Knowledge Extraction from Auto-Encoders on Anomaly Detection Tasks Using Co-activation Graphs

Daniyal Selani
Vrije Universiteit Amsterdam
Amsterdam, Netherlands
d.selani@student.vu.nl

Ilaria Tiddi
Vrije Universiteit Amsterdam
Amsterdam, Netherlands
i.tiddi@vu.nl

ABSTRACT

Deep neural networks have exploded in popularity and different types of networks are used to solve a multitude of complex tasks. One such task is anomaly detection, that a type of deep neural network called auto-encoder has become extremely proficient at solving. The low level neural activity, produced by such a network, generates extremely rich representations of the data, which can be used to extract task specific knowledge. In this paper, we built upon previous work and used co-activation graph analysis to extract knowledge from auto-encoders, that were trained for the specific task of anomaly detection. First, we outlined a method for extracting co-activation graphs from auto-encoders. Then, we performed graph analysis to discover that task specific knowledge from the auto-encoder was being encoded into the co-activation graph, and that the extracted knowledge could be used to reveal the role of individual neurons in the network.

CCS CONCEPTS

• **Networks** → **Network dynamics**; • **Computing methodologies** → *Semantic networks*.

KEYWORDS

Auto-encoders, Anomaly Detection, Co-activation Graphs

ACM Reference Format:
Daniyal Selani and Ilaria Tiddi. 2021. Knowledge Extraction from Auto-Encoders on Anomaly Detection Tasks Using Co-activation Graphs. In *Proceedings of the 11th Knowledge Capture Conference (K-CAP '21), December 2–3, 2021, Virtual Event, USA*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3460210.3493571

1 INTRODUCTION

Deep Neural Networks (DNNs) are extremely powerful tools that can leverage large amounts of data to perform complex tasks. However, despite their success, they have seen slow adoption in key industries such as healthcare, because they typically operate as black boxes. In other words, they are not easily interpretable [1].

Interpretability is crucial in circumstances where the ability to explain the reasoning behind an action is as important as the action itself. For example, when doing differential diagnosis on a patient, it is not enough to prescribe a treatment for the patient, but also to identify the underlying cause of the patient's illness, and the reasoning behind the prescribed medicine. This can be expanded to other key fields such as criminal investigations and drafting new laws. Additionally, opening up the DNN black box can allow us to more efficiently optimize and debug models [2], by finding insights on the way the DNN deals with the input data.

One such complex task is anomaly detection. Identifying data samples that do not fit the overall data distribution is the principal task in anomaly detection. Anomalies can arise due to various reasons such as noise in the data capture process, changes in underlying phenomenon, or due to new or previously unseen conditions in the captured environment. Therefore, anomaly detection (AD) is a crucial task in medical signal analysis [3].

A popular way of tackling anomaly detection is to use auto-encoders (AEs). Auto-encoders are a DNN architecture that aim to learn a low-dimensional feature representation space on which the given data instances can be well reconstructed. The difference between the reconstruction produced by the AE and the input data is referred to as the reconstruction error, which is used as the heuristic for the task of anomaly detection [4]. Despite their success, these models are hard to interpret, therefore hindering their adoption in life-critical contexts such as healthcare.

This paper builds on previous work done on extracting knowledge from DNNs using co-activation graphs [5]. The method took inspiration from studies in neuro-science that represent neural activity as a network, and perform graph analysis to interpret brain activity. We expanded on this idea and use the same principles of neuron co-activation and graph analysis to interpret the inner workings of an auto-encoder performing anomaly detection. Our assumption is that co-activation graphs can be used as a model agnostic method for knowledge extraction from AEs, and that it can be particularly useful for the task of anomaly detection. By knowledge extraction, we mean eliciting knowledge about the model's inner working, towards a more interpretable neural architecture. Differently from [5], we aim to provide a general mechanism for extracting knowledge from generative neural network architectures, with a specific focus on auto-encoders, and further investigate the applications of that knowledge. We first fit an AE onto a time series dataset of ECG readings, and then use a reconstruction error threshold to detect anomalies. We then do preliminary investigation on how to apply the concept of co-activation graphs onto AEs. We further study not only how to extract knowledge from these graphs using graph analysis techniques, such as centrality analysis and community detection, but also how interpretable and re-usable such elicited knowledge is.

