Link Prediction based on Graph Neural Networks using Textual Data for Recommending Dutch News Articles

submitted in partial fulfillment for the degree of master of science

Naomi Rood
12666866

Master Information Studies
data science
Faculty of Science
University of Amsterdam

Submitted on 30-06-2023

	UvA Supervisor	External Supervisor
Title, Name	Hongyun Liu	Felix van Deelen
Affiliation	University of Amsterdam	Nederlandse Omroep Stichting
Email	h.liu@uva.nl	felix.van.deelen@nos.nl

ABSTRACT

Write your abstract here.

KEYWORDS

graph neural network, Dutch, link prediction, SEAL

GITHUB REPOSITORY

https://github.com/Naomirood/Thesis

1 INTRODUCTION

News articles are published online often. Each day, hundreds of news articles are added. People want to read what they are interested in and want to get a good understanding of a news topic. Online news platforms recommend news items to a user that is somehow related to a news article that a user reads at that moment. The explosion of news articles nowadays goes beyond the view limit of normal people and the chances of a reader missing news within his interest increases. It is important that the information overload problem in news articles will be tackled and news recommendations can play an important role to to help [10]. The NOS (a Dutch broadcasting organization) wants to recommend users certain articles efficiently. Currently, they suggest related news articles underneath most of the news articles which are recommended by manually assigning similar articles to the target article by the editorial team of the NOS. However, this is a time-consuming task.

A lot of research is available regarding news recommender systems and the literature on that is increasing in the past years [16]. Most of the recommender systems techniques are not interesting for this research, because it requires user reading history data. This research focuses on recommending news articles based on the content of the article since the news articles are publicly available. The popularity of graph-based recommendation models grows [4]. Graph-based learning methods see the recommender system's information from a graph's perspective. It has the ability to use structured data [24]. However, most of the state-of-the-art recommender systems using graph neural networks (GNN) require user input [22–24]. The relationship between users and items can be represented in a graph. On the other side, a graph structure that uses only information from items can also be used for recommendation, the links between items can be predicted using link prediction and the predicted links can be presented as recommendations to a user [18]. Three main traditional link prediction approaches are heuristic methods, latent-feature methods, and content-based methods [27]. Heuristic methods computed similarity scores as the likelihood of links (for example Common neighbors [13]). Latent-feature methods factorize matrix representations of a graph to learn the node embeddings (for example node2vec [5]). Content-based methods only focus on node attributes rather than the structure of a graph. However, Zhao et al [9] showed that combining graph features with node attributes can improve the performance of link prediction. Zang & Chen [28] created a new framework for link prediction that uses a subgraph enclosing technique around a link to make node embeddings. However, the input of the text is missing in this framework, but it is possible to give attributes about the articles to the nodes. They mention that their framework can be used in further research for recommender system applications. Furthermore, a Context-Aware Node Embedding method (CANE) is presented by Tu et al. [21] that uses the text of a node. It models the semantic relationships and graph structures between nodes into a context-aware node embedding. Their research focuses also on link prediction and is evaluated on three different datasets, two of them with English text and one in Chinese. However, it is not clear which framework they used to predict the links with their embedding method.

This thesis will focus on a new approach for a recommender system for Dutch news articles using data about the news articles. This means the Dutch text and metadata of the news article. The new link prediction framework of Zhang & Chen [28] will be combined with the CANE Context-Aware Node Embedding method that will use the text of Dutch news articles for node embeddings. The predicted links will be the recommended Dutch news articles.

1.1 Research question

The research question that follows up on this research gap is as follows: *To what extent can GNN-based link prediction using textual data recommend Dutch news articles?* Some subquestions are created for this research question:

- What kind of graph neural network framework can be used to predict links of Dutch news articles?
- How can we use textual information in a graphical representation of Dutch news articles to predict links?
- How can we process metadata from the news articles to be node attributes in the SEAl framework?
- How does a GNN link prediction model enriched with textual information compare to a text-only model such as TF-IDF in terms of AUC and recall?

2 RELATED WORK

This section will provide an overview of current techniques used for recommender systems and some new frameworks that have been implemented over the past few years. This section ends with a clear overview of the research gap.

