
Graph Level Anomaly Detection

Jonathan Gomes-Selman
Department of Computer Science
Stanford University
jgs8@stanford.edu

Nikita Demir
Department of Computer Science
Stanford University
ndemir8@stanford.edu

1 Introduction

The key motivation behind our project is to expand upon the traditional approach of node and link based anomaly detection within graphs by considering full graph level anomaly detection. Graph level anomaly detection looks to broaden the scope of anomaly detection by distinguishing different classes of entire graphs. Namely, we look to learn a representation over a particular family of graphs, such that we can identify anomalous graphs from different graph families. This problem formulation separates itself from traditional graph classification problems as it is framed as an unsupervised learning problem over semi-supervised data. We propose the use of unsupervised learning models to learn distributions over classes of graphs utilizing semi-supervised data where examples of anomalous graphs are not easily accessible, defined, or could cover a large range of possibilities. This anomaly detection task holds value in many different domains such as drug discovery, molecular engineering, and anomaly detection in dynamically evolving networks. For example, one could model different molecules as graphs and look to learn a representation over a set of molecules that perform a certain function while being able to distinguish molecules from other families which can come from a very large pool of potential other molecule classes.

Although many methods have been developed to identify anomalous nodes or links within a network, such as [1; 2], there has been relatively little exploration into anomaly detection at a graph level. Therefore, we believe this research presents a unique perspective on anomaly detection applied to graphs. Moreover, this research presents both an interesting problem, as it requires modeling a comprehensive representation of a graph's local and high level structural features, as well as a challenging problem because of the unique properties of graph based data, such as long dependencies and size variability, that prevent application of more traditional machine learning methods for anomaly detection on structured data.

As described earlier, a key component in graph level anomaly detection in a semi-supervised data setting (i.e. having access only to "normal"/"non-anomalous" data) is defining a specific unsupervised model/objective to learn a distribution over the normal graph class. In this project, we formulate two unsupervised learning objectives for graph level anomaly detection. Namely, we compare **1**) generative modeling for graph likelihood estimation and **2**) a novel method based on masked graph representation learning. In applying generative models to the problem of anomaly detection we utilize GraphRNNs and GRANs - Graph Recurrent Attention Networks as auto-regressive models capable of learning likelihood distributions over arbitrary classes of graphs [3; 4]. We also propose what we believe to be a novel method for anomaly detection based on masked representation learning for graphs. Inspired partly by masked language models [5] and unsupervised learning methods for GNNs such as GraphSage [6], we look to learn meaningful representations over a family of graphs by modeling one step edge completion problems. Through self-supervised learning, we look to model important structural features and motifs of a family of graphs, such that when given an out of distribution graph we can identify the graph as anomalous by analyzing our models predictions on many masked completion problems. The goal is that in learning to predict masked out graph features on a particular graph class, when presented an anomalous class our model will flag anomalous structures as unlikely "completions" based on the normal distribution.

Our Contributions: **1)** We define and analyze the task of anomaly detection at the graph level, a task we believe to be under researched. **2)** We present two different unsupervised learning objectives as potential methods for anomaly detection in this setting: **2a)** likelihood based autoregressive models defined on graphs and **2b)** a novel masked representation learning method for anomaly detection. **3)** We present detailed empirical evaluation over synthetic and real world datasets, demonstrating promising potential and points of future progress for both methods.

2 Related Works

Surprisingly, we have not found much work in the domain of graph-level anomaly detection. However, we have found several works in related domains that have been influential for our approaches. One of the key areas we have taken inspiration from is generative modeling of graphs, a domain that seeks to learn distributions over arbitrary classes of graphs. In [3], You et al. propose an autoregressive approach to graph generation that is trained sequentially on existing graphs and then generates them at inference time by breaking the process into a sequence of node and edge formations. We found their autoregressive architecture and its success to be a super practical model for our graph level likelihood prediction as discussed later. As a further step, Renjie et al.'s work in graph generation with graph recurrent attention networks [4] and Velickovic et al.'s work with general graph attention networks [7] could provide interesting more powerful model architectures for calculating graph level likelihoods.

Another approach to graph generation we found highly relevant was [8] You et al.'s work in an RL approach to goal-directed molecular graph generation. We found their approach to structuring a learning problem using partially observed subgraphs (discussed later) and an action-based generation framework very interesting. Similar to the autoregressive approach, an RL approach could also be used to extract graph likelihoods. Another interesting related domain is graph-level representation learning. As authors Bai et al. demonstrate in [9] one can generate graph-level embeddings in an unsupervised manner. In particular, the authors propose a novel Multi-Scale Node Attention mechanism and pretrain their models on large amounts of data and then fine-tune on downstream tasks like classification and similarity ranking. Ideally, we should be able to use these graph level embeddings for anomaly detection by comparing different embeddings against each other.

