Predicting Future collaborations between Authors

Chitrank Gupta*
Department of Computer Science
The University of Texas at Austin
chitrank@utexas.edu

Ying Ding
School of Information Science
The University of Texas at Austin
email

Abstract

In this paper we tackle the problem of predicting future collaboration between academicians. The problem is hard because the number of candidates is huge (in millions) and so manual examination is impractical. In this report, we propose to apply graph representation algorithms on a heterogeneous graph consisting of various kinds of nodes including author nodes to learn dense representations for those author nodes. Later, we use these dense representations to identify possibility of collaboration between two author nodes which haven't co-authored before.

1 Introduction

Predicting future collaborations between academicians is an active ongoing research. Several academicians work on similar topics and publish in similar venues but because they are wide-spreaded out around the globe and belong to different stages of their research career, they are not able to find each other and that leads to sub-optimal collaboration.

In this report we use two state-of-the-art graph machine learning algorithms to learn dense representations of the nodes of a heterogeneous graph which consist of several kinds of nodes including researcher nodes. We then later use the embeddings of the researcher nodes to calculate the possibility of them collaborating with each other in the future.

2 Dataset

In this work, we use the PubMed®knowledge Graph (PKG) [1]. PubMed®is a large online repository of research articles and journals in the medical domain. PKGon the other hand is a knowledge graph which is a structured version of PubMed®containing not only articles and their authors but also their metadata like article's venue, MeSH (Medical Subject Headings) and bioentites. Similarly authors are associated with their affiliations. One can represent all these entities as nodes and associations as edges of a graph. An example is provided in Fig. Such a graph is aptly called heterogeneous graph since not all nodes of this graph are the same. For example, paper nodes are inherently different from the author nodes both of which are in turn different from MeSH nodes. A heterogeneous graph can be described by its corresponding metagraph as depicted in Fig. The metagraph describes the underlying graph completely in the sense that there cannot be any other kinds of edges any more than what is already present in the metagraph.

For our experiments we rely on the "Cancer" subset of the PKG. To obtain this subset we filter in only those articles which have one of the MeSH term as D009369 [2]. And then we retain only those of the rest of the types of nodes which are associated with those set of articles.

*

[2]https://www.ncbi.nlm.nih.gov/mesh/?term=D009369

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

3 Models

In this section we describe the two graph machine learning algorithms that we use to learn dense representation of the nodes of PKG.

3.1 MetaPath2Vec

MetaPath2Vec or MP2V [2] is a self-supervised graph machine learning model. It learns embeddings of nodes in such a way so that it increases the similarity between two nodes which are in local neighborhood of each other while decreasing that of others. The similarity is captured by the following function s

$$s(u, v) = e_u.e_v \tag{1}$$

where u and v are the two nodes in question. During learning, since we want v to be more similar to u than any other node, we actually want to maximize the following from the u's perspective

$$f(u, v) = \frac{e^{s(u,v)}}{\sum_{v' \in V} e^{s(u,v')}} \tag{2}$$

3.1.1 Negative Sampling

In reality, eq 2 is computationally expensive to compute because the denominator involves summing over millions of nodes, so instead we sample a very small set of nodes V' which we will actually use in the denominator.

$$f(u, v) = \frac{e^{s(u,v)}}{\sum_{v' \in V', |V'| \ll |V|} e^{s(u,v')}} \tag{3}$$

Another thing to note is that we use eq3 in the log domain, instead of the $exp(.)$ domain to make training more stable as indicated in the following equation.

$$log\, \sigma(s(e_u.e_v)) + \sum_{i=1}^{M} \mathbb{E}_{v' \sim P(v')}[log\, \sigma(-s(e_u, e_{v'}))] \tag{4}$$

where $P(v')$ is the probability distribution over all the nodes of the graph.

3.2 Heterogeneous Graph Transformer

Heterogeneous Graph Transformer or HGT[3] is a supervised machine learning algorithm that learns a transformation function on input features of the nodes. In contrast to MP2V, HGTis an inductive model. It does not learn embeddings but rather a transformation that can be applied on some input node features. HGTis an example of Graph Attention Networks (GATs) [4] wherein the message from the neighboring nodes is first re-weighted via the attention score computed by the target node before getting aggregated together to update the target's node representation. But the main difference is that unlike GATs, HGT uses different attention weights for different kinds of nodes as explained in eq

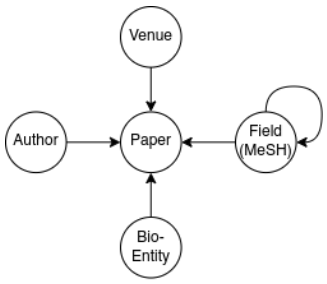

Figure 1: Metagraph of the subset of PKGthat we consider in this work.

