QUBAHAN ACADEMIC JOURNAL
Publication Date: 21 Jan 2024
https://doi.org/10.58429/qaj.v4n1a333

Developing a Method for Building Business Process Models Based on Graph Neural Networks in the Absence of Task Identifier Data

Oleg Kazakov* [1], Natalya Azarenko [2], Irina Kozlova [3]
[1] Bryansk State University of Engineering and Technology, Russia;
[2] Bryansk State University of Engineering and Technology, Russia;
[3] Bryansk State University of Engineering and Technology, Russia;

Corresponding author: (e-mail: it.kazakov@yandex.ru).

ABSTRACT The contemporary methodology of business process modeling is closely tied to process mining. The aim of the study is to develop a method of creating business process models through the restoration of links between events recorded in logs in the absence of CaseID data based on graph neural networks. The problem is solved by applying the graph convolutional networks architecture. The study employs a combination of a weighted adjacency matrix and an adjacency matric accounting for the graph data structure. Textual information about the tasks involved in the business process is considered when implementing the feature matrix using embeddings. The Navec embedding model is chosen to represent task titles as numerical vectors. The study was based on parsing the technological log of the 1C:Enterprise system. The obtained solutions make it possible to restore the required connection (Sequence flow) in the model of the "Approval of a commercial offer" business process in the absence of Case ID data in the event log as part of the "Reset request" task.

Keywords: Business Process Modeling, Graph Convolutional Networks, Process Mining, Restoring Graph Connections.

I. INTRODUCTION

Business process modeling assumes the creation of a graphic representation of the business process to understand its structure, the interaction of its participants, and the sequence of acts [1,2]. This method allows visualizing the processes, discovering their weaknesses, optimizing the order of tasks, and improving the standards of works [3-7].

Today's business process modeling methodology is closely tied to process mining. In the paper titled "Conformance Checking of Processes Based on Monitoring Real Behavior", A. Rozinat and W.M.P. van der Aalst [8] examine the alignment of business processes with observed behavior. The authors present methods and algorithms for analyzing log data and checking the correspondence of the process model to actual performance. Process mining is iterative and generally consists of the following stages [9]:

1. Data preparation: collection of the necessary data on the business process. The sources may include activity registers, logs, databases, etc.
2. Data extraction: the data are processed and converted into a structured format to carry out further analysis. This stage may require the use of text processing or machine learning methods.
3. Process modeling: creation of business process models based on the obtained data. These models can come in the form of graphic representations of performance flow, Petri networks, and other formal languages [10].
4. Process analysis: the data are explored and analyzed to determine the features of process execution. This stage may employ statistical analysis methods, machine learning, and data visualization.
5. Result visualization: the obtained results and insights are presented to the user in an understandable and clear fashion. This can take the form of charts, graphs, or other visual elements.
6. Process optimization: relying on the data analysis and the discovered problem areas, process improvements and optimizations are proposed. These recommended modifications may include changes in process organization, automation, or redistribution of tasks.

In their paper "Process Mining with the Heuristics Miner Algorithm", A.J.M.M. Weijters, J.T. Ribeiroand, and W.M.P. van der Aalst [11] describe how the Heuristics Miner algorithm is used to extract process models

from log data. Process mining relies on data from the event log, which in its minimal composition should include time stamps, the identifier of the case or the process of restoring a connection in the graph neural network (GNN), and event title (Event).

In practice, CaseID data can seldom be found in logs. In this case, process mining may face certain limitations and challenges. CaseID typically presents a unique identifier for each process case, linking the events and data relevant to a particular case.

The following classic methods are helpful in case of lacking CaseID data:

1. Heuristic methods, which can be used to identify relationships between events even without an explicit CaseID. For example, one can look for similar characteristics between events, such as time intervals, distances between points, or frequency of occurrence.
2. Clustering techniques to group similar events together. This can help in identifying patterns and sequences of actions that can be treated as separate cases.
3. Using contextual data: analyzing additional data available along with process data to grasp the context of events, including location, transaction type, etc. In the article "Automated Discovery of Process Models from Knowledge-Intensive Processes", C. Di Francescomarino et al. [12] explore process modeling in the framework of knowledge-intensive processes. The researchers offer a method for discovering process models based on analyzing structured knowledge.

