MetaFill: Text Infilling for Meta-Path Generation on Heterogeneous Information Networks

Zequn Liu[1], **Kefei Duan**[2], **Junwei Yang**[1], **Hanwen Xu**[3], **Ming Zhang**[1]*, **Sheng Wang**[3]*
[1]School of Computer Science, Peking University, Beijing, China
[2]School of EECS, Peking University, Beijing, China
[3]Paul G. Allen School of Computer Science and Engineering, University of Washington, Seattle, WA
zequnliu,dkf4116,yjwtheonly,mzhang_cs@pku.edu.cn
xuhw,swang@cs.washington.edu

Abstract

Heterogeneous Information Network (HIN) is essential to study complicated networks containing multiple edge types and node types. Meta-path, a sequence of node types and edge types, is the core technique to embed HINs. Since manually curating meta-paths is time-consuming, there is a pressing need to develop automated meta-path generation approaches. Existing meta-path generation approaches cannot fully exploit the rich textual information in HINs, such as node names and edge type names. To address this problem, we propose MetaFill, a text-infilling-based approach for meta-path generation. The key idea of MetaFill is to formulate meta-path identification problem as a word sequence infilling problem, which can be advanced by Pretrained Language Models (PLMs). We observed the superior performance of MetaFill against existing meta-path generation methods and graph embedding methods that do not leverage meta-paths in both link prediction and node classification on two real-world HIN datasets. We further demonstrated how MetaFill can accurately classify edges in the zero-shot setting, where existing approaches cannot generate any meta-paths. MetaFill exploits PLMs to generate meta-paths for graph embedding, opening up new avenues for language model applications in graph analysis.[1]

1 Introduction

Heterogeneous Information Network (HIN) is an effective framework to model complicated real-world network data (Sun et al., 2011; Wang et al., 2019; Dong et al., 2017; Shi et al., 2016; Yang et al., 2020; Fu et al., 2017; Yun et al., 2019; Hu et al., 2020). In contrast to a conventional network (Tang et al., 2015; Grover and Leskovec, 2016; Perozzi et al., 2014; Hamilton et al., 2017; Kipf and Welling, 2016; Veličković et al., 2018), HIN supports multiple node types and edge types, thus facilitating the integrative analysis of multiple datasets (Chen et al., 2012; Himmelstein and Baranzini, 2015; Zhao et al., 2020). One of the most important applications on HIN is to discover interactions between different node types by framing it as a link prediction problem (Fu et al., 2020; Zhang et al., 2014; Cao et al., 2017). Link prediction is particularly challenging on heterogeneous and long-distance node pairs, which often do not share any neighbors, thus presenting substantial false-negative predictions by conventional network-based approaches (Shi et al., 2014; Fu et al., 2016; Daud et al., 2020).

HIN exploits meta-paths to address link prediction, especially for heterogeneous and long-distance node pairs. A meta-path is a sequence of node types and edge types. A good meta-path often consists of paths that frequently appear in a HIN, thus guiding the HIN to focus on these important paths in a large network. Since manually curating meta-paths requires domain expertise, automated meta-path generation approaches have become essential for link prediction on HINs (Yang et al., 2018; Wan et al., 2020; Shi and Weninger, 2014a; Meng et al., 2015; Wang et al., 2018; Deng et al., 2021; Wei et al., 2018; Zhong et al., 2020; Ning et al., 2021). Despite their sophisticated design to leverage the network topological features, existing approaches largely overlook the rich textual information on nodes and edges. In fact, real-world HINs contain rich textual information (Yang et al., 2022; Liu et al., 2022; Xu and Wang, 2022), such as node name, node type name and edge type name, which are often the key evidence for human experts to curate meta-paths.

In this paper, we propose to identify meta-paths using pretrained language models (PLMs) (Kenton and Toutanova, 2019; Radford et al., 2019; Liu et al., 2019; Beltagy et al., 2019; Gu et al., 2020) in

*Corresponding author
[1]Our code is available at https://github.com/zequnl/MetaFill

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 5110 - 5122
December 7-11, 2022

order to explicitly utilize the rich textual information in HINs. The key idea of our method MetaFill is to form the meta-path identification problem as a text infilling problem (Zhu et al., 2019). In effect, this converts a graph-based approach to an NLP problem, enabling us to enjoy a variety of new techniques developed along with PLMs. Specifically, MetaFill samples many paths from the HIN according to the PLM-based probability for a word sequence consisting of node names and edge type names on that path, then aggregates these paths into meta-paths by a novel context-aware node type classifier.

We evaluated our method on two large-scale HINs and observed substantial improvement on link predictions compared to existing meta-path generation approaches under two meta-path-based link prediction frameworks. Furthermore, we found that the improvement of our method is larger with the decreasing of training data size, indicating the ability of compensating data sparsity using textual information. In addition to link prediction, our method also achieved prominent results on node classification. Collectively, we demonstrate how language model can be used to accurately generate meta-paths on HINs, opening up new venues for heterogeneous graph analysis using language models.

2 Preliminaries

Heterogeneous information network (HIN) is a network that contains multiple node types and edge types (Shi et al., 2016; Yang et al., 2020). Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a HIN, where $\mathcal{V} = \{v_i\}$ is the set of nodes and $\mathcal{E} = \{e_i\} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges. Each node $v \in \mathcal{V}$ is associated with a node type $a \in \mathcal{A}$. Each edge $e \in \mathcal{E}$ is associated with an edge type $r \in \mathcal{R}$.

There are three kinds of textual information in most HINs. 1) **Node name**: the textual description of a node v (e.g., *breast cancer*). 2) **Node type name**: the textual description of a node type a (e.g., *disease*). 3) **Edge type name**: the textual description of an edge type r (e.g., *treated by*). Some HINs might also have edge names. While we do not consider edge names in this paper, they can be easily incorporated into our framework. Most of the conventional HIN embedding approaches do not fully exploit this rich textual information. We aim to use language models to incorporate textual information into HIN modeling.