2.1 News recommendations

Recommendation systems are systems that try to make relevant recommendations to users using a machine learning approach. Different approaches for making news recommendations already exist in the literature [16]. Usually, these are content-based recommendations that are recommended according to a similarity measure between articles already known to be preferred by a user [15]. This research focuses on non-personalized recommendations.

A common approach for recommendation without user input is using TF-IDF [14]. Each word gets a weight that shows how important a word is in a specific article. In the end, the cosine similarity is computed and this gives the similarity between two feature vectors of two articles. However, this method approaches all words independently and does not use the context of the words. A large similarity score between two articles will not directly mean that it will be a good recommendation, it only tells that the articles have a great similarity in the words. BERT is an approach that uses the context of words. Juarto & Girsang [7] used BERT with embedded sentences of news articles for a news recommender system. BERT [3] focuses on learning the context of words or sentences and creates embedding of the articles which can be used for recommendation. The latest research shows some new implementations of recommender systems for news recommendation [27]. Graph neural networks (GNN) became more popular over the years. GNNs have the ability to use previous recommendations to learn from. Liu et al [11] have discovered that a PMGT (a GNN-based method) outperformed the graph-BERT approach for recommending based on textual information.

2.2 Graph neural networks for link prediction

Graph neural networks have an increasing popularity in link prediction and are a powerful tool to learn from graph structure as well as from node structures together [27]. It has shown that it has big advantages over traditional methods for link prediction. A GNN usually consists of graph convolution layers that extract substructure features for nodes and a graph aggregation layer that uses node-level features to aggregate them into a graph layer feature. Mainly two paradigms for GNN-based link prediction exist. Node-based methods aggregate node embedding of the pairs of nodes connected to a link learned by a GNN, an example of such a method is Variational Graph AutoEncoder (VGAE) [8]. The second method is a subgraph-based method, this extracts a local subgraph around a link and uses the subgraph representation learned by a GNN to predict a link. The most important subgraph-based method currently is SEAL [28]. This method uses an enclosing subgraph for a link that needs to be predicted and then applies GNN to predict if the subgraph classifies to link existence.

2.3 Link prediction using SEAL

As explained in the previous section, the SEAL method extracts local subgraphs to learn subgraph representations which are learned by a GNN for link prediction [28]. Zhang & Chen have implemented a SEAL framework to predict links. The SEAL framework can learn together from Subgraphs, Embeddings, and Attributes for link prediction. The new element is that they enclose local subgraphs from a certain link and do not look at a total graph. This is because a subgraph already contains enough information to learn good graph structure features for link prediction. The SEAL framework consists of three steps. The first one is enclosing subgraph extraction, the second one is node information matrix construction and the third step is GNN training. This method combines graph structure features with side information about individual nodes. This side information is an embedding vector of a node and other node attributes (for example groups). h-hop enclosing subgraphs around two nodes connected to a link is used to define subgraphs. SEAL automatically learns graph structure features from the network using the enclosing subgraph method, the learning process is done using a GNN. The next step is node labeling which gives a feature about the role of a node in a subgraph, for example, the nodes to which a link is connected are the target nodes. These are different from the nodes that are further away from the link but still in the subgraph. A good node labelling of different roles around a link is important for GNNs to predict link existence. In their experiment, they used Node2Vec [5] to define node embeddings, however, they mention that any kind of network node embedding method is possible. Node embedding is a latent feature method that factorizes matrix representations of a network to learn a low-dimensional representation of a node. Besides latent feature methods, explicit features are used, this is available in the form of node attributes and describes any kind of side information about individual nodes. This framework is tested on eight different datasets with links and outperformed methods that used only graph-structured features. The SEAL method had a higher AUC score for link prediction on all eight different datasets in comparison with the VGAE [8] method, the VGAE method is a latent feature method that uses a GNN to learn node embedding that constructs the network best. This framework only looks at link prediction but suggestions for further research are given to include link prediction research in machine learning problems like recommender systems.