Since the efficiency of the described methods as applied to automated process discovery is rather low, we attempted to implement graph convolutional networks (GCN) to restore connections between events.

GCNs are a method of deep learning on graphs that uses convolution operations to distribute information across the graph and extract node features. The study "Graph convolutional networks" [13] demonstrates that GCNs can efficiently model graph structures and use node features for the purposes of classification and regression on graphs. In the paper "Semi-Supervised Classification with Graph convolutional networks" [14], which is an extension of the previously cited study, the CGN algorithm is proposed to classify graph nodes. The authors consider node classification when labels are available for only a small fraction of nodes. They propose a method that combines information about the node's neighbors for more accurate classification. The paper "Graph Neural Networks: A Review of Methods and Applications" [15] presents an overview of the methods and applications of GNNs. The authors cover various architectures and GNN training techniques and explore their applications in various fields, such as social networks, bioinformatics, and recommender systems.

The study "Graph Neural Networks for Recommender Systems" [16] investigates the use of GNNs in recommender systems. The authors propose a method that uses graph representations for modeling relationships between users, items, and their properties. This approach can be used to improve the accuracy and personalization of recommendations.

The purpose of the present study is to develop a method of building business process models through the recovery of connections between logged events in the absence of CaseID data based on GNNs.

II. MATERIALS AND METHODS

In a general case, the problem statement for restoring connections on a GNN can be as follows:

A graph is given, oriented G = (V, E), where V is the set of nodes and E is the set of edges of the graph. Each node $v \in V$ is assigned a feature vector that describes its properties. It is also known that some pairs of vertices have connections (edges) in the graph. The task is to predict missing links or to predict the probability of the existence of a link between pairs of vertices based on the available information about vertex properties and relying on the existing links.

There are several approaches to solving this problem. Most of them involve the construction of an adjacency matrix. The adjacency matrix A is a binary matrix of the size of $|V| \times |V|$, where A[i][j] = 1, if the nodes i and j have an edge between them and, otherwise, A[i][j] = 0. For a directed graph, the adjacency matrix can be asymmetric. The value A [i, j] will indicate the presence of a directed edge from vertex i to vertex j. In this case, A [j, i] can equal 1 or 0 depending on the presence or absence of a directed edge from vertex j to vertex i. If a graph contains loops (edges connecting a vertex to itself), the corresponding elements of the adjacency matrix A [i, i] will have the value 1 for an undirected graph and may have any value (1 or 0) for a directed graph given whether or not a loop is present.

In our study, a GCN was constructed to solve the edge restoration problem. This type of network can process the graph structure and vertex attributes to predict the probability of an edge existing between a pair

Publication Date: 21 Jan 2024
https://doi.org/10.58429/qaj.v4n1a333

of vertices. GCNs take as input the adjacency matrix A and the feature matrix X of graph G and generate a prediction about the presence or absence of an edge between each pair of nodes. The feature matrix X is a matrix of size |V| × d, where d is the dimensionality of the feature vector for each node.

GCNs are based on the idea of applying convolution on graphs, similar to the way GCNs apply convolution on images. The convolution operation on graphs produces new feature representations for graph nodes based on their neighbors. The feature representations are updated and iteratively propagated through the graph, considering the structure of the links between nodes. The updated feature representations can be computed as follows:

$$Z = f(A, X, W)$$

where Z is the new feature representation of nodes, f – graph convolution function, A – the adjacency matrix, X – current feature representations of nodes, and W – weight matrices (model parameters).

In a basic case, the GCN architecture takes the following form:

1. Input layer: the node feature matrix X and the adjacency matrix A.
2. Normalization layer: normalization of the adjacency matrix A to account for different degrees of node connectivity. This can be achieved through symmetric or row-wise normalization.
3. Graph convolution: application of graph convolution to feature representations of nodes. Graph convolution updates the features of nodes using information about the features of neighboring nodes and their weights.
4. GCN output layer: the softmax activation function is typically used to determine the probability of an edge being present.

To restore transitions in the business process model, we experimented with different compositions of neural network layers and settled on the following version (Figure 1).