$$
\begin{aligned}
Attention(s,t) &= K(s)W^{ATT}_{\phi(e)}Q(t) \quad \text{...(HGT)} \\
K(s) &= Linear_{\tau(s)}(H^{(l-1)}[s]) \\
Q(t) &= Linear_{\tau(t)}(H^{(l-1)}[t])
\end{aligned} \tag{5}
$$

$$
\begin{aligned}
Attention(s,t) &= K(s)Q(t) \quad \text{...(GAT)} \\
K(s) &= Linear(H^{(l-1)}[s]) \\
Q(t) &= Linear(H^{(l-1)}[t])
\end{aligned} \tag{6}
$$

In addition HGTconsiders several other modifications inspired from past research– including temporal information in the embeddings and multi-head attention [5]. The transformer's parameters are learnt via the cross-entropy loss between the predicted labels and ground-truth labels. Also since we want this transformer network to generalize to many kinds and size of graphs, during training the transformer is learnt only on a small random subgraph around the node whose label we want to predict which is re-sampled every iteration.

4 Training Details

4.1 MP2V

For MP2Vwe used 1 million random walks in total, each of length of 300. These random walks are uniformly divided across different kinds of starting nodes. i.e., the number of random walks starting from every kind of node is more or less the same. The list of metapaths used, categorized according to the type of the starting node type is given in Table 1.

We use the DGL implementation hosted here [3] and we keep the same hyperparameters as theirs. To summarize, we sample only 5 negatives for loss eq 4, use context window of size 7 (that is the farthest context node is 7 hops away), use SparseAdam optimizer and train for 5 epochs.

	Metapaths
Field(F)	FPVPF, FPBPF, FPAPF, FPFFPF, FPFFFPF, FPFFFFPF
Bio-entity(B)	BPVPB, BPAPB, BPFPB
Author(A)	APVPA, APBPA, APFPA
Venue (V)	VPAPV, VPBPV, VPFPV

Table 1:

4.2 HGT

We use the official codebase provided here [4] and use mostly the same hyperparameters except for few. We use Biolink-BERT-large[6] on title of papers to create input representations of paper nodes. We also skip including number of citations information in the input node features as it might add to the bias. The input features of rest of the nodes is computed by averaging the node features of the paper nodes they are connected with.

Since HGTis a supervised model, we need a supervised task to train the model. In this work, we use Paper-Label Classification task. This is a multi-label multi-class classification task wherein the labels are specific MeSH terms. To be precise, these MeSH terms are level 1 or level 2 in the hierarchy of MeSH terms. During training, we remove all the label MeSH terms from the training subgraphs.

5 Results

Coming back to our original task, we predict the extent of future collaboration between authors using eq 7 and then pick the top 3 recommendations with the largest scores.

$$
norm\text{-}sim(u,v) = \frac{e_u.e_v}{\|e_u\|_2.\|e_v\|_2} \tag{7}
$$

[3] https://github.com/dmlc/dgl/tree/master/examples/pytorch/metapath2vec

[4] https://github.com/UCLA-DM/pyHGT

$$\text{AJIOF}(u, v) = \frac{\text{Journal}(u) \cap \text{Journal}(v)}{\text{Journal}(u)} \tag{8}$$

where Journal means the set of journals the author has published in.

However since we don't have any ground truth, we can't objectively evaluate quality of our predictions. Instead we use some some other evaluation metric, specifically Author-Journal-IOU (AJ-IOU). AJ-IOUis defined as follows

For every author in a set of 10k most cited authors, we make top 3 recommendations and then compute the AJ-IOUscore between the author for which recommendation is made (client author) and each of the 3 recommendations. We then average all these AJ-IOUscores and report in this table 2.

As one can see, MP2Vends up performing better than HGT for both top-1 and top-3 recommendations.

	AJ-IOU Top-1	AJ-IOU Top-3
MP2V	0.168	0.163
HGT	0.115	0.109

Table 2: Averaged AJ-IOUmetric scores on both MP2Vand HGT. The left column means that we only consider top-1 recommendation for every of the most cited 10k authors. While in the right column we consider top-3 recommendations for every author.

Acknowledgments and Disclosure of Funding

References

[1] Jian Xu et al. "Building a PubMed knowledge graph". In: *Scientific data* 7.1 (2020), pp. 1–15.

[2] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. "metapath2vec: Scalable Representation Learning for Heterogeneous Networks". In: *KDD '17*. ACM. 2017, pp. 135–144.

[3] Ziniu Hu et al. "Heterogeneous Graph Transformer". In: *Proceedings of The Web Conference 2020*. WWW '20. Taipei, Taiwan: Association for Computing Machinery, 2020, pp. 2704–2710. ISBN: 9781450370233. DOI: 10.1145/3366423.3380027. URL: https://doi.org/10.1145/3366423.3380027.

[4] Petar Veličković et al. "Graph Attention Networks". In: *International Conference on Learning Representations* (2018). URL: https://openreview.net/forum?id=rJXMpikCZ.

[5] Ashish Vaswani et al. "Attention is All You Need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6000–6010. ISBN: 9781510860964.

[6] Michihiro Yasunaga, Jure Leskovec, and Percy Liang. "LinkBERT: Pretraining Language Models with Document Links". In: *Association for Computational Linguistics (ACL)*. 2022.