Graph Neural Networks for Pedagogical Recommendations of Academic Papers

Bowen Li

May 7, 2024

Abstract

Increasing output in the volume of academic publications over the years will inevitably lead to problems when it comes to allowing a user to be able to pick out papers that are relevant and/or useful to them, with citations being the only sense of structure one can rely on. Incidentally, graph neural networks has seen a recent surge in interest and developments, presenting new ways to apply deep learning techniques on data that can be represented in the structure of a graph. This project explores graph neural networks and the task of link prediction in creating a recommendation system that, given a paper/topic, will provide users with a suggested sequence of papers to read in order to further understand the paper/topic, ultimately assisting users in navigating the ever increasingly complex landscape of academic publications. The code can be found at this repository or at the url: `https://github.com/lib250/publication-recommender`.

Introduction

As interest in the fields of STEM has grown over the past few years, and inevitably will continue to grow in the coming years, the number of academic publications will undoubtedly skyrocket. Although this certainly serves as an indicator of the accelerating advances made across these fields, it could also pose a problem for future generations, hobbyists and academics alike, as it increasingly makes the overhead of searching through and selecting relevant reading material more costly.

Typically, when an interested party happens upon a relevant paper, the references of that paper could serve as a suitable starting point for further reading. However, not only could the references of the paper be dubious due to some of the meta motivations concerning the publication and review process, it is still lacking in terms of reading about future developments. After all, a paper is not physically able to cite something that, as of its writing, has yet to exist.

This leaves the interested party to rely on search engines, scouring relevant conferences (the

volume of which is only increasing), and gaming their own social media recommendations. The last option could involve one following relevant figures on social media, whom in turn follow other relevant figures that may eventually all be taken into account when recommending new papers to users. This is reminiscent of a graph structure, and is suitable for users who may wish to keep up with the latest in terms of the field. In seeking to build a recommendation system that provides more structure to the users however, we can still leverage this idea of graphs.

Graph neural networks (GNNs) have seen a recent surge in popularity as a paradigm that applies deep learning techniques to data with a graph structure. One of the main problems that it addresses is link prediction, wherein a graph is assumed to have missing edges, and the task is, for a given node, to predict which of the other nodes in the graph it would most feasibly share an edge with. With a good choice of nodes and edges, this task steadily lends itself to being applied as a recommendation system.

This project will explore the potential of link prediction to be applied to a recommendation system that will provide users with a suggested reading sequence of academic publications in order to familiarize themselves with a given topic. In the ideal system, if a user is considering a paper, they would be recommended papers to read as prerequisites and/or papers that could serve as further reading.

In this project, we test the feasibility of this by training a Graph Autoencoder for the task of link prediction on a small citation graph and then qualitatively evaluating some of its recommendations. We then try the same training on a larger graph with a larger average node degree. Finally, we explore whether or not the recommendations could be improved by using a modified loss function that penalizes inferring links between two papers that are too different in terms of content. In general, we find that performance in the task of link prediction does not correlate well with the pedagogical quality of recommended papers, and incorporating paper similarity into the loss function does not provide a noticeable improvement in the qualitative evaluations.

Related Work

The task of link prediction is a well-studied problem across various fields. [1] provides a survey on various classical link prediction approaches and comparisons. [2, 3] both provide classical link prediction solutions in the context of recommender systems, one based on vector similarity and the other using bipartite graphs.

In terms of GNNs, [4, 5] provide surveys and taxonomies of GNN techniques applied to recommender systems. [6] presents a particular GNN approach to the recommendation problem based on a new version of the dual-tower model for heterogeneous graphs. [7] presents another GNN approach to creating a recommender system based on a heteroge-

nous graph with users, articles, and article topics as nodes. Most notably, the topic nodes are latent, and are thus learned from the dataset. [8] presents a GNN approach that reconstructs graph components into dense item-item graphs based on a user's history and perceived interests. [9] presents a GNN approach to creating a recommender system for Massively Open Online Courses (MOOCs), though doesn't provide a recommended sequence to take courses in.

Datasets

We will explore two graph datasets: Cora_ML and ogbn-arxiv.

The Cora_ML dataset, from the paper presenting Graph2Gauss [12], is a subset of machine learning related papers extracted from the Cora dataset [13]. The graph contains node embeddings of 2995 papers, connected by 8416 directed edges. With a vocabulary size of 2879, each node embedding is a one-hot-encoding of whether or not a vocabulary word is present in the paper. Directed edges point from citing papers to cited papers. We will perform some initial exploration and visualizations on Cora_ML as it is a reasonably small dataset, but because of that, it only has an average node degree of only around 2.81 (meaning that each paper only has, on average, 3 citations reflected in the graph). As such, further evaluations will be done on ogbn-arxiv as well.

