A Machine Learning Approach for Predicting Hidden Links in Supply Chain with Graph Neural Networks

Edward Elson Kosasih[a,*], Alexandra Brintrup[a]

[a]*Institute for Manufacturing, University of Cambridge, United Kingdom*

Abstract

Supply chain business interruption has been identified as a key risk factor in recent years, with high impact disruptions due to disease outbreaks, logistic issues such as the recent Suez Canal blockage showing examples of how disruptions could propagate across complex emergent networks. Researchers have highlighted the importance of gaining visibility into procurement interdependencies between suppliers to develop more informed business contingency plans. However extant methods such as supplier surveys rely on the willingness or ability of suppliers to share data and are not easily verifiable. In this article, we pose the supply chain visibility problem as a link prediction problem from the field of Machine Learning, and propose the use of an automated method to detect potential links that are unknown to the buyer with Graph Neural Network (GNN). Using a real automotive network as a test case, we show that our method performs better than existing algorithms and provides a complimentary, AI based approach to improve supply chain visibility. Additionally, we use Integrated Gradient, a widely used technique in machine learning to improve the explainability of our approach by highlighting input features that influence GNN's decisions. We also discuss the advantages and limitations of using GNN for link prediction, outlining future research directions that can help improve supply chain visibility through the use of machine learning.

Keywords: Supply Chain, Supply Network, Resilience, Visibility, Link Prediction, Machine Learning, Data Analytics, Artificial Intelligence, Explainability

1. Introduction

Whilst outsourcing provides extensive cost benefits to manufacturers, the practice also creates emergent inter-dependencies between companies, which has increased in complexity and scale over the past two decades [16]. Numerous studies highlighted the need for improved supply chain visibility and how visibility can help reduce risks and increases responsiveness to disruptions ([50], [29]).

Several approaches have been proposed to improve visibility. These mainly fall into three categories: manufacturers questioning their first tier suppliers and enforcing partial control over their supply choices via approved vendor lists, subscription to third party databases, and deployment of product tracking technology such as RFID. The primary issue with these approaches is that they rely on the willingness and ability of companies to share information, which is often hindered due to confidentiality and a lack of incentive to disclose information. A secondary issue is that OEMs have no means of verifying supplier survey data, especially when changes occur in supply chain structure.

Researchers have recently experimented with machine learning (ML) technology to infer procurement relationships between companies. In the field of ML, this task is known as link prediction. Machine learning presents a desirable improvement upon existing supply chain visibility improvement methods as it mitigates companies' reliance on the explicit disclosure of supply chain members. [51] suggested the use of Natural Language processing for automatically extracting supply chain maps from world wide web. Whilst being able to collate previously unknown dependencies, this approach is dependent on the availability of natural language data that is publicly available. Further, the approach necessitates manual training of NLP classifiers for the extraction of text corpus that highlights supply chain relationships. [12] suggested a link prediction framework that may help infer dependencies using minimal amount of data that is not natural language based, but is topologically driven. Being so, the approach necessitates encoding of supply chain expertise for extracting relational features such as competition and cooperation, that would inform likely dependencies.

In this paper we contribute to the growing field of Supply Chain AI, and in particular, the improvement of supply chain visibility with AI. We do so by presenting an alternative supply chain link prediction method using Graph Neural Networks (GNN). GNN is a type of neural network particularly designed to extract information from graph data structures [24], such as supply chain. Using GNN, we could predict the existence of hidden links that are physically present yet missing in the dataset. This additional knowledge may help companies understand their supply chain dependencies better, hence increasing visibility into what risks they are exposed to. In this respect our research contributes to the wider literature on complex supply

*Corresponding author: eek31@cam.ac.uk

networks and resilience.

The use of GNN alleviates the reliance on manual training of algorithms, as it is fully automated. The proposed approach also circumvents the need for explicit encoding of supply chain expertise. Our results show that the GNN approach performs significantly better than extant supply network link prediction methods in three industrial case studies. We postulate that the use of automated Machine Learning methods such as GNN may help towards fulfilling supply chain visibility gaps in near future.

We also find that the use of GNN brings about an interpretability problem as often observed in the application of AI to solve industrial use cases. Hence, we additionally propose a mathematical approach to improve the interpretability of GNN so that supply chain executives who may use this tool can gain more understanding about the decision making process of the algorithms deployed.

This paper is organised as follows. Chapter 2 discusses existing literature in the field of Supply Chain visibility and the use of Machine learning. Chapter 3 explains how the supply chain visibility problem can be formulated as a link prediction problem task and how Graph Neural Network (GNN) could be used to tackle it. Chapter 4 elaborates on the mathematical model that can explain the GNN process, in order to improve the interpretability of the approach. Chapter 5 summarises our experimental results providing a comparative benchmark to [12]). Chapter 6 discusses how our work on link prediction could improve supply chain visibility. We also provide managerial insights, including limitations and extensions of our approach.

2. Literature Review

2.1. Supply Chain Visibility and Extant Approaches

Supply Chain Visibility has been defined in numerous ways in the literature [22]. In this work, we follow [5] who define Supply Chain Visibility as the extent to which supply chain actors possess and share key or useful information with each other. [51] noted that this definition includes an implicit knowledge of the supply network topology, that is the dependency structure between suppliers and buyers in the system, defined as *structural* supply chain visibility.

Studies have attributed the lack of supply chain visibility due to confidentiality and the difficulty in extracting and verifying an evolving, dynamic network [51] .

[7] highlighted that enhancing supply chain visibility is an essential capability that must be acquired. Supply chain managers are advised to focus on obtaining views into their supply network structure, if not in its entirety, at least partially. They argued that structural visibility is key component to communicate and manage risk and complexity in supply networks.

Understanding the structure/topology of the supply network is also critical to detect the presence of nexus suppliers [52]. These are companies who might not traditionally be defined as strategic actors as they are not necessarily top-tier suppliers and often have no direct impact to the firm's financial profits and risk. However, Nexus suppliers are critical due to their unique structural positions in interorganizational networks.

In order to increase visibility, supply chain mapping technologies have been proposed. [21] explained that supply chain maps should represent linkages and membership of companies in the network alongside information about the overall nature of the entire system.

While existing technology such as track-and-trace RFID helps to increase general visibility of information flow, it did not address the problem of structural visibility directly as it was not designed to discover the supply chain structure [51].