3 Methods

To analyze the efficacy of the two proposed unsupervised learning methods for graph level anomaly detection, namely generative modeling and masked representation learning, we present two different experimental approaches.

3.1 GraphRNN and GRAN for Graph Level Anomaly Detection

Both GraphRNNs and GRANs are auto-regressive models defined on sequential representations of graphs [3; 4]. As GRANs are an explicit extension of GraphRNNs, both share the same learning objective and, thus, look to model the likelihood distribution over an arbitrary set of graphs. Therefore, for a given graph G , we can estimate the likelihood of that graph, $P(G)$, with GraphRNNs and GRANs. Moreover, given a set of graphs F we can obtain a *distribution over the likelihoods* for that set of graphs, which aims to capture our model's belief about the likelihood of a graph from the family F . With this intuition, we define the following experimental procedure:

1. Train GraphRNN or GRAN on a specific class of graphs dubbed the "normal class" (i.e. the non-anomalous graph).
2. Compare the Negative Log Likelihood (NLL) distribution over the set of training graphs to the distribution over a held out test set of graphs from the **same / normal** graph class. Given successful training of the model, these distributions should be similar (\approx) showing the model generalizes properly to the test set and has learned a meaningful distribution over the likelihoods of graphs from the normal class.
3. For out-of-distribution graph classes (anomalous classes), compare the NLL distribution given GraphRNN or GRAN trained on the normal class. The desired result is a separation

between the NLL distribution of normal class graphs and out-of-distribution anomalous graphs such that anomalous graphs are less likely.

Note: When estimating the NLL of a given graph G under both GraphRNN and GRAN, there are two key points of consideration that we address.

Permutation Invariance: because many different sequences S^π exist for a single graph, even when only considering BFS sequences as in GraphRNN and GRAN, it is intractable to consider all such sequences; therefore, we approximate the likelihood $P(G)$ as the average over a fixed set of randomly sampled S^π 's, for a given graph G (i.e., average $P(S^\pi)$).

Size Dependence: When modeling sequences in an autoregressive manner, there is an important dependence on sequence length when calculating the NLL. Specifically, longer sequences are more likely to have higher NLLs (i.e. appear less likely) when compared to shorter sequences. To mitigate this effect, we normalize by averaging over the number of nodes in the graph.

3.2 Masked Representation Learning for Graphs

We present masked representation learning for anomaly detection as a novel approach to graph level anomaly detection. Masked representation learning for graph level anomaly detection is inspired partly by the success of masked language models for NLP [5], prior work by You et al. in utilizing supervised one-step trajectories for boosting policy gradient algorithms for graph generation [8], and unsupervised learning methods for GNNs [6]. The motivation for this method is primarily based on the idea that although generative models such as GraphRNN and GRAN are able to model likelihoods of graphs, we are not actually interested in the generative nature of these models. Moreover, we really care about learning structural characteristics/representations of a family of graphs.

In order to address this problem, we look to the area of self-supervised learning or masked modeling, where we aim to learn meaningful representations over a set of graphs by learning to predict properties of those graphs. Therefore, a key aspect of the learning algorithm is generating self supervised data examples that represent meaningful features for our model to predict about the given graph data. We propose 2 different self-supervised approaches.

The first method is explicitly based on masked representation learning. Namely, rather than modeling the probability $P(G)$ over an entire graph, we look to model the transition probability between a graph G and G' where we perform a single edge completion action to generate G' from G . Instead of modeling the complex conditional probability of generating the next portion of a graph based the full trajectory of the graph generation procedure (i.e. $P(s_t|s_{t-1}, \dots, s_0)$) we look to model a simpler one step completion, where we assume our transition dynamics follow the Markov principle $P(s_t|s_{t-1}, \dots, s_0) = P(s_t|s_{t-1})$. Given this masked learning objective, we define our masked completion task with respect to completion examples defined on the entire graph G , referred to later as *full-mask*.

The second method is inspired by the unsupervised edge/proximity prediction objective used in GraphSage, but now applied across multiple graphs:

$$J_G(z_u) = -\log(\theta(z_u^t z_v)) - \mathbb{E}_{v_n \sim P_n(u)} \log(\sigma(-z_u^t z_{v_n}))$$

where we define u and v to be neighboring nodes, and $P_n(u)$ to be a negative sampling distribution. We see that this objective encourages neighboring nodes to have similar representations and non-adjacent nodes to be distinct. The key difference compared to the first two methods is that we are now **not** explicitly breaking the graph by removing an edge, which while useful in graph generation may hinder self-supervised property prediction models by changing the underlying graph dynamics. We refer to this method later as *full-edge-prediction*

Based on the specific goals defined above, we define the following algorithm for generating self-supervised data samples for the two proposed methods.