K-CAP '21, December 2–3, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-8457-5/21/12…$15.00
https://doi.org/10.1145/3460210.3493571

2 RELATED WORK

In this section, we look at related work done on auto-encoders and the task of anomaly detection, as well as work done on ways to interpret such networks for that specific task.

2.1 Auto-Encoders and Anomaly Detection

Auto-encoders (AEs) have been a popular concept for decades. It was in the 1980s that the first applications were made [6]. Their most common usage was for dimensionality reduction or feature learning. However, the concept has become widely used for learning generative models of data.

AEs work by learning to represent the input data in low dimensional space, and then learning to reconstruct the low dimension representation of the data, to resemble the input data. AEs have 2 parts, the encoder and the decoder. A basic auto-encoder is a feed-forward network, of which the input and output layers have the same dimensions. These layers are connected by one of more hidden layers of reduced dimensions.

One of the tasks that AEs gained popularity doing is anomaly detection (AD). AD is a critical technique for detecting fraud, suspicious activity, network intrusion, and other unusual events that are significant yet difficult to detect. It has wide ranging applications, such as detecting manufacturing defects and network intrusions [7].

AEs are simple to implement and have simple intuitions for detecting anomalies. The intuition being that the model learns to reconstruct the normal data, and will encounter large reconstruction errors when trying to do the same with anomalous data. As a result, they have received a great deal of attention in literature. For example, RandNet [8] improves on basic AEs by using an ensemble of AEs. The idea of using AEs on time-series data is further explored by [9], in which they use an ensemble of recurrent auto-encoders to further decompose the data.

AEs do carry some advantages for this specific task, chiefly that they are generic to different types of data, and that different types of powerful AE variants can be leveraged to perform anomaly detection for specific domains.

However, the learned feature representations can be biased by infrequent irregularities and the presence of outliers or anomalies in the training data. AEs are generally vulnerable to data noise presented in the training data as they can be trained to remember the noise, leading to severe overfitting and small reconstruction errors of anomalies. Additionally, the objective function of the data reconstruction performed by AEs is designed for dimension reduction or data compression, rather than anomaly detection. The resulting representations are a generic summarization of underlying regularities, which are not optimized for detecting irregularities [10].

2.2 Interpretability of DNNs and Co-activation Graphs

The popularity of DNNs also ushered in a need of making them interpretable. Multiple strategies have been explored to fulfill this task. Decompositional strategies outline methods to extract rules by focusing on weights and activations in the DNN [11]. Pedagogical strategies are model agnostic, they treat DNNs as a black box, but use decisions made by the DNN to infer its inner workings [12].

Our approach for AE interpretability is based on the concept of co-activation graphs, which was explored in detail on DNNs in [13]. This approach uses the activation of neurons to create a fully connected graph, on which graph analysis methods can be applied to. The idea is to explore interesting relationships between strongly co-activating neurons. Each neuron, regardless of its location in the AE, is connected to every other neuron. The weight of the connection between each neuron is dictated by the correlation of their activations.

There have also been attempts to work on interpretability as it relates to auto-encoders and anomaly detection specifically. [14] proposed an approach for explaining Variational Auto-Encoders for anomaly detection using model gradients. The paper proposes to apply its method on clustering and fingerprinting tasks. By clustering similar gradient results from detected anomalies into distinct bins, domain experts can further investigate the similarities between anomalies and extract information from them. Additionally, the clustering efforts can be used to create "fingerprints" to identify different types of anomalies. [15] proposed to improve a model's interpretability by using SHAP values to calculate how much each input feature, for any given input X_i, contributed to the input being categorized as an anomaly. This method is also intended to be used in collaboration with domain experts for further knowledge extraction.

Centrality analysis and community detection will be key to our efforts, as they provide insights on the structure of the graph, and on the importance of each individual node [5]. We aim at observing how the knowledge acquired from the graph analysis relates directly to the auto-encoder and to the anomaly detection task. Our key innovation is to extend the analysis of the network to include the input and output layers of the AE. This is different than [5, 13], where the input neurons were omitted when extracting the co-activation graph. Another key difference between our work and the previous work is the type of neural network architecture that is being interpreted. In previous work, the efforts were concentrated on a traditional image classification models, our work focuses on a generative model architecture.