Researchers have recently started to investigate machine learning for supply chain mapping. [51] uses deep neural networks to extract supply chain maps from news articles. They showed that Natural Language Processing techniques could generate rudimentary maps. [42] uses neural networks to improve supply chain visibility, particularly focusing on predicting order fulfillment capacity of the network. [12] uses logistic regression and naive bayes classifier to infer supplier interdependencies using incomplete knowledge of the company's supply network.

In this work, we build on the work of [12], treating the link prediction problem as a subsebt of the general supply chain visibility challenge.

2.2. Supply Chain Visibility as a Link Prediction Problem

Supply chains can be seen as complex networks [15] that emerge out of companies buying from and selling goods and services to each other. Mathematically, one can model this network as a graph $G(V, E)$ where V is a set of companies and E is procurement relations between them.

While theoretically speaking there are $O(N^2)$ possible links in the graph, in practice a supply chain is topologically not fully connected [10]. This means that one can model the likelihood of a link (i.e. a relation) between a randomly selected pair of nodes as a probability problem. In machine learning, this task is referred to as the link prediction problem.

Various techniques have been proposed to tackle link prediction in domains other than supply chains. Some of the most popular techniques are based on computing similarity between each pair of nodes, which are mostly applicable to Social Networks, and Recommender systems in eCommerce or streaming services. Similarity could be based on different measures such as number of common neighbors or node degree. The similarity score serves as a proxy for likelihood of observing a link. Algorithms that fall under this class of methods include Jaccard Coefficient [36], Katz [30], LHN Index [35], Preferential Attachment [4], Adamic-Adar [1], Resource Allocation [56], path-based similarity [38] and Bayesian estimation [37]. Although previous research postulated that in supply chains similarity based methods may not be as relevant because two firms that are similar may be competing with one another, competition association between firms were then found to contain information that helped determine link existence. The authors attributed this to dual-sourcing arrangements.

Other methods rely on characterising the network topology (e.g. the presence of large hubs, hierarchy or communities) and using maximum likelihood, derive a set of probabilities that would have the highest probability of producing the given

network structure, which then helps detect missing links in the structure. A popular example in this category is the hierarchical random graph model proposed by [18]. Previous studies ([25], [46], [10], [11]), that examined supply chains have shown that building reliable statistics of network topology is difficult due to the lack of adequate empirical examples, making this approach less applicable to supply chains as a standalone method.

Other methods attempt to extract node features that are derived from topological structure. These node features can then be used to calculate the likelihood of links formed. These include Matrix factorisation [33], Stochastic Block Model [2], DeepWalk [41], LINE [44] and node2vec [23].

Application of link prediction in supply networks, however, has been rather scarce. A recent work called Supply Network Link Prediction (SNLP) [12], was proposed and applied on the global automotive supply chain network. The authors defined four handcrafted heuristics i.e. heuristics that enable the extraction of features which are thought to to extract information/embedding characterising a relation between each company and its potential supplier. These included: the number of existing suppliers, overlaps between both companies' product portfolios, product outsourcing associations and likelihood of having common buyers. The embedding is then fed into a binary classifier in order to predict the existence of link. The highest score achieved was an AUC score of 0.76 (where the maximum performance score is 1 and minimum score is 0).

While the SNLP model works well, it relies on domain experts to define the four aforementioned handcrafted heuristics. As we work with larger datasets with more attributes, manually defining such formulae might be expensive. In addition, while handcrafted heuristics might work well for a particular domain, they might not be directly transferrable to different contexts. For instance, [34] shows that while Common Neighbours (CN) heuristics work for a social network, it fails to perform in protein graph network. This is because in the latter, CN's assumption on homophily (similar nodes are connected) are violated. Similar issues have been observed in supply chains [12]. Therefore, further investigation on link prediction for supply network that does not rely on handcrafted heuristics is needed, which is our primary contribution in this work.

Automated alternatives to handcrafted heuristics has been studied widely in the wider link prediction literature [24]. Researchers have identified a class of models called GNN as a good candidate ([13], [19], [31], and [39]). GNN has been shown to outperform many existing methods in different networks such as airline networks, citation networks, political blogs, protein interactions,power grid, router-level internet and E. coli metabolites reactions ([53], [54], [55], [26], [45]).

Although GNN is a promising approach it has not been studied for inferring links in a supply network. In this work, we propose to investigate the efficacy of GNN for supply chain link prediction and compare our analysis with handcrafted heuristics using the same dataset that was previously analysed by [12].

Moreover, we also use techniques from the field of Explainable AI to interpret the GNN output. This is because, unlike handcrafted heuristics, GNN is a black box. Interpreting why certain decisions are made by the algorithm is not straightforward. The issue of interpretability has also been raised in the domain of Supply Chain Management AI ([6], [9]). As with any other neural network, this challenge is known as the explainability problem. A recent study [8] found that explainability has at least 20 multiple dimensions. In this work, we address interpretability, using attribution-based techniques.

The objective of attribution algorithms is akin to sensitivity analysis, which helps identify which features in the input dataset have the largest influence on a model's output. Researchers have studied various attribution techniques [32], such as Integrated Gradients, Saliency, DeepLift, DeepLiftShap, GradientShape, Input X Gradient, Guided Backprop, Guided GradCAM, Deconvolution, Feature Ablation, Occlusion, Feature Permutation, Shapley Value Sampling, Lime and KernelShap. Some of these are based on gradient methods, whilst others are pertubation based. In this work, we will focus on one of the most widely used gradient-based technique, named Integrated Gradients.

3. Link Prediction with Graph Neural Network

Given a graph $G(V, E)$ where V denotes the set of nodes and E links. We assume that there are hidden links that are not shown in E due to data incompleteness. Link prediction is defined as the task of predicting the existence of a link between two nodes $(u, v) \in V, (u, v) \notin E$. We assume that graph is undirected. In practice, supply chains are directed, but most link prediction approaches simplify this and convert it into an undirected graph. We stick to this paradigm as the directionality of the links may become obvious to the practitioner once the relationship is known. However, working on the original directed graph would be an interesting future extension.

Our proposed approach is based on SEAL [54], GraIL [45] and G-META [26]. The model is composed of three steps: (i) enclosing subgraph extraction, (ii) structural node labeling and (iii) subgraph scoring.

3.1. Enclosing Subgraph Extraction

Consider a pair of nodes (x, y) between which we would like to predict the existence of a link. We assume that the majority of information needed to infer a link lies within the neighborhood of the link under question, following studies that showed this premise to be generalisable property in many link prediction heuristics i.e. the γ-decaying theory by [54] and decaying property of node influence theorem by [26].