Data Generation:

1. Sample a random graph G from the training graphs.
2. Sample a random edge within G between nodes (u, v) .
 - (a) For *full-mask*: remove this edge from G .

3. Sample a random "negative" edge (u, w) involving the node u from above that does not exist in the general graph G and is not the edge (u, v) .
4. For the data pair (u, v) compute the shortest path distance between u and v that is not the direct edge (u, v) - (i.e. the second shortest path = sp_p). Do the same for the pair (u, w) - sp_n . Note, This decision is discussed after this section.
5. The data sample is now defined as $(G, (u, v), (u, w), 1/sp_p, 1/sp_n)$, including a positive trajectory sample, and a negative completion. Note, if either shortest path does not exist in the graph G than set the resulting inverse value to 0.

Before, presenting the full masked representation learning algorithm, we first discuss step 4 in the data generation procedure. The motivation behind including the "second shortest path" between the positive pair and the shortest path between the negative pair is to address the lack of positional awareness inherent in GNNs. Motivated by the exploration of Position-aware Graph Neural Networks [10], we believe that this added positional information provides added predictive power in the context of edge completion as it allows for differentiation of similarly or identically embedded nodes with different relative positions with respect to a node of interest. Now we present the general algorithm for masked representation learning over a set of graphs.

Algorithm:

1. Given a training example as described above $(G', (u, v), (u, w), 1/sp_p, 1/sp_n)$, first run graph G' through a GNN to generate node embeddings for the nodes.
2. Given the node embeddings for the nodes $u, v,$ and w generate positive example as $[u, v, 1/sp_p]$ and negative example $[u, w, 1/sp_n]$, where $[.]$ represents the concatenation operation. With these examples, run the concatenated embeddings through an MLP to produce a binary output prediction for whether an edge should exist between the two nodes or not (i.e. predict the trajectory).
3. Lastly, perform gradient descent to update the GNN parameters and the MLP based on the binary cross entropy loss applied to the MLP edge predictions.

Now given the above training procedure for our masked representation learning model, we look to indirectly model the probability $P(G)$ of a given graph G by analyzing the distribution of positive prediction scores (i.e. the prediction made by our model for the likelihood of a given positive completion) for many different randomly sampled positive completion examples. Based on this idea we develop the following experimental procedure for anomaly detection with a masked representation learning model for graph:

1. Train a masked representation learning model as described above on a specific class of graphs dubbed the "normal class".
2. For each masked completion, we can think of the MLP output as a score for that completion. Namely, a positive score indicates a belief that an edge should exist under our "graph model" and vice versa for a negative score. Therefore, on the held out test dataset we can look at the distribution of scores over many masked completion examples as a representation of our models learned distribution over a class of graphs.
3. To evaluate if a test time graph is anomalous or not we sample many random masked completion examples and compute the positive score distribution for that graph (i.e. only look at the scores associated with positive / existing trajectory completions). If this score distribution matches that of our normal graph family than the graph is not anomalous, but if it differs substantially, such as if our model nearly always predicts that certain edge completions wouldn't exist in our family of graphs, then the graph is an anomaly.

4 Dataset

We include experimental analysis on two classes of datasets: synthetically generated and real world datasets. We use synthetically generated data to allow for very structured experimentation and analysis, where we are able to control the level of similarity between graphs classes (through shared structure, statistical properties, and added randomness) to help inform the analysis and interpretation of results. Figure 1 shows the different types of structure for these synthetic graph classes.

