CAPER: Coarsen, Align, Project, Refine A General Multilevel Framework for Network Alignment

Jing Zhu
University of Michigan, Ann Arbor
jingzhuu@umich.edu

Danai Koutra
University of Michigan, Ann Arbor
dkoutra@umich.edu

Mark Heimann
Lawrence Livermore National Laboratory
heimann2@llnl.gov

ABSTRACT

Network alignment, or the task of finding corresponding nodes in different networks, is an important problem formulation in many application domains. We propose CAPER, a multilevel alignment framework that Coarsens the input graphs, Aligns the coarsened graphs, Projects the alignment solution to finer levels and Refines the alignment solution. We show that CAPER can improve upon many different existing network alignment algorithms by *enforcing alignment consistency across multiple graph resolutions: nodes matched at finer levels should also be matched at coarser levels.* CAPER also accelerates the use of slower network alignment methods, at the modest cost of linear-time coarsening and refinement steps, by allowing them to be run on smaller coarsened versions of the input graphs. Experiments show that CAPER can improve upon diverse network alignment methods by an average of 33% in accuracy and/or an order of magnitude faster in runtime **Paper Type: Research Paper**

ACM Reference Format:
Jing Zhu, Danai Koutra, and Mark Heimann. 2022. CAPER: Coarsen, Align, Project, Refine A General Multilevel Framework for Network Alignment. In *Proceedings of 17th International Workshop on Mining and Learning with Graphs (KDD MLG '22).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Graphs or networks are foundational representations for relational structure and their analysis is useful in innumerable scientific and industrial applications. In many diverse tasks, such as recommendation across multiple social networks, protein-protein interaction analysis, and database schema matching [8], it is necessary to discover meaningful correspondences between nodes in multiple networks. This general problem is called network alignment.

Network alignment methods in general have two main limitations. First, they may overfit to local structural similarity and fail to preserve higher-order measures of matching consistency [2, 5]. Second, they tend to rely on solving challenging optimization problems with high computational complexity. Some of the most successful graph alignment algorithms require quadratic or cubic time in the number of nodes in one of the input graphs [2, 16, 20].

We argue that multilevel network analysis is a powerful technique for improving network alignment algorithms on both fronts. Accordingly, we design a general framework that turns any network alignment method into a multilevel method. Our framework consists of the following steps, which we call CAPER: (1) Coarsening a graph into multiple levels of varying coarseness, (2) Aligning at the coarsest level, and (3) Projecting back to finer levels, and (4) Refining the solution at each level. First, we repeatedly leverage graph coarsening to shrink the sizes of the input graphs. Then, we run a network alignment algorithm on the resulting coarsened graphs, which allows alignment algorithms that are computationally expensive to be run on smaller input graphs. Finally, we project the coarse alignment solution back to the original input graph, using a new soft refinement step to ensure a high-fidelity resolution at each intermediate level. By performing the base matching based on global structural similarity in the coarsened graph and refining it with successively finer-grained local structural information, the final matchings found by CAPER reflect all levels of structural information.

Our contributions can be summarized as follows:

- **General-Purpose Framework**: We propose an intuitive multilevel framework (CAPER) in which any network alignment method can be used.
- **Design Choices and Empirical Success**: We propose and study specific design choices and parameter settings that work well within CAPER.
- **Study of Accuracy and Runtime Tradeoff**: Through complexity analysis and experiments, we show that CAPER is able to improve accuracy by 33% on average across multiple datasets and/or is 10x faster runtime than baselines, depending on the properties of the base methods employed.

We provide code and additional supplementary material at https://github.com/GemsLab/CAPER.

2 RELATED WORK

Graph Coarsening and Multilevel Methods. Graph coarsening [13] is the process of shrinking a large graph into a similar smaller one, such that some properties or structures are preserved, e.g. spectral graph properties or cliques. The typical approach for coarsening involves grouping neighboring nodes in the original graph into supernodes and connecting them with superedges. It has been used to accelerate many graph mining tasks, including graph clustering [4], node embedding [3, 12] and graph neural networks [17].

