International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Machine Learning Methods with Node-Level Features

Koffka Khan

Department of Computing and Information Technology, The University of the West Indies, St. Augustine Campus, TRINIDAD AND TOBAGO

Abstract

In this paper we describe several methods for obtaining node features. They can also be grouped based on structure-based features, which again, do not require degrees, such as the simplest one, and which count edges, clusters, and triangles, as well as significance features like node degree and various centrality measures. A generalization that counts additional structures in which a particular node of interest participates is the Graphlet Degree Vector (GDV).

Keywords:node, features, edges, clusters, centrality, measures

1. Introduction

This paper discusses conventional approaches to machine learning in graphs [3]. Specifically, we're going to look into the various job levels that we can include in the graph. We can focus on the node-level prediction jobs in particular, and the link-level or edge-level prediction tasks that take into account pairs of nodes and attempt to forecast whether the pair is connected or not. Additionally, we may consider making predictions at the graph level, or for the complete graph. For instance, for a whole molecule or for a whole piece of code.

The focus of the conventional machine learning process is on creating appropriate features [19]. We'll focus on two different kinds of characteristics. We will presume that nodes already have a variety of properties attached to them. As a result, for instance, in a protein interaction network, proteins have diverse chemical structures and properties [17]. We can think of these as attributes that are tied to the network's nodes.

At the same time, we also want to be able to add new features that explain where this specific node is located in relation to other nodes in the network and the local network structure it is part of. We can create more precise predictions thanks to these extra attributes that define the topology of the network in the graph. As a result, we must always consider two different kinds of features: structural features [10] and features that describe the characteristics and traits of the nodes.

The objective of what we want to do is therefore particularly focused on structural features [10] that will describe the structure of a link in the network's broader context, the neighborhood of a given node of interest in the network, as well as features that will describe the structure of the entire graph, in order to feed these features into machine learning models that will use them to make predictions.

In traditional machine learning pipelines [20], we have two steps. We will take our data points, nodes, linkages, and whole graphs in the first stage and represent them with feature vectors. And then, on top of

that, we'll train a model or classifier [4] using traditional machine learning, like a random forest, support vector machine, feed-forward neural network, or anything similar. So that we may utilize this model whenever a new node, link, or graph occurs in the future. We can gather its characteristics, and we can predict. So this is the overall environment in which we will function.

We're going to concentrate, as I said, on feature design [18], where we'll employ efficient features on graphs to get strong predictive performance because you want to be able to capture the network's relational structure [21]. And conventional machine learning pipelines make use of manually created features. These customized qualities will be the focus of the discussion today. Additionally, the talk will be divided into three parts. Before moving on to discuss features that can describe a pair of nodes, we will first discuss features that describe individual nodes and can be used for node-level prediction [7]. You can think of these as features that can describe a pair of nodes for link-level prediction. In addition, we'll discuss features and methodologies that characterize the topology structure of complete graphs, allowing for the comparison and classification of various graphs. For the sake of simplicity, we'll concentrate on undirected graphs [12].

The objective is to determine how to anticipate a set of items of interest when our feature vector is a d-dimensional vector due to design considerations. The things that we are interested in are edges, sets of nodes, and nodes themselves, i.e., full graphs. And the goal function that we'll be considering is, what labels are we attempting to predict? The best way to think of this is to imagine a network as a collection of vertices, edges, and nodes.

This paper consists of seven sections. In Section two we introduce node degree which is a feature or characteristic which tries to capture the structure of the graph. In the next three sections we discuss centrality measurements. In Section three we discusseigenvector centrality, while betweenness centrality is given in Section four. Then proximity centrality is illustrated in Section five. Small, induced subgraphs of a big network known as "graphlets" are said to exist at any frequency. An induced subgraph is one in which, after choosing the nodes in the major network, you also choose all the edges connecting them.Graphlets are discussed in Section six. Finally, we give our conclusion in Section seven.

2. Node features

We want to learn a function that, for instance, will provide us with a real valued forecast for each node. This function would be useful, for instance, if we were attempting to predict the edges of each node in our social network. How can we learn this function f that will make these predictions? Therefore, we will start by discussing node-level tasks and attributes that identify certain nodes.

The way we're thinking about this is that we're thinking about it in a situation known as a semi-supervised case, in which we're given a network and a few nodes that have been labelled with various colors and our objective is to predict the colors of uncolored nodes. And we want to color the grey nodes given the red and green nodes. Imagine that you have such a graph.