While the latter two theorems have shown that information lies within close neighborhood of a node, they do not specify the exact mechanism to extract this enclosing subgraph. This is in fact an open problem in the field that can benefit from future research. In this work, we follow standard approaches used in [54] and [26], that is to extract a m-hop ego graph surrounding the pair of nodes under consideration. Following previous experiments, we set m to be a small value, i.e. 1 (one-hop neighborhood).

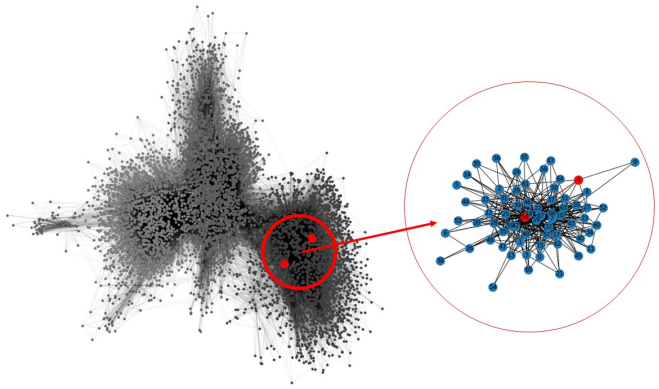

Figure 1: When predicting the existence of link for a given node pair, we first extract the subgraph $G(x, y)$ enclosing the two nodes (x, y). We use the ego graph of the two nodes as enclosing subgraph.

3.2. Structural Node Coloring

Upon obtaining the enclosing subgraph $G(x, y)$ of the node pair of interest, the next step is to capture its topological structure. The underlying hypothesis is that the subgraph that encloses an actual link is topologically different to the one surrounding pairs of disconnected nodes. Extracting information to test whether two graphs are identical is known in the literature as the graph isomorphism problem [24].

One commonly used approach here is to color the nodes within the enclosing subgraph such that it indicates the topological role of the respective node. We use a coloring algorithm that has been shown to work well in other networks called Double Radius Node Labeling / DRNL ([54]). Neighboring nodes are assigned a score based on their shortest path distances to the pair of nodes under consideration. The node's color is then equal to the hash of this distance. [54] uses the following hashing function.

$$f_l(i) = 1 + min(d_x, d_y) + \frac{d}{2}\left[\frac{d}{2} + d\%2 - 1\right] \qquad (1)$$

where i is a neighboring node, x and y are the pair of nodes under consideration, d_x and d_y are shortest paths between i and x, y respectively, and $d = d_x + d_y$.

After each node has been assigned a color, this is converted to a one-hot embedding vector (whose dimension represents color). One limitation of this method is that we assume that the maximum distance (or largest color) is known *a priori* i.e. as a hyperparameter. The embedding dimension size is set to this maximum distance. However, based on the information bound theory provided by ([54], [26]), it can be said that this maximum distance will be small compared to the diameter of the graph.

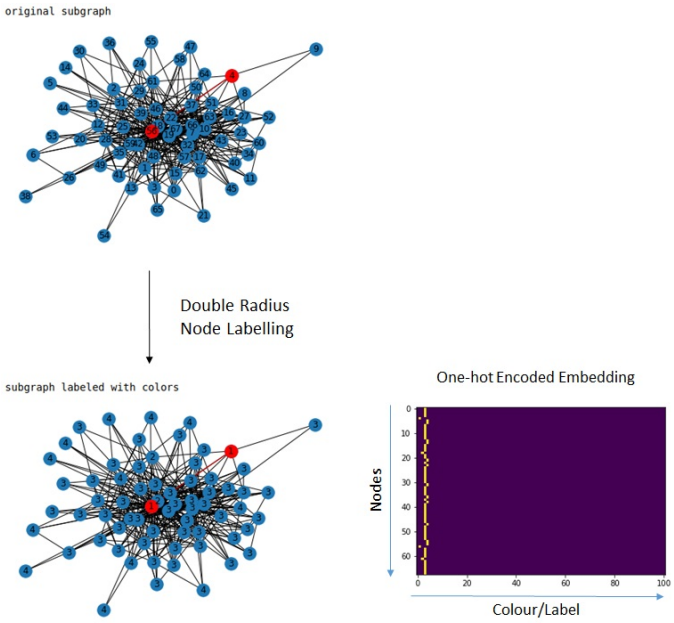

Figure 2: Nodes in the enclosing subgraph are relabeled/colored based on their double radius distance to the pair of nodes under consideration (shown in red). The colored subgraph is then represented as a one-hot encoded matrix.

3.3. Graph Neural Network

Arriving at this step, we now have a matrix of features whose values represent each node's topological role in the subgraph. Our next step is to convert this matrix into a single embedding vector that captures the topological structure of the network. We can then feed the given vector to a classifier to decide if there exists an link between the pair of nodes (x, y) that lie in the center of the given subgraph. This process can be formalised as a binary classification task.

In machine learning literature, the problem of converting this matrix of graph features to an embedding vector is known as the graph representation learning problem [24], which has traditionally been done with handcrafted methods such as graph statistics extraction, kernel methods, neighborhood overlap detection, graph laplacians and spectral methods. As mentioned earlier, the problem with handcrafted techniques is that they require manual designing based on domain expertise, and may not be easily transferable to a different domain. To mitigate this issue, researchers have been exploring alternative, learning-based approaches to extract heuristics in an automated way.

Graph Neural Network (GNN) has been shown as a promising candidate for automated extraction of heuristics. GNN has been shown to be analogous to several techniques that has been empirically shown to perform well in other domains ([24]). First, GNN performs local convolution in a manner that is similar to Convolutional Neural Network, which is the foundation of many state of the art models in various computer vision tasks. Second, GNN aggregates neighborhood information just like message-passing algorithm in probabilistic graphical models, such as Bayesian Network and Markov Random Fields. Lastly, GNN extracts features whose objective is to preserve topological information, akin to the graph isomorphism problem. In this

aspect, GNN was shown to be similar to the Weisfeiler-Lehman coloring algorithm ([24]). Empirical successes from these analogous models motivate us to investigate GNN specifically for supply networks.