Work partially performed while an intern at Lawrence Livermore National Laboratory.

KDD MLG '22, Aug 15, 2022, Washington DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

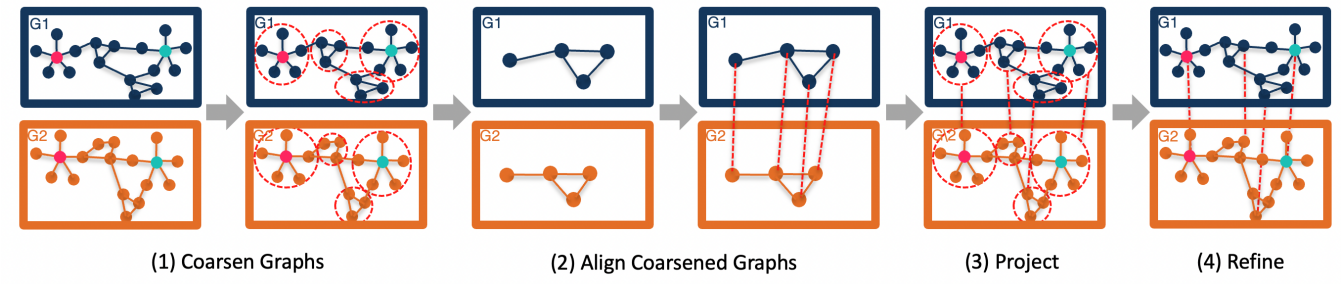


Figure 1: An illustrative example of CAPER. The pink node and blue node in the leftmost figure have exactly the same local structural similarity in the two graphs. So, methods that overfit the local structural similarity cannot differentiate which nodes should be aligned. But with the help of higher-order information (coarsened graphs in step (2)), CAPER is able to eventually correctly align the pink nodes as well as the blue nodes.

Table 1: Qualitative comparison of alignment meta-frameworks.

	General	Multiscale	Improves accuracy	Improves runtime
MOANA [19]	✗	✓	✗	✓
Boosting [10]	✗	✗	✓	✗
RefiNA [5]	✓	✗	✓	✗
CAPER	✓	✓	✓	✓

Network Alignment. Unsupervised graph alignment approaches can be categorized into two groups. (1) Classic graph alignment approaches often formulate an **optimization-based assignment** problem. NetAlign [1] aims to preserve "complete squares" by matching two nodes sharing an edge in one graph to counterparts sharing an edge in the other graph. FINAL [18] optimizes a topological consistency objective which may be augmented with node and edge attribute information. MAGNA [15] simulates the evolution of network populations with a genetic algorithm, maximizing proximity consistency criteria such as edge correctness and is often used in the biological domain. More recently, Zhang *et al.* [20] leveraged kernel-based methods to solve the quadratic-assignment problem, but they suffered from cubic computational complexity. (2) Another line of work relies on **embedding-based methods.** These methods represent each graph as a distribution of node embeddings and match these distributions. For instance, the first work in this space, REGAL [6] matches structural node embeddings [7, 14] that are directly comparable across networks. CONE-Align [2] first uses proximity-based embeddings and aligns the embedding spaces with a subspace alignment procedure, while GWL [16] solves a Gromov-Wasserstein optimization problem to jointly find node embeddings and a graph matching solution. The above approaches are unsupervised, requiring no "anchor" pairs of corresponding nodes to be known a priori. In this work, we focus on unsupervised network alignment given the difficulty of constructing ground-truth anchor pairs in practice.