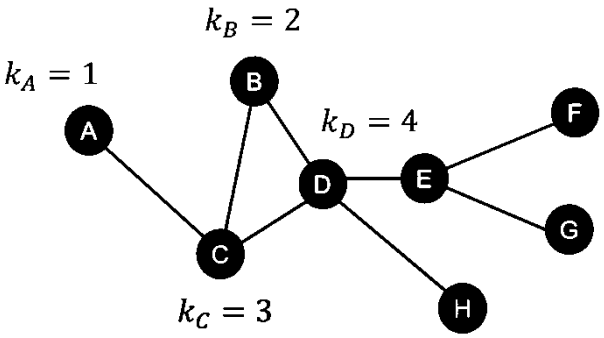
Further, you can create a rule that red nodes must have exactly one edge linked to them, whereas green nodes must have at least two edges immediately adjacent to them. We will be able to learn the model that, in this simple scenario, correctly colors the nodes of the graph if we can now represent the node degree of each node as a structural feature in this graph.

Thus, we require characteristics that will characterize this specific topological pattern. Therefore, the objective is to describe the structure of the network around a specific node as well as, in a sense, the

node's position within the larger network context. And we're going to discuss four different strategies that let us accomplish this. The degree of the node can always be used to characterize the network structure around the node. Next, we may consider the significance and position of the node using the concept of node centrality measures [13]. Then, we'll discuss defining the local network structure [23]. What kind of structure does the node around a specific node have in addition to how many edges it has? The clustering coefficient will be discussed first, and then this will be applied generally to the idea of graphlets.

Let's start by discussing the node degree [22]. It is a very helpful characteristic that is crucial. Basically, we will state that we capture the network node's structure by counting the number of edges it has. Additionally, this has the disadvantage of treating all of the neighbors similarly. However, nodes with the same degree are indistinguishable even if they are located in different areas of the network.

Figure 1: Predictions for nodes in graph.



For instance, the nodes C and E share the same degree, making it impossible for our classifier to tell them apart. Or perhaps nodes A, H, E, F, and G are all degree one nodes. As a result, they will all have the same feature value, which means that our basic machine learning model, which just takes into account the node degree as a feature, will only be able to predict the same value for all of these various nodes because they all have the same degree.

We can then begin to think about how node degree only counts the node's neighbors, without capturing, say, their importance or who they truly are. This very basic idea can then be generalized a little. Therefore, the node centrality metrics aim to, um, represent or characterize this idea of the node's significance in the graph. And there are numerous ways in which we might model or express this idea of importance.

3. Eigenvector Centrality

I'll quickly introduce the Eigenvector Centrality [15] concept. I'm going to discuss both closeness centrality, which attempts to measure how near the network's center a given node is, and centrality, which tells us how significant a conduit a given node is as a connector. There are, of course, several more indicators of centrality or significance. Let's start by explaining what an eigenvector centrality is. When node v is surrounded by other significant nearby nodes u, we say that node v is equally important.

The concept is that we simply add the importance of a given node's neighbors in the network, normalized divided by 1 over Lambda, for a particular node, v, see Equation (1). The concept is that my personal importance increases in direct proportion to the importance of my friends. And if you look at this equation and write it down, you can write it in terms of a straightforward matrix equation where, essen-

tially, Lambda is this positive constant that functions as a normalizing factor, c is the vector of our centrality measures, A is now the graph adjacency matrix, and c is once more that vector of centrality measures.

$$c_v = \frac{1}{\lambda} \sum_{u \in N(v)} c_u$$

λ is some positive constant

$$\lambda \boldsymbol{c} = \boldsymbol{A}\boldsymbol{c}$$

- $\boldsymbol{A}$: Adjacency matrix
  $\boldsymbol{A}_{uv} = 1$ if $u \in N(v)$
- $\boldsymbol{c}$: Centrality vector

(1)

And if you write this in this kind of forum, you'll find that it's a straightforward equation involving an eigenvector and an eigenvalue.