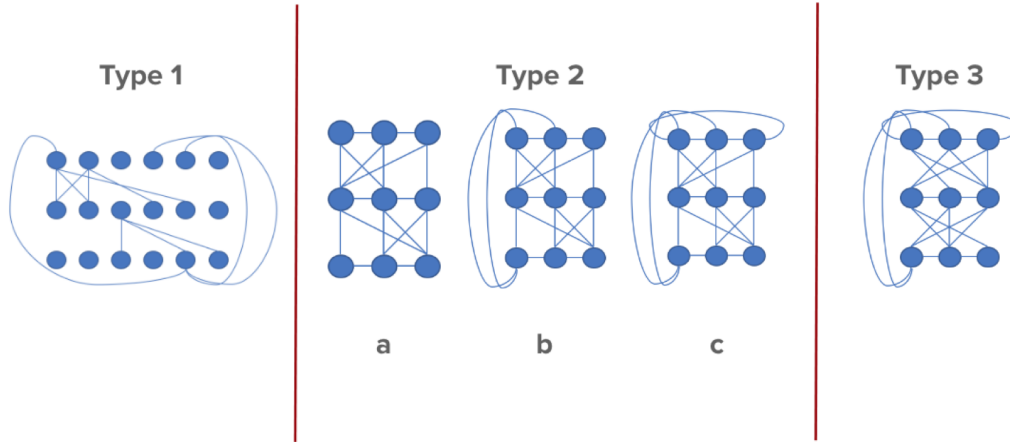


Figure 1: **Type 1)** Three random edges to the next layer for each node with a circular connection between the final and first layers. **Type 2) a)** Grid base structure with an extra random edge to the next layer for each node; **b)** Type 2a with added circular connection between the first and last layer; **c)** Type 2b with added circular individual layers. **Type 3)** Two random edges to the next layer for each node with circular layers and a circular connection between the first and last layers. All experiments use synthetic graphs of size $width = 6$ and $height = 10$, giving 60 node graphs.

4.1 Real World Datasets

In order to further explore the effectiveness of our model, we look to test our methods on real world datasets.

1. D&D [11]: A dataset containing 1178 proteins labeled as enzymes or non-enzymes.
2. AIDS [12]: A dataset with 2000 chemical compounds labeled as active or inactive with regard to the AIDS virus.
3. IMDB-MULI [13]: A collaboration dataset with 1500 ego-networks split among 3 genre classes (Comedy, Romance, Sci-Fi) in which nodes represent actors/actresses and there is an edge between them if they existed in the same genre movie.

5 Experiments

5.1 Evaluation

5.1.1 GraphRNN and GRAN

As described earlier we looked at using GraphRNN and GRAN as generative modeling approaches for graph level anomaly detection. We evaluated each model on the real world DD, AIDS, and IMDB-MULTI datasets by first separating a single class (class 0 for all real world datasets) 80/20 into training and testing graphs. Then, we ran our trained model on a separate class in the same real world dataset (class 1 for all datasets besides class 0 for AIDS). This way our anomalous graphs are similar enough since they belong to the same general dataset family, yet different because they correspond to different classes. We then plotted the resulting log likelihoods for each of the 3 graph types. The model was run 10 times on each graph to generate 10 random sample BFS node orderings and the resulting likelihoods were averaged over. Finally, we plotted a distribution of log-likelihoods for each potential graph type and visually inspect their overlap. We don't evaluate the GRAN on the synthetic dataset because by confirming GraphRNN's performance on the synthetic dataset we think that confirms a similar architecture like GRAN's performance.

5.1.2 Masked Representation Model

When evaluating the performance of our masked representation learning model on the anomaly detection task, we consider two avenues of evaluation:

1. To compare the model’s representation of the normal and anomalous class of graphs, we qualitatively compare distributions of predicted scores for positive completion examples. Specifically, we hope to see a difference in the distributions of scores between the normal and anomalous class when considering a large collection of sampled positive completion examples for each graph class.
2. As a second and quantitative approach, we use clustering algorithms to evaluate our models ability to differentiate between normal and anomalous graph classes. In this setting, we generate a score vector for a given graph G (either normal or anomalous) as a vector of predicted value for 100 sampled positive completion examples. Since we want these score vectors to be permutation invariant, we then compute a 10 dimensional histogram vector with evenly spaced buckets between 0 and 1 (as the sigmoid predicted output lies in this fixed range). Thus, given a collection of graphs F containing normal and anomalous graphs, we can compute the score vector for each individual graph and apply a clustering algorithm that searches for two clusters and gives a corresponding clustering accuracy. The hope, is that the two clusters discover represent a normal and anomalous cluster. Note, because we are clustering distribution vectors we use an Agglomerative clustering algorithm, which is a hierarchical clustering algorithm, rather than the standard k-means.

When showing results, we default to presenting clustering results; however, in the cases where the model performs quite poorly, indicated by a low clustering accuracy, we include a discussion of the qualitative score distributions to help reveal insights into the low performance. Additionally, in the masked representation learning setting we compare the performance between the two masked learning models 1) *full-mask* 2) *full-edge-prediction*.

5.2 Synthetic Experiments

5.2.1 Similarity Sensitivity

Motivation: In this experiment we test the sensitivity of our models to graph families with varying structural similarity to the training distribution. Specifically, we take as our training graphs (the "normal class") Type 1 graphs, and we compare against Type 2a, 3, and Random 6-regular graphs.

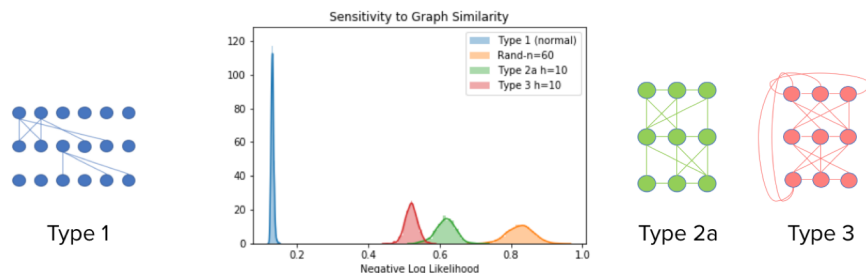


Figure 2: Comparison between graphs of relative similarity where the normal class is in blue using GraphRNN.

