CS590-MLG, Course Projects

Pan Li

Please keep the following topics within our course groups and do not distribute! Register your topics via this link. Note that first come first served!

Course projects form the very important section of this course. Please be serious of them! You should get prepared to take risks while thinking in-depth of the problem and the methods that you are going to work on. There are four milestones regarding the course projects.

1. **Project proposal (20%, due at 11:59 PM, Oct. 8th)**. A project proposal is a 2-page write-up that introduces the motivation of the project, the related works, the potential challenges, and several tentative options of the methods to handle those challenges. I suggest you read at least 5 papers to consolidate your idea.
2. **Milestone report (10%, due at 11:59 PM, Nov. 5th)**. A milestone report is a 1-or-2-page write-up that summarizes the preliminary results you have obtained, and introduces the potential steps to achieve some success in the final report. Importantly, if you have encountered any challenges regarding your proposed methods, you probably need to figure out some new options to solve the problems. You may slightly modify the goal of your project to some extend. Please record all you thoughts. Remember the experiences of failure are also important, which I will also take into account in the final evaluation.
3. **Course presentation (20%, 4:30-5:50pm, (perhaps Nov. 26th) Dec. 01st and Dec. 03st)**. Prepare a **20min talk** (17min talk + 3min Q&A) to introduce your project to the entire class. You are allowed to have only a part of (at least 50%) the experimental results at this point. However, you should have a concrete method and some experimental results to support your arguments.
4. **Final report (50%, due at 11:59 PM, Dec. 10th)**. This is where you include all your findings in this project. These is no page limitation but the report should be self-contained. I suggest you use NeurIPS template [1]. I will evaluate your report based on some criterion similar to the workshop paper in top-ranked conferences, for example, Graph mining workshops [2], Graph neural network workshops [3], and some graph computation-related papers [1].

You may think of one problem that is related to graphs and networks by yourself. The potential directions include an extensive empirical analysis of the network data (like [2]), algorithms and models for either a well-known task or new applications (like [3] or you know many others), scalable algorithms for large graphs (like [4], [5] and [6]), and theoretical analysis for graph algorithms (like [7] or [8]).

In the following, I list a few projects as the extension of some previous research or topics in my mind. I group them into several categories according to the problem types: **T**-theory, **C**-computation, **M**-machine learning model. I will also list the suggested knowledge, if needed, to conduct this project. Note that since this is a group work, you should decide which project to work on based on both your knowledge background and your teammates'. Also, please note that you may also need to read at least 2-4 extra papers for each project besides what I list in the introduction of each project.

Also, please do not discriminate these projects based on the length of my introduction. It is just related to how specific that project would be in my mind. Just choose or think of one project that you

[1] https://neurips.cc/Conferences/2020/PaperInformation/StyleFiles

[2] http://www.mlgworkshop.org/2020/

[3] https://grlearning.github.io/papers/

are most excited about! Moreover, feel free to choose projects with some topics in your mind, which I even more encourage. However, for your own project, please make sure you could make a solid contribution to achieve the quality I mentioned above!

1. (C + T) Local (Parallel) Alternative Projection Method to Compute Personalized PageRank over Hypergraphs.

   Knowledge: convex optimization (necessary), parallel computing (suggested)

   Check the reference [9] (temporarily ignore the proof therein and focus on the algorithms), which can be used to compute PageRank with any transportation vectors for the most general hypergraphs. To simplify the problem, you may use a specific definition of hyperedge (some examples shown in Table 1 in [9]). Try to generalize Alg. 2 (alternative projection) to a local algorithm whose complexity does not depend on the whole graph but a local subgraph. An example of local algorithms for Personalized PageRank over graphs can be found [10]. Evaluate your approach over the datasets in [11]. See if you may make your algorithm parallel (in general, alternative projection can be easily parallelized [12]). Also, see if you may prove the convergence rate (optional).

2. (C + T) Local Random Coordinate Descent Method to Compute Personalized PageRank over Hypergraphs.