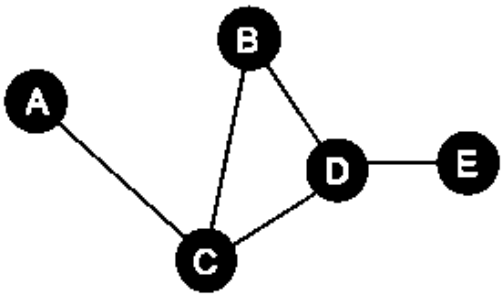
Consequently, this means that the supplied eigenvalue and the corresponding eigenvector are what solve this problem and its associated equation, respectively. And the eigenvector linked to the biggest eigenvalue is what is used to measure the centrality of nodes. Due to the Perron-Frobenius theorem [1] and the fact that we believe the graph to be undirected, the biggest eigenvalue in this instance, if it exists, is Lambda max and is always positive and unique.

The related leading eigenvector c max is then typically employed as a centrality score for the network's nodes. Again, it makes sense that a node's importance is equal to the normalized sum of the importance of the nodes it links to. Therefore, it is not how many connections you have but rather who those connections point to and how significant those individuals are. Therefore, this is the idea of nodes' centrality as it is represented by the eigenvector of centrality.

4. Betweenness centrality

Betweenness centrality [6] is a new kind of centrality that captures a different feature of the position of the nodes in the network and has a totally different intuition. According to betweenness centrality, a node is significant if it is situated along many of the shortest routes between other pairs of nodes. The concept is that a node has significant relevance if it acts as an essential link, bridge, or type of transit hub. In order to calculate betweenness centrality, we must first state that the betweenness centrality of a given node v is the sum across pairs of nodes, s and t. The number of shortest pathways that pass via the node v between s and t is then counted, and that number is normalized by the overall number of shortest paths of the same length that connect s and t.

Figure 2: Betweenness centrality.



In essence, a given node is more significant the more shorter paths it appears on. Therefore, it indicates that this evaluates how effective a node is as a connecting or transit point. The centrality of these nodes that are on the on the edge of the network, like A, B, and E, is zero, see Figure 2.
But because the shortest path from A to B passes via C, the shortest path from A to D also passes through C, and the shortest path between A and E also passes through C, the betweenness centrality of node C equals three. Given that these are the three shortest routes via node C, it has a centrality of three. The node D's betweenness centrality will be the same, equaling three, according to a similar logic. Here are the related shortest routes that actually pass through this node D, between various pairs of nodes.

5. Proximity centrality

The third sort of centrality, known as proximity centrality [14], again captures a different component of the position of the node after we discussed how significant a transportation hub a given node is captured by betweenness centrality. According to the idea of centrality relevance, a node is significant if its shortest path lengths to all other nodes in the network are short. In other words, the closer to the center you are, the shorter your path to everyone else is, and therefore, the more significant you are.
We operationalize this by saying that the proximity centrality of a particular node v is one over the sum of the shortest paths between the node of interest three and all other nodes in the network, see Equation (2).

$$e_v = \frac{\#(\text{edges among neighboring nodes})}{\binom{k_v}{2}} \in [0,1] \quad (2)$$

The principle behind this, well, is that the more in the middle you are, the smaller the sum will be. Therefore, one over a small number will be a large number.
Furthermore, betweenness centrality will be low because the sum of the shortest path lengths will be high if someone is, example, very far from the center of the network and needs many long paths to reach other nodes. For instance, you know, the closest centrality of node A (see Figure 2) equals 1/8 because it has the shortest path of length two to node B, the shortest path of length one to node C, the shortest path of length two to node D, and the shortest path of length three to node E, so it's 1/8, see Equation (3).

$$c_A = 1/(2+1+2+3) = 1/8$$
$$\text{(A-C-B, A-C, A-C-D, A-C-D-E)} \quad (3)$$

As an illustration, the node D, which is a little more in the middle of the network, has shortest pathways of length 2 to node A and length 1 to every other node, see Equation (4).

$$c_D = 1/(2+1+1+1) = 1/5$$
$$\text{(D-C-A, D-B, D-C, D-E)} \quad (4)$$

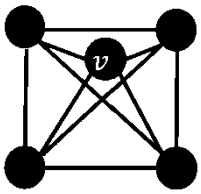
Therefore, since this is one over five, its betweenness centrality—is higher. And now I'm going to change the subject slightly. In my previous sentence, I discussed centralities in terms of how significant

a node's position in a network is. We will now begin to discuss and reflect on the node degree and the local structure surrounding the node.