The objective of GNN is to extract a new embedding for each node in the graph, such that it captures information from its neighborhood. In standard GNN, this is done in two stages: AGGREGATE and UPDATE. We first start with feature vector for each node, which could be obtained from given node attributes and/or topological coloring such as our Double Node Radius Label. Next, we AGGREGATE all neighbors' vectors. In practice, AGGREGATE is often a weighted sum function. Lastly, the aggregated neighborhood vector is combined with the current node embedding and passed through an UPDATE function (in practice this is a typical neural network activation function such as sigmoid, tanh or ReLU) to obtain a new embedding.

$$h_u^{k+1} = UPDATE^k(h_u^k, AGGREGATE^k(h_v^k, v \in N(u))) \quad (2)$$

where k is the kth layer of the GNN, u is a node in the graph, h_u is the embedding vector of node u and $N(u)$ are neighbors of u.

The framework that we use is flexible as it can work with different GNN models. In order to illustrate how it is used in practice, we implement a baseline model called Graph Convolution Network (GCN) [31] that has been widely used as a benchmark in the field of graph learning. In this model, we use sum weighted by node degrees as AGGREGATE and linear transformation W followed by sigmoid activation σ as UPDATE.

$$h_u^{k+1} = \sigma\left(W^k \sum_{v \in N(u) \cup \{u\}} \frac{h_v^k}{\sqrt{|N(u)||N(v)|}}\right) \quad (3)$$

After applying K layers of GNN, we will obtain h_u^K embedding for every node u in the subgraph. In this work, we use 3 layers of GNNs with node embedding dimension of 64. We choose 3 layers as previous studies have shown that GNN does not perform well with deeper layers due to oversmoothing effect [24], with 3-4 layers as being standard depth of a well-functioning GNN. Meanwhile, we pick 64 node embedding dimension following recent graph neural network implementations ([49] and [48]).

The next task is to combine these into a graph level embedding. Similar to CNN, we could perform POOLING on all nodes v in the given enclosing subgraph $G(x, y)$ to obtain a graph level representation $h_{G(x,y)}$.

$$h_{G(x,y)} = POOLING\left(h_v^K, v \in G(x, y)\right) \quad (4)$$

In this work, we use an average as the POOLING function as shown below. There are different pooling mechanisms, such as sum, max, min or set2set. Depending on the complexity of the graph, some might preserve more isomorphism information in the graph embedding than others. However, in this work, we pick average since it is a standard pooling that is used with the GCN baseline model that we implement. We represent the number of nodes in subgraph $G(x, y)$ as $|V_{G(x,y)}|$.

$$h_{G(x,y)} = \frac{1}{|V_{G(x,y)}|} \sum_{v \in G(x,y)} h_v^K \quad (5)$$

We have now converted a graph of arbitrary size to an embedding vector $h_{G(x,y)}$ with fixed-size dimension. This embedding could be used for various purposes in graph learning. In our specific task of link prediction, we can feed this into a binary classifier to decide whether there is an link in the original pair of node (x, y) that is enclosed by the subgraph $G(x, y)$.

In order to make our model end-to-end trainable by backpropagation, we use two layers of fully connected neural network as the binary classifier. We implement our GNN using a widely used software package called pytorch geometric [20]. Experiments were performed on a Dell laptop with Intel i9-9980HK 2.4 GHz CPU (16 GB RAM), and NVIDIA GeForce GTX1650 GPU (4 GB RAM).

4. Explainability

In the previous section, we have shown how GNN can be used as an automated alternative to handcrafted heuristics for supply chain link prediction. Unfortunately, unlike their handcrafted counterparts, GNN is a black box made of weight matrices that cannot be directly interpreted. It is thus difficult to explain why GNN decides that a particular enclosing subgraph corresponds to the existence of a link, while another does not. This issue is prevalent across all machine learning tasks that use neural network based approaches, and is often referred to as the problem of Explainability [8].

In this work, we focus on a particular aspect of Explainability called Interpretability [8]. We choose interpretability instead of the more generic explainability term as it is more quantitatively defined. In particular, we investigate a well-defined task of interpretability called sensitivity analysis. The objective is to see how much the output decision changes if the input is varied. The higher the output sensitivity for a given input change, the more important is that particular change.

In neural networks, sensitivity analysis is also often referred to as an attribution method. In this work, we focus on a widely used gradient-based attribution method called Integrated Gradients / IG [43]. We implemented the code in Captum [32]. Given a neural network F and input $X = [x_1 \dots x_N]$ to be explained, IG will feed in m different versions of the modified input (which are combinations of original X and a chosen neutral baseline $X^{'}$ e.g. vector of all zeros), and calculate the total output changes. IG then attributes the output changes with each respective feature dimension x_i via gradient. Total gradient of the m changes attributed to feature x_i serves as a number that represents the importance of that dimension.

$$\text{IG}_i = (x_i - x_i^{'}) \cdot \sum_{k=1}^{m} \frac{\delta F(X' + \frac{k}{m} \cdot (X - X'))}{\delta x_i} \cdot \frac{1}{m} \quad (6)$$

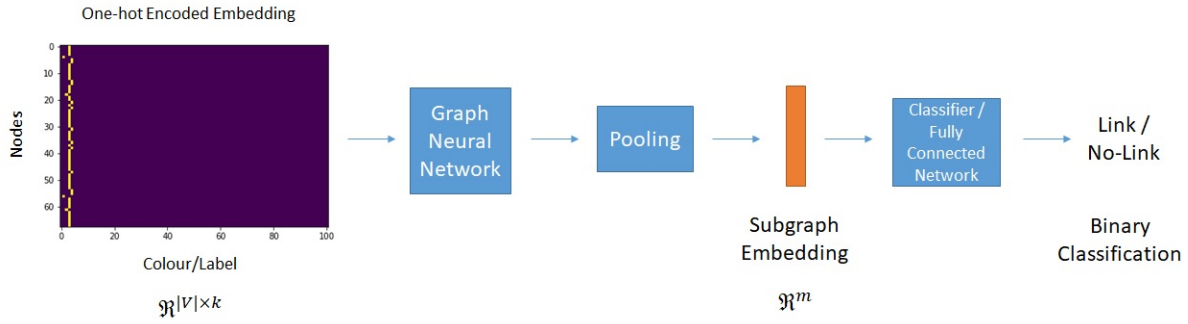

Figure 3: The overall Graph Neural Network pipeline is shown above. Feature Matrix obtained from Structural Node Embedding is fed into a GNN and Pooled into a fixed size embedding vector. This could then be used together with a binary classifier to perform link prediction

where $x^{`}$ is the chosen neutral baseline, $X' + \frac{k}{m} \cdot (X - X')$ is the combined modified input (in this case it is linear interpolation), and $\frac{\delta F(X' + \frac{k}{m} \cdot (X - X'))}{\delta x_i}$ is the gradient of output (given modified input) with respect to i_{th} feature.