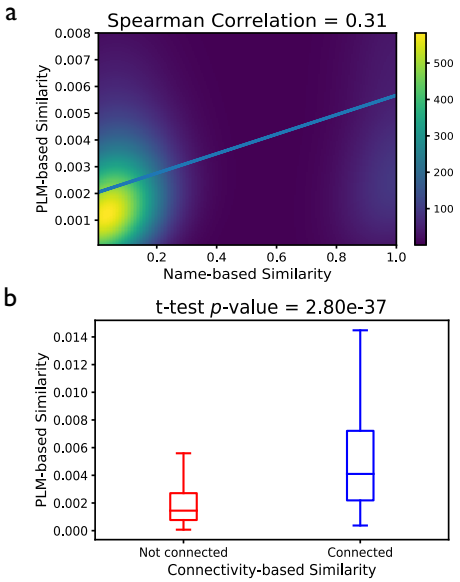

Figure 1: **Hypothesis validation.** Pretrained language model can be used to predict node name similarity (**a**) and connectivity (**b**).

Meta-path is one of the most effective techniques for embedding HINs through explicitly modeling heterogeneous and long-distance semantic similarity (Dong et al., 2017; Wang et al., 2019; Fu et al., 2020). An $l-$hop meta-path Ω is defined as a sequence $a_1 \xrightarrow{r_1} a_2 \xrightarrow{r_2} \cdots \xrightarrow{r_l} a_{l+1}$, where a_i is a node type and r_i is an edge type. Each meta-path could have many path instances on a HIN. Let $\mathcal{P} = v_1 \xrightarrow{e_1} v_2 \xrightarrow{e_2} \cdots \xrightarrow{e_l} v_{l+1}$. Then $\mathcal{P}$ is a path instance of Ω, if a_i is the node type of v_i and r_i is the edge type of e_i. HIN embedding is often performed through repeatedly sampling a path in the graph and then optimizing the embedding of each node along this path. Meta-paths could be used to record prior knowledge and then encourage the sampling process to focus on the path that is an instance of curated meta-paths.

3 Hypothesis Validation

Our hypothesis is that PLMs can be used to identify important meta-paths based on the textual information along each path. Since PLMs have shown to contain a great amount of real-world knowledge (Petroni et al., 2019), they might also be used to extract meta-paths similar to expert curations. We sought to validate this hypothesis using a widely-used HIN dataset NELL(Mitchell et al., 2018). In particular, we randomly sampled 1000 2-hop paths in NELL. For each path, we calculated a PLM-based similarity score, a name-based

similarity score and a connectivity-based similarity score. The PLM-based similarity score concatenated node names and edge type names along a given path as a word sequence and then obtained a probability for this sequence using GPT-2. The name-based similarity score calculated the textual similarity between the node names of the starting node and the end node using GPT-2 embeddings. The connectivity-based similarity score is 1 if two nodes are connected and 0 otherwise.

We first compared PLM-based similarity score and name-based similarity score and observed a substantial agreement of Spearman correlation 0.31 (**Fig. 1 a**). This indicates that PLM is able to find nodes with similar node names. Next, we found that PLM-based similarity score is also highly predictive of the connectivity-based similarity score (**Fig. 1 b**), demonstrating the possibility to predict missing links using a PLM. Collectively, the language model probability of a path is predictive of nodes similarity and connectivity, suggesting the opportunity to find meta-paths using PLMs.

4 Methodology

The key idea of our approach is to form meta-path identification as a text infilling problem (Zhu et al., 2019) and then exploit PLMs to fill in the node names and edge type names that best reflect the graph information. These word sequences then form paths on the HIN and are then aggregated into meta-paths using a context-aware node type classifier (**Fig. 2**).

4.1 Sampling paths through text infilling

To sample a path, we first sample two connected nodes v_h and v_t from the HIN. For notation simplicity, we denote v_h and v_t as the associated node name on v_h and v_t. We then sample an l-hop path from v_h to v_t by using the following templates:

$$v_h\ [\texttt{MASK}_1^E]\ [\texttt{MASK}_1^V].\ \text{It}\ [\texttt{MASK}_2^E]\ [\texttt{MASK}_2^V].\ \text{It}\ \cdots\ [\texttt{MASK}_l^E]\ v_t,$$

where $[\texttt{MASK}_i^E]$ is the edge type name mask for the i-th edge and $[\texttt{MASK}_i^V]$ is the node name mask for the i-th node. v_h and v_t guide the model to fill in the edge type name for $[\texttt{MASK}_i^E]$ and node name for $[\texttt{MASK}_i^V]$. To fill in $[\texttt{MASK}_i^E]$, we initialize it with a common edge type "relates to" and then update it iteratively according to the nearby $[\texttt{MASK}_{i-1}^V]$.

We fill in $[\texttt{MASK}_i^V]$ based on two intuitions. First, the node name that is filled in by the language model is preferred to be an existing node in the HIN. Second, the type of node i should form a valid path connecting $[\texttt{MASK}_{i-1}^V]$ and $[\texttt{MASK}_{i+1}^V]$ through $[\texttt{MASK}_i^E]$ and $[\texttt{MASK}_{i+1}^E]$. For example, for path [breast cancer][is treated by][$\texttt{MASK}^V$], we need to fill in a drug name rather than a disease name. For the first intuition, we propose to fine-tune the PLM using the HIN. For the second intuition, we propose to train a context-aware node type classifier to predict the node type on-the-fly.

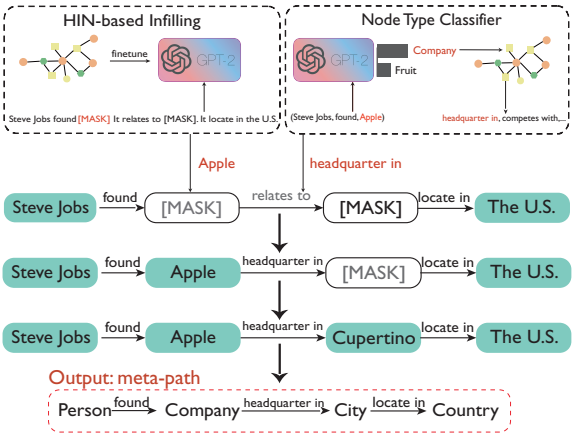

Figure 2: **Flowchart of MetaFill.** MetaFill leverages two GPT-2 to sample paths from heterogeneous information network. One GPT-2 fills in the masked node name and is fine-tuned using the HIN. The other GPT-2 classifies the node type for the filled token. These paths are then aggregated into meta-paths.

4.2 Fine-tuning the PLM using the HIN

We fine-tune the PLM using the HIN to increase the probability that the infilled text is a valid node name in the HIN. Let e be an edge in the HIN. v_h and v_t be the node names of the two nodes connected by e, and r be the edge type name of e. We construct four templates as:

$$v_h \text{ relates to } [\texttt{MASK}],$$
$$[\texttt{MASK}] \text{ relates to } v_t,$$
$$v_h\ r\ [\texttt{MASK}],$$
$$[\texttt{MASK}]\ r\ v_t.$$

The first two templates are edge-type-agnostic templates and the last two templates are edge-type-specific templates.