Network Alignment Meta-Frameworks. Because of the challenge of unsupervised network alignment, a few recent works have proposed meta-frameworks to refine network alignment algorithms' solutions. This includes MOANA, the only other multilevel network alignment approach [19]. MOANA uses multiresolution matrix factorization to accelerate the method FINAL [18] at the cost of some accuracy. However, it produces negative-valued adjacency matrices that do not work well with other network alignment methods. RefiNA [5] refines any network alignment solutions by the matched neighborhood consistency principle (nodes that are close in one graph should be matched to nodes that are close in another graph). Such an approach makes the opposite tradeoff, increasing accuracy of several base methods at the cost of adding additional runtime. Another meta-framework [10] specifically studying how design choices of recent embedding-based network alignment methods can be combined to increase accuracy via boosting. Meanwhile, our approach is orthogonal to boosting approaches, inherits all these benefits, as we show in a qualitative comparison in Tab. 1.

3 PRELIMINARIES

Graphs. We consider two graphs G_1 and G_2 with nodesets $\mathcal{V}_1, \mathcal{V}_2$ and adjacency matrices $\mathbf{A}_1, \mathbf{A}_2$ containing edges between nodes. A graph G_i has a coarsened counterpart $\tilde{G}_i$ with a smaller nodeset $\tilde{\mathcal{V}}_i$ of $\tilde{n} < n$ nodes. Each node in the original graph corresponds to a node in the coarsened graph, with assignments being stored in an assignment matrix $\mathbf{P} \in \{0,1\}^{n \times \tilde{n}}$. For clarity, we drop the $\tilde{G}$ notation unless it is necessary to distinguish coarsened and uncoarsened counterparts.

Alignment. An alignment between the nodes of two graphs can be represented by a matrix $\mathbf{S}$, where s_{ij} is the (real-valued or binary) similarity between node i in G_1 and node j in G_2.

Problem Statement. Given two graphs G_1 and G_2 with meaningful node correspondences, but none known *a priori*, we seek to shrink them into coarsened versions $\tilde{G}_1$ and $\tilde{G}_2$, and recover their alignment $\mathbf{S}$ from the alignment $\tilde{\mathbf{S}}$ obtained by aligning their coarser versions $\tilde{G}_1$ and $\tilde{G}_2$.

4 METHOD

Next, we detail our CAPER framework, the first general-purpose multilevel framework for unsupervised network alignment that can accommodate any base network alignment approach. It consists of four steps that are carefully designed in order to achieve higher accuracy and/or lower runtime compared to its base alignment methods: Coarsen, Align, Project, Refine (CAPER). In Fig. 1, we provide an example of how CAPER can implicitly enforce higher-order structural consistency that improves network alignment.

4.1 Graph Coarsening

Given an input graph G_i, we want to obtain a coarsened graph $\tilde{G}_i$ using grouping-based coarsening methods. We leverage the normalized heavy-edge matching (NHEM) heuristic [4] for graph coarsening. This approach repeatedly combines pairs of adjacent nodes into a supernode, until no node is left uncombined or the uncombined nodes do not have uncombined neighbors (isolated nodes). Each supernode becomes a new node in the resulting coarser graph, and two supernodes share an edge if any of the constituent nodes in one supernode shared an edge in the original graph with any of the constituent nodes in the other supernode.

Nodes are combined in decreasing order of degree-normalized edge weight [12], which for edge (u,v) with weight w_{uv} connecting nodes u and v with degrees d_u and d_v respectively is given by $\frac{w_{uv}}{\sqrt{d_u d_v)}}$. As in many network alignment settings, our graphs are unweighted (so $w_{uv} = 1$ for all edges). This has the effect of combining low-degree nodes first, a useful heuristic since we first coarsen the parts of the graph that are "safer" to coarsen and are also *harder* to align since there is less topological structure to lose.

Graph coarsening turns each input graph G_i into a coarsened graph $\tilde{G}_i$. We may iteratively repeat this coarsening procedure up to $\mathbf{L}$ times to produce a sequence of coarsened graphs $\tilde{G}_i^{(0)}, \ldots, \tilde{G}_i^{(L)}$, where the first level is the input graph ($\tilde{G}_i^{(0)} = G_i$), and the coarsest (smallest) graph is $\tilde{G}_i^{(L)}$. The assignments of nodes between nodes at consecutive levels $\ell - 1$ and ℓ are contained in a matrix $\mathbf{P}_i^{(\ell)}$ for $\ell \in [1, \ldots, L]$ as defined in Sec. 3.