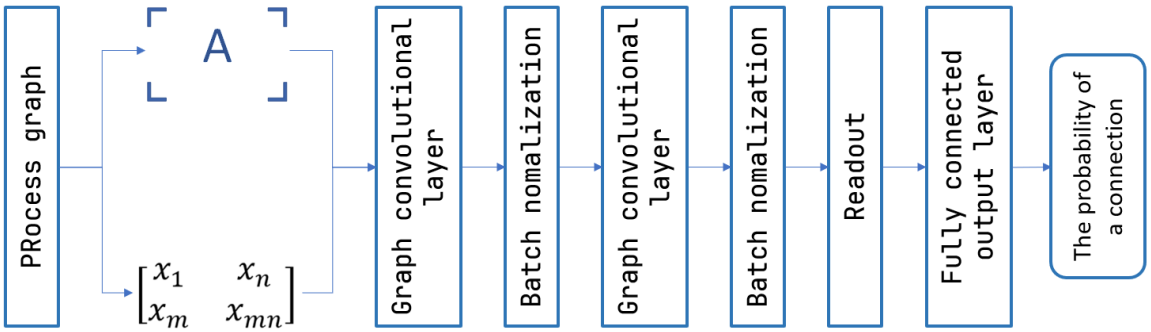

FIGURE 1. Structural composition of the layers of the GNN to recover transitions in the business process model.

As the adjacency matrix, we use a combination of a weighted matrix and an adjacency matrix accounting for the graph structure. In this case, each element of the adjacency matrix (i, j) is assigned a weight reflecting the frequency of token passing along the graph edge, i.e., in BPMN terms, the control flow (Figure 2).

Publication Date: 21 Jan 2024
https://doi.org/10.58429/qaj.v4n1a333

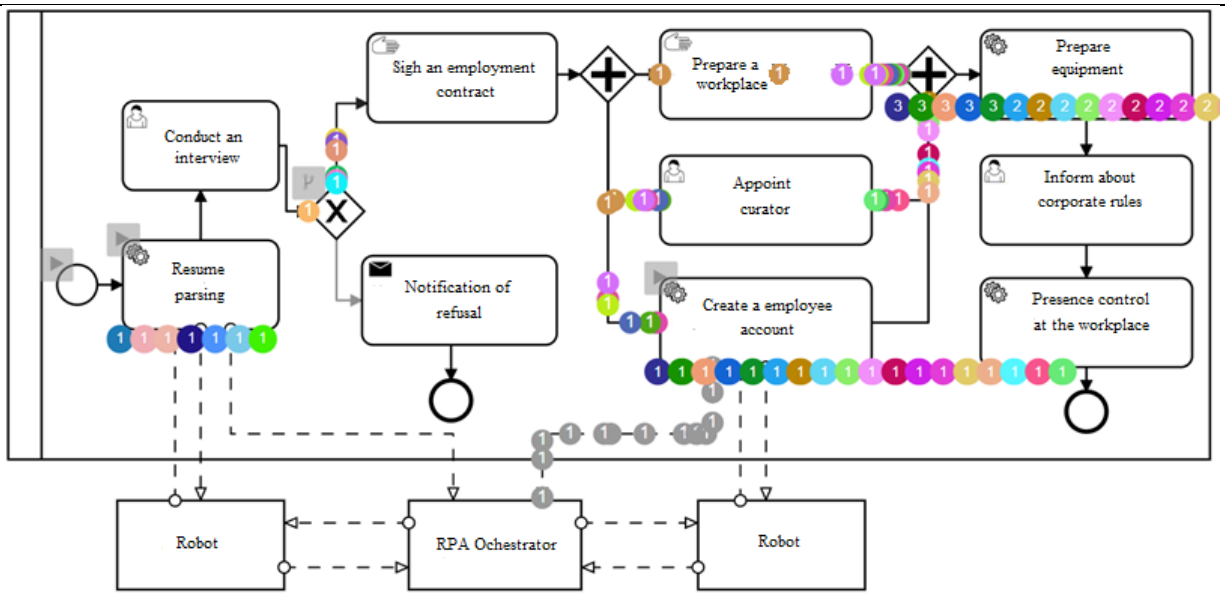

FIGURE 2. Analysis of business process tokens to determine weights in the adjacency matrix.

This considered the local graph structure by using windowing to determine which nodes to consider adjacent to each node in the graph.