In this work, X are the links in the enclosing subgraph. The objective is to find links that produce sensitive changes in the output when they are modified. This serves as a proxy to find out paths in the subgraph that the GNN consider important when making a decision.

5. Experimental Results

To illustrate how GNN can be used for link prediction in a supply chain, we perform an experiment on a real automotive dataset from a third party database provider, called Marklines. This dataset has been used in the previous link prediction study from [12]. This network has 18750 firms and 89175 procurement relationships from companies around the world (see Figure 4). The nodes are colored based on the community that they belong to, using Clauset-Newman-Moore greedy modularity maximization community detection algorithm. Following general link prediction approaches, we first convert the network into an undirected graph.

Figure 4: Automotive Network Visualisation.

In order to construct our experiment dataset, for every company we need to obtain at least 2 edges, 1 for training and 1 for testing set. Moreover, we need to ensure that every edge is only used once across training and testing set. Given this requirement, we end up with 8663 firm nodes and 68812 edges out of the original 18750 firms and 89175 procurement relationships. The large decrease of the number of nodes is attributed to the long-tail degree distribution of the network as seen in Figure 5 where many companies have less than or equal to 1 degree. Nevertheless, we still retain most of the edges in the original network.

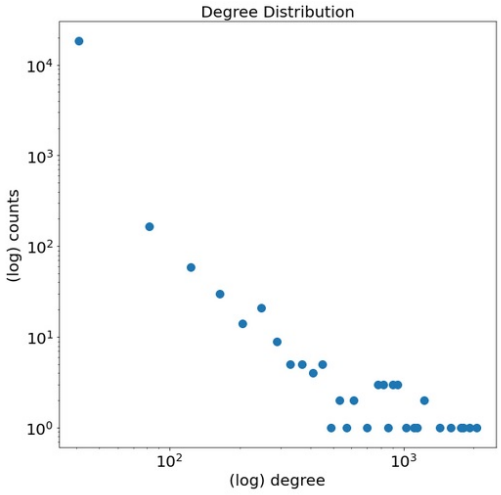

Figure 5: Degree Distribution / Number of Neighbors per Companies

Given that we are dealing with a link prediction task, we need to obtain both positive and negative samples. The former is a set of edges, while the latter is non-edges. For positive samples, we equally split the 68812 edges into training and testing set (each 34406). Meanwhile, for negative samples, we randomly sampled non-edges from the graph, and experimented with different data sizes to test how robust are the algorithms with respect to various negative/positive data ratio (from balanced 1:1 to imbalanced 1:4). Negative sampling is done by picking random non-edges for each of the 8663 company node. The number of such samples are equal to the given nega-

tive/positive ratio multiplied by the number of positive samples for each respective company node. Table 1 shows our experimental setup.

Table 1: Training and Testing Data

	Links	Non Links	Ratio
Training	34406	34406	1:1
Testing	34406	34406 / 68788 / 103254 / 137668	1:1 / 2 / 3 / 4

Five algorithms are compared: our proposed GNN method, SNLP [12], and three commonly used link prediction heuristics i.e. Preferential Attachment (PA), Resource Allocation index (RA) and Jaccard Coefficient (JC) as benchmarks. The algorithms are evaluated using three metrics. Two of them are commonly used metrics for link prediction: Area Under Receiver Operating Characteristic Curve (AUC) and Average Precision (AP) [54]. The other one, f1-score is included for the sake of completeness as this is often used for normal classification task. We measure the performance of the models of each negative/positive test ratio and visualise the result in the plots shown on Figure 6, 7 and 8. Each experiment is repeated three times from different random initial conditions, and the 90% confidence interval (CI) are shown on the same figures. Note that the CI is quite tight for most of the results as the performance scores are quite stable (hence the shaded results are not too obvious on the figures), except for GNN at test ratio of 1:4 where there are more variations.

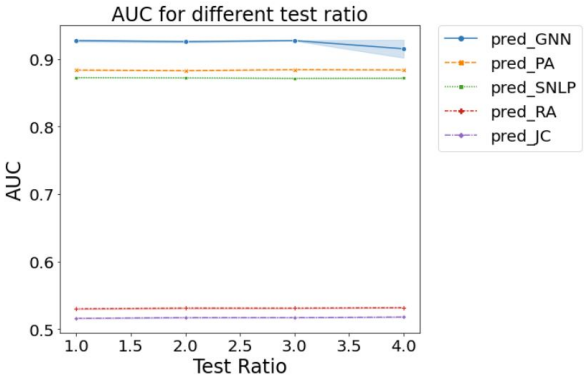

Figure 6: Area Under Curve across different test ratio

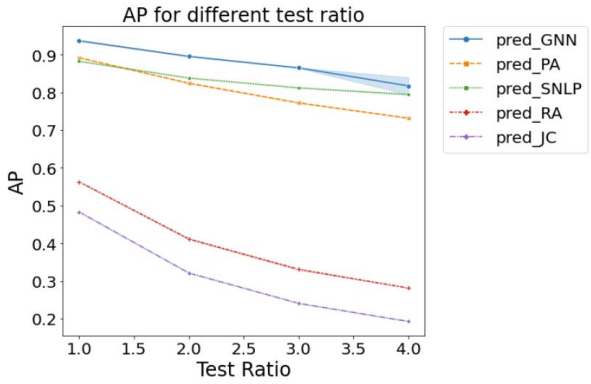

Figure 7: Average Precision across different test ratio

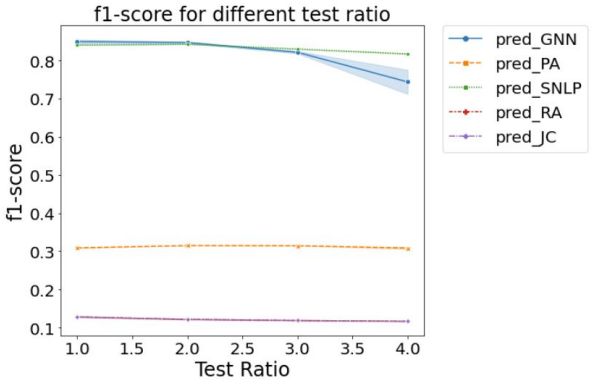

Figure 8: f1-Score across different test ratio

Figure 6 and 7 show that GNN outperforms all of the other algorithms in terms of AUC and AP, across different test ratio. AP significantly decreases as negative/positive ratio increases as generally expected, with more negative samples the precision rate goes down. Meanwhile, Figure 8 shows that GNN outperforms the others (slightly better than SNLP) for more balanced dataset i.e. ratio close to 1. On the other hand for a more imbalanced dataset i.e. larger ratio, the baseline SNLP works better. Next, we use Integrated Gradient to perform sensitivity analysis and see which input links in a given enclosing subgraph affects the GNN output decision the most, by randomly sampling a pair of nodes that are predicted to have links (GNN score close to 1) and those without (GNN score close to 0). Figure 9 shows four such samples. Thickness of the links represent higher importance as calculated by IG.

