Graph Edit Distance Evaluation Datasets: Pitfalls and Mitigation

Eeshaan Jain[*†] **Indradyumna Roy**[*‡]
Saswat Meher[‡] **Soumen Chakrabarti**[‡] **Abir De**[‡]
[†]EPFL [‡]IIT Bombay
eeshaan.jain@epfl.ch
{saswatmeher,soumen,indraroy15,abir}@cse.iitb.ac.in

Abstract

Graph Edit Distance (GED) is a powerful framework for modeling both symmetric and asymmetric relationships between graph pairs under various cost settings. Due to the combinatorial intractability of exact GED computation, recent advancements have focused on neural GED estimators that approximate GED by leveraging data distribution characteristics. These estimators map the structural information of graphs into an embedding space while preserving essential graph invariances and equivariances. However, the datasets commonly used to benchmark such neural models exhibit two critical flaws: (1) significant isomorphism bias leading to high likelihood of train-test leakage, with only a small fraction of graphs being structurally unique (8.9% in Linux, 25.8% in IMDB, and 41% in AIDS data sets), and (2) reliance on uniform edit costs for GED ground truths. These limitations constrain the evaluation of learning and generalization capabilities of competing methods, casting doubt on the validity of existing results and suggesting potential biases in comparative studies.

In this work, we introduce and release a comprehensive suite of datasets specifically designed to rectify these shortcomings. Our datasets eliminate isomorphism leakage and incorporate a range of edit costs, facilitating more accurate assessment of GED methods. We conduct benchmarking evaluations of state-of-the-art methods using these datasets, providing insights into their true generalization capabilities. By making these datasets available as open-source resources, we offer a robust foundation for advancing research in GED estimation.

1 Introduction

Graph Edit Distance (GED) measures the minimum cost of transforming one graph into another through a series of edit operations like node and edge insertions, deletions, and substitutions. Each operation can have a different cost, allowing GED to represent both symmetric and asymmetric relationships between graph pairs. This flexibility makes GED suitable for various graph comparison tasks, such as identifying the Maximum Common Subgraph and verifying Subgraph Isomorphism [1]. However, computing GED is computationally challenging, especially with variable edit costs. Recent neural network-based approaches, typically leveraging Graph Neural Networks (GNNs), aim to approximate GED efficiently. Still, many of these methods struggle with incorporating variable operation costs, limiting their ability to generalize across diverse scenarios.

1.1 GED Evaluation Datasets

Exact GED computation is notoriously intractable, and while neural approaches offer promising approximations, training these models requires ground truth data. Generating this data is computationally expensive, especially for larger graphs. Early neural GED papers relied on datasets with small graphs to address this limitation. For instance, SimGNN [2] and other early works focused on

Jain et al., Graph Edit Distance Evaluation Datasets: Pitfalls and Mitigation (Extended Abstract). Presented at the Third Learning on Graphs Conference (LoG 2024), Virtual Event, November 26–29, 2024.

datasets like AIDS, LINUX, and IMDB, which mostly feature graphs with fewer than 10 nodes. This trend continued with subsequent models like GED-GNN [3], EGSC [4], and ERIC [5], which also utilized small graph datasets such as ALKANE and NCI109, in addition to the original three datasets.

This computationally intensive prerequisite poses a significant barrier to entry for new research. This difficulty is further exacerbated by the fact that many studies do not release their generated datasets. Consequently, most recent work relies heavily on the original three datasets introduced by SimGNN—AIDS, LINUX, and IMDB. We investigate these three main datasets used for GED estimation by the current state-of-the-art neural GED methods [2–9], also provided in the torch-geometric's GEDDataset class [1]. These datasets consists of a collection of graphs $\mathcal{D}$, which are further divided into train $\mathcal{D}_{\text{train}}$, validation $\mathcal{D}_{\text{val}}$, and test $\mathcal{D}_{\text{test}}$ splits, and appropriately paired either by (1) taking $|\mathcal{D}_{\bullet}|^2$ combinations within the split, or (2) as a retrieval setup, pairing test graphs with train graphs. Upon further inspection, we identified a high prevalence of isomorphic graphs, reducing the unique dataset $\mathcal{D}_{\text{unq}}$ to a small fraction of the original size, as shown in Table 1.

