$
$\widetilde{\mathbf{f}} = (\lambda \mathbf{L}_\mathcal{H} + \mathbf{I})^{-1}\mathbf{y}$	$\mathcal{O}(m\log^2(n))$	$\mathcal{O}(n\log^2(n))$	$\mathcal{O}(n\log(n))$

Large computational improvement
↳ accuracy guarantees! [Sadhanala et al. 2016]

Need to approximate spectrum only up to regularization level λ

Ridge Graph Spectral Sparsification

Definition (This paper)

An (ε, γ)-sparsifier of $\mathcal{G}$ is a reweighted subgraph $\mathcal{H}$ whose Laplacian $\mathbf{L}_{\mathcal{H}}$ satisfies

$$\boxed{(1-\varepsilon)\mathbf{L}_{\mathcal{G}} - \varepsilon\gamma\mathbf{I} \preceq \mathbf{L}_{\mathcal{H}} \preceq (1+\varepsilon)\mathbf{L}_{\mathcal{G}} + \varepsilon\gamma\mathbf{I}} \qquad (3)$$

Mixed multiplicative/additive error

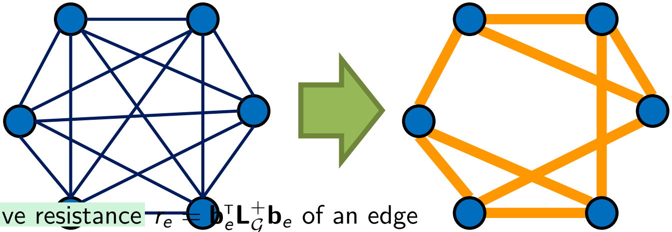
- large (i.e. $\geq \gamma$) directions reconstructed accurately
- small (i.e. $\leq \gamma$) directions uniformly approximated ($\gamma\mathbf{I}$)

Adapted from Randomized Linear Algebra (RLA) community
↳ PSD matrix low-rank approx. [Alaoui and Mahoney 2015]

RLA $\rightarrow$ Graph: Improve over O(n log(n)) size exploiting regularization
Graph $\rightarrow$ RLA: Exploit $\mathbf{L}_{\mathcal{G}}$ structure for fast (ε, γ)-sparsification

How to construct an ε-sparsifier

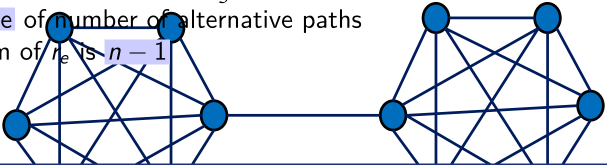
For complete graphs, sample $\mathcal{O}(n\log(n))$ edges uniformly and reweight For generic graphs, sample $\mathcal{O}(n\log(n))$ edges uniformly? For generic graphs, sample $\mathcal{O}(n\log(n))$ edges uniformly? For generic graphs, sample $\mathcal{O}(n\log(n))$ edges using effective resistance



Effective resistance $r_e = \mathbf{b}_e^\top \mathbf{L}_{\mathcal{G}}^{+} \mathbf{b}_e$ of an edge

↳ inverse of number of alternative paths

↳ sum of r_e is $n-1$



How to construct an (ε, γ)-sparsifier

Definition

γ-effective resistance: $r_e(\gamma) = \mathbf{b}_e^\mathsf{T}(\mathbf{L}_\mathcal{G} + \gamma\mathbf{I})^{-1}\mathbf{b}_e$

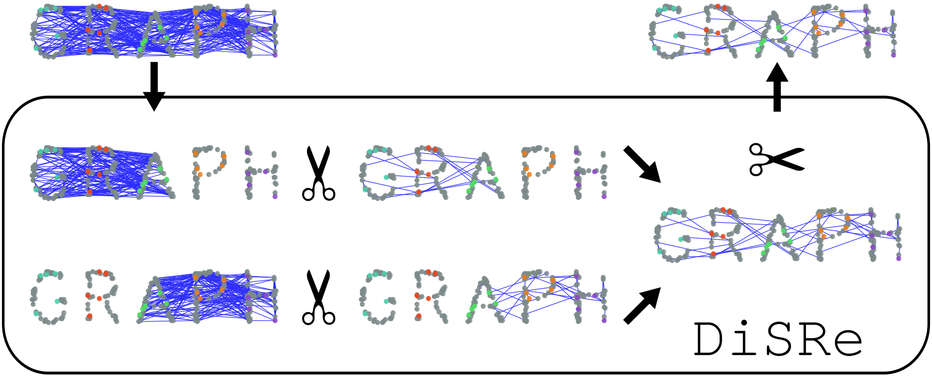
Effective dim.: $\mathbf{d}_{\text{eff}}(\gamma) = \sum_e r_e(\gamma) = \sum_{i=1}^{n} \frac{\lambda_i(\mathbf{L}_\mathcal{G})}{\lambda_i(\mathbf{L}_\mathcal{G})+\gamma} \leq n$