GraphRNN

Based on the results shown in Figure 4, we see clear separation between “normal” and “anomalous” classes demonstrating the ability for GraphRNNs to differentiate graphical structure and learn a well defined distribution over the normal class. Moreover, the relative ordering of distributions matches the relative similarity to the “normal” class.

Masked Representation Learning Using clustering as an evaluation metric, we are able to achieve perfect clustering separation / accuracy for each of the three anomalous graph classes (type 2a, type 3, and rand-n) with both masked learning methods. This indicates that both models accurately solves the

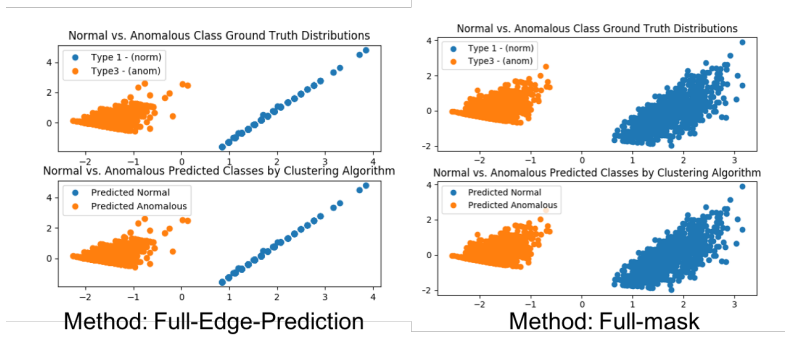


Figure 3: 2D representation of the 10D graph distribution vectors using PCA comparing Type1 (normal class) and Type3 (anomalous class). The top row shows the ground truth value for each data point, and the bottom row shows clustering assignment.

anomaly detection problem, by assigning significantly different score distribution vectors to graphs from the normal and anomalous classes. Figure 3 shows the clustering assignments when comparing the normal class (Type 1) to the anomalous (Type 3) - which of the anomalous classes is the most similar to the normal class. Interestingly, we see that the *full-edge-prediction* model learns a more certain representation over the normal class, as we see significantly less variation in the distribution vectors.

5.2.2 Fine Grained Sensitivity

Motivation: Type 2c and Type 3 graphs share many structural and statistical properties. While in Type 3 graphs, layers in the ladder are connected by randomly distributed edges, what is special about the Type 2c graph is that the linking between layers consist of a one-to-one mapping of the nodes, together with 6 random edges. Given this close similarity in structure, we explore whether our different models consider the setting where Type 2c graphs are the "normal" class and Type 3 are the "anomalous" class.

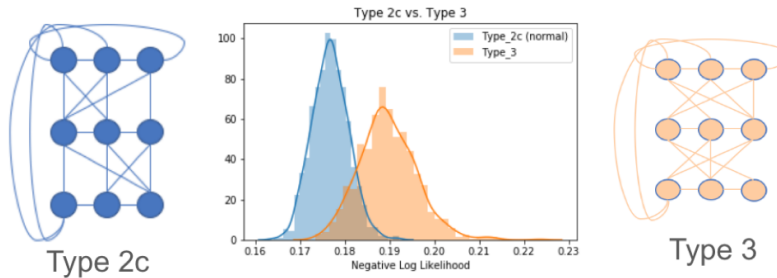


Figure 4: Normal Class in Blue, Anomalous Class in Orange. Using GraphRNN we see separation in the distributions but not exact separation.

GraphRNN

We see that GraphRNN is able to accurately predict a separation between the "normal" and "anomalous" class distributions. While overlap does still exist, these results provide great promise for the ability of GraphRNNs to identify underlying structure in an unsupervised setting, as the graph classes share great similarity. In fact, one hypothesis about the occurrence of the overlapping region is that this region includes Type 3 graphs that by chance have more grid like structure than other Type 3 graphs and thus are truly more similar to Type 2c graphs.