Table 1: Size of datasets before and after filtering isomorphic graphs and the reduction in (%) of the datasets.

Dataset $\mathcal{D}$	$\lvert\mathcal{D}\rvert$	$\lvert\mathcal{D}_{\text{unq}}\rvert$	$\downarrow$ %
LINUX	1000	89	91.1%
IMDB	1500	387	74.2%
AIDS (w/o label)	700	290	58.6%
AIDS (w/ label)	700	670	4.3%

Consequently, irrespective of the methodology employed for forming pairs within the GED datasets, substantial train-test leakage remains unavoidable, significantly undermining the validity of the results reported by current neural approaches for GED estimation. Additionally, these datasets only provide graph edit distance values based on uniform costs (where all edit operations incur a cost of 1), restricting the broader applicability of these methods. To this effect, there is a need in literature for new benchmark datasets which:

1. Eliminate train-test leakage, allowing for robust evaluation of learning and generalization capabilities of competing methods,
2. Incorporate varied edit costs across datasets for comprehensively evaluating the capabilities of neural models to exploit the underlying flexibility of the GED framework,
3. Are open-source, with publicly available ground truth data to lower the barrier of entry and advancing research in GED estimation.

2 Proposed Datasets

In response to the aforementioned challenges, we introduce a comprehensive benchmarking suite for GED estimation, encompassing seven datasets and four distinct edit cost combinations. The datasets, namely Mutagenicity (Mutag), Ogbg-Code2 (Code2), Ogbg-Molhiv (Molhiv), Ogbg-Molpcba (Molpcba), AIDS (AIDS), Linux (Linux), and Yeast (Yeast), have been extracted from standard graph benchmarks [2, 10–12], and are further tailored to the task of computing the GED. We define the costs associated with node addition, deletion, and substitution as $\mathcal{C}_n = (c_n^{\text{add}}, c_n^{\text{del}}, c_n^{\text{sub}})$, and edge costs as $\mathcal{C}_e = (c_e^{\text{add}}, c_e^{\text{del}}, c_e^{\text{sub}})$. The benchmark variants are summarized as follows:

1. **Uniform-GED (U-GED):** Consisting of graphs from the following datasets – *Mutag, Code2, Molhiv, Molpcba, AIDS, Linux, Yeast* with $\mathcal{C}_n = (1, 1, 0)$, and $\mathcal{C}_e = (1, 1, 0)$.
2. **Non-uniform-GED (NU-GED):** Consisting of graphs from the following datasets – *Mutag, Code2, Molhiv, Molpcba, AIDS, Linux, Yeast* with $\mathcal{C}_n = (1, 3, 0)$, and $\mathcal{C}_e = (1, 2, 0)$.
3. **Label-GED (L-GED):** Consisting of graphs from the following datasets – *Mutag, Code2, Molhiv, Molpcba, AIDS* with $\mathcal{C}_n = (1, 1, 1)$, and $\mathcal{C}_e = (1, 1, 0)$.
4. **Non-uniform-edge-GED (NUE-GED):** Consisting of graphs from the following datasets – *Mutag, Molhiv, Linux* with $\mathcal{C}_n = (0, 0, 0)$, and $\mathcal{C}_e = (1, 2, 0)$.

For **Uniform-GED** and **Non-uniform-GED**, the graph edit distance is computed between two graphs without any node substitution cost. In **Label-GED**, a label substitution cost of 1 is introduced. Additionally, we consider a special case, **Non-uniform-edge-GED**, where all node-related costs are set to 0, making the edit distance exclusively focused on the cost of edge alignment.

