PATTON: Language Model Pretraining on Text-Rich Networks

Bowen Jin[1], **Wentao Zhang**[1], **Yu Zhang**[1], **Yu Meng**[1],
Xinyang Zhang[1], **Qi Zhu**[1], **Jiawei Han**[1]
[1]University of Illinois at Urbana-Champaign, IL, USA
{bowenj4,wentao4,yuz9,yumeng5,xz43,qiz3,hanj}@illinois.edu

Abstract

A real-world text corpus sometimes comprises not only text documents, but also semantic links between them (*e.g.*, academic papers in a bibliographic network are linked by citations and co-authorships). Text documents and semantic connections form a *text-rich network*, which empowers a wide range of downstream tasks such as classification and retrieval. However, pretraining methods for such structures are still lacking, making it difficult to build one generic model that can be adapted to various tasks on text-rich networks. Current pretraining objectives, such as masked language modeling, purely model texts and do not take inter-document structure information into consideration. To this end, we propose our ***P**retr**A**ining on **T**ex**T**-Rich **N**etw**O**rk* framework PATTON. PATTON[1] includes two pretraining strategies: network-contextualized masked language modeling and masked node prediction, to capture the inherent dependency between textual attributes and network structure. We conduct experiments on four downstream tasks in five datasets from both academic and e-commerce domains, where PATTON outperforms baselines significantly and consistently.

1 Introduction

Texts in the real world are often interconnected through links that can indicate their semantic relationships. For example, papers connected through citation links tend to be of similar topics; e-commerce items connected through co-viewed links usually have related functions. The texts and links together form a type of network called a *text-rich network*, where documents are represented as nodes, and the edges reflect the links among documents. Given a text-rich network, people are usually interested in various downstream tasks (*e.g.,* document/node classification, document retrieval, and link prediction) (Zhang et al., 2019; Wang et al., 2019; Jin et al., 2023a). For example, given a computer science academic network as context, it is intuitively appealing to automatically classify each paper (Kandimalla et al., 2021), find the authors of a new paper (Schulz et al., 2014), and provide paper recommendations (Küçüktunç et al., 2012). In such cases, pretraining a language model on a given text-rich network which can benefit a great number of downstream tasks inside this given network is highly demanded (Hu et al., 2020b).

While there have been abundant studies on building generic pretrained language models (Peters et al., 2018; Devlin et al., 2019; Liu et al., 2019; Clark et al., 2020), they are mostly designed for modeling texts exclusively, and do not consider inter-document structures. Along another line of research, various network-based pretraining strategies are proposed in the graph learning domain to take into account structure information (Hu et al., 2020a,b). Yet, they focus on pretraining graph neural networks rather than language models and cannot easily model the rich textual semantic information in the networks. To empower language model pretraining with network signals, LinkBERT (Yasunaga et al., 2022) is a pioneering study that puts two linked text segments together during pretraining so that they can serve as the context of each other. However, it simplifies the complex network structure into node pairs and does not model higher-order signals (Yang et al., 2021). Overall, both existing language model pretraining methods and graph pretraining methods fail to capture the rich contextualized textual semantic information hidden inside the complex network structure.

To effectively extract the contextualized semantics information, we propose to view the knowledge encoded inside the complex network structure from two perspectives: token-level and document-level. At the *token* level, neighboring documents can help facilitate the understanding of tokens. For example, in Figure 1, based on the text information of neigh-

[1]Code is available at https://github.com/PeterGriffinJin/Patton

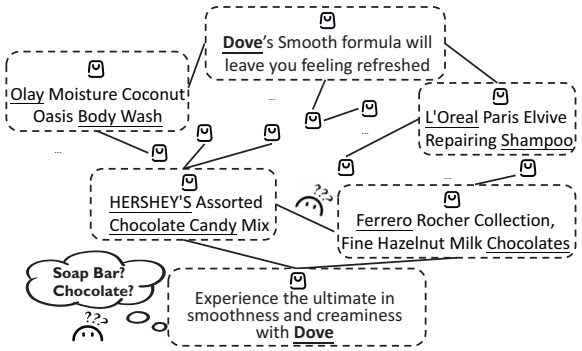


Figure 1: An illustration of a text-rich network (a product item co-viewed network). At the *token* level, from network neighbors, we can know that the "Dove" at the top is a personal care brand and the "Dove" at the bottom is a chocolate brand. At the *document* level, referring to the edge in the middle, we can learn that the chocolate from "Hershey's" should have some similarity with the chocolate from "Ferrero".

bors, we can know that the "Dove" at the top refers to a personal care brand, while the "Dove" at the bottom is a chocolate brand. At the *document* level, the two connected nodes can have quite related overall textual semantics. For example, in Figure 1, the chocolate from "Hershey's" should have some similarity with the chocolate from "Ferrero". Absorbing such two-level hints in pretraining can help language models produce more effective representations which can be generalized to various downstream tasks.

