Combining Structure and Property Values is Essential for Graph-based Learning

[Position Paper]

David J. Haglin
Pacific Northwest National Laboratory
Richland, Washington 99352, USA
Email: david.haglin@pnnl.gov

Lawrence B. Holder
Washington State University
Pullman, WA, 99164, USA
Email: holder@wsu.edu

Abstract

Graph mining algorithms that seek to find interesting structure in a graph are compelling for many reasons but may not lead to useful information learned from the data. This position paper explores the current graph mining approaches and suggests why certain algorithms may provide misleading information whereas others may be just what is needed. In particular, algorithms that ignore the rich set of node and edge properties that are prevalent in many real-world graphs are in danger of finding results based on the wrong information.

1. Introduction

Graph theory has a long and rich history going back more than 250 years [1]. It is therefore natural to explore how to employ *machine learning* (ML) techniques on graph data. We define a graph to be a collection of structural information in the form of nodes connected by edges combined with property values associated with both the nodes and the edges. Throughout this paper we refer to the two types of data as *structural* and *value*. While many ML algorithms learn from either graph structure or the edge and node property data, it is our belief that ML algorithms that draw from both structural and value information are needed to gain insight into our ever-growing complex graph data.

We note that in real-world graphs, between any single pair of nodes there may be zero, one or many (perhaps even millions of) edges. Where multiple edges between a given pair of nodes exist, the expectation is that the combination of edge properties on each of the edges will delineate one edge from the others.

This paper presents a brief survey of ML techniques on graph data, emphasizing the reality of current ML graph algorithms trying to learn from structural or value data, but not both. We then present two use cases where the representative data has a rich combination of both structural and value data. Some natural questions are presented for each of these use cases where learning from only one or other form of information might lead to incorrect conclusions.

2. Survey of Supervised Machine Learning in Graphs

There are two main scenarios for supervised machine learning in graphs. One is the *graph transaction* scenario, where you are given a set of disconnected, relatively small graphs, and each graph is assigned a class (e.g., positive or negative). And the other scenario is one large graph where some parts of the graph are given a class designation. For example, each node might be assigned to one of two classes.

2.1. Graph Transaction Scenario

One approach to learning in this setting is to define a set of features over these graphs, and then each graph is represented by a feature vector. The features can be global graph features [2] or frequent subgraphs across the whole set of graphs [3]. At this point any standard (non-graph) machine learning method can be applied. Support Vector Machines (SVMs) using a radial basis function (RBF) kernel seems to generally work well. Of course, it is difficult to understand the classifier

in terms of characteristics of the graph that makes it positive or negative. Another approach to the graph transaction scenario is to use a graph kernel based SVM, bypassing the need to define features [4], [5]. One final approach is to simply look for subgraphs that occur often in the positive graphs and not in the negative graphs. This is the approach used by the SUBDUE method [6].

2.2. One Large Graph Scenario

The second scenario for supervised machine learning in graphs is where the input is one large graph. In this scenario, nodes (sometimes edges, sometimes subgraphs) are labeled positive or negative, and then learning is applied. One approach is to extract subgraphs from around the neighborhood of these nodes, and then treat the problem as in the Graph Transaction scenario.. Silva *et al*. present an approach to find correlations between subgraphs induced by vertices with specific attributes and the density of the subgraph patterns [7]. But the edges in their graphs are without any properties. Another approach, which also assumes some nodes have unknown classes, is to use semi-supervised learning, i.e., classify unknown nodes based on the classes of their neighbors and iterate [8]. Their definition of the input graph has a vector of attributes on each node, but the edges are not enriched with any properties.

A recent position paper by Turkett *et al.* asserted that nodes and edges ought to be enriched with information to provide better machine learning accuracy from the data [9]. Their application domain is computer network activity, and they described aggregating all traffic between two servers and representing connectivity with a single edge. The enrichment they suggest is to add properties to the edges such as "average number of packets per connect" between each of the nodes. Why not have an edge for each connection and an edge property of "number of packets"?