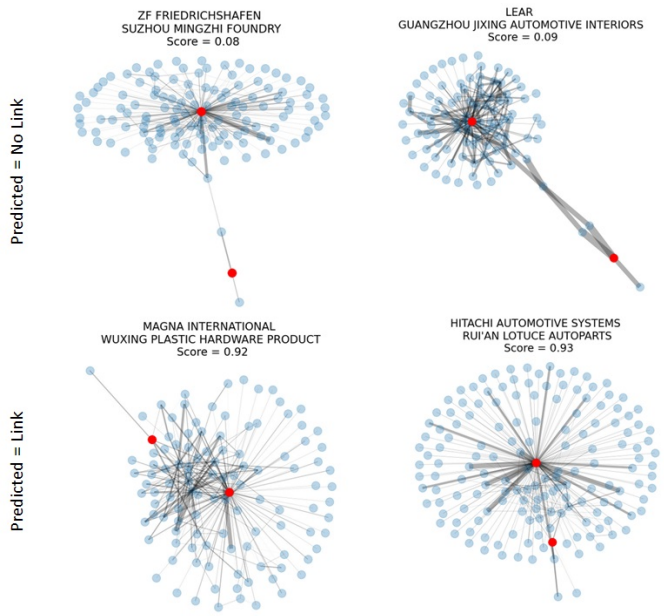

Figure 9: Selected pair of nodes is colored in red. Thickness of the links correspond to importance attributed by the IG model for a given decision.

Figure 9 shows the links that IG deems as being more relevant than the others in making predictions. We can see that GNN captures the importance of links lying in the paths between the two selected pair of nodes. It seems that the further apart the two nodes are, the more likely it is for the GNN to infer that there are no direct links formed between the two. This seems to be fairly aligned with our traditional handcrafted notions of closeness and communities. Path distance, however, seems to not be the only pattern that the GNN sees as there are other links within the vicinity of the nodes that are not directly within the paths connecting the two nodes. This shows that there are remaining mysteries that IG alone could not explain. We recommend researchers to explore other explainability models [32].

6. Discussion and Conclusions

Today, complex, international supply chains underpin the survival of much of our economy, supporting critical sectors such as food, defence, aerospace and automotive. Both high-impact-low-frequency and Low-impact-high-frequency events that cause business interruption are on the rise due to increased complexity and interdependence. In a recent 2021 survey, GEP reported that the total SN disruption cost in 2020 due to COVID-19 was 4 trillion dollars [17]. Supply chain business interruption has been identified as the top business risk factor in 2021 (Allianz), and has been prioritised in numerous government briefs including the UK's Industrial Strategy paper which set out plans to identify where investment and the development of technology can accelerate growth and resilience in the value chain.

As organisations outsource production to one another they create economies-of-scale and reduce prices but also increase risk of disruption cascades (ripple effects) if any member of the chain fails. While Supply chain resilience is a long studied topic, only in the past decade researchers have emphasised the relationship between network topology and the ripple effect [14], highlighting the need to identify " Nexus" firms that are beyond the periphery of the focal organisation but are potentially instrumental to instigating ripple effects [52]. There exists a large body of literature investigating the relationship between the robustness and resilience of SNs and the topology with which they are structured (see [10] for a recent review), digital tools that may help in modelling potential disruption cascades ([28], [27], [3]).

However, before an exploration of topology can be made, a firm would need to know which firms are members of its extended network and how it is connected to them. Supply chain visibility is defined as 'the extent to which actors within the supply chain have access to or share timely information about supply chain operations, other actors and management which they consider as being key or useful to their operations' [5]. Whilst the term encompasses a large body of literature including timeliness, quality and completeness of supply chain related data, we are concerned with the knowledge of the topology of the supply network; i.e. knowledge of the actors and the network of their dependencies, so as to ascertain "hidden" relations that may inform risk management.

To this extent, recently a number of data driven methods have been proposed to extract and codify supply chain knowledge ([51]) and predict missing information ([12]). Whilst the former can extract only publicly available information, the latter focuses on creating predicting supply chain links by designing features that may inform link formation such as competition and product dependencies. Whilst a successful method in itself, designing such features may become cumbersome as importance of features may vary in different industrial sectors. Thus, automated approaches may become more favourable in settings where informative features are unknown.

Hence, in our work we extended this particular body of work by developing an automated, GNN based approach that predicts links that are invisible in a network, but are likely to exist; thereby improving practitioners' completeness of knowledge. Upon comparison with [12]'s approach, this work has shown that as an automated alternative to, GNN performs better and more consistently in different supply networks and thus can be posed as a more generalisable alternative.

Moreover, our model is suitable for supply chain practitioners who only have limited local information about their companies' suppliers and buyers (up to tier-N). During inference, the GNN only requires information from the local enclosing subgraph, which is aligned with the local information constraints that one has to work with.

From a practitioner perspective, our work can inform supply network re-design efforts, where high risk dependencies are detected through scenario-planning, and mitigated through strategies such as multi-sourcing, inventory build up, increased resource redundancy, re-configuration or expediting. The selection of a subsequent strategy would then depend on domain specific constraints, and cost-benefit analyses prevalent in the large body of supply chain resilience literature.

Our proposed approach also has a number of limitations. Despite the utilisation of interpretability methods, GNN still is a black box and thus interpreting the GNN's decision rules are not straightforward. Emerging explainability research such as Integrated Gradients algorithm does help uncover some parts of the model's decision process. However, as shown in our experiments, IG could not verify nor explain the whole story behind the algorithm's decision making process. Unfortunately, this is a general problem across all machine learning tasks and further research is needed.