3 APPROACH

The section here presents the two steps of our methodology, i.e. building the auto-encoder and its co-activation graph. All code to reproduce our method is available online[1].

3.1 Auto-Encoder

The first step of our approach is to build an auto-encoder. As the AE will be used specifically for the task of anomaly detection, it will be trained to reconstruct only the normal data. Once the AE is trained, the reconstruction error threshold is then calculated based on the distribution of the reconstruction errors (Fig 1). This threshold is then used to classify an input as either normal or anomalous. The AE's output will have the same dimensionality as its input. However, we need the output in the form of classification in order to perform analysis on the extracted co-activation graph. Hence, we add a lambda layer to our AE which is fed by the input and output layer (Fig 2). This lambda layer is non differentiable, i.e. it

[1] https://github.com/daniyal9538/coactivation_AE_AD

cannot be trained using back propagation, and is only added after the AE is trained. This lambda layer performs the function of using the reconstruction error threshold t to output a classification, by comparing the output of the model x to the input $\hat{x}$. Equation 1 shows the activation function of the neuron in the lambda layer. The output of the neuron in the lambda layer are used much in the same way as the outputs of the neural-network in [5].

$$f(x) = \begin{cases} 1 & MSE(\hat{x}, x) < t \\ 0 & MSE(\hat{x}, x) \geq t \end{cases} \quad (1)$$

The neuron in the lambda layer only activates when the reconstruction error is below the threshold t, hence signifying a prediction that the input belongs to the normal class.

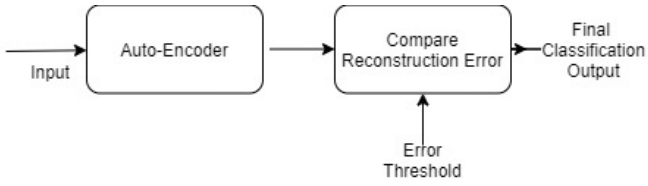

Figure 1: Model architecture for Anomaly Detection.

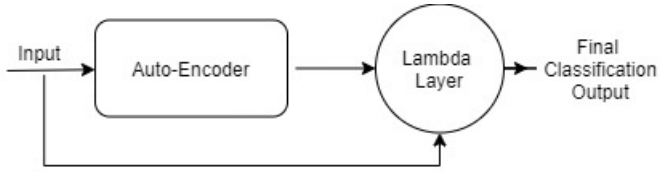

Figure 2: Model architecture for AD with Lambda Layer.

3.2 Co-activation Graph

Once the AE has been trained, we have to extract co-activation values from it in order to build a co-activation graph. The co-activation values have to be extracted over a set of inputs S. A co-activation value (W_{ij}) between 2 neurons (i, j) is defined as the correlation of the activation values (A) of the 2 neurons for a set of inputs S.

$$W_{ij} = Corr(A(i, S), A(j, S)) \quad (2)$$

A co-activation graph is a fully connected undirected network, in which each node corresponds to a neuron in the neural network. The weights in the network are the co-activation values between each connected node. For a more rigorous definition of co-activation values and co-activation graphs, as well as empirical proof that a co-activation graph encodes the same knowledge as the neural-network it was extracted from, we refer the reader to [5]. The procedure to build a co-activation graph from the ground up is as follows:

(1) Train AE using train set.
(2) Use validation set to calculate threshold error for AD task.
(3) Add lambda layer to AE using calculated threshold.
(4) For each datapoint in experimental set, extract activation values for all neurons in AE.
(5) For each set of neuron activation values, find Spearman correlation to every other set of neuron activation values.
(6) Generate a fully connected undirected graph, in which each node represents a unique neuron in the AE.
(7) The correlation value between any given pair of neurons, becomes the weight of the edge that connects the 2 nodes associated with those neurons in the co-activation graph.
(8) Filter in edges greater than 0.3 and less than -0.3, as to preserve only significant connections in the co-activation graph.

After extracting the co-activation values from the AE and building a co-activation graph, we can then perform analysis on the graph. The aim is to investigate if the co-activation graph encodes the same knowledge present in the AE, and if graph analysis techniques can be used to better interpret auto-encoders in an anomaly detection task.