2.3. Temporal Aspects of Graphs

The ML algorithms discussed thus far assume a static graph. While this may be good for analysis of historical operations, finding trends relies on the existence of temporal information. This can be important to some analysis that the graph data structures themselves—not just the learning algorithms—need to accommodate ingesting new information and aging off old information. McGregor describes an approach to graph mining on streams [10], but this work is on graphs whose nodes do not carry properties and edges that have only a single weight. Bifet *et al.* explore mining frequent closed graphs on evolving data streams [11]. Their work focuses on a stream of graphs consisting of graphs with node labels (i.e., a single node property) and edges with no properties (but multiple edges between two nodes are supported). However, they assert that any type of graph would be supported by their algorithm provided it is possible to define a partial ordering among all possible graphs. While their work does not explicitly capture the notion of temporal information, an implicit temporal relationship is implied by the ordering of the input graphs within the data stream.

Another temporal issue is with recording time as a property on either nodes or edges. Several options exist such as a date and time stamp, seconds (or milliseconds) elapsed since some "start" time, a time interval, and a duration. Learning from these data may be impacted by the data format.

3. Use Cases

3.1. eBay

Suppose you are managing a company like eBay where your customers sell things to other customers using your infrastructure. You track the transactions and try to make sure that all of your customers are happy.

Unfortunately for you, some of your customers may not be abiding by all of the laws in their country. You are handed a list of customers that some law enforcement agency asserts are all selling stolen merchandise using your services. You are asked to find other customers who may be doing similar activities. It is in your best interest to do so in order to maintain good customer and public relations. How do you find them?

Some observations about this use case:

1) This is (potentially) a supervised machine learning activity. You are given a small percentage of your customers as being labeled "bad", and you assume the vast majority of the rest are labeled "good".
2) Ideally, you would like to find some "signature" that has a high correlation to the "bad" customers, but with few false positives.
3) The signature could be structural (e.g., having one- or two-hop neighbor substructures that contain certain graph connectivity) or they may be property-based (e.g., the bad customers only sell product X in category Y for price P, and that price is well below the average of all product X's

Table 1. Node properties for the eBay Use Case

Property	Notes
Account	A unique identifier for a particular customer/user. This is the unique name used for each node in the graph.
Address	A postal address associated with this account. It should be possible to compute a geographic distance between two customers.
Age of Account	Some indication as to how long the account has been active, perhaps computable if the date of opening the account is recorded.

Table 2. Edge properties for the eBay Use Case

Property	Notes
Seller	A customer account.
Buyer	A customer account.
Date/Time	The date/time the transaction was established. This may not be the date/time of the actual exchange of the item being sold.
Price	This numeric value may need to include a currency (US Dollar, Euro, etc.) so that aggregation and comparisons can be done.
Item Category	It is likely helpful to your company to have a taxonomy of item categories for tracking purposes.

sold on your system). More likely, the signature is a combination of structural and property-based.

4) The signature might be community-based. For example, it might be that bad customers buy things from only other bad customers and sell things only to good customers.
5) It is clear that the more properties available on each node (customer) and edge (transaction) the more likely you are to find valuable insight.

3.1.1. Data Layout. Each node in the graph will correspond to one of the customer accounts as shown in Table 1. We note that a single person might have multiple accounts, so entity disambiguation may be useful.

Each edge of the graph corresponds to one buy/sell order between two customers. Several examples of edge properties are shown in Table 2.

In addition to these two tables of data, there may be other data available to enrich the graph structure. One example is a collection of information about each item. Another example might be some way to associate several item transactions into a single "sale" where a buyer and seller may agree on a transaction of multiple items. Still others include: capturing shipping information (e.g., tracking numbers), bank or credit card account information associated with a customer, and possibly even tracking model and serial numbers of items being sold.

3.1.2. Malicious Activity Description. Suppose the customers engaged in selling stolen merchandise were all operating together in the following way. Some of the perpetrators brought stolen goods into your system and posted them for sale at prices higher than the average price to avoid enticing innocent participants. We call these the *suppliers*. Within two days, a *broker* then offers something closer to the average price and the supplier accepts. Some time later (perhaps 10 or more days later) the broker then posts the item for sale at a low price and waits for an innocent buyer to arrange a sale.