2.4 Text-based node embedding

Link prediction using GNN needs a node embedding method to learn graph structures. Zhang & Chen did compare their SEAL framework with methods that use only node2vec embeddings. Node embeddings are a way of representing nodes as vectors in a low-dimension space, these embeddings are learned from observed graph nodes. Their SEAL framework which uses a combination of learning from graph structures, as well as latent features, is better than using latent features alone. The SEAL framework makes it possible to use any kind of node embedding. In their implementation, node2vec is used. But this project will focus on textual data. Yang et al. mention that graph embedding methods like node2vec are great for low-dimensional dense vectors but not for using textual information [26]. CANE is a method that can make context-aware text node embeddings [21]. This embedding method uses the text of a node as information to make embeddings. It does not look only at graph structures like node2vec to make embeddings. It can model more semantic relations between nodes. Tu et al. did some experiments with different kinds of textual datasets. The AUC values for link prediction for CANE compared to baselines that used node embedding based on only graph structures were higher. Moreover, the AUC values of CANE compared to other textual node embedding methods (TADW [25], CENE [17]) were also significantly higher for three different datasets.

2.5 Research gap

This research will use a dataset that has never been used. This will be a dataset from the NOS with information about Dutch news articles and data about links between them. The SEAL framework will be implemented on the NOS dataset which is never done before. By revisiting the aforementioned works, this work will implement the SEAL framework on the Dutch news articles and links datasets for the first time. Furthermore. This research will integrate the SEAL framework with the context-aware CANE node embeddings and also apply this to the NOS dataset to do link prediction, which is also novel.

3 METHODOLOGY

3.1 Data description

This research will be carried out in collaboration with the NOS. For this research, two data sets provided by the NOS will be used.

3.1.1 NOS links data. The first data set contains all links between articles. This dataset is hand-labelled by the editorial teams of the NOS. The links represent the recommended news articles underneath the text of a news article (the 'bekijk-ook' articles) or represent a news article that is mentioned in the text of an article (like a citation). For the purpose of this project, only the 'bekijk-ook' articles will be used because these are the recommended article. The dataset contains three attributes. The first one is the *parent_article* which contains the article id of that article, the second one is the *child_article* which contains the id of the recommended article, the last one is *relation_type*, which is in this case always 'bekijk-ook'. Besides filtering on 'bekijk-ook' articles, there is filtered if both article ids are in the filtered article dataset (NOS article data), this means that a parent and child article must be an online news article. In the exploratory data analysis, we found that most of the articles have 0,1,2 or 3 recommended articles, this is shown in figure 1. Before 2014, none of the articles had any labelled recommended articles. From 2014, the number of articles which has linked articles increased over time. But still, a significant part of the articles has no recommended articles. The total number of links is 287049. The mean number of recommended articles (degree) is 2.17.

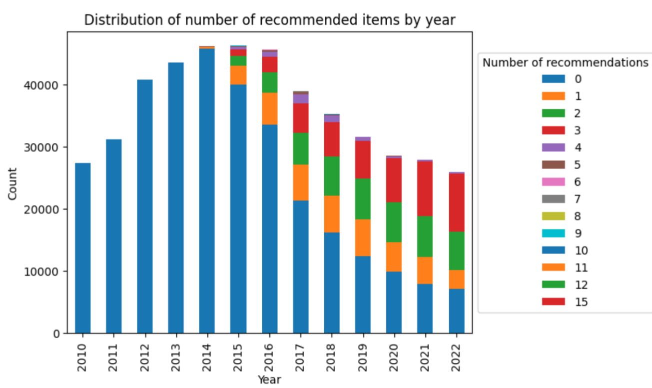

Figure 1: Number of recommended articles for a news article per year

3.1.2 NOS article data. The second data set contains data about all news pages from the NOS from 2010 till the end of 2022, these pages can contain a video, article, live stream, or live blog. For the purpose of this project to predict links between news articles, only the news page type *article* will be used, and only the articles that are online. The data consists of the following attributes: *article_id*, *title*, *description*, *text*, *published_at*, *modified_at*, *editorial team*, *system_tag*, *sub_category_list*, and *keywords*. All the information is written in Dutch. In order to use this data for recommending news articles, only articles that have one or more recommended articles ('bekijk-ook' articles) are used. This results in 131894 Dutch news articles. Figure 2 shows the distribution of the text length in words of the used news articles since this is a new dataset. The median text length is 288 words.