We follow (Donahue et al., 2020) to use GPT-2 to infill this template. GPT-2 takes the concatenated $x[\texttt{SEP}]y$ as input, where x is the masked template and y is the target tokens. For each template, we define x as the masked sentence templates and y as the correct word tokens. All edges in the HIN

are used to fine-tune the PLM. We decide to use "relates to" in the first two templates in order to make the model be compatible with the initialization using "relates to" in the previous path sampling infilling.

4.3 Context-aware node type classifier

For the second intuition, we propose to train a context-aware node type classifier. First, despite fine-tuning the HIN using a PLM, the infilled node name is still not guaranteed to be the name of an existing node in the HIN. Second, even if the infilled node name is in the HIN, it might have an ambiguous node type (e.g., apple as a fruit or a company). Therefore, we need a node type classifier to obtain its node type according to the nearby edge type names and node names. To this end, we train a GPT-2-based classifier which can predict a node type given the textual content.

Specifically, for node v_i, we calculate its node feature embedding as:

$$\mathbf{h}_i = \mathrm{GPT}(v_i). \tag{1}$$

Let $\mathcal{E}_i$ be all the edges that connect v_i. Let $e \in \mathcal{E}_i$ connect v_j and v_i. Let a be the edge type of e. We calculate a context feature embedding $\mathbf{h}_e$ as:

$$\mathbf{h}_e = \mathrm{GPT}(v_j \texttt{ [SEP] } a \texttt{ [SEP] } v_i). \tag{2}$$

A context-aware node type classifier is then trained using $\mathbf{h}_i$ and $\mathbf{h}_e$ as:

$$\begin{aligned} \mathbf{c}_{i,e} &= \mathrm{Softmax}(\mathbf{W}_1(\mathbf{h}_i||\mathbf{h}_e) + \mathbf{b}_1), \\ \mathcal{L} &= -\sum_{v_i \in \mathcal{V}} \sum_{e \in \mathcal{E}_i} \sum_{k \in K} \mathbf{a}_{i,e}[k] \log \mathbf{c}_{i,e}[k], \end{aligned} \tag{3}$$

where $\mathbf{W}_1$ and $\mathbf{b}_1$ are trainable parameters. $\mathbf{c}_{i,e}$ is a predicted vector of node types on node v_i according to edge e. $\mathbf{a}_{i,e}$ is the observed one-hot vector of node types on node v_i. K is the number of node types. We fine-tune GPT-2 while training this classifier.

To better fine-tune GPT-2, we further introduce a similar task of neighbour node type prediction, which predicts the node type of the neighbor v_j:

$$\begin{aligned} \mathbf{c}_{j,e} &= \mathrm{Softmax}(\mathbf{W}_2(\mathbf{h}_j||\mathbf{h}_e) + \mathbf{b}_2), \\ \mathcal{L}_{ngh} &= -\sum_{v_i \in \mathcal{V}} \sum_{e \in \mathcal{E}_i} \sum_{k \in K} \mathbf{a}_{j,e}[k] \log \mathbf{c}_{j,e}[k], \end{aligned} \tag{4}$$

where $\mathbf{W}_2$ and $\mathbf{b}_2$ are trainable parameters. $\mathbf{a}_{j,e}$ is the observed one-hot vector of node types on node v_j.

The final loss function is:

$$\mathcal{L}_{node} = \mathcal{L} + \lambda \mathcal{L}_{ngh}, \tag{5}$$

where λ is a hyperparameter.

4.4 Meta-path induction

We can now sample many paths using text infilling. Each time, we first sample a valid edge type for $[\texttt{MASK}_1^E]$ and $[\texttt{MASK}_l^E]$. Then we start the text infilling from v_h to v_t. When a node name is filled in, node type classifier is used to predict the node type based on this node name, the previous node name and the previous edge type name. This node type is then used to guide the sampling for the next edge type in order to maintain a valid path on the path.

We sample paths of a variety of length. After many paths are sampled, we will use the node type classifier to convert each path to a meta-path. We then rank all meta-paths by the frequency and select the most frequent q paths. For a fair comparison, we set q to the number of meta-paths that comparison approaches have generated in our experiments.

4.5 Meta-path-based predictions

After obtaining meta-paths, we can apply them to meta-path-based graph embedding frameworks (Dong et al., 2017; Wang et al., 2019). These frameworks take a heterogeneous graph $\mathcal{G}$ and q generated meta-paths as inputs, and then output an embedding vector $\mathbf{e}$ for each node. The learned node embeddings can be used for link prediction and node classification.

Link prediction. To classify edges into edge type r, the loss function is defined as:

$$\begin{aligned} \mathcal{L} = &-\sum_{(u,v) \in E_r} \log \sigma\left(\mathbf{e}_u^{\mathrm{T}} \cdot \mathbf{e}_v\right) \\ &-\sum_{(u',v') \in {E_r}^-} \log \sigma\left(-\mathbf{e}_{u'}^{\mathrm{T}} \cdot \mathbf{e}_{v'}\right), \end{aligned} \tag{6}$$

where $\sigma(\cdot)$ is the sigmoid function, $\mathbf{e}_u$ is the learned node embedding for node u, E_r is the node pairs with edge type r, ${E_r}^-$ is the set of negative node pairs.

Node classification. The loss function for node classification is the cross entropy loss:

$$\begin{aligned} \mathbf{z}_v &= \mathrm{Softmax}(\mathbf{W}_3\mathbf{e}_v + \mathbf{b}_3), \\ \mathcal{L} &= -\sum_{v \in \mathcal{V}_L} \sum_{c=1}^{C} \mathbf{z'}_v[c] \log \mathbf{z}_v[c], \end{aligned} \tag{7}$$

where $\mathcal{V}_L$ is the set of nodes that have labels, C is the number of classes, $\mathbf{e}_v$ is the learned node embedding for node v, $\mathbf{z'}_v$ is the ground-truth label vector and $\mathbf{z}_v$ is the predicted probability vector, and $\mathbf{W}_3$ and $\mathbf{b}_3$ are trainable parameters.

