Group Anomaly Detection via Graph Autoencoders

Pierluca D'Oro[1,2], Ennio Nasca[2], Jonathan Masci[1], Matteo Matteucci[2]
[1]NNAISENSE, Switzerland [2]Politecnico di Milano, Italy
{pierluca, jonathan}@nnaisense.com
ennio.nasca@mail.polimi.com, matteo.matteucci@polimi.it

Abstract

Anomaly detection has usually the objective of finding single anomalous samples in the data. However, when faced with important real world problems such as fraud detection and anti-money laundering, we are often more interested, given a dataset of groups (e.g., batches of financial transactions), in assessing whether a group in its entirety constitutes an outlier (e.g., a collective money laundering activity). This paper introduces Group Anomaly Detection via Graph Autoencoders (GADGA), harnessing recent progress in graph representation learning to detect anomalous groups of points by exploiting their graph structure, rather than their raw set representation. We experimentally analyze some properties of graph autoencoders, informing the design decisions behind our method, and then empirically evaluate it, showing superior performance than set-based and graph-based baselines.

1 Introduction

Anomaly detection is an extremely important task in machine learning. It consists of discriminating inliers from outliers in a given dataset, and is paramount in countless real-world problems such as intrusion detection, fault detection, and medical imaging [8], where annotated data is hard or impossible to obtain. Recent progress in deep learning [33] paved the way for several breakthroughs on this task [22, 30, 32]. However, most of the attention of the research community has been directed at solving *point anomaly detection*, in which a single sample at a time (e.g., an image, a transaction) is scrutinized in the search of anomalous behaviors. While this can be desirable for many applications, there exist many others where an anomaly can be defined only by a *group* of data points. For instance, imagine the case of money laundering detection, in which data points could correspond to different transactions: a single transaction per-se is rarely an anomaly, or it is very difficult to be evaluated as such; instead, one is usually more interested in detecting an anomalous group of transactions as a whole. Another example is in high energy physics, where new particles (e.g., the Higgs boson [3]) can be found by detecting anomalous groups of collisions.

Generalizing point anomaly detection methods to the group anomaly detection task is not straightforward and can require either very unpractical assumptions about the data generating process or huge amounts of prior knowledge, in order to take advantage of the inherent structure of a group. Indeed, a central question in the design of a group anomaly detection method concerns the best way to leverage data structure. Very often, the relationship between members of a group can be formalized as a graph, for instance by assessing the similarity between its elements or by using some information about other types of interactions among the samples. Intuitively, the graph representation is richer than a standard set-based representation for the group. However, there is a lack of general methods for detecting group anomalies by taking advantage of such structure in the data, and we argue that geometric deep learning [5] can stand as a powerful tool in this sense. Indeed, recent work showed remarkable potential on domains related to group anomaly detection; for instance, graph neural networks were used for performing anti-money laundering [37, 38] and fake news detection [23]. These attempts framed the problem as a classification task, and therefore they rely on the unrealistic assumption of easy availability of labels.

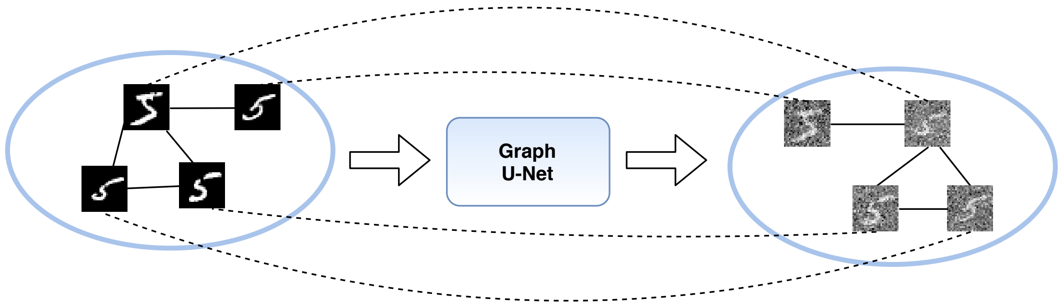

Figure 1: Schematic representation of our method. Given a group, a graph is built and then given as an input to a Graph U-Net autoencoder, that exploits this structured representation to reconstruct each one of the group samples (depicted as noisy digits in the figure). At training time, the loss function is computed by summing the reconstruction error on individual instances of the group, highlighted in the figure by using dashed lines. At test time, the loss is used as an anomaly score for the group.