4.2 Alignment of Coarsened Graphs

In this step, the input is two coarsened graphs $\tilde{G}_1$ and $\tilde{G}_2$, and the output is an alignment solution $\mathbf{S}$. We can apply any existing unsupervised network alignment method to align the nodes of the graphs at the coarsest level to produce a matching $\mathbf{S}^{(\mathbf{L})}$. We observe that the coarsening procedure sometimes generates slightly different numbers of nodes for the same graph even if the input graphs have the same size, so the proposed formulation must be able to handle graphs of different sizes. A workaround suggested for graph matching formulations designed for equally sized graphs is to add singleton nodes to the smaller graph [2].

4.3 Projection

We project the alignment solution at the coarsest level $\mathbf{S}^{(\ell)}$ to a mapping between the nodes at the next finer level using the assignment matrices: $\mathbf{S}^{(\ell-1)} = \mathbf{P}_1^{(\ell)^\top} \mathbf{S}^{(\ell)} \mathbf{P}_2^{(\ell)}$. Note that this solution is coarse, and all nodes in level $\ell - 1$ mapped to the same supernode in level ℓ will have the same match. Thus, we need to use the finer graph structure to refine this coarse solution. To do so, we use soft refinement in the next step.

4.4 Soft Refinement

In this step, we aim to refine the coarse alignment solution we get in the projection step to obtain a more fine-grained alignment solution. Recent work for refining network alignment [5] operates on "hard" initial solutions, where each node is mapped to at most one other node. Here, we propose a new refinement operator that uses the "soft" initial alignments, which better models the various strengths of several potential matches for each node, as shown in Fig. 5. Given an initial soft alignment $\mathbf{S}$, we updates the alignment matrix $\mathbf{S}$ by iteratively applying the update rule $\mathbf{S} = \text{NORMALIZE}(\mathbf{S} \circ \mathbf{A}_1 \mathbf{S} \mathbf{A}_2 + \epsilon)$, where $\circ$ denotes Hadamard product, ϵ is a small positive minimum matching score to any pair of nodes to prevent over-reliance on the initially discovered matches (we set $\epsilon = 10^{-\lceil \log_{10} \max(n_1, n_2) \rceil}$) and NORMALIZE is a single round of row-wise normalization followed by column-wise normalization, as in [5]. This "soft refinement" allows us to correct misaligned nodes during refinement as well as to preserve higher-order structural consistency.

We iteratively apply this project-and-refine procedure between successive levels until we arrive back at the input level, giving us the mapping between nodes in the original graph.

4.5 Computational Complexity

For clarity, we assume both graphs have n nodes. Below we present the time complexity of each method as a function f dependent on n. Then the computational complexity of our CAPER framework is $L f_{\text{coarsen}}(n) + f_{\text{align}}(\frac{n}{2^L}) + L\left(f_{\text{project}}(n) + f_{\text{refine}}(n)\right)$. The first, third and fourth terms represent the overhead introduced by coarsening, projection and refinement, respectively. The coarsening time applied to each of L levels, $f_{\text{coarsen}}(n)$, is linear in the number of edges using heavy-edge matching [4], which is $O(n)$ for the sparse graphs that are typically considered. The refinement consists of projection f_{project} by multiplying matching and assignment matrices, and refinement f_{refine} using RefiNA: these may take quadratic time for dense matrices, but by maintaining a sparse matching matrix consisting of only a small number of top matches for each node as proposed in [5], these operations can also run in $O(n)$ time for sparse graphs.

This leads us to the runtime of the base alignment method, f_{align}. With the heavy edge matching heuristic shrinking the graph by approximately a factor of 2 at each level [4], note that we are able to run the base alignment step on a smaller graph, incurring a runtime of $f_{\text{align}}(\frac{n}{2^L})$ as opposed to $f_{\text{align}}(n)$ by applying the base alignment algorithm to the full input graphs.