5 Results

5.1 Experimental setup

Datasets and tasks. We evaluated our method on two text-rich HINs, HeteroGraphine and NELL (Mitchell et al., 2018). HeteroGraphine is a biomedical HIN constructed based on expert-curated biomedical ontology collection (Liu et al., 2021a). We combined ontologies from five subdisciplines, including "uberon", "pato", "cteno", "ceph" and "ro", and treated each subdiscipline as a node type. There could be edges between the same node type and between two different node types. HeteroGraphine consists of 17,317 nodes, 41,329 edges, 5 node types and 118 edge types. NELL is a HIN extracted from internet web text. We follow Wan et al. (2020) to remove the triples with the relation "generalizations", which correspond to redundant node type information. NELL consists of 77,455 nodes, 384,275 edges, 281 node types and 830 edge types. Given the large number of node types and edge types in both datasets, manually curating meta-paths is hard, necessitating automated meta-path generation approaches.

We studied both link prediction and node classification. For link prediction task, we chose to predict edge type "develops from" in HeteroGraphine and chose to predict "competes with" in NELL, following previous work (Wan et al., 2020). The ratio of positive edges is 50%. Since NELL does not have node labels, we studied node classification on HeteroGraphine only. We used the "subset" information in each ontology as the label and evaluate node classification on uberon and pato. Subset labels can be regarded as the scientific field for each node. uberon has 6 classes and pato has 7 classes.

Comparison approaches. We compared our method with five meta-path generation methods. **Discrmetapath** (Shi and Weninger, 2014a) is a searching-based method, which prioritizes meta-paths that can separate a path from its sibling paths. **GTN** (Yun et al., 2019) and **HGT** (Hu et al., 2020) are attention-based methods. They don't explicitly output meta-path, but meta-paths could be induced from the combination of edge attention scores. **AutoPath** (Yang et al., 2018) and **MPDRL** (Wan et al., 2020) are reinforcement learning-based (RL-based) methods which train an agent to traverse on the graph. Discrmetapath, MPDRL and Autopath did not use textual information. We aim to assess the importance of using textual information by comparing MetaFill to them. GTN and HGT explicitly considered the textual information by pooling PLM-based word embeddings. We aim to examine the effectiveness of text infilling against simple embedding pooling by comparing MetaFill to them. We further implemented a variant of our model **MetaFill w/o fine-tuning** to investigate the impact of fine-tuning the PLM using HIN. We did not report the meta-path generation results of GTN on both datasets because it cannot generate any valid meta-paths there, and did not report the results of HGT on NELL because it cannot scale to such a schema-rich HIN.

Meta-path-based link prediction framework. Our method and the comparison approaches can automatically generate meta-paths. We then fed these meta-paths to a meta-path-based HIN embedding framework. We evaluated two widely-used frameworks: **Metapath2Vec** (Dong et al., 2017) and **HAN** (Wang et al., 2019). Each of them provides us the node embeddings, which are then used for link prediction and node classification. **Metapath2Vec** formalizes meta-path-based random walks and then leverages a skip-gram model to optimize node embeddings. **HAN** aggregates node features from meta-path-based neighbors to get node embeddings.

Comparison approaches that do not use meta-paths. We also compared to methods that do not use meta-paths to demonstrate the importance of meta-path. For link prediction, we compared our method to heterogeneous graph neural network embedding **GTN** (Yun et al., 2019). For node classification, we compared our method to multi-layer perceptron (**MLP**), homogeneous graph neural network embedding **GraphSAGE** (Hamilton et al., 2017), and heterogeneous graph neural network embedding **GTN**. We used **AUC** and **AP** to evaluate link prediction. We used **micro-F1** and **macro-F1** to evaluate node classification. More implementation details can be found in **Appendix A.2**.

5.2 Improved performance on link prediction

We summarized the performance on link prediction in **Table 1**. We found that our method obtained the best results on both datasets under both meta-

Method	HeteroGraphine		NELL	
	AUC	AP	AUC	AP
GTN	0.6352	0.6558	-	-
Metapath2Vec				
Discrmetapath	0.5425	0.5711	0.4899	0.5596
HGT	0.5322	0.5869	-	-
AutoPath	0.5158	0.5723	0.5420	0.6120
MPDRL	0.4833	0.5188	0.5133	0.5949
MetaFill w/o fine-tuning	0.6412	0.7148	0.5162	0.5905
MetaFill	**0.6598**	**0.7189**	**0.5597**	**0.6239**
HAN				
Discrmetapath	0.7621	0.7901	0.5448	0.5652
HGT	0.7690	0.7932	-	-
AutoPath	0.7556	0.7824	0.5582	0.5695
MPDRL	0.7323	0.7685	0.5423	0.5604
MetaFill w/o fine-tuning	0.7904	0.8155	0.5708	0.5833
MetaFill	**0.7985**	**0.8214**	**0.5764**	**0.5913**

Table 1: Link prediction performance on two HIN datasets (HeteroGraphine, NELL). Metapath2vec and HAN are two meta-path-based link prediction frameworks. GTN is not a meta-path based approach.

path-based link prediction frameworks. For example, MetaFill obtained at least 22% AP improvement against other meta-path generation methods on HeteroGraphine under Metapath2Vec framework. MetaFill also outperformed GTN by 22% AUC under HAN framework, indicating the importance of using meta-paths to model HINs. Under HAN framework, other meta-path generation approaches are also better than GTN, again necessitating the generation and utilization of meta-paths. We noticed that Metapath2Vec is in general worse than HAN, partially due to HAN's explicitly aggregation of neighbor features. Importantly, HGT considering textual information performed substantailly better than those do not consider textual information (e.g., Discrmetapath, AutoPath, MPDRL), confirming the benefits of incorporating textual information into HIN modeling. Finally, we observed decreased performance by our variant, where the PLM is not fine-tuned using the HIN, indicating how fine-tuning can ease the text infilling procedure and later derive more accurate meta-paths. Despite having a less superior performance, this variant is still better than all other comparison approaches on both datasets, reassuring the effectiveness of using text infilling to find meta-paths.

5.3 Improved performance on node classification

We next investigated the performance of our method on node classification. Since all of the meta-path generation comparison approaches except HGT are designed for link prediction task, we only compared to HGT and two other non-meta-path-based approaches GraphSAGE and GTN. We found that our method achieved the best performance under both the framework of Metapath2Vec and HAN (**Table. 2**). For example, the micro-F1 and macro-F1 of our model are 6.7% and 6.3% higher than HGT on pato under HAN framework. The performance of HAN is also in general better than non-meta-path-based approaches, especially on macro-F1, again demonstrating the effectiveness of meta-paths on HIN embedding. The superior performance of MetaFill on both link prediction and node classification collectively proves the effectiveness of using text filling to generate meta-paths.