And when I say local structure, I really mean that for a particular node, we just consider the pros, cons and characterize the attributes of the network in and around it. The clustering coefficient is a traditional metric of this. Additionally, the clustering coefficients gauge the degree of ties between neighbors. How interconnected node v's friends are. And to describe this, we'll state that the clustering coefficient [8] of a node, v, is equal to the product of the number of edges between v's neighbors, divided by the degree of v, in this case, 2, see Equation 2.

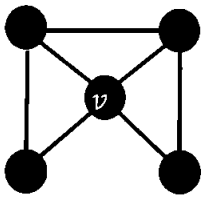
In other words, this number- this is k choose 2 — indicates how many pairings you can choose from among k different things. In other words, how many node pairings are there in your neighborhood? How many possible edges exist in your neighborhood's network? -"How many edges truly exist," among other things. Thus, this metric falls naturally between zero and one, where one would indicate that all of your friends are also friends with one another and zero would indicate that none of your connections or friends are acquainted with one another.

Figure 3:e_v = 1, see Equation 2.



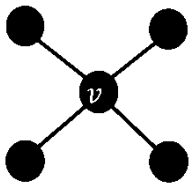
So, let me give you an example. Imagine a basic five-node network where the node labelled v in this case, is the node of interest, see Figure 3. As an illustration, node v in this instance has a clustering value of one because all four of its buddies are linked to one another. Thus, clustering is one in this instance.

Figure 4:e_v = 0.5, see Equation 2.



For instance, the clustering in this specific instance is 0.5, see Figure 4. The reason for this is that only three of the six possible connections between the four nodes that make up node v's neighbours are really in place.

Figure 5:e_v = 0, see Equation 2.



The clustering in this example is zero (see Figure 5), in contrast to the last one, where all four of node v's neighbors are not related to one another. The intriguing finding that prompts us to extend this idea from the clustering coefficient to the concept of graphlets is that the clustering coefficient essentially counts the number of triangles in a node's ego-network. Let me now elaborate on what I mean by that. First off, the ego-network of a specific node is just a network that is created by that node and its surrounding nodes. In essence, a degree 1 neighborhood network is present surrounding a particular node.
What do I mean by "counts of triangles"? By this, I mean that I can now count how many triples of nodes are connected if I have this ego-network of a node. The clustering coefficient of this node is 0.5 (see Figure 4) in this specific use case, which indicates that I can discover three triangles in the network, neighborhood, and ego-network of my node of interest, see Figures 6, 7 and 8. As a result, the clustering coefficient is actually counting these triangles, which are crucial in social networks when I am friends with someone who is a friend of a friend.

Figure 6: 1st triangle out of 6 node triplet from Figure 4.

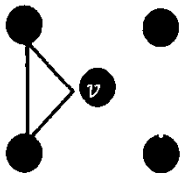


Figure 7: 2nd triangle out of 6 node triplet from Figure 4.

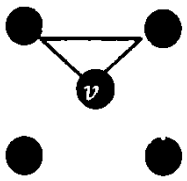
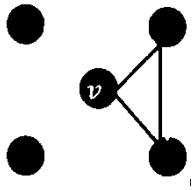


Figure 8: 3rd triangle out of 6 node triplet from Figure 4.