Thus, CAPER can offer computational speedup particularly for slow base alignment methods, where f_{align} may be asymptotically large (such as $O(n^3)$ as some recent, effective methods have [16]). For fast base alignment methods on smaller graphs, the overhead of coarsening and refinement may cause a modest linear-time total increase in runtime. However, the faster approaches tend to stand the most to benefit in accuracy from CAPER ensuring alignment consistency at multiple levels.

5 EXPERIMENTS

In our experiments, we investigate how much CAPER can improve some popular base methods in network alignment both in terms of accuracy and runtime. We first describe our experimental setup and the datasets and baseline methods used in our empirical analysis, and then show quantitative results and discuss our observations.

Data. We use both semi-synthetic and real-world data (Tab. 2). Following prior works [6, 18], we simulate a network alignment scenario with known ground truth: a graph with adjacency matrix $\mathbf{A}$

Table 2: Datasets statistics: These four datasets represent various phenomena such as communication network, social network and protein-protein interactions.

Name	Nodes	Edges	Description
Arenas [9]	1,133	5,451	communication network
Hamsterster [9]	2,426	16,613	social network
Facebook [11]	4,039	88,234	social network
Magna [15]	1,004	8,323	protein-protein interaction

is aligned to a noisy permuted copy $\mathbf{A}^* = \overline{\mathbf{S}}\mathbf{A}\overline{\mathbf{S}}^\top$ and $\overline{\mathbf{S}}$, for which we generate a random permutation matrix $\overline{\mathbf{S}}$; we then randomly remove edges from $\mathbf{A}^*$ with probability $p \in [0.05, 0.10, 0.15, 0.20, 0.25]$. We perform this procedure on graphs representing various phenomena as shown in Table 2. The MAGNA [15] networks are protein-protein interaction (PPI) networks that are aligned to versions of themselves with various percentages of low-confidence PPIs (edges) added; thus, all edges in this graph represent real-world phenomena and we do not need to synthesize an alignment scenario.

Baselines. We consider both optimization- and embedding-based alignment methods. We use **(1)** FINAL [18] and **(2)** REGAL [6] as representative and popular network alignment methods for unsupervised network alignment, respectively. They have publicly available and easy-to-use codebases, and represent different classes of techniques, (optimization and node embeddings) demonstrating the wide applicability for our framework. We also compare with the more recent approach, **(3)** GWL [16], which combines optimization and node embeddings, and achieves good task performance but has slow runtime due to its $O(n^3)$ computational complexity. Moreover, we consider the refinement approach RefiNA applied to each of the network alignment methods. We refer to these refined variants as **(4)** FINAL-RefiNA, **(5)** REGAL-RefiNA and **(6)** GWL-RefiNA. Additionally, we also use **(7)** MOANA [19] as a baseline, the only other multilevel network alignment method.

For FINAL's prior alignment information, we take the top $k = \lfloor \log_2 n \rfloor$ most similar nodes by degree for each node [2, 6]. We set other parameters for REGAL [6] and GWL [16] using the defaults recommended by the authors.

CAPER variants. For our framework, we consider three base alignment approaches: FINAL, REGAL and GWL. We refer to these variants as CAPER(FINAL), CAPER(REGAL), CAPER(GWL), respectively. We use the exact same setup for the base methods as described above. We use 3 coarsening levels and 100 refinement iterations, as in [5], to balance accuracy and computational efficiency (we found that further iterations may increase performance if that is desired and increased runtime is acceptable).

Evaluation metrics. We measure **alignment accuracy**, or the proportion of correctly aligned nodes. We also study the runtime. We average over 5 independent trials on each dataset (with different random permutations and noise additions) and report the average accuracy and standard deviation.