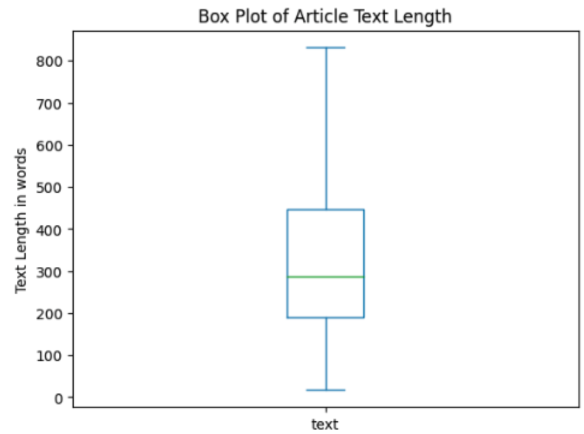

Figure 2: Distribution of the text length of the NOS news articles (outliers not shown)

3.2 Model implementation

3.2.1 Proposed method. Since the success of the SEAL method [28] that combines three different kinds of input for link prediction, we propose a new method that uses the SEAL framework as the basis and can predict links from Dutch textual data as well. This project uses a dataset that is never been shown before, which consists of Dutch textual data. Since the SEAL framework predicts links from node embeddings that do not use textual data, we added the CANE [21] method to the existing SEAL framework. The context-aware text embeddings are used to define the node embeddings of each node in a graph [21]. Figure 3 demonstrates the pipeline of this approach. The nodes in the graph represent the Dutch news articles, whereas the edges represent the links to the recommended articles. To predict whether there exists a link between two nodes, three kinds of inputs are gathered about the graph. These are: enclosing subgraphs, latent features and explicit features. The CANE framework makes node embedding by concatenating context-aware node embeddings and structural node embeddings. All this information is fed into a GNN to train and this predicts the likelihood of an existing link between two nodes. The approach solves the problem of recommending news articles given a graph where Dutch news articles represent the nodes and the recommended items represent the nodes where the text of the articles is also used to predict a link.

Focusing on the new approach in more detail, the GNN used three different kinds of input from the graph to learn from to predict links, these are the local enclosed subgraphs, node embeddings (latent features), and node attributes (explicit features). The local enclosed subgraphs around two nodes are defined by the parameter h (number of hops). The explicit features are node attributes representing any side information of the node, these must be discrete or continuous vectors. To make use of the text in the articles, the CANE embeddings are used instead of embeddings that only capture information about the network structure, these embeddings are context-aware of their neighbors. The CANE embedding is a

