Vec2Struc: A Method Towards Explainable Structural-Based Node Embeddings

Srishti Tomar, Ryan Compton
Quantiply, 333 W San Carlos St. #1650, San Jose, California,
{srishti, compton}@quantiply.com

Abstract

Network embeddings are a popular new method for encapsulating the complexity of networks in a reduced feature space. However, embeddings obfuscates the complexity of networks and make explainability of such models difficult. We propose Vec2Struc, a new method aimed at providing more explainable representations of network models. This approach examines nodes similar to one another in the generated embedding space and highlights common structures among them. This method provides a means to explore embeddings distributionally and visually, leading towards more transparent and interpretable AI systems utilizing state-of-the-art structural-based node embeddings.

Introduction

Networks have long been used in analysing complex systems with entity-to-entity relationships. For example, biologists have used networks to represent Protein-to-Protein interactions whereas social scientist's usage has been in understanding of friendship or community relationships (Goyal and Ferrara 2018). A growing consensus on the lack of common metrics and modeling of networks has led to a more representative approach in building condensed vectorized representations of network properties.

Network (or Node) embeddings are an approach that provide scalable learning within networks. Typically, networks are very rich in information and the size of them explored by researchers has increased into the billions of nodes. Through embeddings, a reduced feature space can be created to encapsulate both network and node information. With this reduced space comes more scalable and generalizable methods without needing to reduce the amount of information being stored in the network (Goyal and Ferrara 2018).

There are many variants to embedding approaches (Goyal and Ferrara 2018; Ribeiro, Saverese, and Figueiredo 2017), but there remains to be a method for assessing what these embeddings represent as they are an ambiguation of the original network, thus leaving the question *how can researchers determine the real-world representation of the node associations in such an ambiguous space?* There is a need to map the embedding space to real-world representations if we are to better understand models that rely on network embeddings.

This short paper proposes *Vec2Struc*, an algorithmic approach for discovering common structures or topologies between nodes in an embedded space. While this is not a universal approach for network embeddings (more on that below), this provides a first step toward interpreting network embeddings, paving the way for such state-of-the-art techniques to be more usable within AI systems that are dependent on explainability.

Network/Node Embeddings

Network embeddings are formally defined from (Goyal and Ferrara 2018):

Given a network G = (V,E), a network embedding is a mapping f : $v_i \rightarrow y_i \in R^d; \forall i \in [n]$ *such that* $d \ll |V|$ *and the function f preserves some proximity measure defined on network G*

Embeddings have been oriented to two different vectorized representations. 1) a representation of a network as a whole (referred to as network or graph embedding) and 2) a representation of node properties within a network (referred to as node embeddings). This work focuses on the node level representation and hence forth in this paper when referring to network embeddings this is the prospective taken.

(Goyal and Ferrara 2018) categorize embedding methods into taxonomy of three categories: *Factorization, Random Walk, and Deep Learning*. But these methods miss the importance of the local structures surrounding nodes. Understanding local structure properties is vital for analyzing complex networks as they showcase higher level properties beyond the one-to-one interactions commonly examined.

Additional methods have been implemented in which the structural identity of a node's neighbor is considered (Ribeiro, Saverese, and Figueiredo 2017). Struc2Vec, proposed by (Ribeiro, Saverese, and Figueiredo 2017), constructs a structural context for nodes through a multi-layer network which encodes structural similarities within the embeddings. Previous network embedding methods tend to encapsulate information around node distances, however Struc2Vec is noted to be distance agnostic and can embed structural information between nodes very far apart in the network (Ribeiro, Saverese, and Figueiredo 2017). While other types of structural based embeddings exist (Goyal and Ferrara 2018), this work utilizes Struc2Vec embeddings.

Need for Explainable AI

The need for such complex methods to be explainable is argued by (Weld and Bansal 2019) that most computer-based produced behavior is actually alien to humans and can fail in many unexpected ways. People can't trust nor control complex systems that lack the ability to verify why they produced a specific output. The authors propose multiple criterion for which an AI system to reach such an ability:

1. It is clear what factors caused the system's action, allowing users to predict how changes to the situation would have led to alternative behaviors
2. Permits effective control of the AI by enabling interaction

To approach such a system, they propose the need to ensure that the underlying reasoning is interpretable (Weld and Bansal 2019).

Vec2Struc fulfills the first criterion proposed by (Weld and Bansal 2019) through visual representations of the network topologies. This allows for high-level heuristic evaluation of the embedded space, highlighting the factors for which nodes are associated in structural-based embeddings.