5.4 Connectivity in HIN improves PLMs

We further visualized the embedding space of GPT-2 before and after fine-tuning to understand the superior performance of our method on node classification (**Fig. 3**). We observed that the quality of GPT-2 embedding improved substantially after fine-tuning on the HIN. In particular, we calculated the word embeddings of node names for nodes whose type is among "website", "clothing" and "disease". GPT-2 without fine-tuning didn't show a visible pattern for these nodes. In contrast, both GPT-2 fine-tuned using HIN and GPT-2 fine-tuned using context-aware node type classifier presented a clear pattern for these three node types. In summary, the HIN provides valuable information about the node names and node types for the language model during the fine-tuning. While PLMs facilitate the HIN embedding, the rich connectivity information in the HIN also improves the word embedding of PLMs.

5.5 Consistent improvement using fewer training data

We next investigated the performance of our method and comparison approaches using fewer training data. On both datasets, we randomly sampled 25%/50%/75% of node pairs in the original training data, and then used them to find meta-paths. We first noticed that the performance of comparison approaches dropped substantially when there are fewer training data points, indicating that they require enough training data to derive accurate meta-paths **Fig. 4**. In contrast, our method demonstrated a stable performance when fewer training data was provided. We attributed this superior performance to MetaFill's ability to exploit the rich textual infor-

Data	Metrics	Non-meta-path-based			Metapath2Vec-based		HAN-based	
		MLP	GraphSAGE	GTN	HGT	MetaFill	HGT	MetaFill
pato	micro-F1	0.5309	0.5855	0.5890	0.3734	**0.4032**	0.5747	**0.6107**
	macro-F1	0.2099	0.1236	0.1242	0.3068	**0.3177**	0.3073	**0.3266**
uberon	micro-F1	0.5679	0.5377	0.5692	0.3178	**0.4038**	0.5586	**0.5760**
	macro-F1	0.2907	0.1794	0.2914	0.2897	**0.3485**	0.3405	**0.3591**

Table 2: Node classification performance on pato and uberon. Non-meta-path-based are methods that do not use meta-paths. HAN and Metapath2Vec are two meta-path-based embedding framework.

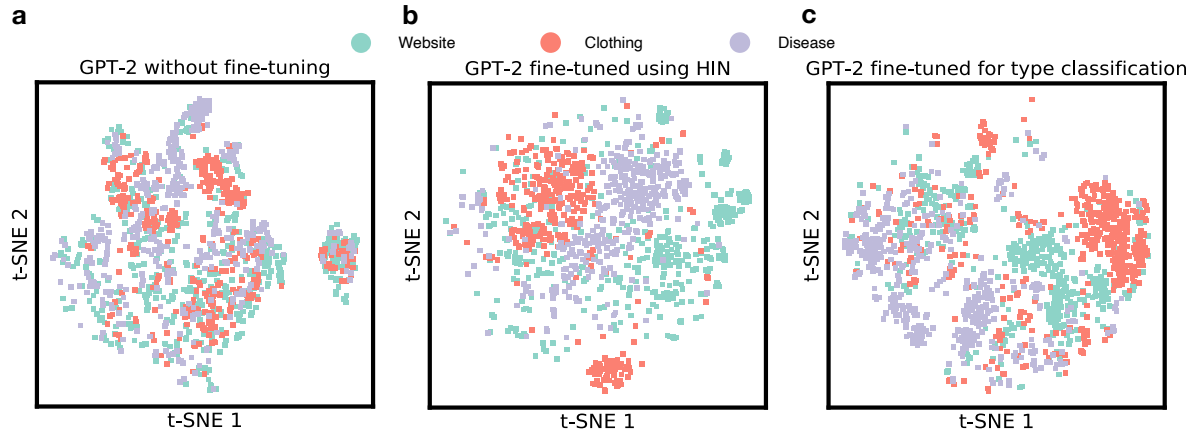

Figure 3: t-SNE plots visualizing the embedding space of GPT-2 without fine-tuning (**a**), GPT-2 fine-tuned by HIN (**b**), and GPT-2 fine-tuned by context-aware node type classifier (**c**). The fine-tuned GPT-2 present more visible patterns for three classes.

mation from the HIN, again confirming the importance of using text infilling to identify meta-paths.

5.6 Zero-shot link prediction

Our method is also able to perform link prediction in the zero-shot setting. Here, we aim to classify edges into the edge type "develops from". To study the zero-shot setting, we held-out all edges belong to this edge type in the training data. To do the link prediction, we first fed "`[MASK]` develops from `[MASK]`" into the fine-tuned GPT-2 to generate many pseudo training node pairs. These pairs are then used by MetaFill to generate meta-paths. We observed a desirable performance on both Metapath2Vec (AUC=0.6567, AP=0.7068) and HAN (AUC=0.7733, AP=0.8024) (**Fig. 4**c). Notably, these results are only slightly worse than the supervised learning setting (**Fig. 4**), highlighting the strong applicability of our method.

5.7 A case study of the generated meta-paths

Finally, we presented a case study for the link prediction task on HeteroGraphine to examine the meta-paths generated by MetaFill. For a test edge *(ceratobranchial 5 tooth, develops from, tooth enamel organ)*, our model generated the following meta-path:

$$\text{uberon} \xrightarrow{\text{is a}} \text{uberon} \xrightarrow{\text{has developmental contribution from}} \text{uberon}.$$

This meta-path helps the meta-path-based framework found the following path:

$$\text{ceratobranchial 5 tooth} \xrightarrow{\text{is a}} \text{calcareous tooth} \xrightarrow{\text{has developmental contribution from}} \text{tooth enamel organ},$$

which enables us to correctly predict this edge. None of the comparison approaches identify this meta-path. Thus, they cannot correctly predict this edge. $uberon$ represents biological structures in anatomy. This meta-path conforms the domain knowledge that a biological structure v_i is more likely to develop from another structure v_j if its parent node has developmental contribution from v_j, indicating the consistency between the meta-paths generated by our model and those curated by domain experts.