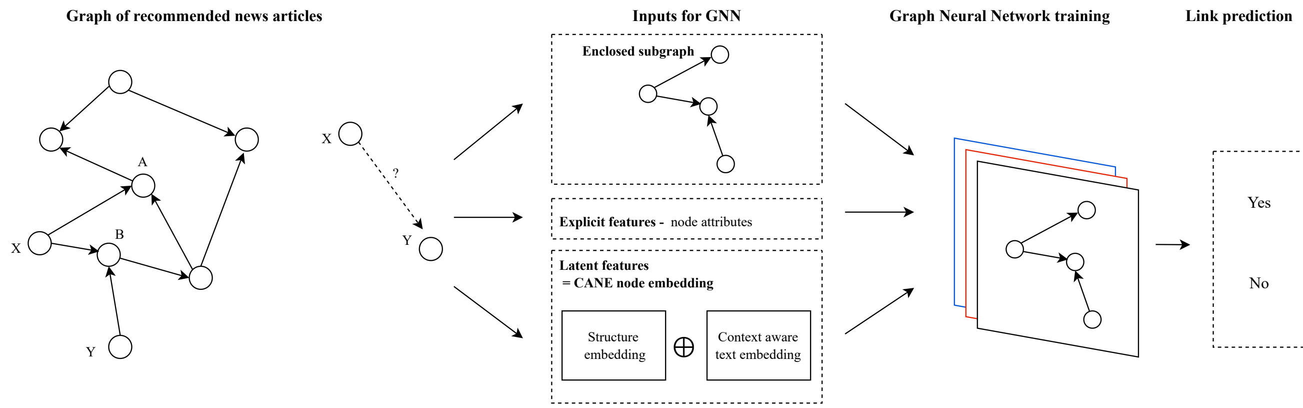

Figure 3: Overview of the new approach for link prediction using textual data

concatenation of the structure-based embedding which captures information about the network structure and the text-based embedding which captures the textual meaning. We created the structural embedding by LINE [19], which calculates a conditional probability of a node generated by another node. The CANE embedding uses mutual attention to enable the pooling layer in a convolutional neural network (CNN) to be aware of the node pair, so the text can affect the embedding of the other node and vice versa. The process of generating a context-aware text embedding is as follows 4. The text belonging to a node goes through a convolutional layer and the output is a matrix. From these matrices from two connected nodes, a correlation matrix is computed. Afterward, a mean-pooling operation is conducted (row- and column-wise) to make vectors, these vectors are transformed into attention vectors by a softmax function. Finally, the context-aware text embedding is computed by multiplying the attention vector with their respective matrix. From both embeddings, the CANE embedding can be computed. We used a GNN to train the graph because it consists of graph convolutional layers which can extract the local substructure features for the individual nodes and a graph aggregation layer, which aggregates the node-level features into a graph-level feature.

Link predictions using SEAL happen in three steps. The first one is enclosing subgraph extraction, the second one is node information construction (from the three inputs), and the last one is GNN training. The GNN takes as input (A, X). X is the node information matrix and A is the adjacency matrix of the enclosed subgraph. The node information X consists of three components: structural node labels, node embeddings and node attributes. We marked the nodes in the enclosed subgraph with a structural node label to point out the different roles of a node inside the enclosed subgraph. The GNN needs to know between which nodes the link should be predicted and has extra information about the relative position of the other nodes to the target nodes. For training, we used negative links beside the positive links to generalize the GNN model. In this way, it learns to distinguish between the features of nodes that are linked and nodes that are not. The framework outputs a score that represents the likelihood of an existing link between two nodes. For each node, only the three links to nodes with the highest scores are considered the predicted links. These nodes are the recommended items, we only used three of the highest scores because each article needs to get the same number of article recommendations.

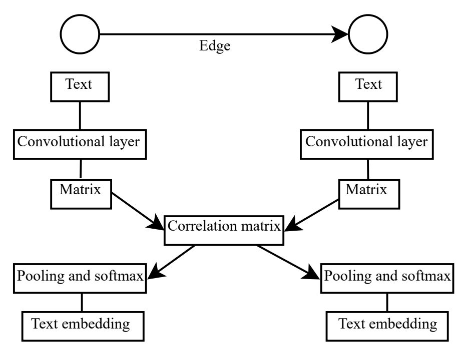

Figure 4: Overview of the process of creating the context-aware text embedding

3.2.2 Baselines. To validate the performance SEAL framework with the CANE node embedding on Dutch news articles, it will be compared against several baseline methods. The first one will be a random link predictor, where the prediction shows three random nodes that will be the recommended items. The second method is the SEAL [28] implementation without the CANE context-aware text node embeddings. This method is the same as described above, the difference is that a node2vec [5] embedding is used as node embedding instead of CANE. This is the same implementation as the original SEAL framework described in the paper of Zhang & Chen [29]. The third method which the new approach will be compared to is not based on link prediction but is a textual approach. This is TF-IDF. The used text of an article will be vectorized by a TfidfVectorizer [14]. Based on this, the cosine similarity will be calculated for two different articles, the three news articles with the highest scores for a certain article will be the recommended items. TF-IDF uses the text but does not use the context of the text or any kind of graph structure.

4 EXPERIMENTAL SETUP

This section describes how the method above is used in this project. The linked NOS dataset is used as input for the SEAL framework and the NOS articles set is used for the node attributes.

4.1 Experiment

The code of the SEAL implementation as well as the code for the CANE implementation is available on GitHub. Several experiments will be conducted to see which part or parts of the available text are the best for the final model. The available text for each item is the title, a short description of the article text, and the whole article text.

4.1.1 GNN. Zhung & Chen stated that SEAL is flexible with which GNN or node embedding method to use. Like in their implementation, the architecture Deep Graph Convolutional Neural Network (DGCNN) [29] as GNN is used. We used the default setting of DGCNN, it consists of four convolution layers with 32, 32, 32, 1 channels, a SortPooling layer, two 1-D convolution layers (16 and 32 output channels), and a dense layer (128 neurons) [29]. The DCGNN is trained for 50 epochs, the model with the smallest loss for the validation set is selected to predict the links in the test set.