   Knowledge: convex optimization (necessary)

   Check the reference [9] (temporarily ignore the proof therein and focus on the algorithms), which can be used to compute PageRank with any transportation vectors for the most general hypergraphs. To simplify the problem, you may use a specific definition of hyperedge (some examples shown in Table 1 in [9]). Try to generalize Alg. 1 (random coordinate descent) to a local algorithm whose complexity does not depend on the whole graph but a local subgraph. An example of local algorithms for Personalized PageRank over graphs can be found [10]. Evaluate your approach over the datasets in [11]. See if you may prove the convergence rate (optional).

3. (C) Parallel local random walks for node ranking and community detection investigation over large graphs.

   Knowledge: parallel computing (suggested)

   PageRank has random-walk explanations. The personalized PageRank (PPR) also has correspondingly random-walk explanations [4]. PPR can be used to detect communities while computing even approximating PPR are not easy [13]. Previous methods to approximate PPR [13] are parallel. If one has parallel computing units, say multi-core CPUs, using random walks to approxiamte PPR may be a better way, as random walks can be easily parallelized. Moreover, for the community detection task, the ranks of nodes based on the values of PPR are more important than the exact values of PPR. Random walks are good to approximate the ranks of nodes. Therefore, in parallel systems, it seems that using random walks instead of computing the values of PPR may be more efficient [14]. This project is to investigate the power of different configurations of parallel random walks (number of seeded nodes, random walk lengths, etc.) to compute node ranks. Consider evaluating your parallel random-walk methods in the datasets [15].

4. (C) Time evolving motif clustering/hypergraph clustering

   Knowledge: no

   Use high-order-graph reduction techniques in [3, 16] + PageRank Nibble [13] to replace the tensor method used in [17] for motif clustering/hypergraph clustering in dynamic networks. You should get a more efficient algorithm with better clustering performance.

5. (C + M) Efficient sampling for GNN training over dynamic graphs

   Knowledge: No

   Traditional full-batch training of GNNs cannot be used over large networks due to the limitation of GPU memory. GraphSAGE [18] is able to process large networks by doing subgraph-sampling to form them into mini-batches. However, GraphSAGE may have a lot of computation redundancy. Recent works [4, 19] propose different methods to reduce such redundancy. This goal of this project is to generalize this methods over dynamic networks to reduce the computational redundancy of methods such as [20].

[4] https://www.r-bloggers.com/from-random-walks-to-personalized-pagerank/

6. (C + M + T) Federated learning for GNNs

   Knowledge: distributed optimization (suggested)

   Learn the concept of federated learning [21]. Consider if you may train GNNs in the setting with distributed data [4]. Specifically, consider you have a really large network. Each node only has the access to the features related to its local graph (e.g. within 2 hops or even 1 hop). You also have a center. However, this center does not allow collecting the data but its allows to collect model parameters. Therefore, federated learning for GNNs is to train several GNNs in a distributed fashion by digesting local-graph features and then use the center to synthesize the GNN parameters of all these distributed models after several iterations. Of course, after the center synthesizes the models into one model, it will broadcast the model to every nodes for another round of distributed training. Can you build a GNN model that follows this procedure? Can you empirically or even theoretically prove whether your training procedure may converge? Also, how about its performance? Check more about federated learning.

7. (M) Deep graph generation

   Knowledge: generative models and Bayesian methods (suggested)

   Learn the deep set generation methods [22]. Check previous graph generation approaches [23, 24]. Think about ways to generate graph structures by not using RNN like previous works but the generation procedure guarantees node permutation invariant property, which is a generalized property of the item permutation invariant property in the set generation problem [22]. Try to evaluate your method over both synthetic and real networks.

8. (M) Improve graph similarity computation via more powerful GNNs

   Knowledge: graph isomorphism (suggested)

   Compute graph similarity is an NP-complete problem. Recently, people study to use GNN to do so [25]. However, GNNs have limited power [26]. More powerful GNN models have been proposed recently [27, 28]. This project is to think about whether these more powerful GNNs or other your proposed GNN models may better the computation of graph similarity.