There may be a couple of variations on how the items get moved from supplier to broker. One might be that they are geographically close to each other and the transfer is taken care of offline. Another is that they arrange for a shipment.

3.1.3. Trying to Hide in the Data. Given our assumed malicious activity description, what could the perpetrators do to hide in the data? If our ML algorithm were

going to be aimed at learning only from the graph structure, the suppliers would need to periodically buy items from other (non-broker) customers. And the brokers need to periodically buy items from non-supplier customers. That way, the following graph signatures of illicit activity would be hard to detect.

- **Supplier**: only sells items to one of a small group of customers. These have only out-degree, no in-degree.
- **Broker**: only buys items from one of the small group of sell-only customers and sells to many. These have only incoming edges from "suppliers", recognizable as zero in-degree nodes, and never sells items to one of the "supplier" types.

3.1.4. False Positives. There may be lots of non-supplier customers who only sell items and refuse to consider buying. Yet, these customers might be flagged as possible suppliers. To reduce the risk of alerting on false positives, it would be helpful to consider more than the zero-in-degree structure pattern as an indication of a supplier. For example, a supplier may repeatedly sell lots of items in the same category of your taxonomy. They may sell items only above some significant price threshold. They may sell items only to buyers within a certain distance.

3.2. Electronic Stock Exchange

In this scenario you are managing an electronic trading service like the NASDAQ. Your customers are traders, and the transactions are exchanges of equities for money. Like the eBay scenario, the customers are nodes in a graph with lots of properties (e.g., account number, address, equity holdings, permissions such as whether trading options are allowed), and the transactions are directed edges from one node to another with lots of properties as well (e.g., timestamp, equity, number-of-shares, price). Some of the customers will have special designations such as being a principle in one of the publicly traded companies.

Like the eBay scenario, you are interested in maintaining good public relations and abiding by the laws governing your business. If the SEC steps in and says they are going to arrest 100 of your customers for fraudulent activity, what could you do to look out for this type of activity in the future without waiting for the SEC to investigate?

3.2.1. Data Layout. This graph is a blend of nodes representing different kinds of entities (i.e., node types): customer accounts, and publicly-traded companies. Each customer account node has property values as shown in Table 3 and each company node property values are shown in Table 4.

Each edge of the graph corresponds to one trade transaction between two account holders. Several examples of edge properties are shown in Table 5. We note that in this graph there may be edges whose types represent relationships other than trades. For example, there may be edges from a company node to an account node indicating the account owner is a key executive. Clearly, these non-trade edges do not have the same vector of property values as the trade edges.

3.2.2. Malicious Activity Description. Suppose the customers engaged in fraudulent activity are somehow finding a way to arrange for the sale of a security at a higher price than a purchase of that security before initiating the purchase followed nearly immediately by the sale. This type of activity shows up as a two-path through the graph, where the time difference between the two edges must be less than a couple of seconds, the equity must be the same (i.e., the same number of shares of the same company), and the price must be higher on the second edge than on the first edge.

While this pattern may be considered good trading, if a customer has only this pattern of trades, then the governing body (and the exchange) gets curious.

3.2.3. Trying to Hide in the Data. Legitimate traders may elect to trade with this rapid turnover strategy. But they would surely make a profit only some of the time, perhaps 60% of their trades if they are really good. If their success rate exceeds 90%, that seems suspicious. For malicious traders who have a way to game the system, they can select to buy and hold some shares once in a while, selling portions of these shares at different times. These trades can hide their structure pattern and, if the equity does not move up or down very much, the trader would not risk losing very much on this tactic.

3.2.4. False Positives. There may be lots of traders who trade rapidly. That is, they try to sell very shortly after every purchase. If a person does this only once in a while and for a few times they are lucky to nearly always gain rather than lose money, they could be flagged as a fraud risk. There could be other indicators of fraudulent trading such as always trading a lot of shares, repeatedly trading a small number of companies, or the price difference might always be above some threshold in spite of the very fast trade.