To this end, we propose PATTON, a method to continuously pretrain language models on a given text-rich network. The key idea of PATTON is to leverage both textual information and network structure information to consolidate the pretrained language model's ability to understand tokens and documents. Building on this idea, we propose two pretraining strategies: 1) Network-contextualized masked language modeling: We randomly mask several tokens within each node and train the language model to predict those masked tokens based on both in-node tokens and network neighbors' tokens. 2) Masked node prediction: We randomly mask some nodes inside the network and train the language model to correctly identify the masked nodes based on the neighbors' textual information.

We evaluate PATTON on both academic domain networks and e-commerce domain networks. To comprehensively understand how the proposed pretraining strategies can influence different downstream tasks, we conduct experiments on classification, retrieval, reranking, and link prediction.

In summary, our contributions are as follows:

- We propose the problem of language model pretraining on text-rich networks.
- We design two strategies, network contextualized MLM and masked node prediction to train the language model to extract both token-level and document-level semantic correlation hidden inside the complex network structure.
- We conduct experiments on four downstream tasks in five datasets from different domains, where PATTON outperforms pure text/graph pretraining baselines significantly and consistently.

2 Preliminaries

Definition 2.1. Text-Rich Networks (Yang et al., 2021; Jin et al., 2023b). A text-rich network can be denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{D})$, where $\mathcal{V}, \mathcal{E}$ and $\mathcal{D}$ are node set, edge set, and text set, respectively. Each $v_i \in \mathcal{V}$ is associated with some textual information $d_{v_i} \in \mathcal{D}$. For example, in an academic citation network, $v \in \mathcal{V}$ are papers, $e \in \mathcal{E}$ are citation edges, and $d \in \mathcal{D}$ are the content of the papers. In this paper, we mainly focus on networks where the edges can provide semantic correlation between texts (nodes). For example, in a citation network, connected papers (cited papers) are likely to be semantically similar.

Problem Definition. (Language Model Pretraining on Text-rich Networks.) Given a text-rich network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{D})$, the task is to capture the self-supervised signal on $\mathcal{G}$ and obtain a $\mathcal{G}$-adapted language model $\mathcal{M}_{\mathcal{G}}$. The resulting language model $\mathcal{M}_{\mathcal{G}}$ can be further finetuned on downstream tasks in $\mathcal{G}$, such as classification, retrieval, reranking, and link prediction, with only a few labels.

3 PATTON

3.1 Model Architecture

To jointly leverage text and network information in pretraining, we adopt the GNN-nested Transformer architecture (called GraphFormers) proposed in (Yang et al., 2021). In this architecture, GNN modules are inserted between Transformer layers. The forward pass of each GraphFormers layer is as follows.

$$\boldsymbol{z}_x^{(l)} = \text{GNN}(\{\boldsymbol{H}_y^{(l)}\texttt{[CLS]} | y \in N_x\}), \quad (1)$$

$$\widetilde{\boldsymbol{H}}_x^{(l)} = \text{Concate}(\boldsymbol{z}_x^{(l)}, \boldsymbol{H}_x^{(l)}), \quad (2)$$

$$\widetilde{\boldsymbol{H}}_x^{(l)\prime} = \text{LN}(\boldsymbol{H}_x^{(l)} + \text{MHA}_{asy}(\widetilde{\boldsymbol{H}}_x^{(l)})), \quad (3)$$

$$\boldsymbol{H}_x^{(l+1)} = \text{LN}(\widetilde{\boldsymbol{H}}_x^{(l)\prime} + \text{MLP}(\widetilde{\boldsymbol{H}}_x^{(l)\prime})), \quad (4)$$

where $\boldsymbol{H}_x^{(l)}$ is token hidden states in the l-th layer for node x, N_x is the network neighbor set of x, LN

is the layer normalization operation and MHA_{asy} is the asymmetric multihead attention operation. For more details, one can refer to (Yang et al., 2021).

3.2 Pretraining PATTON

We propose two strategies to help the language models understand text semantics on both the token level and the document level collaboratively from the network structure. The first strategy focuses on token-level semantics learning, namely network-contextualized masked language modeling; while the second strategy emphasizes document-level semantics learning, namely masked node prediction.