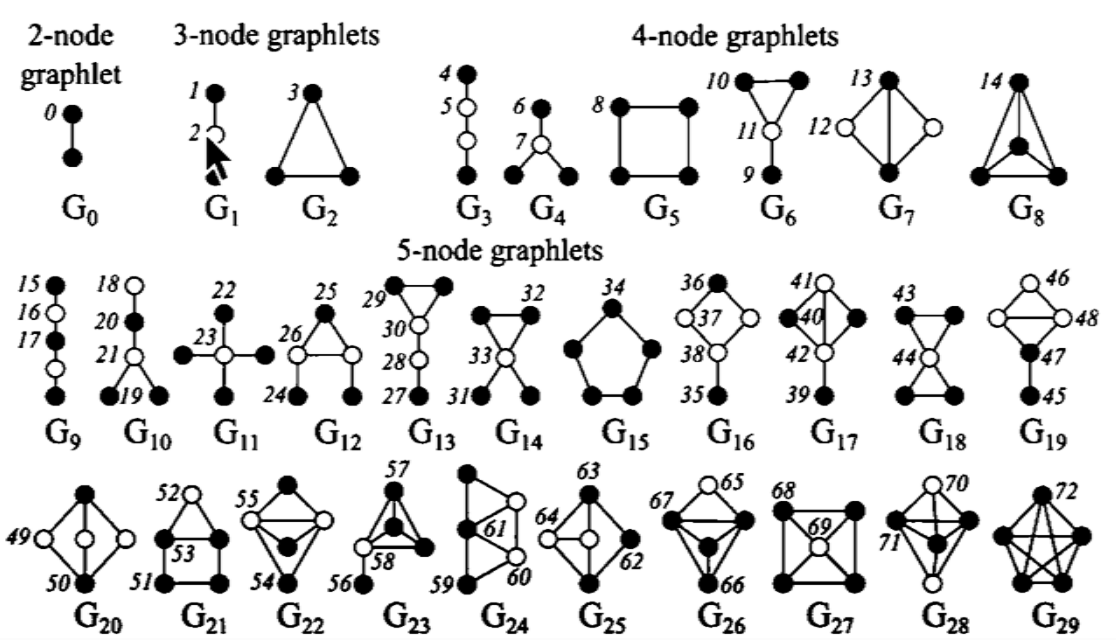


Social networks [2] naturally develop by triangle closing, wherein the intuition is that if two people have friends in common, then more often than not, these two friends will be introduced by this node v and a link will form. As a result, social networks frequently exhibit the "many triangles syndrome," and the clustering coefficient is a crucial indicator. Now that we know this, the question is if we can apply the idea of triangle accounting to other, more intriguing topologies and count the number of pre-specified graphs that are located close to a given node. And the idea of a graphlet precisely captures this.

6. Graphlets

The final method of characterizing the network's structure around a specific node is to use this idea of graphlets [16], which reduces them to nothing more than a triangle count. Counts are of additional types of structures that are nearby the node. So, allow me to explain. Thus, a graphlet is a non-isomorphic, linked, rooted subgraph. What exactly do I mean by that? Here are several examples of all conceivable graphlets, each of which begins with a graphlet on two nodes and has a variety of node counts. In essence, it consists of nodes connected by edges.

Figure 9: 2, 3, 4 and 5-node graphlets.



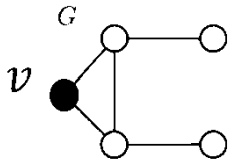
The next query is how many graphs similar to this do you take part in, see Figure 9. Or, you can occupy this other, second-from-the-left position, which essentially asks how many friend pairs you have. Then, since there is only one possible position in the case of a triangle because all of these positions are equivalent and isomorphic, you can be in this position. Similarly, if you look at the four node graphlets, you'll see that there are many more of them. Once more, you have two spots on a chain that has four nodes.

If you travel from the other end, it is simply symmetric; you are either in the edge or one-one away from the edge. You have a choice of two positions in the second star graphlet: either the top on the star's periphery or the star's center within a square, see Figure 9 – 4-node graphlets, G_6. You can be only a portion of the square because all places are isomorphic. Another intriguing example, where you try three alternative positions, is shown here.

You have two options: either you can be in this position, which is isomorphic to the other side in this type of square, or you can be at position 10, which is v-dot diagonal, see Figure 9 – 4-node graphlets, G_6.. And in this final four-node completely connected graph, all nodes are equal, see Figure 9 – 4-node graphlets, G_7.. As a result, there is just one position because they are all equal. This demonstrates that, if you ask how many graphlets there are on five nodes, there are 73 of them. Due to the many graphs and placements in these graphs, the labels are numbered from zero to 72.

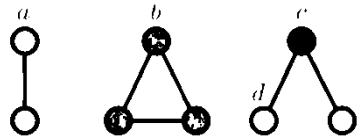
Now that we are aware of what a graphlet is, we can define the term "Graphlet Degree Vector," which refers to a node's attributes that are based on a graphlet [11]. Additionally, graphlet degree counts the instances in which a particular graphlet is rooted at a particular node. The number of edges that a node contacts is counted by degree, the number of triangles it touches or participates in is counted by the clustering coefficient, and the number of graphlets a node is a member of is counted by the graphlet degree vector. A Graphlet Degree Vector, for instance, is just a count vector of graphlets rooted at the specified node.

Figure 10: Graph with node of interest being V.