[1] https://pytorch-geometric.readthedocs.io/en/2.5.3/generated/torch_geometric.datasets.GEDDataset.html

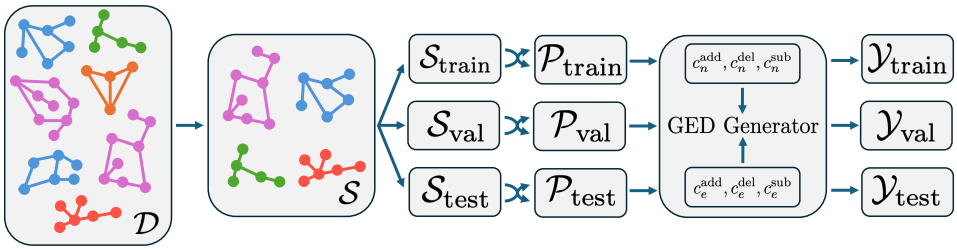

Figure 1: Steps involved to generate the proposed datasets. First, from the collection of graphs $\mathcal{D}$ belonging to a standard graph benchmark, we filter out isomorphic graphs to get the set of graphs $\mathcal{S}$. $\mathcal{S}$ is further split into training, validation, and test splits in the ratio of $60 : 20 : 20$. Then we consider all combinations (including self-combinations) of graphs within each split to generate $|\mathcal{S}_\bullet| \times (|\mathcal{S}_\bullet| + 1)/2$ pairs of graphs. Finally, we calculate the ground truth GED $\mathcal{Y}_\bullet$ through a GED generator using the MIP-F2 [13] with appropriate edit costs.

We generate the datasets as follows: Given the graph benchmark dataset $\mathcal{D}$, we first filter out isomorphic graphs through an extensive, explicit isomorphism check between all pairs, resulting in a set of unique graphs $\mathcal{S}$. This set is then split into $\mathcal{S}_{\text{train}}, \mathcal{S}_{\text{val}}$, and $\mathcal{S}_{\text{test}}$ in a 60:20:20 ratio. For each split, we generate pairs $\mathcal{P}_\bullet$ by considering all combinations (including self-combinations), yielding $|\mathcal{S}_\bullet| \times (|\mathcal{S}_\bullet| + 1)/2$ pairs. The GED ground truth $\mathcal{Y}_\bullet$ for each pair is computed using the MIP-F2 solver [13], with our varying edit cost configurations. Since all graphs within and across splits are unique, we effectively prevent isomorphic leakage. This pipeline is demonstrated in Figure 1. We further present the statistics for **Uniform-GED** and **Non-Uniform-GED** in Table 2. Statistics on **Label-GED** and **Non-uniform-edge-GED** are present in the Appendix.

Table 2: Salient characteristics of datasets for **Uniform-GED** and **Non-uniform-GED**.

	#Graphs	# Train Pairs	# Val Pairs	# Test Pairs	Avg. $\lvert V\rvert$	Avg. $\lvert E\rvert$	Avg. U-GED	Avg. NU-GED
Mutag	729	95703	10585	10878	16.01	15.76	11.15	18.57
Code2	128	2926	325	378	18.77	17.77	10.02	16.43
Molhiv	1000	180300	20100	20100	15.01	15.65	11.77	19.86
Molpcba	1000	180300	20100	20100	17.52	18.67	9.58	15.73
AIDS	911	149331	16653	16836	10.97	10.97	7.38	12.07
Yeast	1000	180300	20100	20100	16.59	17.04	10.65	17.74
Linux	89	1431	153	190	8.71	8.35	4.91	7.94

3 Experiments

We present the evaluation results of nine state-of-the-art methods across all four cost variants of our datasets. Additional experiments and standard error measurements for the methods are provided in the appendix. Upon acceptance, we will publicly release the datasets, along with the code for dataset generation, ground truth computation under all cost variants, and benchmarking of the methods

3.1 Methods

We compare nine state-of-the-art methods, namely GMN-Match [14], GMN-Embed [14], ISONET [15], GREED [7], ERIC [5], SimGNN [2], H2MN [6], GraphSim [8], and EGSC [4]. Among these, GMN-Match and H2MN are early-interaction networks that leverage cross-graph signals during node embedding computation, while the others are late-interaction networks, where node embeddings are computed independently for each graph in the pair. The methods use two approaches to estimate the GED: (1) calculating the Euclidean distance between node or graph embeddings to directly estimate GED, or (2) computing a normalized GED-based similarity score, given by $s = \exp\left(-2y_{G_1,G_2}/(|V_1| + |V_2|)\right)$, where y_{G_1,G_2} is the GED between graphs G_1 and G_2. Additionally, none of the methods incorporate edit costs into their formulations, and hence they are presented as features to the network explicitly.