Can still be computed using fast graph solvers
↳ interpretation as inverse of alternative paths lost

Most existing graph algorithms inapplicable [Kyng, Pachocki, et al. 2016]
Most existing RLA algorithms too slow [Cohen et al. 2017]

Adapt SOA algorithm for kernel matrix approximation
SQUEAK, Calandriello et al. 2017

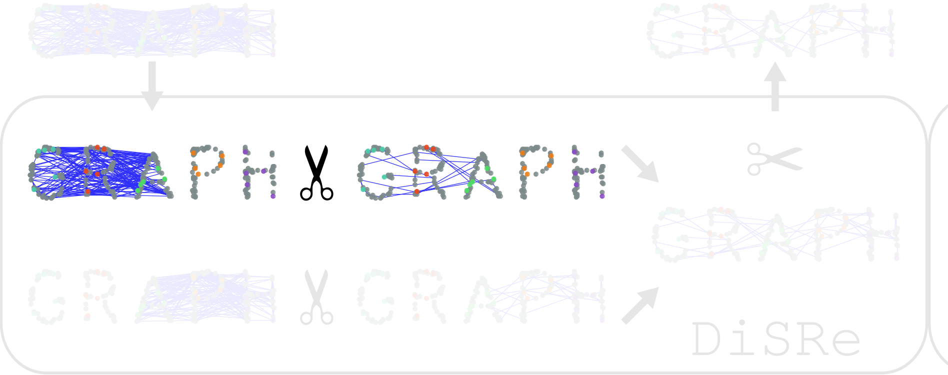
DisRe



arbitrarily split in subgraphs that fit in a single machine

recursively merge-and-reduce until one graph left
↳ additive error cumulates!
  ↳ merge-and-resparsify

Sparsification

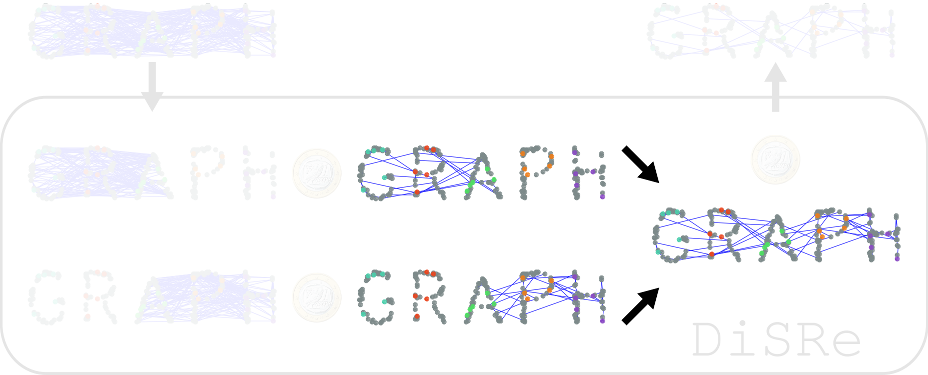


Compute $\widetilde{p}_e^{(1)} \propto \widetilde{r}_e^{(1)}(\gamma)$ using fast graph solver

For each edge e sample with probability $\widetilde{p}_e^{(1)}$

w.h.p. (ε, γ)-accurate and use only $\mathcal{O}(d_{\text{eff}}(\gamma)\log(n)) \leq \mathcal{O}(n\log(n))$ space