4 EXPERIMENTAL SETUP AND RESULTS

4.1 Dataset

We chose a dataset of ECG readings [16, 17] to perform the AD task on. It is a collection of ECG readings from 5000 randomly selected heart beats from a patient with severe congestive heart failure, classified into 5 distinct classes. 4 of those classes are collapsed into 1 super class signifying an instance of an unhealthy (anomalous) heartbeat. While the remaining class indicates an instance of a healthy (normal) heartbeat.

An simple feed-forward AE, comprised of 9 layers, was trained on the ECG5000 dataset. Each datapoint in the dataset represents a single heartbeat, which is a sequence of readings in the shape of 1 dimensional vector of size 140. The data was interpolated so that each heartbeat in the dataset is of the same length. The dataset contains a 58-42 split of normal and anomalous data. The data was split into 4 sets: train, validation, test and anomalous. For all the experiments performed on the AE in this paper, the data from the test and anomalous set was randomly sampled to create a new experimental set, with a balanced distribution of both the normal and anomalous classes.

Table 1: Dataset split.

Dataset	Size (%)	Class	Accuracy (%)
Train	32.5	Normal	-
Validation	10	Normal	-
Test	7.5	Normal	92.4
Anomalous	50	Anomalous	99.5

4.2 Co-activation Value Distribution

The distribution of the co-activation values of the trained auto-encoder deviates quite significantly from a Gaussian distribution. This is to be expected, only a subset of neurons in the AE co-activate strongly with one another as the AE gets better at reconstruction during training. The information encoded in the AE in the form of the neuron activation values, is what gives the co-activation graph its structure, on which we perform graph analysis. This can be seen experimentally, as we compare the distributions of 3 AEs that have identical initial weights and biases:

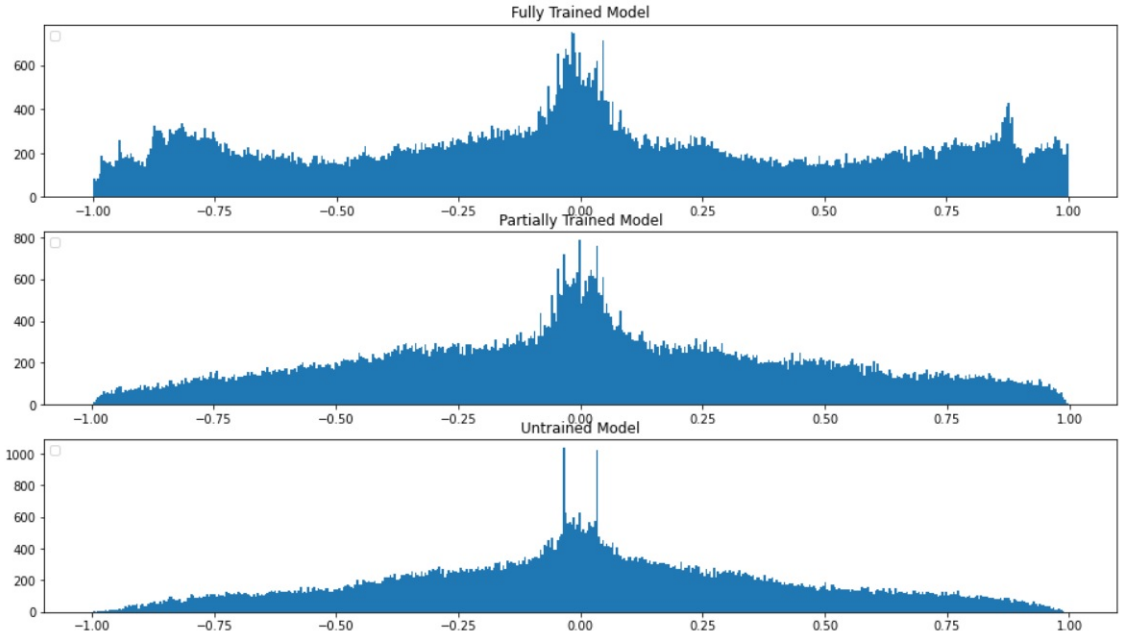

Figure 3: Distribution of co-activation values from different AE models (Fully trained, partially trained, untrained).