3.2 Evaluation.

We train the models using the Mean Squared Error (MSE) between the predicted GED and the ground truth GED as the loss function, and present the MSE on the test set. In each case, we mark the best performing method by yellow.

3.3 Results.

In Table 3, we compare the performance of all methods on the **Uniform-GED** task in terms of MSE. Notably, EGSC and ERIC consistently outperform the other methods across the majority of datasets.

Table 3: MSE of methods for **Uniform-GED** with $\mathcal{C}_n = (1, 1, 0)$, and $\mathcal{C}_e = (1, 1, 0)$.

	Mutag	Code2	Molhiv	Molpcba	AIDS	Linux	Yeast
GMN-Match	0.797	1.677	1.318	1.073	0.821	0.687	1.175
GMN-Embed	1.032	1.358	1.859	1.951	1.044	0.736	1.767
ISONET	1.187	0.879	1.354	1.106	1.640	1.185	1.578
GREED	1.398	1.869	1.708	1.550	1.004	1.331	1.423
ERIC	0.719	1.363	1.165	0.862	0.731	1.664	0.969
SimGNN	1.471	2.667	1.609	1.456	1.455	7.232	1.999
H2MN	1.278	7.240	1.521	1.402	1.114	2.238	1.353
GraphSim	2.005	3.139	2.577	1.656	1.936	2.900	2.232
EGSC	0.765	4.165	1.138	0.938	0.627	2.411	0.950

In Table 4, we compare the performance of all methods on the **Non-uniform-GED** task in terms of MSE. We note that EGSC, ERIC, and ISONET consistently outperform the other methods across the majority of datasets. Moreover the methods which predict the GED through Euclidean distance between graph vectors, namely GMN-Match, GMN-Embed, and GREED do not scale with the edit costs, and performed significantly poorly compared to the other methods.

Table 4: MSE of methods for **Non-uniform-GED** with $\mathcal{C}_n = (1, 3, 0)$ and $\mathcal{C}_e = (1, 2, 0)$.

	Mutag	Code2	Molhiv	Molpcba	AIDS	Linux	Yeast
GMN-Match	69.210	13.472	76.923	23.985	31.522	21.519	63.179
GMN-Embed	72.495	13.425	78.254	28.437	33.221	20.591	60.949
ISONET	3.369	3.025	3.451	2.781	5.513	3.031	4.555
GREED	68.732	11.095	78.300	26.057	34.354	20.667	60.652
ERIC	1.981	12.767	3.377	2.057	1.581	7.809	2.341
SimGNN	4.747	5.212	4.145	3.465	4.316	5.369	4.496
H2MN	3.413	9.435	3.782	3.396	3.105	5.848	3.678
GraphSim	5.370	7.405	6.643	3.928	5.266	6.815	6.907
EGSC	1.758	3.957	2.371	2.133	1.693	5.503	2.157

In Table 5, we compare the performance of all methods on the **Label-GED** task in terms of MSE. No single method consistently outperforms the others: EGSC leads in two cases, while ERIC, ISONET, and GMN-Match excel in the remaining datasets. However, in the case of **Non-uniform-edge-GED**, as reported in Table 6, ISONET is the best performer.

Table 5: MSE of methods for **Label-GED** with $\mathcal{C}_n = (1, 1, 1)$ and $\mathcal{C}_e = (1, 1, 0)$.