6 Related Work

Automatic meta-path selection and identification are emerging research problems due to their importance in modeling HIN. Searching-based methods enumerate all the meta-paths (Wang et al., 2018;

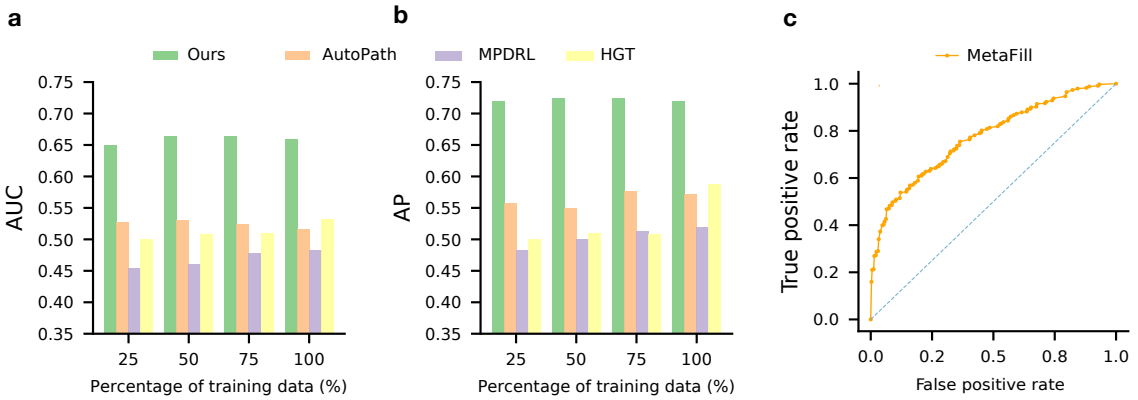

Figure 4: **Evaluation on the low-resource setting. a,b**, Performance on HeteroGraphine using fewer training data, evaluated by AUC (**a**) and AP (**b**). **c**, AUROC curve of zero-shot prediction using MetaFill (AUC=0.7733).

Deng et al., 2021; Wei et al., 2018) or expand meta-paths iteratively with searching algorithm such as greedy tree (Meng et al., 2015; Cao et al., 2016; Zheng et al., 2017), k-shortest path (Shi and Weninger, 2014b) and A* (Zhu et al., 2018), and rank them by some pre-defined metrics. Reinforcement learning-based methods train an agent to walk on the graph and induce meta-paths from the trajectories (Ning et al., 2021; Wan et al., 2020; Yang et al., 2018; Zhong et al., 2020). Attention -based methods use the attention score of edges (Yun et al., 2019; Hu et al., 2020; Wang et al., 2020b) or meta-paths (Li et al., 2021; Hu et al., 2018; Zhang and Zhu, 2021; Liang and Liu, 2020) in GNN with attention mechanism to evaluate the importance of meta-paths. Compared to searching-based methods, our approach can leverage not only the graph structure but also the textual information. In contrast to RL-based methods and attention-based methods, our approach avoid the information loss caused by pooling the textual content into a node embedding vector, and more effectively utilize the knowledge in PLMs.

Language models have been used to build and complement knowledge bases (AlKhamissi et al., 2022). Previous works extract knowledge from PLMs for fact probing (Petroni et al., 2019; Davison et al., 2019; Jiang et al., 2020b,a; Adolphs et al., 2021; Zhong et al., 2021; Qin and Eisner, 2021; Kassner et al., 2021; Shin et al., 2020; Dhingra et al., 2021; Liu et al., 2021b; Sung et al., 2021), semantic probing (Gao et al., 2021; Shin et al., 2020; Beloucif and Biemann, 2021), reasoning (Talmor et al., 2020; Gao et al., 2022), planning (Huang et al., 2022) and knowledge graph construction (Yao et al., 2019; Wang et al., 2020a). The key difference between our work and these studies is that we aim at finding meta-paths rather than a specific path or edge in the knowledge graph. Aggregating paths into meta-paths is non-trivial due to the potential invalid paths and ambiguous node types. Our ablation study that showed the importance of fine-tuning PLMs using HIN also confirmed this. To the best of our knowledge, our work is the first application of language models for meta-path generation.

7 Conclusion and Future Work

In this paper, we have proposed a novel text-infilling-based meta-path generation method. The key idea of our method is to form meta-path generation as a text infilling problem and sample important paths from a HIN by using PLMs to generate sequences of node names and edge type names. We have evaluated our method on two datasets under two meta-path-based HIN embedding frameworks, and obtained the best performance on node classification and link prediction. The improvement of our method is also consistent in low-resource settings. In addition, we found that fine-tuning the PLM using a HIN can further improve word embeddings, again indicating how our method creates an synergistic effort between HINs and PLMs.

To the best of our knowledge, our method is the first attempt to apply PLMs to meta-path generation. Since PLMs are constructed on large-scale real-word text corpus, they often contain rich real-world knowledge. We envision that our method will motivate future research in investigating how PLMs can be used to advance other graph analysis problems.

Acknowledgement

This paper is partially supported by National Key Research and Development Program of China with Grant No. 2018AAA0101902 and the National Natural Science Foundation of China (NSFC Grant Numbers 62106008 and 62276002).

Limitations

We currently have identified three limitations for our paper. First, the edge type names and node names are generated greedily for computational efficiency, introducing accumulative errors. We plan to implement beam search to alleviate this problem in the future. Second, we only generate meta-paths to connect positive connected pairs but overlook the difference between positive and negative pairs. We plan to exploit contrastive learning techniques to maximize the probability of meta-paths that connect positive pairs, while minimizing the probability of meta-paths that connect negative pairs. Finally, our method has currently only been applied to link prediction and node classification. It is also important to evaluate other graph-based tasks such as node clustering, graph-to-text generation, to thoroughly evaluate our method.

References

Leonard Adolphs, Shehzaad Dhuliawala, and Thomas Hofmann. 2021. How to query language models? *arXiv preprint arXiv:2108.01928*.

Badr AlKhamissi, Millicent Li, Asli Celikyilmaz, Mona Diab, and Marjan Ghazvininejad. 2022. A review on language models as knowledge bases. *arXiv preprint arXiv:2204.06031*.

Meriem Beloucif and Chris Biemann. 2021. Probing pre-trained language models for semantic attributes and their values. In *EMNLP*, pages 2554–2559. Association for Computational Linguistics.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. pages 3615–3620.

Xiaohuan Cao, Yuyan Zheng, Chuan Shi, Jingzhi Li, and Bin Wu. 2016. Link prediction in schema-rich heterogeneous information network. In *Pacific-asia conference on knowledge discovery and data mining*, pages 449–460. Springer.