In this paper, we propose Group Anomaly Detection via Graph Autoencoders (GADGA), a method for group anomaly detection based on graph representation learning. The approach uses the reconstruction error of a graph autoencoder, trained on unlabeled data, to assign anomaly scores to groups represented as graphs. In designing GADGA, we leverage the insights on the empirical limitations of structure-only graph autoencoders, obtained by running experiments on toy data, as well as careful adaptation of the best performing graph neural network architectures.

2 Related work

Group Anomaly Detection. Sometimes referred to with the term *collective* [8], anomalies that are defined starting from entire sets of samples have seldom been object of analysis in modern machine learning. Classical approaches [35, 9, 34, 24], usually custom-made for specific types of data (e.g., sequences) are unfortunately not easy to run at scale. Group anomaly detection methods can rely to some extent on anomaly detection techniques created for single samples [21, 4, 20, 13]. For instance, methods based on the reconstruction error of different types of autoencoders [13, 22, 14] have been employed for group anomaly detection using environmental [1] and visual [7] data.

Graph Autoencoders. Building on the idea of learning an identity function, commonly employed in deep learning [31, 2, 22, 13], recent work adapted autoencoders to graph-structured data. A first family of approaches focuses on the reconstruction of the adjacency matrix [16], with applications such as link prediction [16] and graph embedding [26]. Starting from the observation that the sole reconstruction of the structure of the network is often not sufficient, other works integrated node features into the loss function for tasks such as shape completion [19] and node embedding [18].

3 Group Anomaly Detection via Graph Autoencoders

Group Anomaly Detection. Let $\mathcal{X} = \{x_1, x_2, ...x_N\}$ be a dataset of *instances* (e.g., single RGB images), $2^{\mathcal{X}}$ its power set and $\mathbf{G} = \{g_1, g_2, , , , , g_M\} \subseteq 2^{\mathcal{X}}$ a set of *groups*. We assume a group g can be represented as a graph, by means of a graph generation function $h : \mathbf{G} \rightarrow \mathcal{G}$. For instance, h can be the procedure that generates the similarity graph of the instances contained in a group g. After being transformed by h, a group g is characterized by a node feature matrix $\mathbf{X}_g$, containing the vectors describing the members of the group individually, and an adjacency matrix $\mathbf{A}_g$, summarizing the connections among the group elements.

When performing group anomaly detection, we are interested in learning a detector either in the form $f_{\mathbf{G}} : \mathbf{G} \rightarrow \mathbb{R}$ or, given the graph representation provided by h, in the form $f_{\mathcal{G}} : \mathcal{G} \rightarrow \mathbb{R}$. The output, defined as *anomaly score*, is an increasing function of the probability that the corresponding input is an anomaly according to the model. For $f_{\mathbf{G}}$ to be a valid set function, some simple conditions about permutation-invariant representations must hold [39]. Despite a function satisfying these conditions is not difficult to learn in general, when information about the structure of the group is available in

Class		Reconstruction error		AUC
Normal	Anomaly	Normal	Anomaly	
Cycle	Star	1.40 ± 0.00	34.54 ± 0.09	1.00
Star	Cycle	1.08 ± 0.15	2.03 ± 0.00	1.00
Erdos	Watts	0.38 ± 0.04	0.54 ± 0.31	0.54
Watts	Erdos	0.37 ± 0.02	0.44 ± 0.10	0.50

Table 1: Results in terms of reconstruction error and AUC, using structure-only autoencoders [16].

the form of a graph, using more specialized families of functions can lead to a boost in performance. Thus, we investigate the use of models specifically designed for graph-structured data.