9. (T+M) Investigate the ordered space dimension for subgraph matching

   Knowledge: graph isomorphism (necessary)

   Recent subgraph matching work based on GNNs investigated a new angle, termed ordered space embedding, to efficiently perform subgraph matching and subgraph pattern search [29]. However, an in-depth understanding of such an ordered space is missing: Specifically, how many dimensions one needs to embed all types of graph structures with a given number of nodes so that a partial order relation can be guaranteed. This project is to investigate this ordered space in theory. Also, it would be better if it helps GNN for subgraph matching applications. A relevant mathematical concept can be found in wiki [5] and [29].

10. (M) Attack and defense of GNN models for dynamic networks

    Knowledge: Bayesian methods (suggested)

    GNNs seem to be sensitive to adversarial attack, especially graph structured attack [30]. Some variational inference methods by injecting randomness into the model training are able to yield more robust models [31, 32]. Think about analyzing attack or robust training of GNN models for dynamic networks (for example, the model in [20]).

11. (M) Distance encoding + GNNs for dynamic triangle closure prediction

    Knowledge: graph isomorphism (suggested)

    Read the paper [27]. Understand how distance encoding improves the power of GNNs to represent node sets. Leverage this idea to build GNNs for dynamic triangle closure prediction tasks [33]. Think of better ways to further improve your model to beat baselines in [33] and standard GNN models.

12. (M) Distance encoding + GNNs for hyperedge prediction

    Knowledge: graph isomorphism (suggested)

    Read the paper [27]. Understand how distance encoding improves the power of GNNs to represent node sets. Leverage this idea to build GNNs for hyperedge prediction. One can find hypergraph datasets in [11, 34]. Note that there are a few GNNs proposed for hypergraphs [35–39]. So you should review these works first.

[5] https://en.wikipedia.org/wiki/Order$_e mbedding$

References

[1] H. Nassar, A. R. Benson, and D. F. Gleich, "Pairwise link prediction," in *2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2019, pp. 386–393.

[2] A. R. Benson, D. F. Gleich, and J. Leskovec, "Higher-order organization of complex networks," *Science*, vol. 353, no. 6295, pp. 163–166, 2016.

[3] P. Li and O. Milenkovic, "Inhomogeneous hypergraph clustering with applications," in *Advances in Neural Information Processing Systems*, 2017, pp. 2308–2318.

[4] H. Zeng, H. Zhou, A. Srivastava, R. Kannan, and V. Prasanna, "Graphsaint: Graph sampling based inductive learning method," in *International Conference on Learning Representations*, 2019.

[5] K. Dong, A. R. Benson, and D. Bindel, "Network density of states," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 1152–1161.

[6] P. Li, N. He, and O. Milenkovic, "Quadratic decomposable submodular function minimization," in *Advances in Neural Information Processing Systems*, 2018, pp. 1054–1064.

[7] Y. Deshpande, S. Sen, A. Montanari, and E. Mossel, "Contextual stochastic block models," in *Advances in Neural Information Processing Systems*, 2018, pp. 8581–8593.

[8] P. Li, I. Chien, and O. Milenkovic, "Optimizing generalized pagerank methods for seed-expansion community detection," in *Advances in Neural Information Processing Systems*, 2019, pp. 11 710–11 721.

[9] P. Li, N. He, and O. Milenkovic, "Quadratic decomposable submodular function minimization: Theory and practice," *Journal of Machine Learning Research*, vol. 21, no. 106, pp. 1–49, 2020.

[10] M. Liu and D. F. Gleich, "Strongly local p-norm-cut algorithms for semi-supervised learning and local graph clustering," *arXiv preprint arXiv:2006.08569*, 2020.

[11] N. Veldt, A. R. Benson, and J. Kleinberg, "Localized flow-based clustering in hypergraphs," *arXiv preprint arXiv:2002.09441*, 2020.

[12] P. Li and O. Milenkovic, "Revisiting decomposable submodular function minimization with incidence relations," in *Advances in Neural Information Processing Systems*, 2018, pp. 2237–2247.

[13] R. Andersen, F. Chung, and K. Lang, "Local graph partitioning using pagerank vectors," in *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*. IEEE, 2006, pp. 475–486.