There are several possible extensions to this work. From a methodical perspective, we need to compare the usage of directed GNN for this task e.g. [47] as opposed to the undirected path taken in this work. Doing so would inevitably increase data imbalance, but also potentially be more informative. Second, the subgraph extraction model that we used selects neighboring nodes within a fixed distance threshold from the company under consideration (distance here is defined in terms of graph which might not correspond to physical distance). This does not allow long-range dependency to certain nodes that are distant. One may consider exploring other techniques that might extract better neighborhood, perhaps taking motifs or communities into account. Third, the Double Radius Node Labelling algorithm that we used was based on shortest paths. One could borrow emerging coloring techniques from the field of graph isomorphism to improve the topological feature extraction. Fourth, we could investigate uncertainty modeling for the model. The proposed GNN model currently outputs a point prediction without any confidence level, which is a component that is often desired by practitioners. Here one can explore other recent GNN techniques such as [40] based for example on Gaussian Process. Fifth, if one has more information such as material dependencies or financial transactions between firms, one can create a multi-link knowledge graph ([45]), and this additional information would benefit the training process of the GNN. Lastly, advances in explainable AI would benefit our model as we can explore other techniques to complement IG in interpreting the black box GNN.

References

[1] Adamic, L. A. and Adar, E. (2003). Friends and neighbors on the Web. *Social Networks*, 25(3):211–230.

[2] Airoldi, E. M., Blei, D. M., Fienberg, S. E., and Xing, E. P. (2008). Mixed Membership Stochastic Blockmodels. *Journal of machine learning research : JMLR*, 9:1981–2014.

[3] Aldrighetti, R., Battini, D., Ivanov, D., and Zennaro, I. (2021). Costs of resilience and disruptions in supply chain network design models: A review and future research directions. *International Journal of Production Economics*, 235:108103.

[4] Barabási, A.-L. and Albert, R. (1999). Emergence of Scaling in Random Networks. *Science*, 286(5439):509–512.

[5] Barratt, M. and Oke, A. (2007). Antecedents of supply chain visibility in retail supply chains: A resource-based theory perspective. *Journal of Operations Management*, 25(6):1217–1233.

[6] Baryannis, G., Validi, S., Dani, S., and Antoniou, G. (2019). Supply chain risk management and artificial intelligence: state of the art and future research directions. *International Journal of Production Research*, 57(7):2179–2202. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00207543.2018.1530476.

[7] Basole, R. C. and Bellamy, M. A. (2014). Supply Network Structure, Visibility, and Risk Diffusion: A Computational Approach. *Decision Sciences*, 45(4):753–789. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/deci.12099.

[8] Brennen, A. (2020). What Do People Really Want When They Say They Want "Explainable AI?" We Asked 60 Stakeholders. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI EA '20, pages 1–7, New York, NY, USA. Association for Computing Machinery.

[9] Brintrup, A. (2020). Artificial Intelligence in the Supply Chain. ISBN: 9780190066727.

[10] Brintrup, A., Ledwoch, A., and Barros, J. (2016). Topological robustness of the global automotive industry. *Logistics Research*, 9(1):1.

[11] Brintrup, A., Wang, Y., and Tiwari, A. (2017). Supply Networks as Complex Systems: A Network-Science-Based Characterization. *IEEE Systems Journal*, 11(4):2170–2181. Conference Name: IEEE Systems Journal.

[12] Brintrup, A., Wichmann, P., Woodall, P., McFarlane, D., Nicks, E., and Krechel, W. (2018). Predicting Hidden Links in Supply Networks. *Complexity*, 2018:1–12.

[13] Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. (2014). Spectral Networks and Locally Connected Networks on Graphs. *arXiv:1312.6203 [cs]*. arXiv: 1312.6203.

[14] Chauhan, V. K., Perera, S., and Brintrup, A. (2021). The relationship between nested patterns and the ripple effect in complex supply networks. *International Journal of Production Research*, 59(1):325–341. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00207543.2020.1831096.

[15] Choi, T. Y., Dooley, K. J., and Rungtusanatham, M. (2001). Supply networks and complex adaptive systems: control versus emergence. *Journal of Operations Management*, 19(3):351–366.

[16] Christopher, M. and Holweg, M. (2011). "Supply Chain 2.0": managing supply chains in the era of turbulence. *International Journal of Physical Distribution & Logistics Management*, 41(1):63–82. Publisher: Emerald Group Publishing Limited.

[17] CIPS (2021). Total cost of supply chain disruption in 2020 'was $4tn'. https://www.cips.org/supply-management/news/2021/march/total-cost-of-supply-chain-disruption-in-2020-was-4tn/.

[18] Clauset, A., Moore, C., and Newman, M. E. J. (2008). Hierarchical structure and the prediction of missing links in networks. *Nature*, 453(7191):98–101. Number: 7191 Publisher: Nature Publishing Group.

[19] Duvenaud, D., Maclaurin, D., Aguilera-Iparraguirre, J., Gómez-Bombarelli, R., Hirzel, T., Aspuru-Guzik, A., and Adams, R. P. (2015). Convolutional Networks on Graphs for Learning Molecular Fingerprints. *arXiv:1509.09292 [cs, stat]*. arXiv: 1509.09292.

[20] Fey, M. and Lenssen, J. E. (2019). Fast Graph Representation Learning with PyTorch Geometric. *arXiv:1903.02428 [cs, stat]*. arXiv: 1903.02428.

[21] Gardner, J. T. and Cooper, M. C. (2003). Strategic Supply Chain Mapping Approaches. *Journal of Business Logistics*, 24(2):37–64. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/j.2158-1592.2003.tb00045.x.

[22] Goh, M., Souza, R. D., Zhang, A. N., Wei He, and Tan, P. S. (2009). Supply Chain Visibility: A decision making perspective. In *2009 4th IEEE Conference on Industrial Electronics and Applications*, pages 2546–2551. ISSN: 2158-2297.

[23] Grover, A. and Leskovec, J. (2016). node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 855–864, San Francisco California USA. ACM.

[24] Hamilton, W. L. (2020). Graph Representation Learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3):1–159.

[25] Hearnshaw, E. J. and Wilson, M. M. (2013). A complex network approach to supply chain network theory. *International Journal of Operations & Production Management*, 33(4):442–469. Publisher: Emerald Group Publishing Limited.

[26] Huang, K. and Zitnik, M. (2021). Graph Meta Learning via Local Subgraphs. *arXiv:2006.07889 [cs, stat]*. arXiv: 2006.07889.

[27] Ivanov, D. and Dolgui, A. (2021a). A digital supply chain twin for managing the disruption risks and resilience in the era of Industry 4.0. *Production Planning & Control*, 32(9):775–788. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/09537287.2020.1768450.