Strategy 1: Network-contextualized Masked Language Modeling (NMLM). Masked language modeling (MLM) is a commonly used strategy for language model pretraining (Devlin et al., 2019; Liu et al., 2019) and domain adaptation (Gururangan et al., 2020). It randomly masks several tokens in the text sequence and utilizes the surrounding unmasked tokens to predict them. The underlying assumption is that the semantics of each token can be reflected by its contexts. Trained to conduct masked token prediction, the language model will learn to understand semantic correlation between tokens and capture the contextualized semantic signals. The mathematical formulation of MLM is as follows,

$$\mathcal{L}_{\text{MLM}} = -\sum_{i \in M_t} \log p(w_i | \boldsymbol{H}_i), \tag{5}$$

where M_t is a subset of tokens which are replaced by a special `[MASK]` token and $p(w_i|\boldsymbol{H}_i)$ is the output probability of a linear head f_{head} which gives predictions to w_i (from the vocabulary W) based on contextualized token hidden states $\{\boldsymbol{H}_i\}$.

Such token correlation and contextualized semantics signals also exist and are even stronger in text-rich networks. Text from adjacent nodes in networks can provide auxiliary contexts for text semantics understanding. For example, given a paper talking about "Transformers" and its neighboring papers (cited papers) in the academic network on machine learning, we can infer that "Transformers" here is a deep learning model rather than an electrical engineering component by reading the text within both the given paper and the neighboring papers. In order to fully capture the textual semantic signals in the network, the language model needs to not only understand the in-node text token correlation but also be aware of the cross-node semantic correlation.

We extend the original in-node MLM to network-contextualized MLM, so as to facilitate the language model to understand both in-node token correlation and network-contextualized text semantic relatedness. The training objective is shown as follows.

$$\begin{aligned} \mathcal{L}_{\text{NMLM}} &= -\sum_{i \in M_t} \log p(w_i | \boldsymbol{H}_x, \boldsymbol{z}_x), \\ p(w_i | \boldsymbol{H}_x, \boldsymbol{z}_x) &= \text{softmax}(\boldsymbol{q}_{w_i}^\top \boldsymbol{h}_i), \end{aligned} \tag{6}$$

where $\boldsymbol{z}_x$ denotes the network contextualized token hidden state in Section 3.1 and $\boldsymbol{h}_i = \boldsymbol{H}_x^{(L)}[i]$ (if i is inside node x). L is the number of layers. $\boldsymbol{q}_{w_i}$ refers to the MLM prediction head for w_i. Since the calculation of $\boldsymbol{h}_i$ is based on $\boldsymbol{H}_x$ and $\boldsymbol{z}_x$, the likelihood will be conditioned on $\boldsymbol{H}_x$ and $\boldsymbol{z}_x$.

Strategy 2: Masked Node Prediction (MNP). While network-contextualized MLM focuses more on token-level semantics understanding, we propose a new strategy called "masked node prediction", which helps the language model understand the underlying document-level semantics correlation hidden in the network structure.

Concretely, we dynamically hold out a subset of nodes from the network ($M_v \subseteq V$), mask them, and train the language model to predict the masked nodes based on the adjacent network structure.

$$\begin{aligned} \mathcal{L}_{\text{MNP}} &= -\sum_{v_j \in M_v} \log p(v_j | \boldsymbol{G}_{v_j}), \\ p(v_j | \boldsymbol{G}_{v_j}) &= \text{softmax}(\boldsymbol{h}_{v_j}^\top \boldsymbol{h}_{N_{v_j}}) \end{aligned} \tag{7}$$

where $\boldsymbol{G}_{v_j} = \{\boldsymbol{h}_{v_k} | v_k \in N_{v_j}\}$ are the hidden states of the neighbor nodes in the network and N_{v_j} is the set of neighbors of v_j. In particular, we treat the hidden state of the last layer of `[CLS]` as a representation of node level, that is, $\boldsymbol{h}_{v_j} = \boldsymbol{H}_{v_j}^{(L)}$`[CLS]`.

By performing the task, the language model will absorb document semantic hints hidden inside the network structure (*e.g.*, contents between cited papers in the academic network can be quite semantically related, and text between co-viewed items in the e-commerce network can be highly associated).

However, directly optimizing masked node prediction can be computationally expensive since we need to calculate the representations for all neighboring nodes and candidate nodes for one prediction. To ease the computation overload, we prove that the masked node prediction task can be theoretically transferred to a computationally cheaper pairwise link prediction task.