	Mutag	Code2	Molhiv	Molpcba	AIDS
GMN-Match	1.057	5.224	1.388	1.432	0.868
GMN-Embed	2.159	4.070	3.523	4.657	1.818
ISONET	0.876	1.129	1.617	1.332	1.142
GREED	2.876	4.983	2.923	3.902	2.175
ERIC	0.886	6.323	1.537	1.278	1.602
SimGNN	1.160	5.909	1.888	2.172	1.418
H2MN	1.277	6.783	1.891	1.666	1.290
GraphSim	1.043	4.708	1.817	1.748	1.561
EGSC	0.776	8.742	1.273	1.426	1.270

Table 6: MSE of methods for **NUE-GED** with $\mathcal{C}_n = (0, 0, 0)$ and $\mathcal{C}_e = (1, 2, 0)$.

	Mutag	Molhiv	Linux
GMN-Match	11.276	13.586	4.893
GMN-Embed	13.627	16.482	4.363
ISONET	1.468	2.142	1.930
GREED	11.906	13.723	3.847
ERIC	1.900	2.154	3.361
SimGNN	3.138	3.771	5.089
H2MN	3.771	3.735	5.443
GraphSim	4.696	5.200	6.597
EGSC	1.871	2.187	2.803

4 Conclusion

In this work, we first examined the current datasets for GED estimation and identified two key issues: (1) significant train-test leakage and (2) a lack of variation in edit costs. To address these challenges, we introduced a comprehensive suite of datasets incorporating four distinct edit cost configurations, aimed at advancing neural GED research. Lastly, we benchmarked nine state-of-the-art methods on these newly proposed datasets.

References

[1] Horst Bunke. On a relation between graph edit distance and maximum common subgraph. *Pattern Recognition Letters*, 18(8):689–694, 1997. 1

[2] Yunsheng Bai, Haoyang Ding, Song Bian, Ting Chen, Yizhou Sun, and Wei Wang. Simgnn: A neural network approach to fast graph similarity computation. *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 2018. URL https://api.semanticscholar.org/CorpusID:59528402. 1, 2, 3

[3] Chengzhi Piao, Tingyang Xu, Xiangguo Sun, Yu Rong, Kangfei Zhao, and Hong Cheng. Computing graph edit distance via neural graph matching. *Proc. VLDB Endow.*, 16(8): 1817–1829, apr 2023. ISSN 2150-8097. doi: 10.14778/3594512.3594514. URL https://doi.org/10.14778/3594512.3594514. 2

[4] Can Qin, Handong Zhao, Lichen Wang, Huan Wang, Yulun Zhang, and Yun Fu. Slow learning and fast inference: Efficient graph similarity computation via knowledge distillation. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=Q4SdMvWMxb. 2, 3

[5] Wei Zhuo and Guang Tan. Efficient graph similarity computation with alignment regularization, 2024. URL https://arxiv.org/abs/2406.14929. 2, 3

[6] Zhen Zhang, Jiajun Bu, Martin Ester, Zhao Li, Chengwei Yao, Zhi Yu, and Can Wang. H2mn: Graph similarity learning with hierarchical hypergraph matching networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, page 2274–2284, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383325. doi: 10.1145/3447548.3467328. URL https://doi.org/10.1145/3447548.3467328. 3

[7] Rishabh Ranjan, Siddharth Grover, Sourav Medya, Venkat Chakravarthy, Yogish Sabharwal, and Sayan Ranu. Greed: a neural framework for learning graph distance functions. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9781713871088. 3

[8] Yunsheng Bai, Haoyang Ding, Ken Gu, Yizhou Sun, and Wei Wang. Learning-based efficient graph similarity computation via multi-scale convolutional set matching. In *AAAI Conference on Artificial Intelligence*, 2018. URL https://api.semanticscholar.org/CorpusID:210928750. 3

[9] Runzhong Wang, Tianqi Zhang, Tianshu Yu, Junchi Yan, and Xiaokang Yang. Combinatorial learning of graph edit distance via dynamic embedding. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5237–5246, 2021. doi: 10.1109/CVPR46437.2021.00520. 2

[10] Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs, 2020. URL https://arxiv.org/abs/2007.08663. 2

[11] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 22118–22133. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/fb60d411a5c5b72b2e7d3527cfc84fd0-Paper.pdf.