(1) Fully Trained AE (weighted accuracy score: 95%)
(2) Partially trained AE whose training was stopped prematurely (weighted accuracy score: 77%)
(3) Untrained AE (weighted accuracy score: 61%)

The distribution of the co-activation values (Fig 3) gets more skewed towards the tails and the peak, as accuracy improves.

4.3 Centrality Analysis

Node centrality in graphs is a useful measure of node importance. Typically in graph analysis, a node is said to be highly central if it has a large influence on the topology of the network. The concept of importance varies significantly depending on the domain of the graph and the type of centrality measure used. For the purposes of using co-activation graphs to analyse and interpret neural networks, node importance, and hence node centrality, should directly reflect the importance of the corresponding neurons in the neural network. Neurons of high importance should have a higher influence on the neural network, when compared to neurons of low importance. Using this assumption, we can design an experiment in which we systematically "turn off" neurons based on their calculated importance, and observe the loss in accuracy to determine its effect. We can "turn off" neurons by masking the activation values of the selected neurons.

[5] showed that degree centrality is a worse node importance measure when compared to Pagerank centrality. Degree centrality is the measure of importance by the number of connections a node has to other nodes. Pagerank centrality is a better measure of centrality for co-activation graphs, as it not only uses the weight of the edges connected to a specific node, but also the centrality of the neighbors of that node. In a complex networks like a co-activation graph, this provides a better assessment of importance, as not all nodes might have the same implicit importance, and connections to more important nodes have to weigh more heavily than connections to less important nodes.

We use Pagerank centrality to calculate neuron importance, and sequentially turn off the 100 most and least important hidden neurons. Fig 4 shows that removing higher Pagerank neurons (green line) has a much higher affect on model performance, when compared to removing lower Pagerank neurons (red line). This result further strengthens the claim that the co-activation graph of the AE does encode information that is present in the neural network itself. Centrality measures can also be used to prune large networks for purposes of optimization, by removing neurons of low importance, that do not significantly affect model performance.

4.4 Input Feature Importance

By using centrality analysis, we can now try to determine which input features are most important at describing normality. Unlike the method proposed in [15], which provides feature importance for each input fed to an AE, the method proposed here aims to provide a description for a generic input that the AE was trained to reconstruct. A human parallel to this problem would be trying to describe what makes a generic apple normal/healthy, as opposed to describing what makes a specific individual apple normal/healthy.

The experiment is designed as follows:

(1) Use Pagerank centrality to assign neuron importance for input neurons
(2) Extract the $n = 70$ most and least important neurons
(3) Sequentially mask input features corresponding to those input neurons
(4) Observe the affect of the modified inputs on the model performance

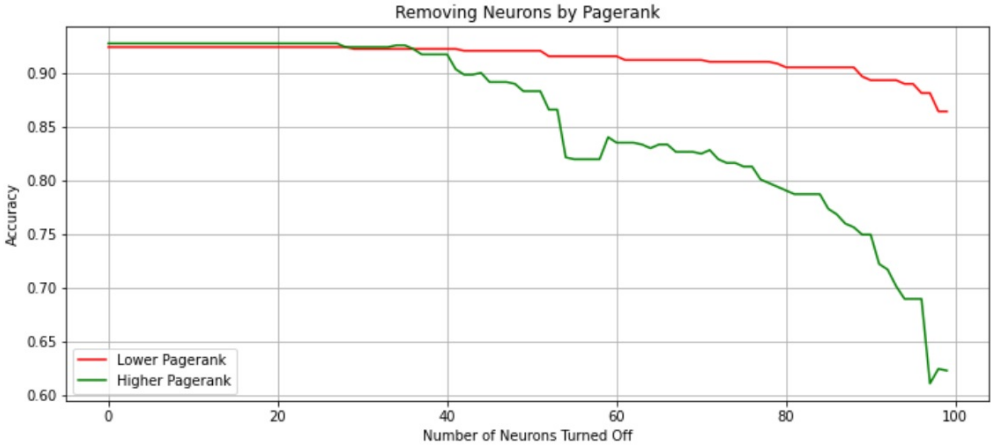

Figure 4: Model accuracy after neuron removal.