Graph Autoencoders. The idea of using the reconstruction error of encoder-decoder neural networks trained to learn the identity function as anomaly score dates back to 1995 [13], but had no chance of being generalized to non-euclidean data until the last years, when graph autoencoder architectures have been proposed for different applications. A first family of graph autoencoder works by reconstructing the adjacency matrix $\mathbf{A}_g$. We consider a simple method, proposed in [16], that consists in using one or more Graph Convolutional Network (GCN) layers [15] in the encoder to obtain an embedding $\mathbf{Z}_g$, then directly used for reconstructing $\mathbf{A}_g$. The model is trained by minimizing the negative log-likelihood (NLL), that can then be used as anomaly score $f_{\mathrm{GAE}}(g) = \mathrm{NLL}(\mathbf{A}_g, \widehat{\mathbf{A}}_g)$, where $\widehat{\mathbf{A}}_g = \sigma\left(\mathbf{Z}_g \mathbf{Z}_g^T\right)$ (where σ is the sigmoid function) and $\mathbf{Z}_g = \mathrm{GCN}(\mathbf{X}_g, \mathbf{A}_g)$. However, this reconstruction error only considers the structural information contained in a group, without making advantage of the node information. Hence, in building our method for group anomaly detection, we make use of the recently proposed Graph U-Nets [12], which bring the ideas behind the U-Net architecture [28] to the graph setting, interleaving a modified version of GCN [15] with *graph pooling* and *graph unpooling* operations. This model was originally proposed for supervised learning and evaluated on node classification, but we adapt it to be the backbone of our unsupervised autoencoder-based anomaly detection method. We train the model by using the Frobenius norm of the difference between the original and the reconstructed node features, using the same loss as anomaly score for a group $f_{\mathrm{UNET}}(g) = \|\mathbf{X}_g - \widehat{\mathbf{X}}_g\|_{\mathrm{F}}$.

4 Experiments

4.1 A problem with structure-based autoencoders

As a preliminary study for the use of graph autoencoders in anomaly detection, we carry out an analysis whose objective is to evaluate whether the reconstruction error in structure-only autoencoders (i.e., trained only for the reconstruction of the adjacency matrix) is an appropriate anomaly score. We test the method on two types of data. For the first one, we generate graphs by sampling them from two fixed-topology schemes, *star* (in which every node apart a center one has a single connection with the center itself) and *cycle* (in which every node is connected to two adjacent nodes). For the second one, we generate random graphs using standard methods, namely the Erdős–Rényi [10] and the Watts-Strogatz [36] models, that have been shown to share properties with real world data such as social and biological networks.

The results of the experiment for different normal/anomalous combinations are reported in Table 1. On the one hand, structure-only graph autoencoders seem to be able to perfectly learn a representation for very simple graphs: when dealing with cycle and star graphs, they are able to reach perfect performance in terms of AUROC, when, in turn, both graphs are considered as anomalies. On the other hand, when this family of methods is faced with the more challenging, and more realistic, anomaly detection task using random graphs, performance degrades almost to the one of random guessing. This gives hints for a possible fundamental problem with this kind of autoencoder: we hypothesize that both the simple structure of the decoder and the loss function are not sufficient for learning a representation that is appropriate for anomaly detection.

GADGA			*Local Outlier Factor*		
Ours	LDP ablation	Structure-only	Graph2Vec	Local Degree Profile	Feature Averaging
0.81	0.58	0.23	0.49	0.52	0.70

Table 2: Results for group anomaly detection on the MNIST-based dataset, in terms of AUROC. GADGA is evaluated using the U-Net model (*Ours*), using local degree profile node features (*LDP ablation*) and a loss based on the adjacency matrix (*Structure-only*). LOF is tested using different representations.

4.2 Group Anomaly Detection with Image Data

Motivated by our experiment on toy data, in GADGA, our method for group anomaly detection, we use the more complex U-Net architecture and, most importantly, a loss function that considers the node features. We now illustrate our experiments on image data.

Dataset. Due to the lack of standard benchmarks, we build a dataset for group anomaly detection starting from MNIST [17]. We use the encoder of an MSE-trained MLP autoencoder in order to extract a 128-dimensional representation for the samples. Afterwards, we create non-anomalous groups by sampling a non-fixed number of samples from homogeneous classes. For instance, a non-anomalous group can be composed by the representations of random images belonging to the class 5, while another one can contain only images belonging to the class 6, and so on. Anomalous groups are created by randomly sampling images from at least two of the available classes.