Table 3: Number of nodes in the coarsened graph after 2,3,4 levels of coarsening for Hamsterster and Facebook dataset

Name	2	3	4
Hamsterster	1,288	702	418
Facebook	2,078	1,078	572

5.1 Alignment Accuracy

Setup. In Fig. 2, we report the average and standard deviation for each metric over five trials for each experimental setting, except for Magna. Magna is a real-world dataset, so we do not generate any synthetic data for different noise levels. The + sign in Fig. 2 means that the standard deviation here is larger than 0.05.

Results. While the existing multilevel alignment method MOANA has accuracy below its single-level counterpart FINAL as expected, our multilevel framework, CAPER, significantly outperforms different base alignment methods as well as their single-level refined variants using RefiNA. Moreover, we can see that **CAPER is more robust to noise**; specifically, the performance for REGAL-CAPER is very stable even when the noise level increases. This is mainly due to the fact that CAPER forces the nodes to maintain high-order consistency even when the noise level increases.

5.2 Alignment Runtime

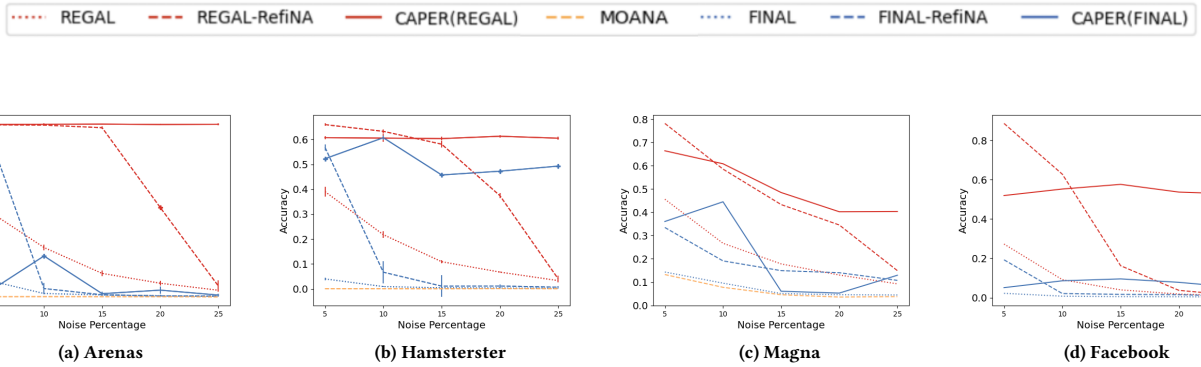
Runtime Setup & Evaluation. Due to GWL's slow runtime, we only run it for one trial on the largest Facebook dataset. Others are averaged over five trials in Figure 3.

Results. For faster base methods such as FINAL and especially REGAL, our improvements are mainly in accuracy. This is because the overhead of coarsening the graphs and refining the coarsened alignment solution does not outweigh the computational savings of performing the alignment on smaller graphs when the base alignment method is fast. However, when the base alignment method is slow, as is the case for GWL, our framework results in considerable computational savings, while we see that it can largely preserve accuracy. Compared with MOANA, CAPER is able to obtain better accuracy.

5.3 Ablation studies

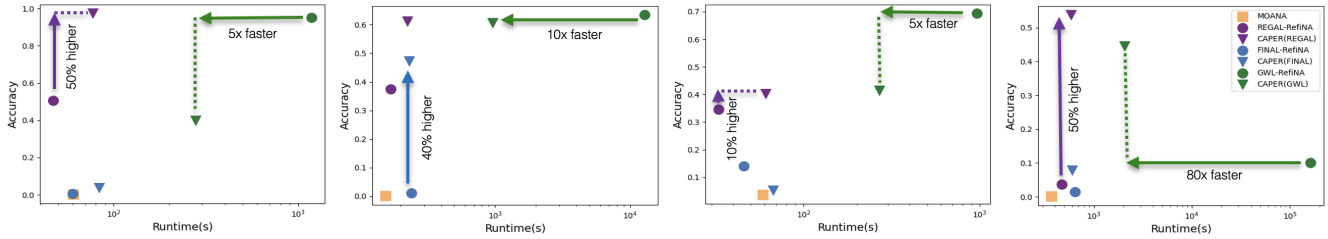
Number of levels. Figure 4 compares the performance of CAPER using REGAL as the base alignment method and different numbers of coarsening levels on the Hamsterster and Facebook datasets. We observe that having 2 levels works best while having a larger number of coarsening levels (e.g. 4) actually hurts the performance of CAPER. However, the number of coarsening levels to have is also relevant to the tradeoff between accuracy and runtime: more coarsening levels leads to fewer nodes in the coarsened graph and faster runtime at a cost of an eventual drop in accuracy, as shown in Tab. 3. Similar trends are observed for other datasets as well. As mentioned, for our main experiments, we used 3 levels of coarsening for all datasets to balance this tradeoff.