Table 3. Customer Account Node properties for the Stock Exchange Use Case

Property	Notes
Account	A unique identifier for a particular customer/user. This is the unique name used for each node in the graph.
Address	A postal address associated with this account. It should be possible to compute a geographic distance between two customers.
Age of Account	Some indication as to how long the account has been active, perhaps computable if the date of opening the account is recorded.
Holdings	Some way to capture the number of shares of the various companies owned by this account.
Cash	Some way to represent cash held (available for stock purchases) in this account.

Table 4. Company Node properties for the Stock Exchange Use Case

Property	Notes
Basic Info	This includes such things as company name, address, web site, etc.
Key Executives	The people considered by the governing body to be "insiders". These might be in the form of pointers to account nodes.
Financial Info	Accounting that is required by the governing body.
Number of Shares	Total number of shares held by the public (an aggregation of all account node's values).

Table 5. Edge properties for the Stock Exchange Use Case

Property	Notes
Seller	A customer account.
Buyer	A customer account.
Date/Time	The date/time the transaction was established. This may not be the date/time of the actual exchange of the item being sold.
Price	This numeric value represents the basic unit (share) price.
Security	A count of the number of shares and the identity (e.g., ticker) of the company.

4. Summary

We have presented a case for designing and implementing graph-based machine learning algorithms that include both Structure and Property Values. These examples suggest large datasets, perhaps from a streaming source, that will clearly require HPC techniques in order to provide insight into the data within a useful time.

There are many additional benefits to supporting a graph representation that allows multiple attributes on nodes and edges. The ability to learn using this type of data allows for the analysis of data from heterogeneous sources, which can either be analyzed in parallel or fused together into one graph. This is important when we want to, e.g., combine data from company annual reports with transaction data on the company stock, and possibly even information about the company appearing in news sources.

Acknowledgment

A part of this work was funded by the Center for Adaptive Super Computing Software - MultiThreaded Architectures (CASS-MT) at the U.S. Department of Energy's Pacific Northwest National Laboratory. Pacific Northwest National Laboratory is operated by Battelle Memorial Institute under Contract DE-ACO6-76RL01830.

References

[1] L. Euler, "Solutio problematis ad geometriam situs pertinentis," *Commentarii Academiae Scientiarum Imperialis Petropolitanae*, vol. 8, pp. 128–140, 1736.

[2] G. Li, M. Semerci, B. Yener, and M. J. Zaki, "Effective graph classification based on topological and label attributes," *Stat. Anal. Data Min.*, vol. 5, no. 4, pp. 265–283, Aug. 2012. [Online]. Available: http://dx.doi.org/10.1002/sam.11153

[3] M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis, "Frequent substructure-based approaches for classifying chemical compounds," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 17, no. 8, pp. 1036–1050, 2005.

[4] N. Shervashidze and K. Borgwardt, "Fast subtree kernels on graphs," *Advances in Neural Information Processing Systems*, vol. 22, pp. 1660–1668, 2009.

[5] S. Hido and H. Kashima, "A linear-time graph kernel," in *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*. IEEE, 2009, pp. 179–188.

[6] L. Holder, D. Cook, J. Coble, and M. Mukherjee, "Graph-based relational learning with application to security," *Fundamenta Informaticae*, vol. 66, no. 1-2, pp. 83–102, 2005.

[7] A. Silva, W. Meira, Jr., and M. J. Zaki, "Mining attribute-structure correlated patterns in large attributed graphs," *Proc. VLDB Endow.*, vol. 5, no. 5, pp. 466–477, Jan. 2012. [Online]. Available: http://dl.acm.org/citation.cfm?id=2140436.2140443

[8] L. McDowell and D. W. Aha, "Semi-supervised collective classification via hybrid label regularization," in *Proceedings of the 29th International Conference on Machine Learning, ICML*, 2012.

[9] W. Turkett Jr, E. Fulp, C. Lever, and E. Allan Jr, "Graph mining of motif profiles for computer network activity inference," in *Ninth Workshop on Mining and Learning with Graphs*, 2011.

[10] A. McGregor, "Graph mining on streams," *Encyclopedia of Database Systems*, pp. 1271–1275, 2009.

[11] A. Bifet, G. Holmes, B. Pfahringer, and R. Gavaldà, "Mining frequent closed graphs on evolving data streams," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 591–599.