Baselines. The idea behind all the baselines is to obtain a vector representation for the group, therefore reducing the task of group anomaly detection to point anomaly detection. The first approach, directly based on [39], uses as vector embedding for a whole group a permutation-invariant combination of the representations of the individual members: we encode the group g by using the average $\frac{1}{M}\sum_{x \in g} x$, where x is the representation for an image as provided by the pretrained feature extractor. We additionally investigate two methods for obtaining a vector representation from the graph view of a group. First, Graph2Vec [25], which uses an unsupervised procedure for transforming graphs of arbitrary size into fixed-length vectors; then, an 8-bin histogram of the *local degree profile* [6] of nodes, recently proposed as strong node classification baseline. As a point anomaly detection method, we use Local Outlier Factor (LOF), selected as best performing algorithm among a set of widely-used ones such as Isolation Forest [20], OC-SVM [21] and Robust Covariance [29].

Results and discussion. Table 2 shows the group anomaly detection results in terms of AUROC on the test set of the aforementioned dataset based on MNIST. The results clearly display GADGA as superior. We also evaluate two ablations of our method, using local degree profile as only node feature and structure-only autoencoders, showing two interesting points: first, that using appropriate node features is an essential requirement when using graph representations for group anomaly detection; second, that the intuition provided by the experiments on toy data using structure-only graph autoencoders is verified on more difficult data. Our method also performs better than the baselines based on the use of LOF on the different group representations. A simple set representation [39] constitutes a strong baseline for group anomaly detection: the sole use of the graph representation of the data is not enough to improve it. Instead, a careful use of graph representation learning, such as the one we propose in GADGA, is required for taking advantage of this additional structural information and obtain excellent performance.

5 Conclusion

In this paper, we presented GADGA, a group anomaly detection method based on deep graph autoencoders. Future work could focus on gathering and organizing real world data to serve as standard benchmark for group anomaly detection, for instance in anti-money laundering [37]. Moreover, we showed that deep learning on graphs has the capability of leveraging the graph structure of a group, obtaining excellent group anomaly detection performance: therefore, it would be interesting to see the development of other methods, based on graph representation learning, for solving this task.

References

[1] Daniel B Araya, Katarina Grolinger, Hany F ElYamany, Miriam AM Capretz, and G Bitsuamlak. Collective contextual anomaly detection framework for smart buildings. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 511–518. IEEE, 2016.

[2] Dana H. Ballard. Modular learning in neural networks. In *AAAI*, 1987.

[3] Pushpalatha C Bhat. Multivariate analysis methods in particle physics. *Annual Review of Nuclear and Particle Science*, 61:281–309, 2011.

[4] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *ACM sigmod record*, volume 29, pages 93–104. ACM, 2000.

[5] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[6] Chen Cai and Yusu Wang. A simple yet effective baseline for non-attribute graph classification. *arXiv preprint arXiv:1811.03508*, 2018.

[7] Raghavendra Chalapathy, Edward Toth, and Sanjay Chawla. Group anomaly detection using deep generative models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 173–189. Springer, 2018.

[8] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[9] Sanjay Chawla and Pei Sun. Slom: a new measure for local spatial outliers. *Knowledge and Information Systems*, 9(4):412–429, 2006.

[10] Paul Erd6s. On the evolution of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci*, 5:17–61, 1960.

[11] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[12] Hongyang Gao and Shuiwang Ji. Graph u-nets. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2083–2092, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[13] Nathalie Japkowicz, Catherine Myers, and Mark A. Gluck. A novelty detection approach to classification. In *IJCAI*, 1995.

[14] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[15] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[16] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *NIPS Workshop on Bayesian Deep Learning*, 2016.

[17] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[18] Sébastien Lerique, Jacob Levy Abitbol, and Márton Karsai. Joint embedding of structure and features via graph convolutional networks. *arXiv preprint arXiv:1905.08636*, 2019.

[19] Or Litany, Alex Bronstein, Michael Bronstein, and Ameesh Makadia. Deformable shape completion with graph convolutional autoencoders. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1886–1895, 2018.