Hard vs. soft refinement. Recent work for refining network alignment [5] requires a hard prior alignment where each node is mapped to only one counterpart, while in CAPER, we propose to use soft



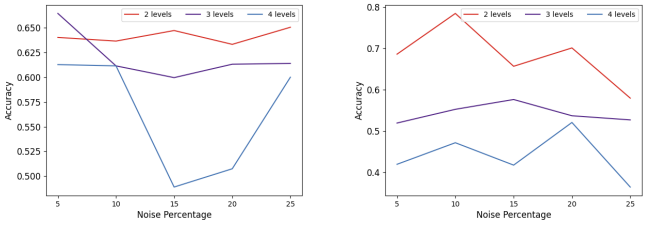
(a) Arenas (b) Hamsterster (c) Magna (d) Facebook

Figure 2: Accuracy (solid lines) and std vs. different noise levels. CAPER significantly outperforms baselines, particularly as noise increases.



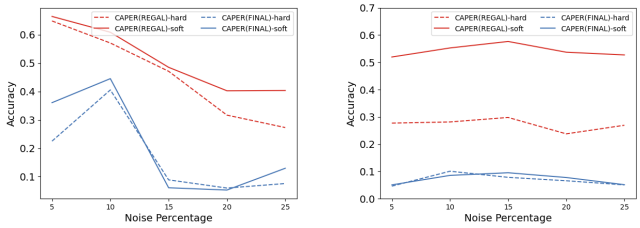
(a) Arenas (b) Hamsterster (c) Magna (d) Facebook

Figure 3: Accuracy vs. runtime plot for CAPER and the baseline approach RefiNA for 20% noise. CAPER is able to yield better accuracy for FINAL and REGAL by enforcing higher-order consistency. For GWL, CAPER is able to run up to 80x faster by shrinking the size of the graphs that need to be aligned.



(a) Hamsterster (b) Facebook

Figure 4: Ablation study for number of coarsening levels for CAPER(REGAL) on the Hamsterster and Facebook dataset. In general, two levels lead to highest accuracy. In our main experiments, we choose 3 levels for the best accuracy runtime tradeoff/consistency.



(a) Magna (b) Facebook

Figure 5: Ablation study for the accuracy for CAPER(REGAL) and CAPER(FINAL) on Magna and Facebook dataset using soft/hard refinements. We observe that soft refinement works significantly better than hard refinement, especially when the base alignment method is REGAL.

refinement to use more detailed alignment information. Here, we study how hard versus soft refinement affect our results.

We found that in CAPER, using soft refinement works better than hard refinement, as shown in Fig. 5. This effect is most noticeable for the base method REGAL, which is more accurate on these datasets. For FINAL, because its initial solution is less accurate, we used hard refinement at the coarsest level (when operating directly on the solution returned by FINAL) and soft refinement at subsequent levels. This also explains the smaller gap in performance.

6 CONCLUSION

We describe the first general-purpose multilevel framework for unsupervised network alignment that can work with a variety of base network alignment algorithms, making them more accurate and robust by incorporating multiscale graph information, and accelerating the runtime of slower network alignment methods by allowing them to operate on smaller input graphs. We have also found that not all coarsening methods work well. Some up-to-date spectral coarsening methods [3] will give clusters with zero nodes in it and thus our multi-level alignment framework could fail. One possible future direction would be to characterize the effect of various coarsening methods on multilevel network alignment.