[12] Asim Kumar Debnath, Rosa L. Lopez de Compadre, Gargi Debnath, Alan J. Shusterman, and Corwin Hansch. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *Journal of medicinal chemistry*, 34 2:786–97, 1991. URL https://api.semanticscholar.org/CorpusID:19990980. 2

[13] Julien Lerouge, Zeina Abu-Aisheh, Romain Raveaux, Pierre Héroux, and Sébastien Adam. New binary linear programming formulation to compute the graph edit distance. *Pattern Recognition*, 72:254–265, 2017. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2017.07.029. URL https://www.sciencedirect.com/science/article/pii/S003132031730300X. 3

[14] Yujia Li, Chenjie Gu, Thomas Dullien, Oriol Vinyals, and Pushmeet Kohli. Graph matching networks for learning the similarity of graph structured objects, 2019. URL https://arxiv.org/abs/1904.12787. 3

[15] Indradyumna Roy, Venkata Sai Baba Reddy Velugoti, Soumen Chakrabarti, and Abir De. Interpretable neural subgraph matching for graph retrieval. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7):8115–8123, Jun. 2022. doi: 10.1609/aaai.v36i7.20784. URL https://ojs.aaai.org/index.php/AAAI/article/view/20784. 3

A Dataset statistics

In Tables 7, 8, 9, and 10 we present the statistics of the proposed datasets for the tasks of **Uniform-GED**, **Non-uniform-GED**, **Label-GED**, and **Non-uniform-edge-GED** respectively.

Table 7: Salient characteristics of datasets for **Uniform-GED** with $\mathcal{C}_n = (1, 1, 0)$ and $\mathcal{C}_e = (1, 1, 0)$.

	#Graphs	# Train Pairs	# Val Pairs	# Test Pairs	Avg. $\lvert V\rvert$	Avg. $\lvert E\rvert$	Avg. U-GED
Mutag	729	95703	10585	10878	16.01	15.76	11.15
Code2	128	2926	325	378	18.77	17.77	10.02
Molhiv	1000	180300	20100	20100	15.01	15.65	11.77
Molpcba	1000	180300	20100	20100	17.52	18.67	9.58
AIDS	911	149331	16653	16836	10.97	10.97	7.38
Yeast	1000	180300	20100	20100	16.59	17.04	10.65
Linux	89	1431	153	190	8.71	8.35	4.91

Table 8: Salient characteristics of datasets for **Non-uniform-GED** with $\mathcal{C}_n = (1, 3, 0)$ and $\mathcal{C}_e = (1, 2, 0)$.

	#Graphs	# Train Pairs	# Val Pairs	# Test Pairs	Avg. $\lvert V\rvert$	Avg. $\lvert E\rvert$	Avg. NU-GED
Mutag	729	95703	10585	10878	16.01	15.76	18.57
Code2	128	2926	325	378	18.77	17.77	16.43
Molhiv	1000	180300	20100	20100	15.01	15.65	19.86
Molpcba	1000	180300	20100	20100	17.52	18.67	15.73
AIDS	911	149331	16653	16836	10.97	10.97	12.07
Yeast	1000	180300	20100	20100	16.59	17.04	17.74
Linux	89	1431	153	190	8.71	8.35	7.94

Table 9: Salient characteristics of datasets for **Label-GED** with $\mathcal{C}_n = (1, 1, 1)$ and $\mathcal{C}_e = (1, 1, 0)$.

	#Graphs	# Train Pairs	# Val Pairs	# Test Pairs	Avg. $\lvert V\rvert$	Avg. $\lvert E\rvert$	Avg. L-GED
Mutag	1287	298378	33153	33411	15.25	15.02	15.68
Code2	181	5886	666	703	18.9	17.9	16.87
Molhiv	1000	180300	20100	20100	14.45	15.03	16.55
Molpcba	1000	180300	20100	20100	17.43	18.59	13.84
AIDS	1588	453628	50403	51040	10.53	10.54	11.12

Table 10: Salient characteristics of datasets for **Non-uniform-edge-GED** with $\mathcal{C}_n = (0, 0, 0)$ and $\mathcal{C}_e = (1, 2, 0)$.