The feature matrix for the project was compiled using textual information associated with each node of the graph, i.e., each task in the business process. Comments, annotations, and other logs tied to the task were analyzed. However, most often such information was missing. In such cases, the textual formulation of the task title was taken as the basis for analysis.

Naturally, the representation of task names as numerical vectors was based on embeddings, which have the following main advantages:

1. Semantic proximity: Embeddings can display semantic proximity between words or texts. Words with similar meanings will have close vector representations.
2. Information compactness: Embeddings allow presenting a large amount of textual information in a more compact way. Vectors of a smaller size contain compressed information about words or texts.

The chosen embedding model for the Russian language is Navec, which is based on Word2Vec and uses the Continuous Skip-gram algorithm. However, unlike the standard Word2Vec model, Navec is smaller in size and can be uploaded and used quicker.

To form the basis for the event log, we performed parsing of the process log of the 1C:Enterprise system of a company operating in the Bryansk region of Russia. The dataset for GNN training was prepared by extracting process data via process mining algorithms with subsequent manual correction of labels with respect to the presence of transitions between tasks.

III. RESULTS

The confusion matrix formed based on 38 experiments is presented in Figure 3.

Publication Date: 21 Jan 2024
https://doi.org/10.58429/qaj.v4n1a333

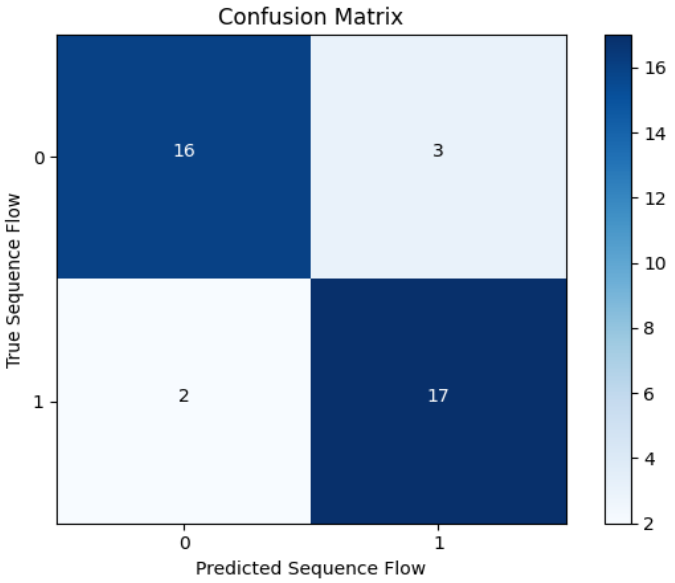

FIGURE 3. Confusion Matrix of trained GCNs.

The values of the respective quality metrics are found to be:
Precision = 0.895;
Recall = 0.850;
F1Score = 0.872;
Accuracy = 0.868.

In our view, with respect to the task of restoring connections between logged events in the absence of CaseID data, the obtained quality metrics are high.

The project baseline is based on a GNN without accounting for additional information derived from embeddings for the Russian language Navec. Data in Table 1 show the decisive role of textual information associated with each graph node, i.e., each task within the business process.

Table 1. Comparison of project baseline and embedding-based quality metrics.

Quality metric	Presented project	Project baseline (without textual data embeddings)
Precision	0.895	0.704
Recall	0.85	0.679
F1Score	0.872	0.701
Accuracy	0.868	0.629

In Figure 4, we provide the model of the business process "Approval of a commercial offer" restored based on the logbook.