Method

The summary of Vec2Struc is as follows: with an embedding representation generated, the nodes similar to each other need to be gathered. As a pair-wise comparison is exhaustive and computationally heavy, a filter is needed to find which pairs are best to compare neighborhoods. Then with these pairs, find if any common structure exists between them and measure the frequency of such structures to associate them with each embedding. The procedure of Vec2Struc is shown in Fig. 1 and is conducted in four stages.

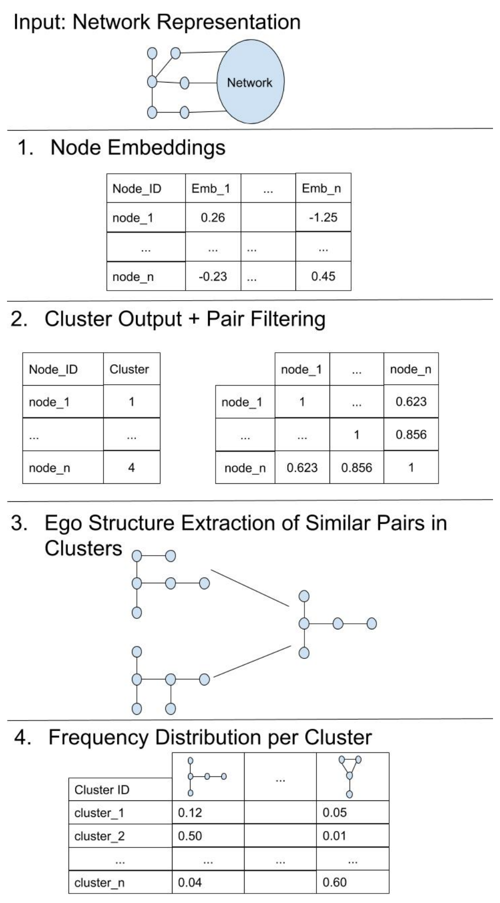
1. Node Embedding Generation

Embeddings are generated using an approach oriented toward representing the structural information of a node's neighborhood. While any structural based embedding should work theoretically, we only tested this with Struc2Vec (Ribeiro, Saverese, and Figueiredo 2017).

2. Cluster Modeling on Embedded Space + Pair-wise Filtering

Once embeddings are generated, a similarity heuristic is used to find those pairs worth comparing. This can be done through clustering nodes using the embeddings. A Gaussian Mixture model was used in our experiment with the number of clusters chosen through minimizing the Bayesian Information Criterion of the number of clusters. Thus nodes have been binned into groups for which pair-wise comparison can be made more efficiently.

However, even with the clusters there is still a potential for the pair-wise search space to be too large. We conducted further filtering through only examining pairs of nodes within


Input: Network Representation
Network
1. Node Embeddings
2. Cluster Output + Pair Filtering
3. Ego Structure Extraction of Similar Pairs in Clusters
4. Frequency Distribution per Cluster

Node_ID	Emb_1	...	Emb_n
node_1	0.26		-1.25
...	...	...	...
node_n	-0.23	...	0.45

Node_ID	Cluster
node_1	1
...	...
node_n	4

	node_1	...	node_n
node_1	1	...	0.623
...	...	1	0.856
node_n	0.623	0.856	1

Cluster ID		...	
cluster_1	0.12		0.05
cluster_2	0.50		0.01
...	...	...	...
cluster_n	0.04		0.60

Figure 1: Vec2Struc Procedure.

a cluster that are above a similarity threshold. Using cosine similarity, we found pairs of nodes that are above a similarity of 0.9, providing a reduced search space of node pairs.

3. Ego Structure Extraction of Similar Pairs in Clusters

Next, using the pairs of nodes that are most similar in their clusters, pair-wise comparisons were conducted on their local neighbor networks. The problem of examining two separate networks and finding common structures has been defined as the Maximum Common Edge Subgraph problem and has been found to be an NP-Hard problem (Bahiense et al. 2012). Hence the need for a reduced space as algorithmic complexity of this problem makes the runtime of current solutions potentially intractable given a large enough network. To further assist with this performance issue, we only examine 2-hop ego networks centered on nodes of interest before conducting the maximum common subgraph search.

Maximum Common Subgraphs To tackle finding common subgraphs, we incorporated the algorithm SailMCS provided by (Larsen et al. 2016). This a heuristic algorithm using a combination of iterative local search and a perturbation strategy. This algorithm is parallelized and handled the ego-network sizes we found efficiently enough to make this

Cluster 1	**Tot. Structs**	*979*
Struc ID	***Freq.***	***Total %***
Fan + Tri	84	8.58%
Fan	45	4.59%
Single Match	24	2.45%
Path	15	1.53%
Fan + Path	3	0.30%
No Match	808	82.53%

Table 1: Distribution of found structures within Cluster 1.