ACKNOWLEDGEMENTS

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 and was supported by the LLNL-LDRD Program under Project No. 21-ERD-012. It was also partially supported by the NSF under Grant No. IIS 1845491, and Adobe, Amazon, and Google faculty awards.

REFERENCES

[1] Mohsen Bayati, Margot Gerritsen, David F Gleich, Amin Saberi, and Ying Wang. Algorithms for large, sparse network alignment problems. In *ICDM*, 2009.

[2] Xiyuan Chen, Mark Heimann, Fatemeh Vahedian, and Danai Koutra. Cone-align: Consistent network alignment with proximity-preserving node embedding. In *CIKM*, 2020.

[3] Chenhui Deng, Zhiqiang Zhao, Yongyu Wang, Zhiru Zhang, and Zhuo Feng. Graphzoom: A multi-level spectral approach for accurate and scalable graph embedding. In *ICLR*, 2020.

[4] Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE transactions on pattern analysis and machine intelligence*, 29(11):1944–1957, 2007.

[5] Mark Heimann, Xiyuan Chen, Fatemeh Vahedian, and Danai Koutra. Refining network alignment to improve matched neighborhood consistency. In *SDM*, 2021.

[6] Mark Heimann, Haoming Shen, Tara Safavi, and Danai Koutra. Regal: Representation learning-based graph alignment. In *CIKM*, 2018.

[7] Junchen Jin, Mark Heimann, Di Jin, and Danai Koutra. Toward understanding and evaluating structural node embeddings. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 16(3):1–32, 2021.

[8] Ehsan Kazemi. Network alignment: Theory, algorithms, and applications. Technical report, EPFL, 2016.

[9] Jérôme Kunegis. Konect: the koblenz network collection. In *WWW*, 2013.

[10] Alexander Frederiksen Kyster, Simon Daugaard Nielsen, Judith Hermanns, Davide Mottin, and Panagiotis Karras. Boosting graph alignment algorithms. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3166–3170, 2021.

[11] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[12] Jiongqian Liang, Saket Gurukar, and Srinivasan Parthasarathy. Mile: A multi-level framework for scalable graph embedding. In *ICWSM*, 2021.

[13] Yike Liu, Tara Safavi, Abhilash Dighe, and Danai Koutra. Graph summarization methods and applications: A survey. *ACM computing surveys (CSUR)*, 51(3):1–34, 2018.

[14] Ryan A Rossi, Di Jin, Sungchul Kim, Nesreen K Ahmed, Danai Koutra, and John Boaz Lee. On proximity and structural role-based embeddings in networks: Misconceptions, techniques, and applications. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 14(5):1–37, 2020.

[15] Vikram Saraph and Tijana Milenković. Magna: maximizing accuracy in global network alignment. *Bioinformatics*, 30(20):2931–2940, 2014.

[16] Hongteng Xu, Dixin Luo, Hongyuan Zha, and Lawrence Carin Duke. Gromov-wasserstein learning for graph matching and node embedding. In *ICML19*, pages 6932–6941, 2019.

[17] Yujun Yan, Jiong Zhu, Marlena Duda, Eric Solarz, Chandra Sripada, and Danai Koutra. Groupinn: Grouping-based interpretable neural network for classification of limited, noisy brain data. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 772–782, 2019.

[18] Si Zhang and Hanghang Tong. Final: Fast attributed network alignment. In *KDD*, 2016.

[19] Si Zhang, Hanghang Tong, Ross Maciejewski, and Tina Eliassi-Rad. Multilevel network alignment. In *The World Wide Web Conference*, pages 2344–2354, 2019.

[20] Zhen Zhang, Yijian Xiang, Lingfei Wu, Bing Xue, and Arye Nehorai. KerGM: Kernelized Graph Matching. In *NeurIPS19*, pages 3330–3341, 2019.