The ogbn-arxiv dataset is a similarly directed graph of 169,343 computer science publications from arXiv indexed by Microsoft Academic Graph (MAG) [15]. Each node is a 128-dimensional embedding vector obtained from averaging the embeddings of all words in the paper's title and abstract. Individual word embeddings are obtained by running the skip-gram model [16] over the MAG corpus. With 1,166,243 edges, the ogbn-arxiv graph has an average node degree of about 6.89, which is more reasonable since the goal is ultimately to learn about content similarities via citation relationships. Another benefit of this dataset is that it contains more recent papers (up to 2019) than the Cora dataset (up to the year 2000).

Dataset Exploration

Cora_ML Dataset Agreement

PyG provides built-in methods to install distributions of certain datasets, however, the issue is that it is only possible to download the graph itself via PyG, meaning that we would only have access to the node embeddings and adjacency information. However, if we want to qualitatively assess the model's performance, it would be pertinent to have a way to retrieve the original papers from the graph and predictions, and as such we wouldn't be able to use PyG's built-in loader alone.

The GitHub repository of [12], contains .npz files for the Cora_ML dataset that we can

use, although it presents a slight problem when compared with the same dataset from PyG's loader. Although the adjacency information between the 2 sources of the same dataset are identical, the node embeddings are not. It is unlikely that the indexing of each publication is different, as the adjacency information is represented using the indices of the original papers, so in all likelihood the node embeddings themselves are different between the graph retrieved from the G2G repository and the one retrieved from PyG's loader. Further investigation is needed to determine why exactly the embeddings do not match, although may not be necessary as we can test the performance of a model on both representations.

Dataset Structure

Although the node embeddings differed between the two sources, the structure of the graph was the same. Both graphs had 2995 nodes represented by 2879-dimensional embeddings, connected by 8416 directed edges. Both graphs had an average node degree of 2.81 and have no isolated nodes (all publications were either cited by another in the dataset or cites another in the dataset).

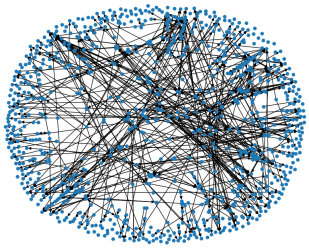
There were attempts made to visualize graphs, however, due to the high-dimensional embeddings and general volume of the graph, creating visualizations were impractical. Several attempts were made using the NetworkX library to visualize a subset of the graph with various different graph drawing techniques, but the visualizations for the most part were not very informative, as shown in Figure 1.

Approach

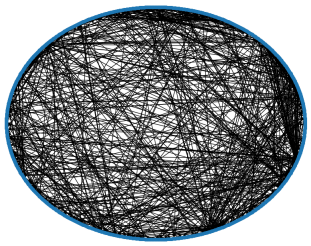
Link Prediction

Given a graph with nodes and edges, the task of link prediction seeks to infer new/previously missing edges between initially unconnected nodes in the dataset. Given two nodes in the graph, a link prediction model would thus output a score representing the likelihood that an edge exists between those two nodes (with the ordering of the two nodes only mattering in the case of a directed graph). To train such a model, we treat the problem as a binary classification task using the graph's existing structure as the ground truth. That is, edges that already exist in the graph would serve as positive examples, and pairs of nodes that do not have an edge between them would be sampled as negative examples.

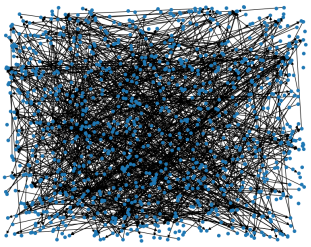
The GNN architecture we chose to experiment with is a Graph AutoEncoder (GAE) based on [14], which lends itself well to the task of link prediction. The GAE is an encoder-decoder model, wherein the encoder learns latent representations for the nodes, and the decoder, for a pair of latent node embeddings, outputs a score corresponding to the likelihood of an edge existing between the node pair (an ordered pair for the directed case). In



(a) Spring



(b) Circular



(c) Random

Figure 1: NetworkX visualizations of a random sample of 1000 nodes from the dataset using different drawing layouts. The visualizations are not very informative. More examples are available in the notebook in the repository.

this architecture, the encoder is a 2-layer Graph Convolutional Network (GCN), and the decoder is an inner product decoder. As such, our model would have the encoder learn latent node representations such that papers that have a citation relationship would be closer together (larger inner product) in the latent space.