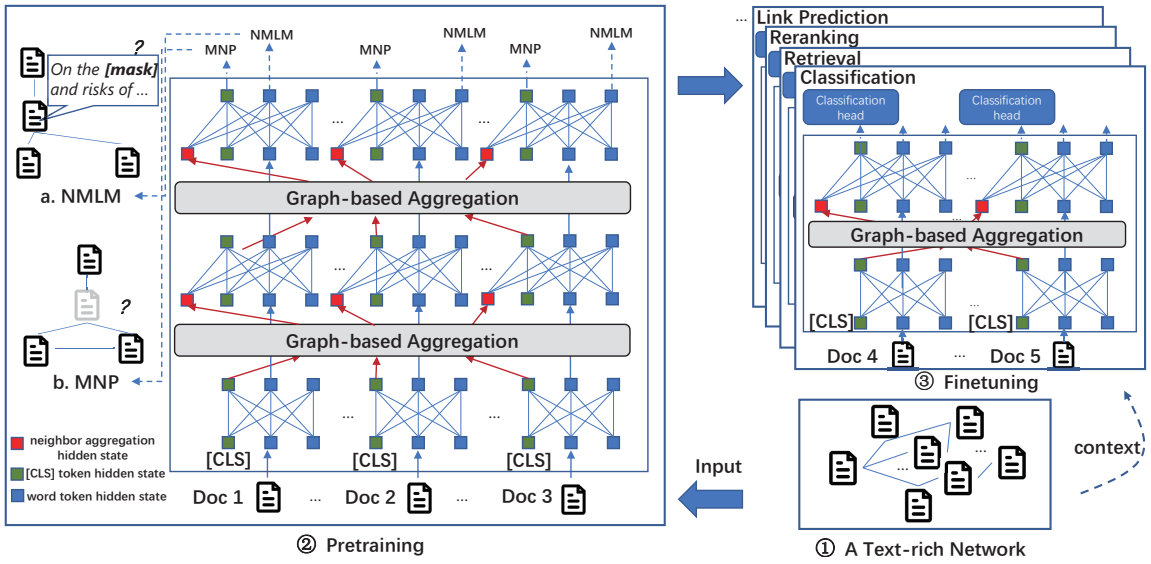


Figure 2: Overall pretraining and finetuning procedures for PATTON. We have two pretraining strategies: network-contextualized masked language modeling (NMLM) and masked node prediction (MNP). Apart from output layers, the same architectures are used in both pretraining and finetuning (in our experiment, we have 12 layers). The same pretrained model parameters are used to initialize models for different downstream tasks. During finetuning, all parameters are updated.

Theorem 3.2.1. *Masked node prediction is equivalent to pairwise link prediction.*

Proof: Given a set of masked nodes M_v, the likelihood of predicting the masked nodes is

$$\begin{aligned}
&\prod_{v_{\texttt{[MASK]}} \in M_v} p(v_{\texttt{[MASK]}} = v_i | v_k \in N_{v_{\texttt{[MASK]}}}) \\
\propto &\prod_{v_{\texttt{[MASK]}} \in M_v} p(v_k \in N_{v_{\texttt{[MASK]}}} | v_{\texttt{[MASK]}} = v_i) \\
= &\prod_{v_{\texttt{[MASK]}} \in M_v} \prod_{v_k \in N_{v_{\texttt{[MASK]}}}} p(v_k | v_{\texttt{[MASK]}} = v_i) \\
= &\prod_{v_{\texttt{[MASK]}} \in M_v} \prod_{v_k \in N_{v_{\texttt{[MASK]}}}} p(v_k \longleftrightarrow v_i)
\end{aligned}$$

In the above proof, the first step relies on the Bayes' rule, and we have the assumption that all nodes appear uniformly in the network, *i.e.*, $p(v_{\texttt{[MASK]}} = v_i) = p(v_{\texttt{[MASK]}} = v_j)$. In the second step, we have the conditional independence assumption of neighboring nodes generated given the center node, *i.e.*, $p(v_k, v_s | v_{\texttt{[MASK]}} = v_i) = p(v_k | v_{\texttt{[MASK]}} = v_i) \cdot p(v_s | v_{\texttt{[MASK]}} = v_i)$.