Publication Date: 21 Jan 2024
https://doi.org/10.58429/qaj.v4n1a333

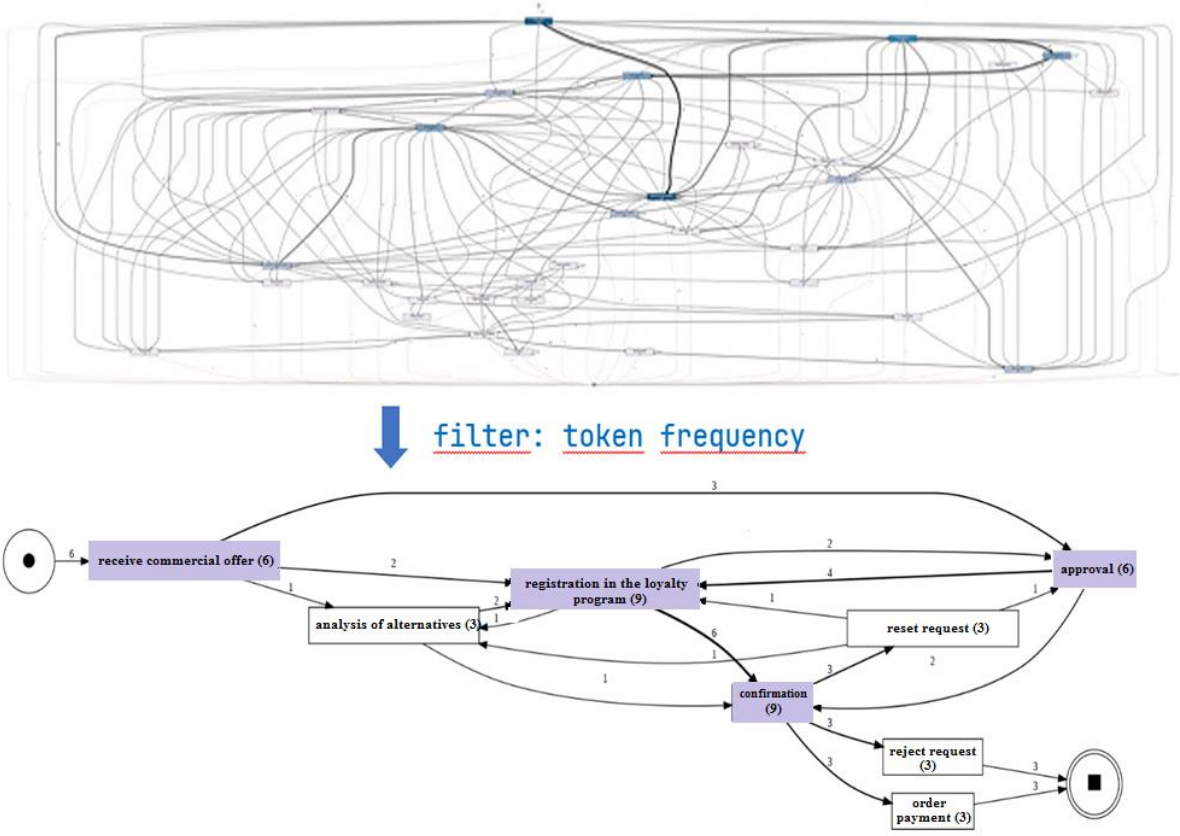

FIGURE 4. Model of the business process "Approval of a commercial offer" restored based on the logbook.

Although the logbook had no CaseID data within the "Reset request" task, the ML-subsystem managed to restore the connection needed (in BPMN terms – Sequence flow).

IV. CONCLUSION

In real-life practice, logbooks seldom contain CaseID data. In such an event, process mining may face certain limitations and challenges. In the present study, we developed a method for building business process models by restoring connections between events recorded in logs in the absence of CaseID based on GNNs.

As an adjacency matrix, we use a combination of a weighted matrix and an adjacency matric accounting for the graph structure. Thus, local graph structure is accounted for using a window to determine what nodes should be considered adjacent to each graph node. The feature matrix as part of the project is constructed based on textual information associated with each graph node, i.e., with each task in the business process. The employed embedding model for the Russian language is Navec, which relies on Word2Vec and uses the Continuous Skip-gram algorithm. Unlike the standard Word2Vec model, Navec is smaller in size and can be installed and used quicker.

To form the basis for the event log, we performed parsing of the process log of the 1C: Enterprise system of a company operating in the Bryansk region of Russia. As a result of process data extraction based on process mining algorithms with subsequent manual correction of labels in view of the presence of transitions between tasks, a dataset for training the GNN was prepared.

The relevant quality metrics are as follows: Precision = 0.895; Recall = 0.850; F1Score = 0.872; Accuracy = 0.868. In our view, as applied to the task of restoring connections between logged events in the absence of CaseID data, the obtained quality metrics can be considered high.

With the help of the obtained solutions, the necessary connection (Sequence flow) in the model of the business process "Approval of a commercial offer" was successfully restored in the absence of CaseID data in the event log as part of the "Reset request" task.