Prerequisite Recommendations

We first gauge how well the task of link prediction on a citation graph correlates with giving pedagogical recommendations. That is, if a model were to perform well on a link prediction task, we also qualitatively evaluate how well the predicted links could serve as a pedagogical (prerequisite) recommendation. In our experiments, we focus on recommending prior readings, as a system that works well on recommending prior readings should feasibly also work well on recommending further readings if the model were to be trained on the same graph with the edge directions reversed.

Experimental Setup

To this end, we train the GAE for link prediction on the Cora_ML dataset. The training is done on Google Colab with a 12.67 GB RAM limit. The encoder component has 2 graph convolutional layers with output channels 128 and 64 to transform the nodes into an embedding space. During training, negative samples are also drawn to serve as negative examples in a 1-to-1 ratio with the positive edge examples for the decoder's classification task. For training, we use an Adam optimizer with learning rate 0.001 and binary cross entropy (BCE) loss as the loss function.

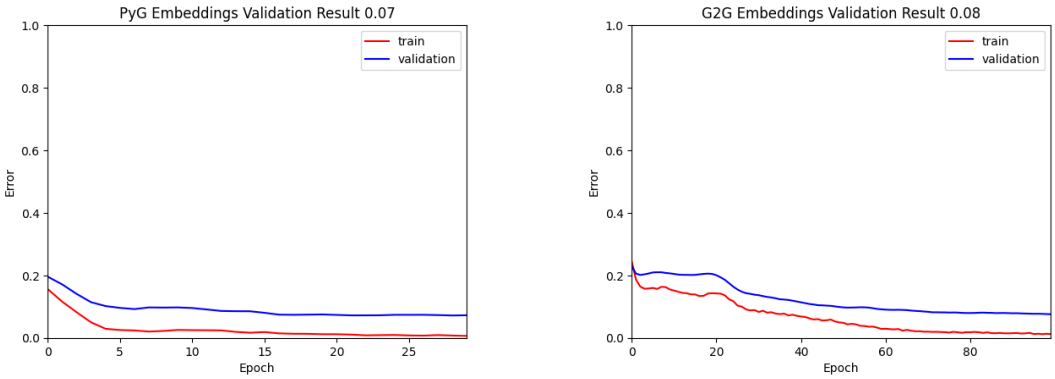
We then choose a select few papers to get recommendations for prerequisites, serving as test papers. For a given test paper, we have the model make predictions on potential edges from that paper to every other paper in the dataset, assigning each potential edge a score based on how likely it is to exist. We then qualitatively evaluate the top 10 scoring edges, reasoning about whether the most likely predicted links point to papers that could serve as sensible prior readings to the test paper.

Afterwards, we perform the same test except with a model trained on the much larger ogbn-arxiv dataset to try to account for dataset size.

We then perform the same test on the ogbn-arxiv dataset again but with a modified loss function that is a weighted sum between the original classification binary cross entropy loss and a new negative cosine similarity component between the original node embeddings for each link sample. This way, the model would also be incentivized to learn latent node representations that satisfy two conditions:

1. Nodes that have a citation relationship between them are close together in the latent space.
2. Nodes that are close together in the original embedding space are close together in the latent space.

Since the original embedding space was based on the contents of the papers' titles and abstracts, the second condition would, in theory, allow for recommendations that are similar in content, and by proxy, more relevant in subject matter. The weights of the weighted sum would thus control the priority balance between these two characteristics of the latent space. Accordingly, we perform the evaluations on a model trained on the loss function with equal weights (both 0.5), and on trained on a loss function that favors content similarity (0.1 for the BCE component and 0.9 for the negative cosine similarity component).



(a) PyG Data

(b) G2G Repo Data

Figure 2: Training loss curves for the GCN model evaluated on both (a) the graph from the PyG loader and (b) the graph from the G2G repository. Note that the x axis scales are different between the two graphs.

Evaluation

Cora_ML

For initial tests, we evaluate model performance using ROC-AUC as is convention with link prediction tasks and the result of the model on the Cora_ML dataset gathered from both sources is shown in Figure 2.