As a result, the masked node prediction objective can be simplified into a pairwise link prediction objective, which is

$$\begin{aligned}
\mathcal{L}_{\text{MNP}} &= -\sum_{v_j \in M_v} \sum_{v_k \in N_{v_j}} \log p(v_j \leftrightarrow v_k) \\
&= -\sum_{v_j \in M_v} \sum_{v_k \in N_{v_j}} \log \frac{\exp(\boldsymbol{h}_{v_j}^\top \boldsymbol{h}_{v_k})}{\exp(\boldsymbol{h}_{v_j}^\top \boldsymbol{h}_{v_k}) + \sum_{u'} \exp(\boldsymbol{h}_{v_j}^\top \boldsymbol{h}_{v'_u})},
\end{aligned} \tag{8}$$

where v'_u stands for a random negative sample. In our implementation, we use "in-batch negative samples" (Karpukhin et al., 2020) to reduce the encoding cost.

Joint Pretraining. To pretrain PATTON, we optimize the NMLM objective and the MNP objective jointly:

$$\mathcal{L} = \mathcal{L}_{\text{NMLM}} + \mathcal{L}_{\text{MNP}}. \tag{9}$$

This joint objective will unify the effects of NMLM and MNP, which encourages the model to conduct network-contextualized token-level understanding and network-enhanced document-level understanding, facilitating the joint modeling of texts and network structures. We will show in Section 4.6 that the joint objective achieves superior performance in comparison with using either objective alone.

3.3 Finetuning PATTON

Last, we describe how to finetune PATTON for downstream tasks involving encoding for text in the network and text not in the network. For text in the network (thus with neighbor information), we will feed both the node text sequence and the neighbor text sequences into the model; while for texts not in the network (thus neighbor information is not available), we will feed the text sequence into the model and leave the neighbor text sequences blank. For both cases, the final layer hidden state of `[CLS]` is used as text representation following (Devlin et al., 2019) and (Liu et al., 2019).

Table 1: Dataset Statistics.

Dataset	#Nodes	#Edges	#Fine-Classes	#Coarse-Classes
Mathematics	490,551	2,150,584	14,271	18
Geology	431,834	1,753,762	7,883	17
Economics	178,670	1,042,253	5,205	40
Clothes	889,225	7,876,427	2,771	9
Sports	314,448	3,461,379	3,034	16

4 Experiments

4.1 Experimental Settings

Dataset. We perform experiments on both academic networks from Microsoft Academic Graph (MAG) (Sinha et al., 2015) and e-commerce networks from Amazon (McAuley et al., 2015). In academic networks, nodes are papers and there will be an edge between two papers if one cites the other; while in e-commerce networks, nodes correspond to items, and item nodes are linked if they are frequently co-viewed by users. Since MAG and Amazon both have multiple domains, we select three domains from MAG and two domains from Amazon. In total, five datasets are used in the evaluation (*i.e.*, MAG-Mathematics, MAG-Geology, MAG-Economics, Amazon-Clothes and Amazon-Sports). The statistics of all the datasets can be found in Table 1. Fine-classes are all the categories in the network-associated node category taxonomy (MAG taxonomy and Amazon product catalog), while coarse-classes are the categories at the first layer of the taxonomy.

Pretraining Setup. The model is trained for 5/10/30 epochs (depending on the size of the network) on 4 Nvidia A6000 GPUs with a total batch size of 512. We set the peak learning rate as 1e-5. NMLM pretraining uses the standard 15% `[MASK]` ratio. For our model and all baselines, we adopt a 12-layer architecture. More details can be found in the Appendix A.

Baselines. We mainly compare our method with two kinds of baselines, off-the-shelf pretrained language models and language model continuous pretraining methods. The first category includes BERT (Devlin et al., 2019), SciBERT (Beltagy et al., 2019), SPECTER (Cohan et al., 2020), SimCSE (Gao et al., 2021), LinkBERT (Yasunaga et al., 2022) and vanilla GraphFormers (Yang et al., 2021). BERT (Devlin et al., 2019) is a language model pretrained with masked language modeling and next sentence prediction objectives on Wikipedia and BookCorpus. SciBERT (Beltagy et al., 2019) utilizes the same pretraining strategies as BERT but is trained on 1.14 million paper abstracts and full text from Semantic Scholar. SPECTER (Cohan et al., 2020) is a language model continuously pretrained from SciBERT with a contrastive objective on 146K scientific papers. SimCSE (Gao et al., 2021) is a contrastive learning framework and we perform the experiment with the models pretrained from both unsupervised settings (Wikipedia) and supervised settings (NLI). LinkBERT (Yasunaga et al., 2022) is a language model pretrained with masked language modeling and document relation prediction objectives on Wikipedia and BookCorpus. GraphFormers (Yang et al., 2021) is a GNN-nested Transformer and we initialize it with the BERT checkpoint for a fair comparison. The second category includes several continuous pretraining methods (Gururangan et al., 2020; Gao et al., 2021). We perform continuous masked language modeling starting from the BERT checkpoint (denoted as BERT.MLM) and the SciBERT checkpoint (denoted as SciBERT.MLM) on our data, respectively. We also perform in-domain supervised contrastive pretraining with the method proposed in (Gao et al., 2021) (denoted as SimCSE.in-domain).