Masked Representation Learning Using clustering as an evaluation metric, we are able to achieve perfect clustering separation / accuracy when using the *full-mask* method and 99.7% accuracy when using the *full-edge-prediction* method. Both show very promising results for the masked representation learning model, as both models outperform the ability of GraphRNN to distinguish

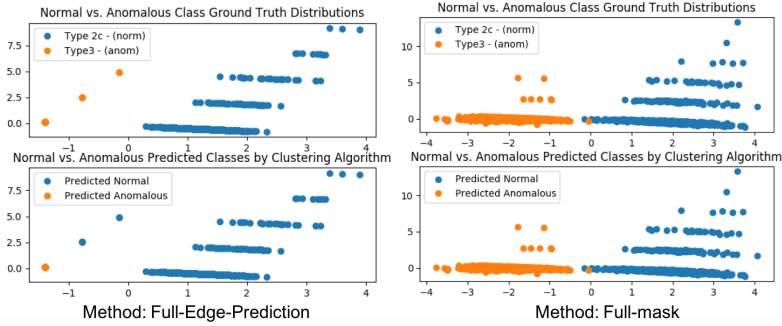


Figure 5: 2D representation of the 10D graph distribution vectors using PCA comparing Type2c (normal class) and Type3 (anomalous class) using masked representation learning.

between the graph classes. It is interesting to again see how the *full-edge-prediction* method tends to generate lower variance prediction vectors. Namely, this suggest that the *full-edge-prediction* method is able to potentially learn a more precise representation of the normal graph class and is a more powerful approach, noting that the accuracy error is essentially negligible.

5.3 Real World Datasets

5.3.1 GraphRNN and GRAN

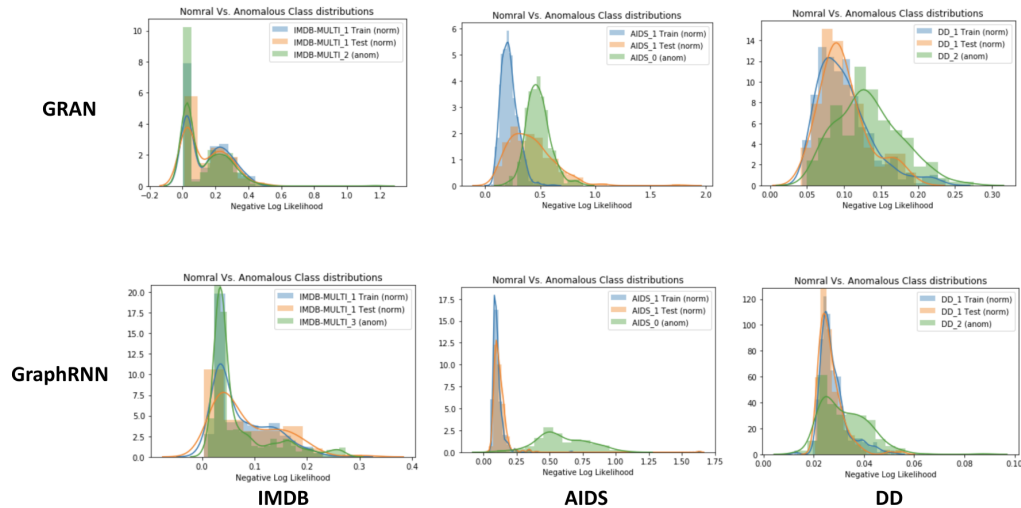


Figure 6: Comparison of the negative log likelihood distributions of the GRAN and GraphRNN approaches to graph anomaly detection.

Moving to the real world datasets, we first examine our negative likelihood distribution plots of the GRAN and GraphRNN approaches to detecting anomaly graphs. We see that the two generative architectures performed roughly similarly with the anomaly class being distinguishable in the AIDS and DD datasets to varying degrees and not distinguishable in the IMDB dataset at all. We notice that in general the GRAN seems to more easily overfit as evidenced by its performance on the AIDS dataset in which the train and test distributions are slightly different. Interestingly, GRAN and GraphRNN show complimentary results, as evidenced by GRAN’s improved performance on the complicated D&D dataset, and GraphRNN’s ability to almost completely distinguish the AIDS dataset classes. In terms of the performance on the D&D, it makes sense that we see improvements in the GRAN model as one of its improvements upon GraphRNN is its ability to better handle large graphs.

Overall, when analyzing these results, it is important to note that the real world datasets are rather complicated even for supervised approaches to distinguish. [14] showed classification accuracies of around 80% on the D&D dataset and [13] show 43% on the IMDB-MULTI. Thus we can reasonably conclude that since the real world datasets are already inherently difficult even in supervised scenarios our unsupervised approaches using GraphRNN and GRAN architectures show good promise for enabling graph level anomaly detection.