Xiaohuan Cao, Yuyan Zheng, Chuan Shi, Jingzhi Li, and Bin Wu. 2017. Meta-path-based link prediction in schema-rich heterogeneous information network. *International Journal of Data Science and Analytics*, 3(4):285–296.

Xing Chen, Ming-Xi Liu, and Gui-Ying Yan. 2012. Drug–target interaction prediction by random walk on the heterogeneous network. *Molecular BioSystems*, 8(7):1970–1978.

Nur Nasuha Daud, Siti Hafizah Ab Hamid, Muntadher Saadoon, Firdaus Sahran, and Nor Badrul Anuar. 2020. Applications of link prediction in social networks: A review. *Journal of Network and Computer Applications*, 166:102716.

Joe Davison, Joshua Feldman, and Alexander M. Rush. 2019. Commonsense knowledge mining from pretrained models. In *EMNLP-IJCNLP*, pages 1173–1178. Association for Computational Linguistics.

Weiwei Deng, Wei Du, and Cong Han. 2021. Incorporating heterogeneous information in deep learning with informative meta-paths for community recommendations. *Journal of Information Science*, page 01655515211047423.

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2021. Time-aware language models as temporal knowledge bases. *CoRR*, abs/2106.15110.

Chris Donahue, Mina Lee, and Percy Liang. 2020. Enabling language models to fill in the blanks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2492–2501.

Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 135–144.

Gang Fu, Ying Ding, Abhik Seal, Bin Chen, Yizhou Sun, and Evan Bolton. 2016. Predicting drug target interactions using meta-path-based semantic network analysis. *BMC bioinformatics*, 17(1):1–10.

Tao-yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1797–1806.

Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. 2020. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of The Web Conference 2020*, pages 2331–2341.

Daniel Gao, Yantao Jia, Lei Li, Chengzhen Fu, Zhicheng Dou, Hao Jiang, Xinyu Zhang, Lei Chen, and Zhao Cao. 2022. KMIR: A benchmark for evaluating knowledge memorization, identification and reasoning abilities of language models. *CoRR*, abs/2202.13529.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *ACL/IJCNLP*, pages 3816–3830. Association for Computational Linguistics.

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*, pages 855–864.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2020. Domain-specific language model pretraining for biomedical natural language processing. *arXiv preprint arXiv:2007.15779*.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Daniel S Himmelstein and Sergio E Baranzini. 2015. Heterogeneous network edge prediction: a data integration approach to prioritize disease-associated genes. *PLoS computational biology*, 11(7):e1004259.

Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S Yu. 2018. Leveraging meta-path based context for top-n recommendation with a neural co-attention model. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1531–1540.

Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous graph transformer. In *Proceedings of The Web Conference 2020*, pages 2704–2710.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. *CoRR*, abs/2201.07207.

Zhengbao Jiang, Antonios Anastasopoulos, Jun Araki, Haibo Ding, and Graham Neubig. 2020a. X-FACTR: multilingual factual knowledge retrieval from pretrained language models. In *EMNLP*, pages 5943–5959. Association for Computational Linguistics.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020b. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Nora Kassner, Philipp Dufter, and Hinrich Schütze. 2021. Multilingual LAMA: investigating knowledge in multilingual pretrained language models. In *EACL*, pages 3250–3258. Association for Computational Linguistics.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks.

Yi Li, Yilun Jin, Guojie Song, Zihao Zhu, Chuan Shi, and Yiming Wang. 2021. Graphmse: Efficient meta-path selection in semantically aligned feature space for graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4206–4214.

Tao Liang and Jin Liu. 2020. Meta-path generation online for heterogeneous network embedding. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Fenglin Liu, Bang Yang, Chenyu You, Xian Wu, Shen Ge, Adelaide Woicik, and Sheng Wang. 2022. Graph-in-graph network for automatic gene ontology description generation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1060–1068.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Zequn Liu, Shukai Wang, Yiyang Gu, Ruiyi Zhang, Ming Zhang, and Sheng Wang. 2021a. Graphine: A dataset for graph-aware terminology definition generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3453–3463.

Zeyu Liu, Yizhong Wang, Jungo Kasai, Hannaneh Hajishirzi, and Noah A. Smith. 2021b. Probing across time: What does roberta know and when? In *EMNLP*, pages 820–842. Association for Computational Linguistics.

Changping Meng, Reynold Cheng, Silviu Maniu, Pierre Senellart, and Wangda Zhang. 2015. Discovering meta-paths in large heterogeneous information networks. In *Proceedings of the 24th International Conference on World Wide Web*, pages 754–764.

Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi, Matt Gardner, Bryan Kisiel, et al. 2018. Never-ending learning. *Communications of the ACM*, 61(5):103–115.

Wentao Ning, Reynold Cheng, Jiajun Shen, Nur Al Hasan Haldar, Ben Kao, Nan Huo, Wai Kit Lam, Tian Li, and Bo Tang. 2021. Reinforced meta-path selection for recommendation on heterogeneous information networks. *arXiv preprint arXiv:2112.12845*.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5203–5212.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Baoxu Shi and Tim Weninger. 2014a. Mining interesting meta-paths from complex heterogeneous information networks. In *2014 IEEE International Conference on Data Mining Workshop*, pages 488–495. IEEE.

Baoxu Shi and Tim Weninger. 2014b. Mining interesting meta-paths from complex heterogeneous information networks. In *2014 IEEE International Conference on Data Mining Workshop*, pages 488–495. IEEE.

Chuan Shi, Xiangnan Kong, Yue Huang, S Yu Philip, and Bin Wu. 2014. Hetesim: A general framework for relevance measure in heterogeneous networks. *IEEE Transactions on Knowledge and Data Engineering*, 26(10):2479–2492.

Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and S Yu Philip. 2016. A survey of heterogeneous information network analysis. *IEEE Transactions on Knowledge and Data Engineering*, 29(1):17–37.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235.

Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment*, 4(11):992–1003.

Mujeen Sung, Jinhyuk Lee, Sean S. Yi, Minji Jeon, Sungdong Kim, and Jaewoo Kang. 2021. Can language models be biomedical knowledge bases? In *EMNLP*, pages 4723–4734. Association for Computational Linguistics.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. olmpics - on what language model pre-training captures. *Trans. Assoc. Comput. Linguistics*, 8:743–758.

Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pages 1067–1077.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Guojia Wan, Bo Du, Shirui Pan, and Gholameza Haffari. 2020. Reinforcement learning based meta-path discovery in large-scale heterogeneous information networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6094–6101.