Ablation Setup. For academic networks, we pretrain our model starting from the BERT-base [2] checkpoint (PATTON) and the SciBERT [3] checkpoint (SciPATTON) respectively; while for e-commerce networks, we pretrain our model from BERT-base only (PATTON). Furthermore, we conduct ablation studies to validate the effectiveness of both the NMLM and the MNP strategies. The pretrained model with NMLM removed and that with MNP removed are called "w/o NMLM" and "w/o MNP", respectively. In academic networks, the ablation study is done on SciPATTON, while in e-commerce networks, it is done on PATTON.

We demonstrate the effectiveness of our framework on four downstream tasks, including classification, retrieval, reranking, and link prediction.

4.2 Classification

In this section, we conduct experiments on 8-shot coarse-grained classification for nodes in the networks. We use the final layer hidden state of `[CLS]` token from language models as the representation of the node and feed it into a linear layer classifier to obtain the prediction result. Both the language model and the classifier are finetuned. The experimental results are shown in Table 2. From the result, we can find

[2] https://huggingface.co/bert-base-uncased

[3] https://huggingface.co/allenai/scibert_scivocab_uncased

Table 2: Experiment results on Classification. We show the mean_{std} of three runs for all the methods.

Method	Mathematics		Geology		Economics		Clothes		Sports	
	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1
BERT	$18.14_{0.07}$	$22.04_{0.32}$	$21.97_{0.87}$	$29.63_{0.36}$	$14.17_{0.08}$	$19.77_{0.12}$	$45.10_{1.47}$	$68.54_{2.25}$	$31.88_{0.23}$	$34.58_{0.56}$
GraphFormers	$18.69_{0.52}$	$23.24_{0.46}$	$22.64_{0.92}$	$31.02_{1.16}$	$13.68_{1.03}$	$19.00_{1.44}$	$46.27_{1.92}$	$68.97_{2.46}$	$43.77_{0.63}$	$50.47_{0.78}$
SciBERT	$23.50_{0.64}$	$23.10_{2.23}$	$29.49_{1.25}$	$37.82_{1.89}$	$15.91_{0.48}$	$21.32_{0.66}$	-	-	-	-
SPECTER	$23.37_{0.07}$	$29.83_{0.96}$	$30.40_{0.48}$	$38.54_{0.77}$	$16.16_{0.17}$	$19.84_{0.47}$	-	-	-	-
SimCSE (unsup)	$20.12_{0.08}$	$26.11_{0.39}$	$38.78_{0.19}$	$38.55_{0.17}$	$14.54_{0.26}$	$19.07_{0.43}$	$42.70_{2.32}$	$58.72_{0.34}$	$41.91_{0.85}$	$59.19_{0.55}$
SimCSE (sup)	$20.39_{0.07}$	$25.56_{0.00}$	$25.66_{0.28}$	$33.89_{0.40}$	$15.03_{0.53}$	$18.64_{1.32}$	$52.82_{0.87}$	$75.54_{0.98}$	$46.69_{0.10}$	$59.19_{0.55}$
LinkBERT	$15.78_{0.91}$	$19.75_{1.19}$	$24.08_{0.58}$	$31.32_{0.04}$	$12.71_{0.12}$	$16.39_{0.22}$	$44.94_{2.52}$	$65.33_{4.34}$	$35.60_{0.33}$	$38.30_{0.09}$
BERT.MLM	$23.44_{0.39}$	$31.75_{0.58}$	$36.31_{0.36}$	$48.04_{0.69}$	$16.60_{0.21}$	$22.71_{1.16}$	$46.98_{0.84}$	$68.00_{0.84}$	$62.21_{0.13}$	$75.43_{0.74}$
SciBERT.MLM	$23.34_{0.42}$	$30.11_{0.97}$	$36.94_{0.28}$	$46.54_{0.40}$	$16.28_{0.38}$	$21.41_{0.81}$	-	-	-	-
SimCSE.in-domain	$25.15_{0.09}$	$29.85_{0.20}$	$38.91_{0.08}$	$48.93_{0.14}$	$18.08_{0.22}$	$23.79_{0.44}$	$57.03_{0.20}$	$80.16_{0.31}$	$65.57_{0.35}$	$75.22_{0.18}$
PATTON	$\mathbf{27.58}_{0.03}$	$\mathbf{32.82}_{0.01}$	$39.35_{0.06}$	$48.19_{0.15}$	$19.32_{0.05}$	$25.12_{0.05}$	$\mathbf{60.14}_{0.28}$	$\mathbf{84.88}_{0.09}$	$\mathbf{67.57}_{0.08}$	$\mathbf{78.60}_{0.15}$
SciPATTON	$27.35_{0.04}$	$31.70_{0.01}$	$\mathbf{39.65}_{0.10}$	$\mathbf{48.93}_{0.06}$	$\mathbf{19.91}_{0.08}$	$\mathbf{25.68}_{0.32}$	-	-	-	-
w/o NMLM	$25.91_{0.45}$	$27.79_{2.07}$	$38.78_{0.19}$	$48.48_{0.17}$	$18.86_{0.23}$	$24.25_{0.26}$	$56.68_{0.24}$	$80.27_{0.17}$	$65.83_{0.28}$	$76.24_{0.54}$
w/o MNP	$24.79_{0.65}$	$29.44_{1.50}$	$38.00_{0.73}$	$47.82_{1.06}$	$18.69_{0.59}$	$25.63_{1.44}$	$47.35_{1.20}$	$68.50_{2.60}$	$64.23_{1.53}$	$76.03_{1.67}$