[20] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE, 2008.

[21] Larry M Manevitz and Malik Yousef. One-class svms for document classification. *Journal of machine Learning research*, 2(Dec):139–154, 2001.

[22] Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. In *International Conference on Artificial Neural Networks*, pages 52–59. Springer, 2011.

[23] Federico Monti, Fabrizio Frasca, Davide Eynard, Damon Mannion, and Michael M Bronstein. Fake news detection on social media using geometric deep learning. *arXiv preprint arXiv:1902.06673*, 2019.

[24] Krikamol Muandet and Bernhard Schölkopf. One-class support measure machines for group anomaly detection. *arXiv preprint arXiv:1303.0309*, 2013.

[25] Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. graph2vec: Learning distributed representations of graphs. *arXiv preprint arXiv:1707.05005*, 2017.

[26] Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang. Adversarially regularized graph autoencoder for graph embedding. *arXiv preprint arXiv:1802.04407*, 2018.

[27] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[28] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[29] Peter J Rousseeuw and Katrien Van Driessen. A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41(3):212–223, 1999.

[30] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *International Conference on Machine Learning*, pages 4393–4402, 2018.

[31] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[32] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International Conference on Information Processing in Medical Imaging*, pages 146–157. Springer, 2017.

[33] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[34] Shashi Shekhar, Chang-Tien Lu, and Pusheng Zhang. Detecting graph-based spatial outliers: algorithms and applications (a summary of results). In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 371–376. ACM, 2001.

[35] Christina Warrender, Stephanie Forrest, and Barak Pearlmutter. Detecting intrusions using system calls: Alternative data models. In *Proceedings of the 1999 IEEE symposium on security and privacy (Cat. No. 99CB36344)*, pages 133–145. IEEE, 1999.

[36] Duncan J Watts and Steven H Strogatz. Collective dynamics of ‘small-world’networks. *nature*, 393(6684):440, 1998.

[37] Mark Weber, Jie Chen, Toyotaro Suzumura, Aldo Pareja, Tengfei Ma, Hiroki Kanezashi, Tim Kaler, Charles E Leiserson, and Tao B Schardl. Scalable graph learning for anti-money laundering: A first look. *arXiv preprint arXiv:1812.00076*, 2018.

[38] Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel Karl I Weidele, Claudio Bellei, Tom Robinson, and Charles E Leiserson. Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics. *arXiv preprint arXiv:1908.02591*, 2019.

[39] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan R Salakhutdinov, and Alexander J Smola. Deep sets. In *Advances in neural information processing systems*, pages 3391–3401, 2017.

A Experiments setups

For implementing the graph neural networks in both experiments, we use Pytorch Geometric [11, 27].

Toy Data Experiment. For both types of graphs, we vary the number of nodes, randomly sampling it in the interval [10,50]. For the autoencoder, we follow the architecture suggested in [16], using a single-layer graph encoder with 32 channels, followed by a decoder entirely based on dot product. We train the autoencoder for 10 epochs on a training set of 700 graph, composed of the 90% of non-anomalous data. We optimize for the reconstruction of the adjacency matrix, quantified by the negative log likelihood on single elements of the matrix. After training, we compute the area under the ROC curve (AUC) on a test set of 300 graphs that is following the same distribution as the training set, using the same loss function used for training as anomaly score. As an optimizer, we employ Adam with default (β_1, β_2) and a learning rate of 0.001.

MNIST-based Experiment. For the dataset, the number of instances in a group is randomly selected from a minimum of 10 to a maximum of 50. We use a training set of 700 groups and a test set of 300 groups, with a fixed proportion of 0.9 for non-anomalies and anomalies. We then implement the graph-generating function h by creating a similarity graph of the images in a group, using a fixed number of 10 nearest neighbours and using a threshold of 0.5 for neglecting the arcs corresponding to distant image representations. We use the U-Net autoencoder as described in the main paper, using 3 layers with 64 channels to reconstruct the original node feature matrix. We train the autoencoder for 100 epochs, and then use the same Frobenius norm used as a loss as an anomaly score for the test set, using Adam with default (β_1, β_2) and a learning rate of 0.001.