Project Proposal
Data-sparse matrix computations (CS 6220)
Prof. **Anil Damle**

Xinran Zhu (NetID: xz584)
Cornell University
Submit Date: April 22, 2020

Summary

In 2015, Xie etc. built a fast approximation of edge-weighted personalized PageRank based on model reduction, which enables learning-to-rank at interactive speeds and scales well to large graphs [1]. In light of this fast approximation of edge-weighted personalized PageRank, we hope to 1) improve some aspects by further theoretical analysis and 2) apply similar fast approximation to large-scale clustering problem, i.e. to tackle the learning to cluster problem.

1 Personalized PageRank

Personalized PageRank, stemming from conventional PageRank, takes into account information beyond graph topology in the sense that nodes or weights are parameterized. With parameterization, PageRank can be *personalized* to specific users or queries. Moreover, real applications have proved that, when other information such as user feedback is incorporated, personalized PageRank models, together with other machine learning methods, could greatly improve ranking results [2, 3, 4, 5].

In edge-weighted personalized PageRank, as in the standard PageRank, the transition matrix $P = AD^{-1}$, where $A \in \mathbb{R}^{n\times n}$ is the weighted adjacency matrix and $D \in \mathbb{R}^{n\times n}$ is the degree matrix which is diagonal with weighted out-degree $d_i = \sum_j A_{ji}$. In edge-weighted personalized PageRank, we solve

$$(I - \alpha P(w))x(w) = (1-\alpha)v \tag{1}$$

where $w \in \mathbb{R}^d$ is a vector of personalization parameters. Letting $M(w) = I - \alpha P(w)$ and $(1-\alpha)v = b$, it is equivalent to solve

$$M(w)x(w) = b \tag{2}$$

When v is dense, (2) can be solved iteratively. However, challenges lie in the cost proportional to the number of edges in the graph since iterative solvers would repeatedly require matrix-vector multiplications involving the matrix $M(w)$.

2 Fast Approximation

To avoid matrix-vector multiplications involving the matrix $M(w)$, we could first seek for a low-rank approximation of the PageRank vector $x(w)$. Specifically, we assume that $x(w)$ can be well approximated in a k-dimensional space $\mathcal{U}$ for any w. Equivalently, we want an approximate PageRank vector $\tilde{x}(w) = Uy(w) \approx x(w)$ for some $y(w) \in \mathbb{R}^k$, where columns of U form a basis of $\mathcal{U}$. In practice U is constructed by applying PCA to a set of PageRank vectors $\{x^{(j)}\}_{j=1}^r$, which are computed from uniform random samples $\{w^{(j)}\}_{j=1}^r$.

Therefore, we replace $x(w)$ by the approximate PageRank vector $\tilde{x}(w)$ and thanks to the low-rank representation, it suffices to solve for $y(w)$ in the framework of Petrov-Galerkin methods,

i.e.

$$W^T[M(w)Uy(w) - b] = 0 \tag{3}$$

where W contains test functions. One of the specific settings in practice is to force a subset of equations in (2), i.e. to solve

$$\min \|[M_{\mathcal{I},:}(w)Uy(w) - b_{\mathcal{I}}]\|^2 \tag{4}$$

Therefore, only a few rows of $M(w)$ need to be computed, which is cheap even for complicated parametrization.

3 Learning to Rank

The fast PageRank approximation enables the learning to rank task, where the parameter w is optimized iteratively based on training data such as user feedback. Specifically, given the training data $T = \{(i_q, j_q)\}$, where $(i, j) \in T$ enforces node i should be ranked higher than node j, the optimization problem is

$$\min_w L(w) = \min_{(i,j)\in T} l(x_i(w) - x_j(w)) + \lambda\|w\|^2 \tag{5}$$

Without low-rank approximation, each evaluation of derivatives requires solving a linear system involving $M(w)$, which is very expensive. With the low-rank approximation, once we have computed $M_{\mathcal{I},:}(w)U$ and its derivatives, computing y only takes $\mathcal{O}(k^2|\mathcal{I}|)$ time. Moreover, evaluation of derivatives could re-utilize previous factorization of $M_{\mathcal{I},:}U$ in solving for $y(w)$ and takes $\mathcal{O}(k|\mathcal{I}|)$ time.

4 Plan for Project

The rough plan is stated here, most of which may need further survey and advice.

4.1 Theory: Model reduction

There are some aspects in the fast approximation of PageRank vector that could be improved potentially. For example, the low-dimensional space $\mathcal{U}$ is constructed from uniformly sampled parameters $\{w^{(j)}\}_{j=1}^r$. In practice, roughly 1000 sample is needed. We could seek for other ways to build the low-dimensional space $\mathcal{U}$ with less evaluation of $x(w)$. Also, we could work on building $\mathcal{U}$ in different ways for different types of parameterization.

In the reduced problem for $y(w)$ (4), when we choose a subset of rows in $M(w)$, we would like $M_{\mathcal{I},:}(w)U$ contains rows that are as linearly independent as possible. The paper applies pivoted QR to the matrix

$$Z = [M(\tilde{w}^{(q)})U, M(\tilde{w}^{(2)})U, \ldots, M(\tilde{w}^{(q)})U)]$$

based on some sampled parameters $\{\tilde{w}^{(j)}\}_{j=1}^q$ and then the number of rows to choose into $\mathcal{I}$ can be as many as kq. We could seek for another way to select rows based on randomized rank-revealing QR directly on $M(w)U$ for a given w and then get rid of sampling $\{\tilde{w}^{(j)}\}_{j=1}^q$.

4.2 Implementation: Learning to cluster

Similar to the learning to rank problem, we could also work on similar fast approximation and solve parameter learning in large-scale (spectrally) clustering problems. Suppose the graph Laplacian is parameterized in some way, i.e. $L(\theta)$, and we could build a k-dimensional invariant subspace $V(\theta)$ and then use k-means on $V(\theta)$ to find clusters. Suppose we have training data $\mathcal{A}_+$ for node pairs to cluster together, and $\mathcal{A}_-$ for node pairs to cluster apart. The loss function could then take the form

$$f(\theta) = \sum_{(i,j)\in\mathcal{A}_+} \|v_i(\theta) - v_j(\theta)\|^2 - \sum_{(i,j)\in\mathcal{A}_-} \|v_i(\theta) - v_j(\theta)\|^2 \tag{6}$$

where $v(\theta)$ is the clustering vector. Using model reduction, we could find an approximation space $\mathcal{V}$ for $V(\theta)$ for any theta and then use similar fast approximation to accelerate the learning.

References

[1] W. Xie, D. Bindel, A. Demers, and J. Gehrke, "Edge-weighted personalized pagerank: Breaking a decade-old performance barrier," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 1325–1334.

[2] B. Gao, T.-Y. Liu, W. Wei, T. Wang, and H. Li, "Semi-supervised ranking on very large graphs with rich metadata," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2011, pp. 96–104.

[3] L. Backstrom and J. Leskovec, "Supervised random walks: predicting and recommending links in social networks," in *Proceedings of the fourth ACM international conference on Web search and data mining*, 2011, pp. 635–644.

[4] Z. Nie, Y. Zhang, J.-R. Wen, and W.-Y. Ma, "Object-level ranking: bringing order to web objects," in *Proceedings of the 14th international conference on World Wide Web*, 2005, pp. 567–574.

[5] W. Feng and J. Wang, "Incorporating heterogeneous information for personalized tag recommendation in social tagging systems," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012, pp. 1276–1284.