Merge



Combine sparsifiers, using $2\mathcal{O}(d_{\text{eff}}(\gamma)\log(n))$ space

twice as large as necessary

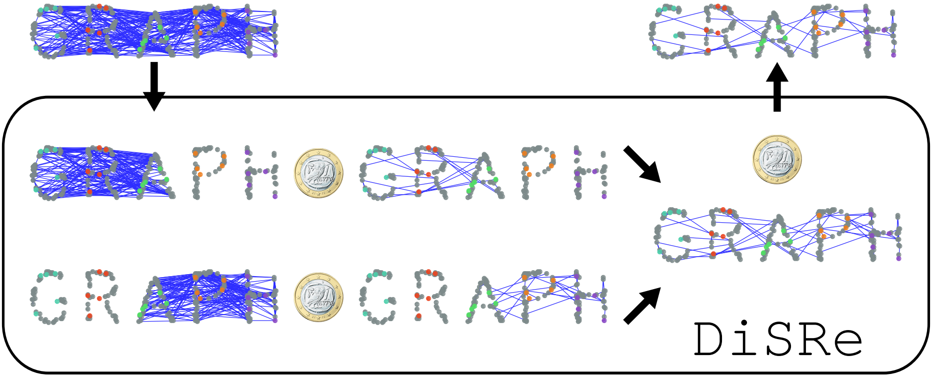
Merge-and-Resparsify

Compute $\widetilde{p}_e^{(2)} \propto \min\{\widetilde{r}_e^{(2)}(\gamma), \widetilde{p}_e^{(1)}\}$ using fast graph solver

For each edge e sample with probability $\widetilde{p}_e^{(2)}/\widetilde{p}_e^{(1)}$

$$\text{survival probability} \quad \frac{\widetilde{p}_e^{(2)}}{\widetilde{p}_e^{(1)}}\,\widetilde{p}_e^{(1)}$$

DisRe guarantees



Theorem

Given an arbitrary graph $\mathcal{G}$ w.h.p. DisRe satisfies

(1) each sub-graphs is an (ε, γ)-sparsifier

(2) with at most $\mathcal{O}(d_{\text{eff}}(\gamma)\log(n))$ edges.

Guarantees for Laplacian smoothing

$$\widehat{\mathbf{f}} = (\lambda \mathbf{L}_{\mathcal{G}} + \mathbf{I})^{-1}\mathbf{y}, \qquad \widetilde{\mathbf{f}} = (\lambda \mathbf{L}_{\mathcal{H}} + \mathbf{I})^{-1}\mathbf{y}$$

Theorem (Sadhanala et al. 2016 This paper)

If $\mathbf{L}_{\mathcal{H}}$ is an $(\varepsilon, 0)$ (ε, γ)-sparsifier of $\mathbf{L}_{\mathcal{G}}$

$$\|\widetilde{\mathbf{f}} - \widehat{\mathbf{f}}\|_2^2 \leq \frac{\varepsilon^2}{1-\varepsilon}(0.25 + \lambda\gamma)\left(\lambda \widehat{\mathbf{f}}^{\mathsf{T}} \mathbf{L}_{\mathcal{G}} \widehat{\mathbf{f}} + \lambda\gamma \|\widehat{\mathbf{f}}\|_2^2\right).$$

$\mathcal{O}(d_{\mathrm{eff}}(\gamma)\log(n))$ space, $\mathcal{O}(d_{\mathrm{eff}}(\gamma)\log^3(n))$ time
↳ exploit regularization: $\mathcal{H}$ sub-linear in n

Recover bound for ε-sparsifier when $\gamma \to 0$
↳ freely cross-validate γ since $d_{\mathrm{eff}}(0) \leq n$
↳ trade-off between smoothness and decay of $\mathbf{L}_{\mathcal{G}}$

Experiments

Dataset: Amazon co-purchase graph [Yang and Leskovec 2015]
↳ natural, artificially sparse (true graph known only to Amazon)
↳ we compute 4-step random walk to recover removed co-purchases [Gleich and Mahoney 2015]

Target: eigenvector $\mathbf{v}$ associated with $\lambda_2(\mathbf{L}_{\mathcal{G}})$ [Sadhanala et al. 2016]

$n = 334,863$ nodes, $m = 98,465,352$ edges (294 avg. degree)

Alg.	*Parameters*	$\|\mathcal{E}\|$ *(x10^6)*	$\|\|\widetilde{\mathbf{f}} - \mathbf{v}\|\|_2^2$ $(\sigma = 10^{-3})$	$\|\|\widetilde{\mathbf{f}} - \mathbf{v}\|\|_2^2$ $(\sigma = 10^{-2})$
EXACT		98.5	0.067 ± 0.0004	0.756 ± 0.006
kN	$k = 60$	15.7	0.172 ± 0.0004	0.822 ± 0.002
DisRe	$\gamma = 0$	22.8	0.068 ± 0.0004	**0.756** ± 0.005
DisRe	$\gamma = 10^2$	11.8	**0.068** ± 0.0002	0.772 ± 0.004