that: 1) PATTON and SciPATTON consistently outperform baseline methods; 2) Continuous pretraining method (BERT.MLM, SciBERT.MLM, SimCSE.in-domain, PATTON, and SciPATTON) can have better performance than off-the-shelf PLMs, which demonstrates that domain shift exists between the pretrained PLM domain and the target domain, and the adaptive pretraining on the target domain is necessary. More detailed information on the task can be found in Appendix B.

4.3 Retrieval

The retrieval task corresponds to 16-shot fine-grained category retrieval, where given a node, we want to retrieve category names for it from a very large label space. We follow the widely-used DPR (Karpukhin et al., 2020) pipeline to finetune all the models. In particular, the final layer hidden states of `[CLS]` token are utilized as dense representations for both node and label names. Negative samples retrieved from BM25 are used as hard negatives. The results are shown in Table 3. From the result, we can have the following observations: 1) PATTON and SciPATTON consistently outperform all the baseline methods; 2) Continuously pretrained models can be better than off-the-shelf PLMs in many cases (SciBERT and SPECTER perform well on Mathematics and Economics since their pretrained corpus includes a large number of Computer Science papers, which are semantically close to Mathematics and Economics papers) and can largely outperform traditional BM25. More detailed information on the task can be found in Appendix C.

4.4 Reranking

The reranking task corresponds to the 32-shot fine-grained category reranking. We first adopt BM25 (Robertson et al., 2009) and exact matching as the retriever to obtain a candidate category name list for each node. Then, the models are asked to rerank all the categories in the list based on their similarity to the given node text. The way to encode the node and category names is the same as that in retrieval. Unlike retrieval, reranking tests the ability of the language model to distinguish among candidate categories at a fine-grained level. The results are shown in Table 4. From the result, we can find that PATTON and SciPATTON consistently outperform all baseline methods, demonstrating that our pretraining strategies allow the language model to better understand fine-grained semantic similarity. More detailed information on the task can be found in Appendix D.

4.5 Link Prediction

In this section, we perform the 32-shot link prediction for nodes in the network. Language models are asked to give a prediction on whether there should exist an edge between two nodes. It is worth noting that the edge semantics here ("author overlap" [4] for academic networks and "co-purchased" for e-commerce networks) are different from those in pretraining ("citation" for academic networks and "co-viewed" for e-commerce networks). We utilize the final layer `[CLS]` token hidden state as node representation and conduct in-batch evaluations. The results are shown in Table 5. From the result, we can find that PATTON and SciPATTON can outperform baselines and ablations in most cases, which shows that our pretraining strategies can help the language model extract knowledge from the pretrained text-rich network and apply it to the new link type prediction. More detailed information on the task can be found in Appendix E.

[4] Two papers have at least one same author.

Table 3: Experiment results on Retrieval. We show the mean_{std} of three runs for all the methods.