Note that the scales of the x axes are different between the two graphs. It seems that the node embeddings are indeed different between the two data sources as the behaviors differ greatly during training. While the PyG graph's error rate converges at around 10 epochs, the error rate for the G2G repository's graph behaves erratically before converging much later at around 60 epochs.

For our qualitative evaluation, we choose a test paper that describes a compression algorithm for probability transition matrices [17] and analyze the top 3 recommendations from the model. The top 3 recommendations were a paper discussing multi-associative memory mapping, a paper that dealt with genetic algorithms, and a paper on evolving 3-D models, none of which concern or utilize stochastic processes or even probabilistic algorithms. Interestingly, due to the small size of the dataset, the test paper only had 3 citations, 2 of which were papers that applied markov models to the field of genetic algorithms, which is likely why a genetic algorithms paper was one of the top recommendations despite being unrelated to probabilistic modeling. As such, the model is clearly held back by a small dataset, and so we turn to a larger dataset.

ogbn-arxiv

We train our model on the ogbn-arxiv dataset in a similar fashion and use the seminal paper "Attention is all you need" [18] as our test paper. Here, we would expect predictions that point to papers concerning machine translation, the attention mechanism, or at the very least deep learning. The model's top predictions included papers that dealt with sentiment analysis, which is reasonable, but also many unrelated papers such as subtask discovery, coding, and computer networks.

Here, it seemed as if the model predictions were not similar enough in content to the test paper, and so we implemented the modified loss function described in our approach section, which penalizes linking papers that are dissimilar in content. Training the model with the loss function that weighs both components equally, we get predictions in the fields of cybersecurity, communications, graph theory, and audio processing. However, one of the papers recommended utilizes the attention mechanism, which could imply that the content-similarity component of the loss function is having an effect.

To further test this, we evaluate another model, this time trained on a loss function that heavily weighs the cosine dissimilarity in order to encourage predictions that are more similar in content. This however, led to predictions that were even less similar than before, recommending papers concerning optical character recognition, branching programs, and computer vision.

Discussion

Results

From our qualitative evaluations, it is most likely the case that link prediction as it would not serve very well as a pedagogical recommender system of academic papers. This is likely due to a multitude of factors and nuances that distinguish the two tasks. From our qualitative evaluations with the Cora_ML dataset, we see one of the issues that arise which is a possible overfitting to the citation relationships. Since the test paper only had 3 citations within the dataset, and the majority of which just so happened to be about genetic algorithms, the model ended up predicting a genetic algorithms paper based solely on that fact, instead of the contents of the actual paper itself. As such, citation relationships alone do not carry enough information about the content of the paper to provide reasonable educational recommendations.

We attempted to remedy this overfitting by training on a larger dataset and introducing a modified loss function. With a larger dataset, each paper would have more citation information, and, in expectation, the majority of a paper's citations should be relevant to the content of the paper itself. As such more citations per paper would likely influence predictions to favor linking papers closer in content. To further incentivize this, the mod-

ified loss function penalizes predictions that stray too far from the contents of the test paper. Even with these modifications, however, the quality of recommendations did not seem to improve much, implying that more work should be done to address other issues when applying link prediction to this problem.

Limitations & Future Work

There are some fundamental limitations that, when addressed, are likely to improve recommendations when it comes to applying link prediction to pedagogical recommendations. One of the major limitations of our experiments is that we tested on a model architecture that may not reflect this problem accurately, especially when it comes to the choice of decoder. For our model, we used an inner product decoder to score whether two nodes would have a link between them, but the inner product operation is symmetrical while a citation relationship is fundamentally asymmetrical. As such, the decoder may have a bias preventing it from accurately capturing citation relationships, and it may be worthwhile to experiment with other, asymmetrical decoders.

Another limitation of this experiment is the restriction to purely academic publications, as academic publications that are expository in nature are a rarity. As such, in many cases, there simply may not be a good prerequisite paper to recommend, and the proper prior reading is, instead, a textbook, for example. This could be addressed with a heterogeneous dataset that contains both expository articles and state-of-the-art academic publications.

A worthwhile endeavour may be to formalize this problem by creating a dedicated dataset to train on to learn pedagogical recommendations and a proper metric for determining the quality of a recommendation. This study has relied heavily upon qualitative evaluations since pedagogical quality is fundamentally subjective due to the nature of human learning, but the search of some objective metric could be worthwhile. In addition, the creation of a dedicated dataset for this problem could also include expository articles/blog posts in addition to smaller optimizations, such as pruning citations that come from less informative sections of a paper.