	#Graphs	# Train Pairs	# Val Pairs	# Test Pairs	Avg. $\lvert V\rvert$	Avg. $\lvert E\rvert$	Avg. NUE-GED
Mutag	729	95703	10585	10878	16.01	15.76	11.16
Molhiv	1000	180300	20100	20100	15.01	15.65	11.87
Linux	89	1431	153	190	8.71	8.35	5.32

B Additional Experiments

In Tables 11, 12, 13, and 14, we report the MSE along with standard error on all methods for the **Uniform-GED**, **Non-uniform-GED**, **Label-GED**, and **Non-uniform-edge-GED** respectively. In each case, yellow denotes the best performer.

Table 11: MSE $\pm$ STD of methods for **Uniform-GED** with $\mathcal{C}_n = (1, 1, 0)$, and $\mathcal{C}_e = (1, 1, 0)$.

	Mutag	Code2	Molhiv	Molpcba	AIDS	Linux	Yeast
GMN-Match	0.797 ± 0.013	1.677 ± 0.187	1.318 ± 0.020	1.073 ± 0.011	0.821 ± 0.010	0.687 ± 0.088	1.175 ± 0.013
GMN-Embed	1.032 ± 0.016	1.358 ± 0.104	1.859 ± 0.020	1.951 ± 0.020	1.044 ± 0.013	0.736 ± 0.102	1.767 ± 0.021
ISONET	1.187 ± 0.021	0.879 ± 0.061	1.354 ± 0.015	1.106 ± 0.011	1.640 ± 0.020	1.185 ± 0.115	1.578 ± 0.019
GREED	1.398 ± 0.033	1.869 ± 0.140	1.708 ± 0.019	1.550 ± 0.017	1.004 ± 0.012	1.331 ± 0.169	1.423 ± 0.015
ERIC	0.719 ± 0.011	1.363 ± 0.110	1.165 ± 0.018	0.862 ± 0.009	0.731 ± 0.008	1.664 ± 0.260	0.969 ± 0.010
SimGNN	1.471 ± 0.024	2.667 ± 0.215	1.609 ± 0.020	1.456 ± 0.020	1.455 ± 0.020	7.232 ± 0.762	1.999 ± 0.043
H2MN	1.278 ± 0.021	7.240 ± 0.527	1.521 ± 0.020	1.402 ± 0.020	1.114 ± 0.015	2.238 ± 0.247	1.353 ± 0.018
GraphSim	2.005 ± 0.031	3.139 ± 0.206	2.577 ± 0.064	1.656 ± 0.023	1.936 ± 0.026	2.900 ± 0.318	2.232 ± 0.030
EGSC	0.765 ± 0.011	4.165 ± 0.285	1.138 ± 0.016	0.938 ± 0.010	0.627 ± 0.007	2.411 ± 0.325	0.950 ± 0.010

Table 12: MSE $\pm$ STD of methods for **Non-uniform-GED** with $\mathcal{C}_n = (1, 3, 0)$, and $\mathcal{C}_e = (1, 2, 0)$.