4.1.2 Validation and test set. The edges (links) will be split into a train, validate and train set. Like in the implementation of Zhung & Chen [29], a split of 80% training, 10% validation, and 10% test is used. All these different splits of the dataset need to have negative examples for training. This is done by selecting randomly non-existent links.

4.1.3 Hyperparameter. An important hyperparameter in the SEAL framework is the number of hops (h). This number does select the enclosed subgraphs around target nodes. In the SEAL framework, h is only selected from 1,2 since they have verified that these subgraphs contain most of the useful information already. The selection procedure for the number of hops is as follows: if the second-order heuristic Adamic-Adar (AA) [1] is better than the first-order heuristic CN on the validation set, then $h = 2$ is chosen, otherwise $h = 1$.

4.1.4 Explicit features. One of the three different kinds of input for the GNN to train on is the explicit features, these need to be given in a continuous or discrete vector. Any kind of side information other than its structure can be used. Attributes from the NOS item dataset will be used. Different attributes and combinations will be tested to see which ones add the most information to the model. The node attributes that will be tested are published_time, editorial_team, system_tag, and subcategory. Also, a vector of the word distribution of an article text will be tested. The published_time can be converted to a vector by assigning a numerical value for each time interval (year, month, day, hour, minutes, seconds), this still captures the natural ordering of the timestamps. The editorial team is one-hot encoded into a discrete vector, the attribute consists of seven distinct categories. The variable system_tag is one-hot encoded in the same way as the variable editorial_team, 27 distinct categories exist but most of the articles do not contain a system tag. The variable subcategory will be multi-hot encoded, this attribute consists of 109 distinct categories where it is also possible that an item has multiple subcategories. We converted the text of an article to a tf-idf vector. These vectors are created by tokenizing the text and creating a vocabulary. The term-frequency was calculated for each word in the text and the idf was calculated for each word in the vocabulary. The idf shows how important a word is in the entire corpus, which is calculated by taking the logarithm of the ratio of the number of articles the word appears in divided by the total number of articles. For each word in the text, the tf-idf score is calculated by multiplying these the tf and idf values, all these scores represent the tf-idf vector for the article text.

4.2 Evaluating metrics

The proposed link prediction model of SEAL combined with CANE for recommending Dutch news articles will be evaluated. Two standard metrics are often used to quantify the accuracy of a predictive link model, these are area under the curve (AUC) and precision [12]. The AUC can be calculated from the area under the ROC curve which shows the true positive rate (TPR, equation 1) with respect to the false positive rate (FPR, equation 2) at different thresholds. The AUC (the area from the ROC curve) can be seen as the probability that a randomly selected missing link is more likely to appear than a randomly chosen non-existent link [6].

$$TPR/recall = \frac{TP}{TP + FN} \tag{1}$$

$$FPR = \frac{FP}{TP + FN} \tag{2}$$

Precision represents the number of good predicted links divided by the total number of predicted links. But concerning the purpose of this research, it will be a limited evaluation metric. The dataset of the hand-labelled links contains possible recommended items. This does not directly mean that these are the only correct articles. False positive articles, could be good recommended articles as well. An evaluation metric that will not directly punish false positives is recall. Recall is used when the objective is to minimize false negatives. Recall is also a commonly used evaluation metric in link prediction [2, 20]. The formula for recall is the same as the TPR and is written down in equation 1. The true positives are divided by the true positives plus the false negatives. Recall is the fraction of the news articles that are successfully recommended.

5 RESULTS

Write about your results here. Good captions to tables and/or figures are key.

6 DISCUSSION

Write your discussion here. Do not forget to use sub-sections. Normally, the discussion starts with comparing your results to other studies as precisely as possible. The limitations should be reflected upon in terms such as reproducibility, scalability, generalizability, reliability and validity. It is also important to mention ethical concerns.

7 CONCLUSION

Write your conclusion here. Be sure that the relation between the research gap and your contribution is clear. Be honest about how limitations in the study qualify the answer on the research question.