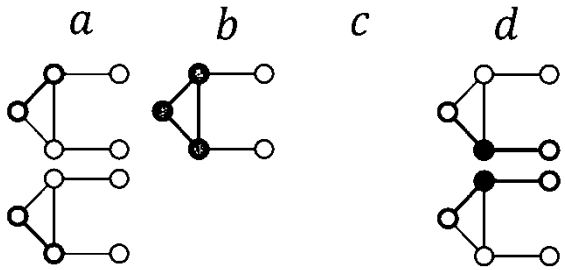
So, as an example, let's have a look at this particular graph where node v is the focus of our attention, see Figure 10. Next, we have a group of graphlets on two and three nodes, see Figure 11.

Figure 11: List of Graphlets derived from Figure 10.



This is the graphlet universe we live in, up to three nodes and no bigger. The graphlets will only be examined up to a size of three nodes, where node V is located. After that, you know, what are the instances of the graphlets, for instance, the graphlets of type a, in which node v is involved in two of them, see Figure 12 – under letter a.

Figure 12: Graphlet instances.



So one of these involves the graphlet of type b node V, see Figure 12 – under letter b. Then you ask, "How many graphlets of type c does node V participate in?" It doesn't take part in any since in Figure 10 the two nodes are connected directly to V and there is an edge existing which connects the two nodes. Therefore, we receive zero. For d, we see two occurrences, Figure 12 – under letter d. This is shown on Figure 10 with V and the top two nodes for one occurrence and the bottom two nodes and V for the oth-

er. Therefore, the graphlet's degree vector for node v would be 2, 1, 0, 2, respectively. This is written as key: value pairs as follows: Graphlet Degree Vector (GDV) of node v = [a: 2, b: 1, c: 0, d: 2].

Now, based on the frequency of these graphlets in which the supplied node of interest participates, this describes the neighborhood structure locally around that node. If we take into account graphlets with, two to five nodes, each node in the network may now be described by a vector with 73 dimensions or 73 coordinates. In essence, this is a node signature that describes the neighborhood's topology of nodes. It also captures its interactions and connections up to a distance of four hops. Because a chain with four edges contains five nodes, if you are at the edge of the chain you should count the number of pathways that branch off of that node that are four edges long. Therefore, it describes the network up to a distance of four.

GDV provides a more thorough measure of local topological similarity by measuring nodes, local network topology, and comparing vectors now of two nodes. For instance, only examining the clustering coefficient or node degree. Thus, this provides a granular method for comparing the neighborhoods, the structure of neighborhoods of two distinct nodes potentially across two different networks.

7. Conclusion

In conclusion, the importance based features accurately reflect a node's significance in a graph. We discussed degree centrality as well as other concepts of proximity, betweenness, and centrality, as well as the centrality of the eigenvector. And these kinds of characteristics are used to forecast, for instance, how significant or influential nodes in the graph are. For instance, one such example would be spotting celebrities on social media.

The other kind of node level features we discussed were structured based features that capture the topological characteristics of the immediate area surrounding the node. We discussed node degree, clustering coefficient, graphlet degree vector, and other indicators that are quite helpful for determining a specific node's role in the network. For instance, if you consider predicting protein function, these types of graphlet features are particularly helpful since they describe the topology of the network around a particular node.

8. References

1. Ahmadinejad, AmirMahdi, Arun Jambulapati, Amin Saberi, and Aaron Sidford. "Perron-frobenius theory in nearly linear time: Positive eigenvectors, m-matrices, graph kernels, and other applications." In Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1387-1404. Society for Industrial and Applied Mathematics, 2019.
2. Benamira, Adrien, Benjamin Devillers, Etienne Lesot, Ayush K. Ray, Manal Saadi, and Fragkiskos D. Malliaros. "Semi-supervised learning and graph neural networks for fake news detection." In 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp. 568-569. IEEE, 2019.
3. Chami, Ines, Sami Abu-El-Haija, Bryan Perozzi, Christopher Ré, and Kevin Murphy. "Machine learning on graphs: A model and comprehensive taxonomy." Journal of Machine Learning Research 23, no. 89 (2022): 1-64.
4. Ding, Ming, Jie Tang, and Jie Zhang. "Semi-supervised learning on graphs with generative adversarial nets." In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 913-922. 2018.