ACKNOWLEDGMENTS

The study was supported by the Russian Science Foundation grant No. 23-28-00180, https://rscf.ru/project/23-28-00180/.

REFERENCES

1. Bagratuni, K.; Kashina, E.; Kletskova, E.; Kapustina, D.; Ivashkin, M.; Sinyukov, V.; Karshalova, A.; Hajiyev, H.; and Hajiyev, E. Impact of socially responsible business behavior on implementing the principles of sustainable development (experience of large business). *Int. J. Sustain. Dev. Plan.* **2023**, *18*(8), 2481-2488. https://doi.org/10.18280/ijsdp.180819
2. Borodina, M.; Idrisov, H.; Kapustina, D.; Zhildikbayeva, A.; Fedorov, A.; Denisova, D.; Gerasimova, E.; and Solovyanenko, N. State regulation of digital technologies for sustainable development and territorial planning. *Int. J. Sustain. Dev. Plan.* **2023**, *18*(5), 1615-1624. http://dx.doi.org/10.18280/ijsdp.180533
3. Hamilton, W.L.; Ying, R.; and Leskovec, J. Inductive representation learning on large graphs, in *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017)*; Long Beach, CA, USA, 2017. https://doi.org/10.48550/arXiv.1706.02216
4. Abdullaev, I.S. and Khamraev, K.I. Modeling factors affecting net assets of investment funds using autoregressive distributed lag (ARDL) model. *J. Crit. Rev*. **2020**, *7*(12), 987-990. http://dx.doi.org/10.31838/jcr.07.12.174
5. Ilyushin, Y. and Afanaseva, O. Modeling of a spatial distributed management system of a preliminary hydro-cleaning gasoline steam column, in *20th International Multidisciplinary Scientific GeoConference SGEM 2020*, Vol. 2.1; STEF92 Technology: Sofia, Bulgaria, 2020; pp. 531-538. http://dx.doi.org/10.5593/sgem2020/2.1/s08.068
6. Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. Graph attention networks, in *Proceedings of the 6th International Conference on Learning Representations (ICLR 2018)*; Vancouver Convention Center: Vancouver, Canada, 2018. https://doi.org/10.48550/arXiv.1710.10903
7. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Yu, P.S. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*(1), 4-24. https://doi.org/10.1109/TNNLS.2020.2978386
8. Rozinat, A.; and van der Aalst, W.M.P. Conformance checking of processes based on monitoring real behavior. *Inf. Syst.* **2006**, *33*(1), 64-95. https://doi.org/10.1016/j.is.2007.07.001
9. van der Aalst, W.M.P. *Process mining: Discovery, conformance and enhancement of business processes*; Springer-Verlag: Berlin; Heidelberg, Germany, 2011.
10. Rybakov, A.V.; Shichkin, I.A.; Tolmachev, O.M.; and Magomaeva, L. The impact of a progressive personal income tax scale on reducing income inequality: Comparative analysis. *Relacoes Internacionais no Mundo Atual* **2022**, *1*(34), 371-395.
11. Weijters, A.J.M.M.; Ribeiro, J.T.; and van der Aalst, W.M.P. *Process mining with the heuristics miner algorithm*, BETA publicatie: Working papers, Vol. 166; Technische Universiteit Eindhoven: Eindhoven, Netherlands, 2011.
12. Di Francescomarino, C.; Ghidini, C.; Rospocher, M.; and Sperduti, A. Automated discovery of process models from knowledge-intensive processes, 2015.
13. Kipf, T.N. Graph convolutional networks; 2016. https://tkipf.github.io/graph-convolutional-networks/
14. Kipf, T.N. and Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint* **2016**. https://doi.org/10.48550/arXiv.1609.02907
15. Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; and Sun, M. Graph neural networks: A review of methods and applications. *AI Open,* **2020**, *1*, 57-81. https://doi.org/10.1016/j.aiopen.2021.01.001
16. Gao, C.; Zheng, Y.; Li, N.; Li, Y.; Qin, Y.; Piao, J.; Quan, Y.; Chang, J.; Jin, D.; He, X.; and Li, Y. Graph neural networks for recommender systems: Challenges, methods, and directions. *ACM Trans. Inf. Syst*. **2021**, *1*(1), 1-46. https://arxiv.org/pdf/2109.12843v1.pdf