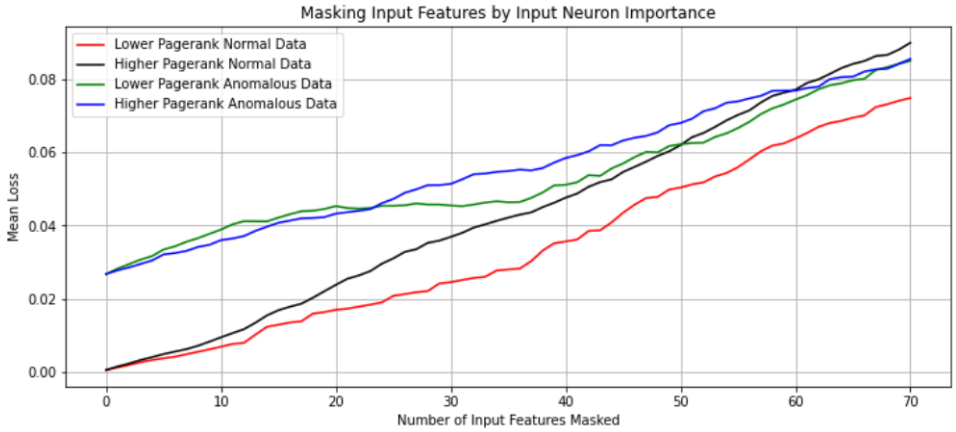

Figure 5: Model performance after input modification.

(5) Average out the difference in performance for each iteration of input modification

We run this experiment on 2 types of data: on normal test data, and on anomalous data. We use mean loss as the metric for model performance instead of accuracy, as we are interested in how the modification to the input affects each given instance in the dataset.

The results (Fig 5) show that, for normal data, masking the input features corresponding to higher importance input neurons (black line) does impact model performance more significantly than masking input features that correspond to lower importance input neurons (red line). The interpretation of these results is domain dependent. In the case of our chosen dataset, we can interpret these results as highlighting the partial sequences within the interpolated sequence of ECG readings that are the most important to detecting a healthy (normal) heartbeat for a specific patient. This information could also be used to compress input data fed to the AE for inference, by removing the least important features. However, there is no observable relationship between neuron centrality and feature importance when processing anomalous data. This is to be expected, as the AE is optimized to only reconstruct normal data. In our experiment, when removing least important input features, the reconstruction error of the normal data never gets larger than that of the corresponding anomalous data.

4.5 Community Analysis

To further analyse the co-activation graph, we investigate the presence of community structures, which is usually defined as a densely connected substructures present in the graph itself. [5] showed how the Louvian community detection algorithm [18] is a relevant and useful method for community detection for this use case. The algorithm outputs 3 distinct communities (Fig 6), with a modularity coefficient of 2.099. Higher modularity values indicate that connections between nodes in the same community, are stronger than the connection between nodes in different communities. We can assign a class to 2 of the communities by observing which communities have the highest and lowest mean co-activation value with the node corresponding to the neuron in the final lambda layer of the AE. The largest community by size, contained the lambda node,

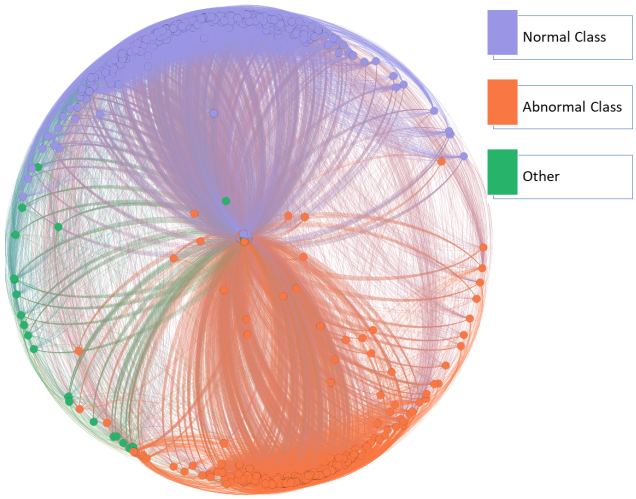

Figure 6: Visualization of the communities within the graph.