5. Furno, Angelo, Nour-Eddin El Faouzi, Rajesh Sharma, and Eugenio Zimeo. "Fast approximated betweenness centrality of directed and weighted graphs." In International Conference on Complex Networks and their Applications, pp. 52-65. Springer, Cham, 2018.
6. Furno, Angelo, Nour-Eddin El Faouzi, Rajesh Sharma, and Eugenio Zimeo. "Fast approximated betweenness centrality of directed and weighted graphs." In International Conference on Complex Networks and their Applications, pp. 52-65. Springer, Cham, 2018.
7. Heindl, Christoph. "Graph Neural Networks for Node-Level Predictions." arXiv preprint arXiv:2007.08649 (2020).
8. Ji, Qingbin, Deyu Li, and Zhen Jin. "Divisive algorithm based on node clustering coefficient for community detection." IEEE Access 8 (2020): 142337-142347.
9. Kwon, Oh-Hyun, Tarik Crnovrsanin, and Kwan-Liu Ma. "What would a graph look like in this layout? a machine learning approach to large graph visualization." IEEE transactions on visualization and computer graphics 24, no. 1 (2017): 478-488.
10. Lee, Jaekoo, Hyunjae Kim, Jongsun Lee, and Sungroh Yoon. "Transfer learning for deep learning on graph-structured data." In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31, no. 1. 2017.
11. Lyu, Tianshu, Yuan Zhang, and Yan Zhang. "Enhancing the network embedding quality with structural similarity." In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 147-156. 2017.
12. Matlock, Matthew K., Arghya Datta, Na Le Dang, Kevin Jiang, and S. Joshua Swamidass. "Deep learning long-range information in undirected graphs with wave networks." In 2019 International Joint Conference on Neural Networks (IJCNN), pp. 1-8. IEEE, 2019.
13. Mendonça, Matheus RF, André MS Barreto, and Artur Ziviani. "Approximating network centrality measures using node embedding and machine learning." IEEE Transactions on Network Science and Engineering 8, no. 1 (2020): 220-230.
14. Mendonça, Matheus RF, André MS Barreto, and Artur Ziviani. "Approximating network centrality measures using node embedding and machine learning." IEEE Transactions on Network Science and Engineering 8, no. 1 (2020): 220-230.
15. Rakaraddi, Appan, and MahardhikaPratama. "Unsupervised Learning for Identifying High Eigenvector Centrality Nodes: A Graph Neural Network Approach." In 2021 IEEE International Conference on Big Data (Big Data), pp. 4945-4954. IEEE, 2021.
16. Rossi, Ryan A., Rong Zhou, and Nesreen K. Ahmed. "Estimation of graphlet counts in massive networks." IEEE transactions on neural networks and learning systems 30, no. 1 (2018): 44-57.
17. Rual, Jean-François, Kavitha Venkatesan, Tong Hao, Tomoko Hirozane-Kishikawa, Amélie Dricot, Ning Li, Gabriel F. Berriz et al. "Towards a proteome-scale map of the human protein–protein interaction network." Nature 437, no. 7062 (2005): 1173-1178.
18. Simsek, Meryem, Oner Orhan, Marcel Nassar, OguzElibol, and Hosein Nikopour. "Iab topology design: A graph embedding and deep reinforcement learning approach." IEEE Communications Letters 25, no. 2 (2020): 489-493.
19. Xia, Feng, Ke Sun, Shuo Yu, Abdul Aziz, Liangtian Wan, Shirui Pan, and Huan Liu. "Graph learning: A survey." IEEE Transactions on Artificial Intelligence 2, no. 2 (2021): 109-127.

20. Xin, Doris, Hui Miao, Aditya Parameswaran, and NeoklisPolyzotis. "Production machine learning pipelines: Empirical analysis and optimization opportunities." In Proceedings of the 2021 International Conference on Management of Data, pp. 2639-2652. 2021.
21. You, Jiaxuan, Jure Leskovec, Kaiming He, and Saining Xie. "Graph structure of neural networks." In International Conference on Machine Learning, pp. 10881-10891. PMLR, 2020.
22. Zhang, Ziwei, Peng Cui, and Wenwu Zhu. "Deep learning on graphs: A survey." IEEE Transactions on Knowledge and Data Engineering (2020).
23. Zhu, Wenwu, Xin Wang, and Peng Cui. "Deep learning for learning graph representations." In Deep learning: concepts and architectures, pp. 169-210. Springer, Cham, 2020.