Cluster 2	**Tot. Structs**	*1026*
Struc ID	***Freq.***	***Total %***
Fan	126	12.28%
Node bt. Fan	49	4.77%
Single Match	44	4.28%
Fan + Path	24	2.34%
Fan + Tri	19	1.85%
Path	19	1.85%
Fan + Path + Tri	4	0.38%
No Match	741	72.22%

Table 2: Distribution of found structures within Cluster 2.

method tractable. Examples of common subgraphs found can be seen within Fig. 2.

4. Frequency Distribution of Structures per Cluster

With the extracted substructures found, we bring this information back to the cluster level by examining the distribution of structures across clusters. Through equivalence checks between each common subgraph (using a graph isomorphism algorithm (Cordella et al. 2001)), we count how often they occur in each cluster. This distributional representation is used to describe the clusters visually through finding the boundaries of the clusters in the embedded space.

To test this method, we implemented it on a network built from a network simulator on financial transactions called AMLSim by (Weber et al. 2018). AMLSim works well for a testing environment as we were able to specify network size and density as well as incorporate specific substructures within the network. We built a network of ~10,000 nodes with an average degree of 23.25.

Results

Using the AMLSim built network, we found 4 clusters using Struct2Vec embeddings as an input to a Gaussian Mixture Model. Examining the two dominant clusters (Cluster 1 contains 53% of nodes and Cluster 2 contains 38% of nodes), we explored a subset of structures within each cluster. We examined around 1000 structures for each cluster, as the number of possible pair-wise comparisons in each cluster, even after filtering, was high (25 million possible pair-wise comparison were possible within Cluster 1). While examining these, many can be considered similar but still vary in the number of nodes and edges, which would cause the strict isomorphic checks within step 3 to fail as the exact number of nodes and edges is needed for these checks to return true. This was a good indication that isomorphic methods are too strict.

To showcase similar structures, we manually examined the commonly found structures and categorized them. Tables 1 and 2 show the frequency in which each category of structure was found. The primary category found was what we refer to as a *Fan* (shown within Fig. 2a), which is a central node acting as the only connection between the surrounding neighborhood. Iterations of the Fan structure was seen in many other types, *Fan+Triangle* (shown in Fig. 2b) has the inclusion of a triangle clique between two nodes and the central node. The number of triangles present were dominantly one, but some structures were found to have up to 3.

The next primary structure was a node acting as a bridge between two Fans, referred to as *Node Between Fan* (shown in 2c). These highlight a more complex structure as a bridge node is the primary connection between two neighborhoods. Lastly, we found a fair amount of path structures which were only a continuous sequence of nodes with those in the middle of the path having exactly 2 edges. Fans were found to have paths extending outside of the central node, which are classified separately from the Fans shown in Fig. 2a.

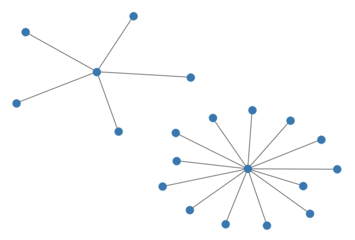
The most common structures overall tended to have a Fan structure, which isn't surprising given that this is a common pattern made from random connections within AMLSim (Weber et al. 2018). Cluster 1 has a higher frequency of Fans with a Triangle, indicating more nuanced relationships. As Cluster 2's second most frequent structure was a node acting as a bridge between two fans (shown within Fig. 2c) which wasn't present within Cluster 1, indicating nodes in Cluster 2 may be acting as neighbor connectors.

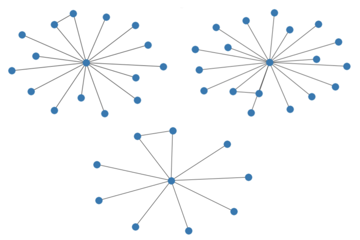
While unique structures were found across each cluster, many of the them were unable to find a match (82.53% within Cluster 1 and 72.22% within Cluster 2). However, after conducting a visual inspection, many of the structures matched similar high level categorization as the categories already reported within each Cluster. Due to the high amount of no matches we were unable to categorize them into the found types. This further indicates a need for a less strict checks on isomorphism and allow for more higher level similarity measures of substructures to automate this process.

There were structural types found within the no match category that were too complex and unique to match any other. Fig. 3 shows an interesting structure as it shows a higher connected pair of Fans as there exist multiple bridges instead of the more common one bridge within Fig 2c. Again highlighting the need for better similarity measures.