	Mutag	Code2	Molhiv	Molpcba	AIDS	Linux	Yeast
GMN-Match	69.210 ± 0.883	13.472 ± 0.970	76.923 ± 0.862	23.985 ± 0.224	31.522 ± 0.513	21.519 ± 2.256	63.179 ± 1.127
GMN-Embed	72.495 ± 0.915	13.425 ± 1.035	78.254 ± 0.865	28.437 ± 0.268	33.221 ± 0.523	20.591 ± 2.136	60.949 ± 0.663
ISONET	3.369 ± 0.062	3.025 ± 0.206	3.451 ± 0.039	2.781 ± 0.029	5.513 ± 0.092	3.031 ± 0.299	4.555 ± 0.061
GREED	68.732 ± 0.867	11.095 ± 0.773	78.300 ± 0.795	26.057 ± 0.238	34.354 ± 0.557	20.667 ± 2.140	60.652 ± 0.704
ERIC	1.981 ± 0.032	12.767 ± 1.177	3.377 ± 0.070	2.057 ± 0.020	1.581 ± 0.017	7.809 ± 0.911	2.341 ± 0.030
SimGNN	4.747 ± 0.079	5.212 ± 0.360	4.145 ± 0.051	3.465 ± 0.047	4.316 ± 0.071	5.369 ± 0.546	4.496 ± 0.060
H2MN	3.413 ± 0.053	9.435 ± 0.728	3.782 ± 0.046	3.396 ± 0.046	3.105 ± 0.043	5.848 ± 0.611	3.678 ± 0.046
GraphSim	5.370 ± 0.092	7.405 ± 0.577	6.643 ± 0.181	3.928 ± 0.053	5.266 ± 0.081	6.815 ± 0.628	6.907 ± 0.137
EGSC	1.758 ± 0.026	3.957 ± 0.365	2.371 ± 0.025	2.133 ± 0.022	1.693 ± 0.023	5.503 ± 0.496	2.157 ± 0.027

Table 13: MSE $\pm$ STD of methods for **Label-GED** with $\mathcal{C}_n = (1, 1, 1)$, and $\mathcal{C}_e = (1, 1, 0)$.

	Mutag	Code2	Molhiv	Molpcba	AIDS
GMN-Match	1.057 ± 0.011	5.224 ± 0.404	1.388 ± 0.018	1.432 ± 0.017	0.868 ± 0.007
GMN-Embed	2.159 ± 0.026	4.070 ± 0.318	3.523 ± 0.040	4.657 ± 0.054	1.818 ± 0.014
ISONET	0.876 ± 0.008	1.129 ± 0.084	1.617 ± 0.020	1.332 ± 0.014	1.142 ± 0.010
GREED	2.876 ± 0.032	4.983 ± 0.531	2.923 ± 0.033	3.902 ± 0.044	2.175 ± 0.016
ERIC	0.886 ± 0.009	6.323 ± 0.683	1.537 ± 0.018	1.278 ± 0.014	1.602 ± 0.036
SimGNN	1.160 ± 0.013	5.909 ± 0.490	1.888 ± 0.031	2.172 ± 0.050	1.418 ± 0.020
H2MN	1.277 ± 0.014	6.783 ± 0.587	1.891 ± 0.024	1.666 ± 0.021	1.290 ± 0.011
GraphSim	1.043 ± 0.010	4.708 ± 0.425	1.817 ± 0.021	1.748 ± 0.021	1.561 ± 0.021
EGSC	0.776 ± 0.008	8.742 ± 0.831	1.273 ± 0.016	1.426 ± 0.018	1.270 ± 0.028

Table 14: MSE $\pm$ STD of methods for **Non-uniform-edge-GED** with $\mathcal{C}_n = (0, 0, 0)$, and $\mathcal{C}_e = (1, 2, 0)$.

	Mutag	Molhiv	Linux
GMN-Match	11.276 ± 0.143	13.586 ± 0.171	4.893 ± 0.527
GMN-Embed	13.627 ± 0.179	16.482 ± 0.188	4.363 ± 0.420
ISONET	1.468 ± 0.020	2.142 ± 0.023	1.930 ± 0.186
GREED	11.906 ± 0.148	13.723 ± 0.136	3.847 ± 0.397
ERIC	1.900 ± 0.028	2.154 ± 0.024	3.361 ± 0.353
SimGNN	3.138 ± 0.052	3.771 ± 0.046	5.089 ± 0.524
H2MN	3.771 ± 0.062	3.735 ± 0.047	5.443 ± 0.566
GraphSim	4.696 ± 0.076	5.200 ± 0.074	6.597 ± 0.697
EGSC	1.871 ± 0.028	2.187 ± 0.025	2.803 ± 0.260