5.3.2 Masked Representation Learning

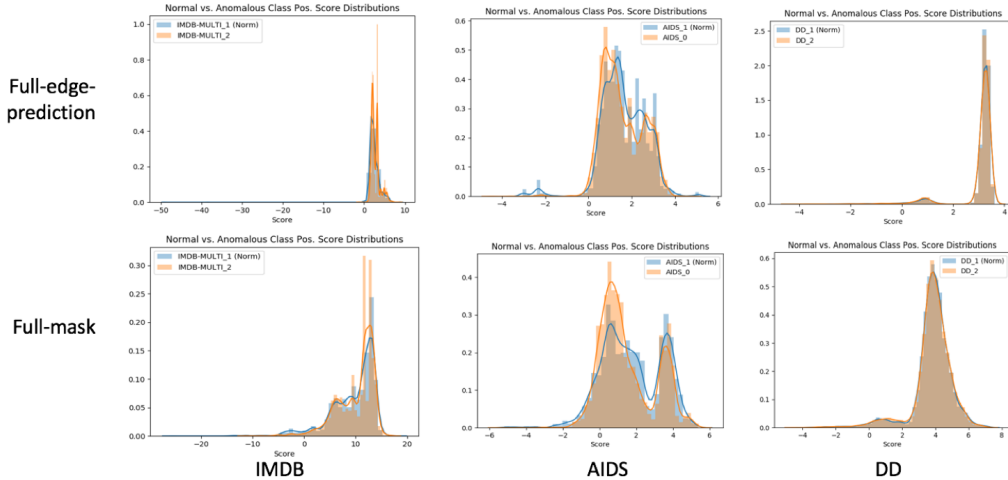


Figure 7: Positive completion score distributions for normal (blue) and anomalous (orange) graph classes using masked representation learning. Each row represents a different modeling approach, and the columns a different dataset.

On the real world datasets, we see the performance of the Masked Representation Learning algorithm diminishes greatly. For all three real world datasets both masked learning models are unable to perform better than $\approx 50\%$ clustering accuracy. Therefore, we look to the distributions over the predicted scores for positive completion examples in both the normal and anomalous datasets. In 7, we see almost no separation between the normal and anomalous score distributions, indicating that both the *full-edge-prediction* and *full-mask* models are unable to learn distinguishing representations for the normal classes. While these results are not as desired, there are still interesting takeaways from these experiments. With respect to the masked representation Learning model, an interesting take-away is that we again see a tendency for the *full-edge-prediction* model to be more certain about the scores it assigns, as we see less overall variance in the score distributions. Moreover, from figure 8 we that the *full-edge-prediction* model seems to learn greater separation between the positive and negative self-supervised examples. These observations provide indication that the choice to not remove the graph edge in order to maintain the graphs internal dynamics is a potentially meaningful decision.

Overall, the results on the real world dataset are interesting and still indicate promise. We see that for both masked learning models, the models are able to learn separation between the positive and negative self-supervised examples (Figure 8), providing an indication that the models are learning important characteristics that define that graphs. In terms of not being able to differentiate between the normal and anomalous classes, one potential direction to explore is considering more positive samples when defining the score distribution for a given test graph. Right now we only consider 100 samples, but in the future, by sampling 1000 or 10000, these distributions could better capture anomalous completions hidden within anomalous test graphs. Another point of concern is that the masked representation model makes predictions primarily based lower level graph features and struggles to effectively capture entire graph level features. Specifically, in the formulation of the Masked Representation model we focus on making edge predictions/completions based primarily on GNN embedding outputs for nodes; therefore, we are explicitly limiting ourselves to the k-hop neighborhoods for the given nodes. Because of this we are explicitly limiting the models ability to detect very high level graph anomalies that for instance consider how many of a particular edge

completions exist within a graph. Additionally, unlike a generative model which looks to model the entire generative process of a graph, the masked learning model does not directly take into account missing features, such as if a house graph is missing its roof. While framing the problem in this way is not necessarily incorrect, this definition of anomaly detection may not apply to certain real-world datasets or may simply need to be further expanded to address the discussed limitations.

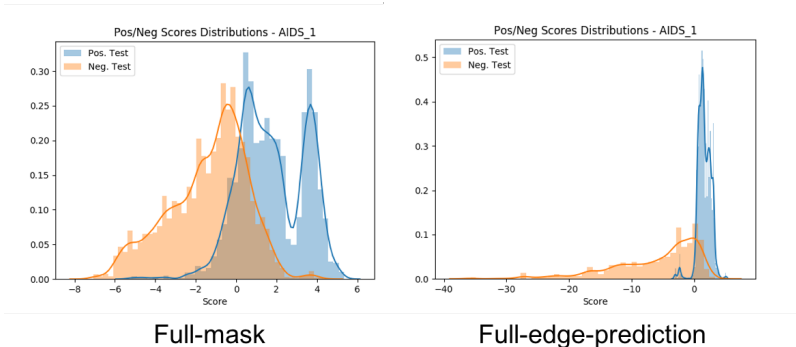


Figure 8: Comparison of scores assigned to positive and negative completion examples. Our learning objective seeks to assign a positive score to positive completion examples and a negative score to negative completion examples