Chenguang Wang, Xiao Liu, and Dawn Song. 2020a. Language models are open knowledge graphs. *CoRR*, abs/2010.11967.

Chenguang Wang, Yangqiu Song, Haoran Li, Ming Zhang, and Jiawei Han. 2018. Unsupervised meta-path selection for text similarity measure based on heterogeneous information networks. *Data Mining and Knowledge Discovery*, 32(6):1735–1767.

Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *The world wide web conference*, pages 2022–2032.

Yifan Wang, Suyao Tang, Yuntong Lei, Weiping Song, Sheng Wang, and Ming Zhang. 2020b. Disenhan: Disentangled heterogeneous graph attention network for recommendation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1605–1614.

Xiaokai Wei, Zhiwei Liu, Lichao Sun, and Philip S Yu. 2018. Unsupervised meta-path reduction on heterogeneous information networks. *arXiv preprint arXiv:1810.12503*.

Hanwen Xu and Sheng Wang. 2022. Protranslator: zero-shot protein function prediction using textual description. In *International Conference on Research in Computational Molecular Biology*, pages 279–294. Springer.

Carl Yang, Mengxiong Liu, Frank He, Xikun Zhang, Jian Peng, and Jiawei Han. 2018. Similarity modeling on heterogeneous networks via automatic path discovery. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 37–54. Springer.

Carl Yang, Yuxin Xiao, Yu Zhang, Yizhou Sun, and Jiawei Han. 2020. Heterogeneous network representation learning: Survey, benchmark, evaluation, and beyond.

Junwei Yang, Zequn Liu, Ming Zhang, and Sheng Wang. 2022. Pathway2text: Dataset and method for biomedical pathway description generation. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1441–1454.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. KG-BERT: BERT for knowledge graph completion. *CoRR*, abs/1909.03193.

Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2019. Graph transformer networks. *Advances in neural information processing systems*, 32.

Jiarui Zhang and Yonghua Zhu. 2021. Meta-path guided heterogeneous graph neural network for dish recommendation system. In *Journal of Physics: Conference Series*, volume 1883, page 012102. IOP Publishing.

Jiawei Zhang, Philip S Yu, and Zhi-Hua Zhou. 2014. Meta-path based multi-network collective link prediction. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1286–1295.

Zhongying Zhao, Xuejian Zhang, Hui Zhou, Chao Li, Maoguo Gong, and Yongqing Wang. 2020. Hetnerec: Heterogeneous network embedding based recommendation. *Knowledge-based systems*, 204:106218.

Yuyan Zheng, Chuan Shi, Xiaohuan Cao, Xiaoli Li, and Bin Wu. 2017. Entity set expansion with meta path in knowledge graph. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 317–329. Springer.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [mask]: Learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5017–5033.

Zhiqiang Zhong, Cheng-Te Li, and Jun Pang. 2020. Reinforcement learning enhanced heterogeneous graph neural network. *arXiv preprint arXiv:2010.13735*.

Wanrong Zhu, Zhiting Hu, and Eric Xing. 2019. Text infilling. *arXiv preprint arXiv:1901.00158*.

Zichen Zhu, Reynold Cheng, Loc Do, Zhipeng Huang, and Haoci Zhang. 2018. Evaluating top-k meta path queries on large heterogeneous information networks. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 1470–1475. IEEE.

A Appendix

A.1 Name-based Score for Hypothesis Validation

For a path we calculate the text similarity between its head node v_h and tail node v_t:

$$sim(v_h, v_t) = \frac{1}{1 + ||e_h - e_t||}$$

where e_h and e_t are the GPT-2 embeddings of the head node name and tail node name, $||\cdot||$ is the Euclidean distance.

A.2 Implementation Details

Metapath2Vec and HAN For Metapath2vec, the walk length is 1 on HeteroGraphine and 10 on NELL. For HAN, we use bag-of-words vector as the initialization of node embeddings. The dimension of node embeddings are both set to 128 for fair comparison and the learning rates are both 0.001. We follow the other hyperparameter settings in the orginal papers.

Link prediction and node classification We use dot product score to do link prediction after getting node embeddings. For node classification, we train HAN with the cross entropy loss function end-to-end. Since Metapath2Vec cannot do node classification directly, we train a 1-layer MLP classifier on top of the node embedding vectors. 12.5% of the training data are randomly sampled as the validation set for early stopping.

Our meta-path generation method We follow the settings in (Donahue et al., 2020) to fine-tune the GPT-2 for text-infilling, and fine-tune based on their fine-tuned model. The context-aware node type classifier is trained using early stopping and the training data and validation data are 4:1, λ is set to 1. We sample paths from 1-hop to 4-hop. Note that for 1-hop paths, no node name needs to be infilled, we only randomly sample an edge to connect v_h to v_t. We run the generation process of each node pair 10 times since there could be more than one paths connecting two nodes. To reduce the computaional cost, we sample connected nodes from a subset of the large-scale HIN. For link prediction task, the subset is the training positive edges. For node classification task, the subset is nodes with similar labels (The label similarity is calculated by the cosine similarity of the muti-hot label vector). For link prediction task, we select top-8 meta-paths for HeteroGraphine and top-23 meta-paths for NELL on all the competing methods for a fair comparison. For node classification task, we select top-6 meta-paths. All experiments are carried out on NVIDIA GeForce RTX 3090. We use 2 GPUs for finetuning and 1 GPU for infilling. The finetuning stage need 1 day and the infilling stage can be finished within 2 hours.

Baselines For **GTN**, we set the number of layers to 3 for link prediction task and set the layers for node classification task to 2, in order to adapt to the scale of the dataset. The learning rates for these two tasks are 5e-4 and 5e-6. All the other hyper parameters are same with the original official code.

For **HGT**, in link prediction task, we set the layers of HGT to 4 and the depth and width of sampling to 6 and 128 respectively. The batch size is set to 256. As to node classification task, we set the layers of HGT to 3 and the depth and width of sampling to 3 and 64. The batch size for node classification is 64. The choices of these parameters are also for the scale of datasets. The corresponding learning rate for these two tasks are 1e-3 and 1e-6. **AutoPath** and **MPDRL** are only used in the link prediction task. We just follow the official implementations without changing of hyperparameters. **MLP** use Bag of Words of nodes' names as the features for nodes to do node classification. We use three layers of MLP and use Tanh as the activation function. We set learning rate to 1e-3. For **GraphSAGE**, we also follow the official implementation. We set learning rate to 1e-1.