REFERENCES

[1] Lada A Adamic and Eytan Adar. 2003. Friends and neighbors on the web. *Social networks* 25, 3 (2003), 211–230.

[2] Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki. 2006. Link prediction using supervised learning. In *SDM06: workshop on link analysis, counter-terrorism and security*, Vol. 30. 798–805.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[4] Chen Gao, Xiang Wang, Xiangnan He, and Yong Li. 2022. Graph neural networks for recommender system. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 1623–1625.

[5] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 855–864.

[6] James A Hanley and Barbara J McNeil. 1982. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology* 143, 1 (1982), 29–36.

[7] Budi Juarto and Abba Suganda Girsang. 2021. Neural collaborative with sentence BERT for news recommender system. *JOIV: International Journal on Informatics Visualization* 5, 4 (2021), 448–455.

[8] Thomas N Kipf and Max Welling. 2016. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308* (2016).

[9] Chaozhuo Li, Senzhang Wang, Dejian Yang, Zhoujun Li, Yang Yang, Xiaoming Zhang, and Jianshe Zhou. 2017. PPNE: property preserving network embedding. In *Database Systems for Advanced Applications: 22nd International Conference, DASFAA 2017, Suzhou, China, March 27-30, 2017, Proceedings, Part I 22*. Springer, 163–179.

[10] Jiahui Liu, Peter Dolan, and Elin Rønby Pedersen. 2010. Personalized news recommendation based on click behavior. In *Proceedings of the 15th international conference on Intelligent user interfaces*. 31–40.

[11] Zhuang Liu, Yunpu Ma, Matthias Schubert, Yuanxin Ouyang, and Zhang Xiong. 2022. Multi-Modal Contrastive Pre-training for Recommendation. In *Proceedings of the 2022 International Conference on Multimedia Retrieval*. 99–108.

[12] Linyuan Lü and Tao Zhou. 2011. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications* 390, 6 (2011), 1150–1170.

[13] Mark EJ Newman. 2001. Clustering and preferential attachment in growing networks. *Physical review E* 64, 2 (2001), 025102.

[14] Constituency Parsing. 2009. Speech and language processing. (2009).

[15] Michael Pazzani and Daniel Billsus. 1997. Learning and revising user profiles: The identification of interesting web sites. *Machine learning* 27 (1997), 313–331.

[16] Shaina Raza and Chen Ding. 2022. News recommender system: a review of recent progress, challenges, and opportunities. *Artificial Intelligence Review* (2022), 1–52.

[17] Xiaofei Sun, Jiang Guo, Xiao Ding, and Ting Liu. 2016. A general framework for content-enhanced network representation learning. *arXiv preprint arXiv:1610.02906* (2016).

[18] Nitish Talasu, Annapurna Jonnalagadda, S Sai Akshaya Pillai, and Jampani Rahul. 2017. A link prediction based approach for recommendation systems. In *2017 international conference on advances in computing, communications and informatics (ICACCI)*. IEEE, 2059–2062.

[19] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding.. In *WWW*. ACM.

[20] Ben Taskar, Ming-Fai Wong, Pieter Abbeel, and Daphne Koller. 2003. Link prediction in relational data. *Advances in neural information processing systems* 16 (2003).

[21] Cunchao Tu, Han Liu, Zhiyuan Liu, and Maosong Sun. 2017. Cane: Context-aware network embedding for relation modeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1722–1731.

[22] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*. 165–174.

[23] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised graph learning for recommendation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*. 726–735.

[24] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. 2022. Graph neural networks in recommender systems: a survey. *Comput. Surveys* 55, 5 (2022), 1–37.

[25] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Chang. 2015. Network representation learning with rich text information. In *Twenty-fourth international joint conference on artificial intelligence*.

[26] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. 2016. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*. PMLR, 40–48.

[27] Muhan Zhang. 2022. Graph Neural Networks: Link Prediction. In *Graph Neural Networks: Foundations, Frontiers, and Applications*, Lingfei Wu, Peng Cui, Jian Pei, and Liang Zhao (Eds.). Springer Singapore, Singapore, 195–223.

[28] Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. *Advances in neural information processing systems* 31 (2018).

[29] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. 2018. An end-to-end deep learning architecture for graph classification. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

Appendix A FIRST APPENDIX

Put your appendices here.