[28] Ivanov, D. and Dolgui, A. (2021b). Stress testing supply chains and creating viable ecosystems. *Operations Management Research*.

[29] Ivanov, D., Dolgui, A., Das, A., and Sokolov, B. (2019). Digital Supply Chain Twins: Managing the Ripple Effect, Resilience, and Disruption Risks by Data-Driven Optimization, Simulation, and Visibility. In Ivanov, D., Dolgui, A., and Sokolov, B., editors, *Handbook of Ripple Effects in the Supply Chain*, International Series in Operations Research & Management Science, pages 309–332. Springer International Publishing, Cham.

[30] Katz, L. (1953). A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43.

[31] Kipf, T. N. and Welling, M. (2017). Semi-Supervised Classification with Graph Convolutional Networks. *arXiv:1609.02907 [cs, stat]*. arXiv: 1609.02907.

[32] Kokhlikyan, N., Miglani, V., Martin, M., Wang, E., Alsallakh, B., Reynolds, J., Melnikov, A., Kliushkina, N., Araya, C., Yan, S., and Reblitz-Richardson, O. (2020). Captum: A unified and generic model interpretability library for PyTorch. *arXiv:2009.07896 [cs, stat]*. arXiv: 2009.07896.

[33] Koren, Y., Bell, R., and Volinsky, C. (2009). Matrix Factorization Techniques for Recommender Systems. *Computer*, 42(8):30–37. Conference Name: Computer.

[34] Kovács, I. A., Luck, K., Spirohn, K., Wang, Y., Pollis, C., Schlabach, S., Bian, W., Kim, D.-K., Kishore, N., Hao, T., Calderwood, M. A., Vidal, M., and Barabási, A.-L. (2019). Network-based prediction of protein interactions. *Nature Communications*, 10(1):1240. Number: 1 Publisher: Nature Publishing Group.

[35] Leicht, E. A., Holme, P., and Newman, M. E. J. (2006). Vertex similarity in networks. *Physical Review E*, 73(2):026120.

[36] Liben-Nowell, D. and Kleinberg, J. (2007). The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7):1019–1031. _eprint: https://asistdl.onlinelibrary.wiley.com/doi/pdf/10.1002/asi.20591.

[37] Liu, Z., Zhang, Q.-M., Lü, L., and Zhou, T. (2011). Link prediction in complex networks: A local naïve Bayes model. *EPL (Europhysics Letters)*, 96(4):48007.

[38] Lü, L., Jin, C.-H., and Zhou, T. (2009). Similarity index based on local paths for link prediction of complex networks. *Physical Review E*, 80(4):046122.

[39] Niepert, M., Ahmed, M., and Kutzkov, K. (2016). Learning Convolutional Neural Networks for Graphs. In *International Conference on Machine Learning*, pages 2014–2023. PMLR. ISSN: 1938-7228.

[40] Opolka, F. L. and Liò, P. (2020). Graph Convolutional Gaussian Processes For Link Prediction. *arXiv:2002.04337 [cs, stat]*. arXiv: 2002.04337.

[41] Perozzi, B., Al-Rfou, R., and Skiena, S. (2014). DeepWalk: online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, New York New York USA. ACM.

[42] Silva, N., Ferreira, L. M. D. F., Silva, C., Magalhães, V., and Neto, P. (2017). Improving Supply Chain Visibility With Artificial Neural Networks. *Procedia Manufacturing*, 11:2083–2090.

[43] Sundararajan, M., Taly, A., and Yan, Q. (2017). Axiomatic Attribution for Deep Networks. *arXiv:1703.01365 [cs]*. arXiv: 1703.01365.

[44] Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. (2015). LINE: Large-scale Information Network Embedding. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1067–1077, Florence Italy. International World Wide Web Conferences Steering Committee.

[45] Teru, K. K., Denis, E., and Hamilton, W. L. (2020). Inductive Relation Prediction by Subgraph Reasoning. *arXiv:1911.06962 [cs, stat]*. arXiv: 1911.06962 version: 2.

[46] Thadakamaila HP, Raghavan, U. N., Kumara, S., and Albert, R. (2004). Survivability of multiagent-based supply networks: a topological perspect. *IEEE Intelligent Systems*, 19(5):24–31. Conference Name: IEEE Intelligent Systems.

[47] Thost, V. and Chen, J. (2021). Directed Acyclic Graph Neural Networks. *arXiv:2101.07965 [cs]*. arXiv: 2101.07965.

[48] Tsai, Y.-C., Guan, M., Li, C.-T., Cha, M., Li, Y., and Wang, Y. (2019). Predicting New Adopters via Socially-Aware Neural Graph Collaborative Filtering. In Tagarelli, A. and Tong, H., editors, *Computational Data and Social Networks*, Lecture Notes in Computer Science, pages 155–162, Cham. Springer International Publishing.

[49] Wang, X., He, X., Wang, M., Feng, F., and Chua, T.-S. (2019). Neural Graph Collaborative Filtering. *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 165–174. arXiv: 1905.08108.

[50] Wei, H.-L. and Wang, E. T. G. (2010). The strategic value of supply chain visibility: increasing the ability to reconfigure. *European Journal of Information Systems*, 19(2):238–249. Publisher: Taylor & Francis _eprint: https://doi.org/10.1057/ejis.2010.10.

[51] Wichmann, P., Brintrup, A., Baker, S., Woodall, P., and McFarlane, D. (2020). Extracting supply chain maps from news articles using deep neural networks. *Int. J. Prod. Res.*, 58(17):5320–5336.

[52] Yan, T., Choi, T. Y., Kim, Y., and Yang, Y. (2015). A Theory of the Nexus Supplier: A Critical Supplier From A Network Perspective. *Journal of Supply Chain Management*, 51(1):52–66. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/jscm.12070.

[53] Zhang, M. and Chen, Y. (2017). Weisfeiler-Lehman Neural Machine for Link Prediction. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 575–583, Halifax NS Canada. ACM.

[54] Zhang, M. and Chen, Y. (2018). Link Prediction Based on Graph Neural Networks. *arXiv:1802.09691 [cs, stat]*. arXiv: 1802.09691.

[55] Zhang, M., Li, P., Xia, Y., Wang, K., and Jin, L. (2020). Revisiting Graph Neural Networks for Link Prediction. *arXiv:2010.16103 [cs]*. arXiv: 2010.16103.

[56] Zhou, T., Lü, L., and Zhang, Y.-C. (2009). Predicting missing links via local information. *The European Physical Journal B*, 71(4):623–630.