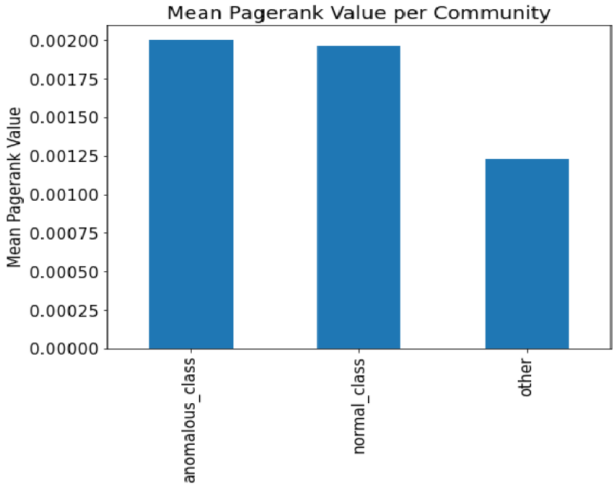

Figure 7: Mean Pagerank value per community.

and also had the highest mean co-activation value with the lambda node. We can attribute the normal class with this community, as the lambda neuron only activates when the classification is normal from the AE. Similarly, we can assign the second largest community to the anomalous class, and the other community contains 4% of the nodes in the graph.

Table 2: Communities in co-activation graph.

Community	Number of Nodes
Normal class	256
Anomalous class	236
Other	21

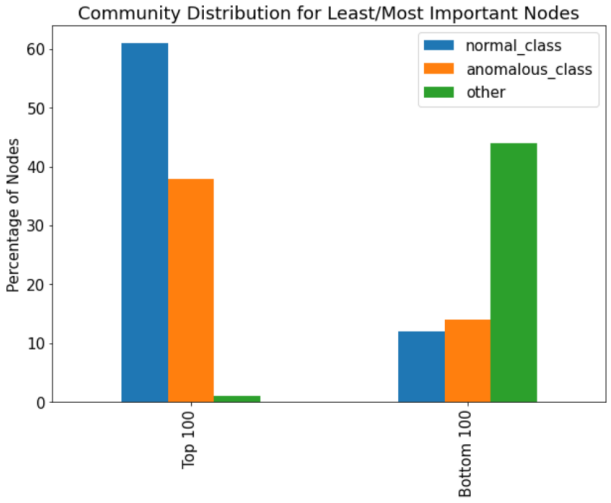

Figure 8: Community distribution for most and least important nodes.

The two largest communities should also be the most important in regards to the co-activation graph. We can asses this by calculating the mean Pagerank values of each community (Fig 7). Fig 8 shows that the top 100 nodes by Pagerank have a majority of the normal class community, even though, normal class and anomalous class communities both have approximately the same mean Pagerank values. This is to be expected as the normal class community co-activates very strongly with the neuron in the lambda layer (which only activates when the AE predicts an input to be normal). We see that the bottom 100 nodes by Pagerank, mostly belong from the other community, which is expected as well, as they have the lowest mean Pagerank values. These observations prove that community analysis can be used to assign nodes to a specific data class when dealing with AEs. This is particularly useful as it can be used to fingerprint different types of anomalies without having to re-train the auto-encoder.

5 FUTURE WORK AND CONCLUSION

In this paper, we outlined the methodology for extracting co-activation graphs from auto-encoders, and performed graph analysis to gain insight into the anomaly detection task. We stated our assumptions and described experiments to verify the hypothesis that analysis of co-activation graphs can be used to extract information from AEs and interpret their inner workings. Our results prove that not only is knowledge from the AEs encoded into its co-activation graph, but also that the co-activation graph can be used to extract that encoded knowledge. That knowledge can be used to optimize the model, or find further insights about the input, such as feature importance with respects to normality.

However, the method used does have some limitations. Because the activation values are aggregated over a set of inputs, graph analysis of the co-activation graph cannot be used for interpreting the model's output for any single given input. Hence, our method does not supersede or replace the interpretability approaches discussed in the related work section. Instead, we believe co-activation graphs can be used to get a deeper understanding of the overall task, rather than finding out specifics of any given input. Another limitation is that the centrality analysis and community detection methods used to perform graph analysis are biased by the hyper-parameters chosen for those methods.