Discussion

This work presents Vec2Struc, a new method pushing more interpretable representations of state-of-the-art network models. As embeddings have continued to highlight their potential for graphical representations in a more compact space (Goyal and Ferrara 2018), this method provides a step forward in being able to explain the associations of nodes. This is through addressing the first criterion mentioned by (Weld and Bansal 2019) by bringing forward the


(a) *Fan* Structures commonly found across both clusters.


(b) *Fan + Triangle* structures more often found within cluster 1.

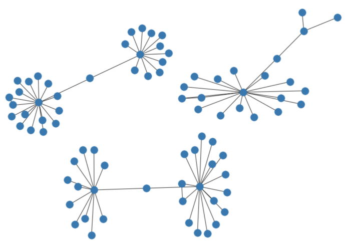

(c) *Node between Fan* structures commonly found within cluster 2.

Figure 2: Examples of commonly found structures across the dominant two clusters.

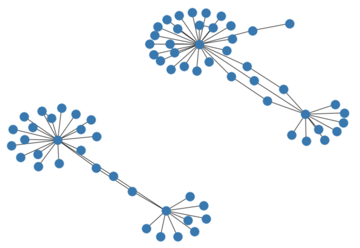


Figure 3: Structures within Cluster 2 containing multiple bridges.

structural factors influencing decisions made using embeddings with visualizations of such structures. Visualizations have been a common representation of other prediction explanations (Ribeiro, Singh, and Guestrin 2016) as well as a common representation of networks. Applications of this work can be within the areas mentioned above in the Introduction of discovering new Protein-to-Protein interactions as well as social network structures, but directly this can be applied along with AMLSim to be used in financial networks for the discovery of new criminal financial activities in money laundering (Weber et al. 2018).

While this work showcases a test case of this method, it still contains many limits in its ability. First being the computational complexity of the algorithm and the main bottleneck being the Maximum Common Subgraph discovery procedure, which is an NP-Hard problem (Bahiense et al. 2012). To work around this, filter techniques were used to only examine the most common pairs, hence the typically assumptions of clustering apply (Clusters within the embedded space are not guaranteed to be separable and goodness of fit measures are needed to audit the cluster model's performance), as well as the thresholds of our similarity check need to be further examined within each space as to what threshold is appropriate. Another issue lies within our experiment, as we needed to conduct further sampling (1000 common subgraphs found per cluster) to allow for a reasonable runtime of SailMCS.

Future work could expand on this experiment through exploring higher frequency of specific structures of interest, as well as explore other network types as this is unclear of its performance given global network properties, (i.e how does density of the network affect runtime?). Real-world networks need to be explored, as the most common type of structure found (Fan shown in Fig. 2a) is easily produced from random edge generation. Lastly, additional work is needed in understanding the dynamics of embeddings through these visualizations to reach the second criterion proposed by (Weld and Bansal 2019).

References

Bahiense, L.; Manić, G.; Piva, B.; and De Souza, C. C. 2012. The maximum common edge subgraph problem: A polyhedral investigation. *Discrete Applied Mathematics* 160(18):2523–2541.

Cordella, L. P.; Foggia, P.; Sansone, C.; and Vento, M. 2001. An improved algorithm for matching large graphs. In *3rd IAPR-TC15 workshop on graph-based representations in pattern recognition*, 149–159.

Goyal, P., and Ferrara, E. 2018. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems* 151:78–94.

Larsen, S. J.; Alkærsig, F. G.; Ditzel, H. J.; Jurisica, I.; Alcaraz, N.; and Baumbach, J. 2016. A simulated annealing algorithm for maximum common edge subgraph detection in biological networks. In *Proc. of the Genetic and Evolutionary Computation Conference 2016*, 341–348. ACM.

Ribeiro, L. F.; Saverese, P. H.; and Figueiredo, D. R. 2017. struc2vec: Learning node representations from structural identity. In *Proc. of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 385–394. ACM.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. Why should i trust you?: Explaining the predictions of any classifier. In *Proc. of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 1135–1144. ACM.

Weber, M.; Chen, J.; Suzumura, T.; Pareja, A.; Ma, T.; Kanezashi, H.; Kaler, T.; Leiserson, C. E.; and Schardl, T. B. 2018. Scalable graph learning for anti-money laundering: A first look. In *NeurIPS 2018 Workshop on Challenges and Opportunities for AI in Financial Services: the Impact of Fairness, Explainability, Accuracy, and Privacy*.

Weld, D. S., and Bansal, G. 2019. The challenge of crafting intelligible intelligence. *Commun. ACM* 62(6):70–79.