[14] B. Bahmani, K. Chakrabarti, and D. Xin, "Fast personalized pagerank on mapreduce," in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, 2011, pp. 973–984.

[15] K. Kloster and D. F. Gleich, "Heat kernel based community detection," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 1386–1395.

[16] N. Veldt, A. R. Benson, and J. Kleinberg, "Hypergraph cuts with general splitting functions," *arXiv preprint arXiv:2001.02817*, 2020.

[17] D. Fu, D. Zhou, and J. He, "Local motif clustering on time-evolving graphs," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 390–400.

[18] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in neural information processing systems*, 2017, pp. 1024–1034.

[19] W.-L. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C.-J. Hsieh, "Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 257–266.

[20] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, "Inductive representation learning on temporal graphs," *arXiv preprint arXiv:2002.07962*, 2020.

[21] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[22] Y. Zhang, J. Hare, and A. Prugel-Bennett, "Deep set prediction networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 3212–3222.

[23] R. Liao, Y. Li, Y. Song, S. Wang, W. Hamilton, D. K. Duvenaud, R. Urtasun, and R. Zemel, "Efficient graph generation with graph recurrent attention networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 4255–4265.

[24] J. You, R. Ying, X. Ren, W. L. Hamilton, and J. Leskovec, "Graphrnn: Generating realistic graphs with deep auto-regressive models," *arXiv preprint arXiv:1802.08773*, 2018.

[25] Y. Bai, H. Ding, S. Bian, T. Chen, Y. Sun, and W. Wang, "Simgnn: A neural network approach to fast graph similarity computation," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 2019, pp. 384–392.

[26] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" *arXiv preprint arXiv:1810.00826*, 2018.

[27] P. Li, Y. Wang, H. Wang, and J. Leskovec, "Distance encoding – design provably more powerful gnns for structural representation learning," *arXiv preprint arXiv:2009.00142*, 2020.

[28] H. Maron, H. Ben-Hamu, H. Serviansky, and Y. Lipman, "Provably powerful graph networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 2156–2167.

[29] Z. Lou, J. You, C. Wen, A. Canedo, J. Leskovec *et al.*, "Neural subgraph matching," *arXiv preprint arXiv:2007.03092*, 2020.

[30] D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial attacks on neural networks for graph data," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2847–2856.

[31] D. Zhu, Z. Zhang, P. Cui, and W. Zhu, "Robust graph convolutional networks against adversarial attacks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 1399–1407.

[32] A. Hasanzadeh, E. Hajiramezanali, S. Boluki, M. Zhou, N. Duffield, K. Narayanan, and X. Qian, "Bayesian graph neural networks with adaptive connection sampling," *arXiv preprint arXiv:2006.04064*, 2020.

[33] A. R. Benson, R. Abebe, M. T. Schaub, A. Jadbabaie, and J. Kleinberg, "Simplicial closure and higher-order link prediction," *Proceedings of the National Academy of Sciences*, vol. 115, no. 48, pp. E11 221–E11 230, 2018.

[34] M. Hein, S. Setzer, L. Jost, and S. S. Rangapuram, "The total variation on hypergraphs-learning on hypergraphs revisited," in *Advances in Neural Information Processing Systems*, 2013, pp. 2427–2435.

[35] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 3558–3565.

[36] S. Bai, F. Zhang, and P. H. Torr, "Hypergraph convolution and hypergraph attention," *arXiv preprint arXiv:1901.08150*, 2019.

[37] N. Yadati, M. Nimishakavi, P. Yadav, V. Nitin, A. Louis, and P. Talukdar, "Hypergcn: A new method for training graph convolutional networks on hypergraphs," in *Advances in Neural Information Processing Systems*, 2019, pp. 1511–1522.

[38] C. Yang, R. Wang, S. Yao, and T. Abdelzaher, "Hypergraph learning with line expansion," *arXiv preprint arXiv:2005.04843*, 2020.

[39] J. Yi and J. Park, "Hypergraph convolutional recurrent neural network," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 3366–3376.