6 Conclusion

In this work we outlined and explored a few novel ways to expand upon the traditional approach of node and link based anomaly detection within graphs by considering full graph level anomaly detection. Namely, we looked to learn a representation over a particular family of graphs such that we can identify anomalous graphs from different graph families using two different unsupervised learning objectives: a likelihood based autoregressive generative model approach and a novel masked representation learning method. To our knowledge, this task has had relatively little research exploration yet has significant real world implications like discovering outlier molecules or highlighting anomalous states of dynamic networks. Our results on controlled synthetic datasets show great potential for our anomaly detection approaches, and while the real world datasets proved to be more difficult, as expected given the relatively low classified accuracies in previous works’ approaches, our results especially in the generative likelihood models highlight again their potential.

7 Future Work and Directions

For our first method of using autoregressive graph generative approaches one future direction would be to use clustering approaches to label new graphs as belonging to either a training dataset or anomaly dataset and provide a supplemental set of quantitative results. This would effectively correspond to classification from an unsupervised approach to semi-supervised data. Another direction that would potentially address issues we observed in real world experiments is to consider training on much larger datasets as well as looking to incorporate adversarial training to enhance generative modeling generalizability. Finally, in using autoregressive modeling, an interesting future direction could be looking into ways to de-emphasize the generation capabilities of the model and focus further on likelihood estimation.

For our masked representation learning model, we believe that this idea is still fairly early along in its infancy. As discussed earlier, one future direction is to include a global representation component when making edge completion predictions. Additionally, an interesting direction could be in the general development of self-supervised learning for tasks such as sub-graph anomaly prediction, node anomaly prediction, or as additional information for more general graph learning tasks.

8 Contributions

Nikita: Coding and running experiments for GRAN, proposal/milestone/final report writing
Jonathan: Coding and running experiments for GraphRNN, masked representation learning formulation, coding of masked representation learning framework, experimental analysis of masked representation learning, writing focus on methods, masked representation learning sections, and introduction

9 Acknowledgements

We would like to especially thank Jiaxuan You for his advise and brainstorming throughout the process. Additionally, we would like to thank Professor Leskovec, Michele, and the fellow course staff for an extremely exciting and well taught course.

References

- [1] S. Kumar, R. West, and J. Leskovec, “Disinformation on the web: Impact, characteristics, and detection of wikipedia hoaxes,” in *Proceedings of the 25th international conference on World Wide Web*, pp. 591–602, International World Wide Web Conferences Steering Committee, 2016.
- [2] S. Kumar, J. Cheng, and J. Leskovec, “Antisocial behavior on the web: Characterization and detection,” in *Proceedings of the 26th International Conference on World Wide Web Companion*, pp. 947–950, International World Wide Web Conferences Steering Committee, 2017.
- [3] 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.
- [4] R. Liao, Y. Li, Y. Song, S. Wang, C. Nash, W. L. Hamilton, D. Duvenaud, R. Urtasun, and R. S. Zemel, “Efficient graph generation with graph recurrent attention networks,” *arXiv preprint arXiv:1910.00760*, 2019.
- [5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.
- [6] W. Hamilton, Z. Ying, and J. Leskovec, “Inductive representation learning on large graphs,” in *Advances in Neural Information Processing Systems*, pp. 1024–1034, 2017.
- [7] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, “Graph attention networks,” *arXiv preprint arXiv:1710.10903*, 2017.
- [8] J. You, B. Liu, Z. Ying, V. Pande, and J. Leskovec, “Graph convolutional policy network for goal-directed molecular graph generation,” in *Advances in Neural Information Processing Systems*, pp. 6410–6421, 2018.
- [9] Y. Bai, H. Ding, Y. Qiao, A. Marinovic, K. Gu, T. Chen, Y. Sun, and W. Wang, “Unsupervised inductive whole-graph embedding by preserving graph proximity,” *arXiv preprint arXiv:1904.01098*, 2019.
- [10] J. You, R. Ying, and J. Leskovec, “Position-aware graph neural networks,” *arXiv preprint arXiv:1906.04817*, 2019.
- [11] P. D. Dobson and A. J. Doig, “Distinguishing enzyme structures from non-enzymes without alignments,” *Journal of molecular biology*, vol. 330, no. 4, pp. 771–783, 2003.
- [12] K. Riesen and H. Bunke, “Iam graph database repository for graph based pattern recognition and machine learning,” in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, pp. 287–297, Springer, 2008.
- [13] P. Yanardag and S. Vishwanathan, “Deep graph kernels,” in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1365–1374, ACM, 2015.

- [14] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, “Hierarchical graph representation learning with differentiable pooling,” in *Advances in Neural Information Processing Systems*, pp. 4800–4810, 2018.