Time: Loading $\mathcal{G}$ from disk 90sec, DisRe 120sec($k = 4 \times 32$ CPU), computing $\widetilde{\mathbf{f}}$ 120sec, computing $\widehat{\mathbf{f}}$ 720sec

Recap and open questions

Remark (Sadhanala et al. 2016)

To the best of our knowledge, [graph sparsification] applications in machine learning have not yet been thoroughly pursued.

- introduction of (ε, γ)-sparsifiers to Graph ML
- DisRe, new distributed algorithm to construct (ε, γ)-sparsifiers
- new results for fast Laplacian Smoothing
- new results for fast SSL using ε-sparsifiers (at poster #76)

Open questions

- other accelerated Graph ML algorithms using (ε, γ)-sparsifiers
- more experiments on dense graphs
- Facebook: 300 average friends [Pew Research Center 2013]
- Twitter 453 average followers, 3.4x denser 2012-16 [Leskovec et al. 2007]

Bibliography I

Ahmed El Alaoui and Michael W. Mahoney. "Fast randomized kernel methods with statistical guarantees". In: Neural Information Processing Systems. 2015. URL: https://papers.nips.cc/paper/5716-fast-randomized-kernel-ridge-regression-with-statistical-guarantees.pdf.

Daniele Calandriello, Alessandro Lazaric, and Michal Valko. "Distributed Sequential Sampling for Kernel Matrix Approximation". In: International conference on Artificial Intelligence and Statistics. 2017. URL: http://proceedings.mlr.press/v54/calandriello17a/calandriello17a-supp.pdf.

Avery Ching, Sergey Edunov, Maja Kabiljo, Dionysios Logothetis, and Sambavi Muthukrishnan. "One trillion edges: graph processing at Facebook-scale". In: Proceedings of the VLDB Endowment 8.12 (2015), pp. 1804–1815.

Bibliography II

- Michael B Cohen, Cameron Musco, and Christopher Musco. "Input sparsity time low-rank approximation via ridge leverage score sampling". In: Symposium on Discrete Algorithms. 2017. URL: https://arxiv.org/pdf/1511.07263.pdf.
- David F. Gleich and Michael W. Mahoney. "Using Local Spectral Methods to Robustify Graph-Based Learning Algorithms". In: Knowledge Discovery and Data Mining. 2015, pp. 359–368. URL: https://www.stat.berkeley.edu/~mmahoney/pubs/robustifying-kdd15.pdf.
- Ioannis Koutis, Gary L. Miller, and Richard Peng. "A Nearly-m log n time solver for SDD Linear Systems". In: Symposium on Foundations of Computer Science. 2011, pp. 590–598. URL: https://arxiv.org/pdf/1102.4842.pdf.

Bibliography III

Rasmus Kyng, Jakub Pachocki, Richard Peng, and Sushant Sachdeva. "A Framework for Analyzing Resparsification Algorithms". In: Symposium on Theory of Computing. 2016. URL: https://arxiv.org/pdf/1611.06940.pdf.

Rasmus Kyng and Sushant Sachdeva. "Approximate gaussian elimination for laplacians-fast, sparse, and simple". In: Foundations of Computer Science. 2016. URL: https://arxiv.org/pdf/1605.02353.pdf.

Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. "Graph evolution: Densification and shrinking diameters". In: ACM Transactions on Knowledge Discovery from Data (TKDD) 1.1 (2007), p. 2.

Bibliography IV

- Veeru Sadhanala, Yu-Xiang Wang, and Ryan Tibshirani. "Graph sparsification approaches for laplacian smoothing". In: International conference on Artificial Intelligence and Statistics. 2016. URL: http://proceedings.mlr.press/v51/sadhanala16.pdf.
- Daniel A. Spielman and Nikhil Srivastava. "Graph Sparsification by Effective Resistances". In: SIAM Journal on Computing 40.6 (2011). URL: https://arxiv.org/pdf/0803.0929.pdf.
- Jaewon Yang and Jure Leskovec. "Defining and evaluating network communities based on ground-truth". In: Knowledge and Information Systems 42.1 (2015), pp. 181–213.