The obvious next step is to explore the use of co-activation graphs on other types of generative models, such as GANs [19]. Additionally, we would like to test our methodology on sparse datasets, and explore the impact of sparsity of the input on the extracted co-activation graphs. As seen above, node centrality is a good measure for ranking input node importance, and using this information to compress input data for neural networks could be a useful real world application. Another possible experiment would be to see what kind of co-activation graph is produced when using only anomalous data. In our experiments, all the significant results pertained directly to normal data. It is worthwhile to explore if community detection can be used to do fingerprinting and identify if sub-classes in the anomalous classes exist, and if they exist, then how to identify them without further training the AE. We could also analyse co-activation graphs extracted from more complicated AEs. It would be interesting how the behavior of convolution or recurrent layers differs from the behavior of the simple feed-forward layers used in our experiments.

6 ACKNOWLEDGEMENTS

The authors would like to thank Vitor A.C. Horta from Insight SFI Research Centre for Data Analytics, Dublin City University, Dublin, Ireland, for his valuable help and discussions during the development of this work.

REFERENCES

[1] Feng-Lei Fan, Jinjun Xiong, Mengzhou Li, and Ge Wang. On interpretability of artificial neural networks: A survey. *IEEE Transactions on Radiation and Plasma Medical Sciences*, page 1–1, 2021.
[2] Roozbeh Yousefzadeh and Dianne P O'Leary. Auditing and debugging deep learning models via decision boundaries: Individual-level and group-level analysis.
[3] Tharindu Fernando, Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. Deep learning for medical anomaly detection – a survey. *ACM Computing Surveys*, 54(7), 2021.
[4] Zhaomin Chen, Chai Kiat Yeo, Bu Sung Lee, and Chiew Tong Lau. Autoencoder-based network anomaly detection. *2018 Wireless Telecommunications Symposium (WTS)*, 2018.
[5] Vitor A.c. Horta, Ilaria Tiddi, Suzanne Little, and Alessandra Mileo. Extracting knowledge from deep neural networks through graph analysis. *Future Generation Computer Systems*, 120:109–118, 2021.
[6] Diederik P. Kingma and Max Welling. An introduction to variational autoencoders. 2019.
[7] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection. *Encyclopedia of Machine Learning and Data Mining*, page 1–15, 2016.
[8] Jinghui Chen, Saket Sathe, Charu Aggarwal, and Deepak Turaga. Outlier detection with autoencoder ensembles. *Proceedings of the 2017 SIAM International Conference on Data Mining*, page 90–98, 2017.
[9] Tung Kieu, Bin Yang, Chenjuan Guo, and Christian S. Jensen. Outlier detection for time series with recurrent autoencoder ensembles. *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019.
[10] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. Deep learning for anomaly detection. *ACM Computing Surveys*, 54(2):1–38, 2021.
[11] Veronica Chan and Christine W. Chan. Development and application of an algorithm for extracting multiple linear regression equations from artificial neural networks for nonlinear regression problems. *2016 IEEE 15th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC)*, 2016.
[12] Enric Junque De Fortuny and David Martens. Active learning-based pedagogical rule extraction. *IEEE Transactions on Neural Networks and Learning Systems*, 26(11):2664–2677, 2015.
[13] Vitor A. C. Horta and Alessandra Mileo. Towards explaining deep neural networks through graph analysis. *Communications in Computer and Information Science Database and Expert Systems Applications*, page 155–165, 2019.
[14] Quoc Phong Nguyen, Kar Wai Lim, Dinil Mon Divakaran, Kian Hsiang Low, and Mun Choon Chan. Gee: A gradient-based explainable variational autoencoder for network anomaly detection. *2019 IEEE Conference on Communications and Network Security (CNS)*, 2019.
[15] Liat Antwarg, Ronnie Mindlin Miller, Bracha Shapira, and Lior Rokach. Explaining anomalies detected by autoencoders using shapley additive explanations. *Expert Systems with Applications*, 186:115736, 2021.
[16] Ecg5000 dataset.
[17] Yanping Chen, Yuan Hao, Thanawin Rakthanmanon, Jesin Zakaria, Bing Hu, and Eamonn Keogh. A general framework for never-ending learning from time series streams. *Data Mining and Knowledge Discovery*, 29(6):1622–1664, 2014.
[18] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10), 2008.
[19] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.