References

[1] Ajay Kumar, Shashank Sheshar Singh, Kuldeep Singh, Bhaskar Biswas. *Link prediction techniques, applications, and performance: A survey*. Physica A: Statistical Mechanics and its Applications, 553, 2020. `https://doi.org/10.1016/j.physa.2020.124289`

[2] Zhan Su, Xiliang Zheng, Jun Ai, Yuming Shen, Xuanxiong Zhang. *Link prediction in recommender systems based on vector similarity*. Physica A: Statistical Mechanics and its Applications, 560, 2020. `https://doi.org/10.1016/j.physa.2020.125154`

[3] T. Jaya Lakshmi, S. Durga Bhavani. *Link prediction approach to recommender systems.* Computing, 2023. https://doi.org/10.1007/s00607-023-01227-0

[4] Shiwen Wu, Fei Sun, Wentao Zhang, X Xie, Bin Cui. *Graph Neural Networks in Recommender Systems: A Survey.* ACM Computing Surveys, 55(5): 1-37, 2022. https://doi.org/10.48550/arXiv.2011.02260

[5] Zijian Liang, Hui Ding, Wenlong Fu. *A Survey on Graph Neural Networks for Recommendation.* International Conference on Culture-oriented Science & Technology (ICCST): 383-386, 2021. https://doi.org/10.1109/ICCST53801.2021.00086

[6] Qiang He, Xinkai Li, Biao Cai. *Graph neural network recommendation algorithm based on improved dual tower model.* Scientific Reports, 14, 3853, 2024. https://doi.org/10.1038/s41598-024-54376-3

[7] Linmei Hu, Chen Li, Chuan Shi, Cheng Yang, Chao Shao. *Graph neural news recommendation with long-term and short-term interest modeling.* Information Processing & Management, 57(2), 2020. https://doi.org/10.1016/j.ipm.2019.102142

[8] Jianxin Chang, Chen Gao, Yu Zheng, Yiqun Hui, Yanan Niu, Yang Song, Depeng Jin, Yong Li. *Sequential Recommendation with Graph Neural Networks.* Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval: 378-387, 2021. https://doi.org/10.48550/arXiv.2106.14226

[9] Jingjing Wang, Haoran Xie, Fu Lee Wang, Lap-Kei Lee, Oliver Tat Sheung Au. *Top-N personalized recommendation with graph neural networks in MOOCs.* Computers and Education: Artificial Intelligence, 2, 2021. https://doi.org/10.1016/j.caeai.2021.100010

[10] Johannes Gehrke, Paul Ginsparg, Jon Kleinberg. *Overview of the 2003 KDD Cup.* SIGKDD Explorations 5(2): 149-151, 2003. http://www.cs.cornell.edu/home/kleinber/kddcup2003.pdf. Retrieved from https://www.kaggle.com/datasets/wolfram77/graphs-citation/data?select=cit-HepTh-abstracts.

[11] Mathurin Aché. *Citation Network Dataset.* March 2021. Retrieved from https://www.kaggle.com/datasets/mathurinache/citation-network-dataset/data

[12] Aleksandar Bojchevski, Stephan Günnemann. *Deep Gaussian Embedding of Graphs: Unsupervised Inductive Learning via Ranking.* February 2018. https://doi.org/10.48550/arXiv.1707.03815

[13] Andrew McCallum, Kamal Nigam, Jason Rennie, Kristie Seymore. *Automating the Construction of Internet Portals with Machine Learning.* July 2000. https://doi.org/10.1023/A:1009953814988

[14] Thomas N. Kipf, Max Welling. *Variational graph auto-encoders*. NIPS Workshop on Bayesian Deep Learning. November 2016. https://doi.org/10.48550/arXiv.1611.07308

[15] Kuansan Wang, Zhihong Shen, Chiyuan Huang, Chieh-Han Wu, Yuxiao Dong, and Anshul Kanakia. *Microsoft academic graph: When experts are not enough.* Quantitative Science Studies, 1(1):396–413, 2020.

[16] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. *Distributed representations of words and phrases and their compositionality.* Advances in Neural Information Processing Systems (NeurIPS), pp. 3111–3119, 2013.

[17] William M. Spears. *A Compression Algorithm for Probability Transition Matrices.* SIAM Journal on Matrix Analysis and Applications, 20(1), 1998.

[18] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. *Attention is all you need.* Advances in neural information processing systems 30 (2017).