Method	Mathematics		Geology		Economics		Clothes		Sports	
	R@50	R@100	R@50	R@100	R@50	R@100	R@50	R@100	R@50	R@100
BM25	20.76	24.55	19.02	20.92	19.14	22.49	15.76	15.88	22.00	23.96
BERT	$16.73_{0.17}$	$22.66_{0.18}$	$18.82_{0.39}$	$25.94_{0.39}$	$23.95_{0.25}$	$31.54_{0.21}$	$40.77_{1.68}$	$50.40_{1.41}$	$32.37_{1.09}$	$43.32_{0.96}$
GraphFormers	$16.65_{0.12}$	$22.41_{0.10}$	$18,92_{0.60}$	$25.94_{0.39}$	$24.48_{0.36}$	$32.16_{0.40}$	$41.77_{2.05}$	$51.26_{2.27}$	$32.39_{0.89}$	$43.29_{1.12}$
SciBERT	$24.70_{0.17}$	$33.55_{0.31}$	$23.71_{0.89}$	$30.94_{0.95}$	$29.80_{0.66}$	$38.66_{0.52}$	-	-	-	-
SPECTER	$23.86_{0.25}$	$31.11_{0.31}$	$26.56_{1.05}$	$34.04_{1.32}$	$31.26_{0.15}$	$40.79_{0.11}$	-	-	-	-
SimCSE (unsup)	$17.91_{0.26}$	$23.19_{0.29}$	$20.45_{0.20}$	$26.82_{0.26}$	$25.83_{0.23}$	$33.42_{0.28}$	$44.90_{0.35}$	$54.76_{0.38}$	$38.81_{0.35}$	$49.30_{0.44}$
SimCSE (sup)	$20.29_{0.41}$	$26.23_{0.51}$	$22.34_{0.49}$	$29.63_{0.55}$	$28.07_{0.38}$	$36.51_{0.37}$	$44.69_{0.59}$	$54.70_{0.77}$	$40.31_{0.43}$	$50.55_{0.41}$
LinkBERT	$17.25_{0.30}$	$23.21_{0.47}$	$17.14_{0.75}$	$23.05_{0.74}$	$22.69_{0.30}$	$30.77_{0.36}$	$28.66_{2.97}$	$37.79_{3.82}$	$31.97_{0.54}$	$41.77_{0.67}$
BERT.MLM	$20.69_{0.21}$	$27.17_{0.25}$	$32.13_{0.36}$	$41.74_{0.42}$	$27.13_{0.04}$	$36.00_{0.14}$	$52.41_{1.71}$	$63.72_{1.79}$	$54.10_{0.81}$	$63.14_{0.83}$
SciBERT.MLM	$20.65_{0.21}$	$27.67_{0.32}$	$31.65_{0.71}$	$40.52_{0.76}$	$29.23_{0.67}$	$39.18_{0.73}$	-	-	-	-
SimCSE.in-domain	$24.54_{0.05}$	$31.66_{0.09}$	$33.97_{0.07}$	$44.09_{0.19}$	$28.44_{0.31}$	$37.81_{0.27}$	$61.42_{0.84}$	$72.25_{0.86}$	$53.77_{0.22}$	$63.73_{0.30}$
PATTON	$27.44_{0.15}$	$34.97_{0.21}$	$34.94_{0.23}$	$45.01_{0.28}$	$32.10_{0.51}$	$42.19_{0.62}$	$\mathbf{68.62}_{0.38}$	$\mathbf{77.54}_{0.19}$	$\mathbf{58.63}_{0.31}$	$\mathbf{68.53}_{0.55}$
SciPATTON	$\mathbf{31.40}_{0.52}$	$\mathbf{40.38}_{0.66}$	$\mathbf{40.69}_{0.52}$	$\mathbf{51.31}_{0.48}$	$\mathbf{35.82}_{0.69}$	$46.05_{0.69}$	-	-	-	-
w/o NMLM	$30.85_{0.14}$	$39.89_{0.23}$	$39.29_{0.07}$	$49.59_{0.11}$	$35.17_{0.31}$	$\mathbf{46.07}_{0.20}$	$65.60_{0.26}$	$75.19_{0.32}$	$57.05_{0.14}$	$67.22_{0.12}$
w/o MNP	$22.47_{0.07}$	$30.20_{0.15}$	$31.28_{0.89}$	$40.54_{0.97}$	$29.54_{0.36}$	$39.57_{0.57}$	$60.20_{0.73}$	$69.85_{0.52}$	$51.73_{0.41}$	$60.35_{0.78}$

Table 4: Experiment results on Reranking. We show the mean_{std} of three runs for all the methods.

Method	Mathematics		Geology		Economics		Clothes		Sports	
	NDCG@5	NDCG@10	NDCG@5	NDCG@10	NDCG@5	NDCG@10	NDCG@5	NDCG@10	NDCG@5	NDCG@10
BERT	$37.15_{0.64}$	$44.76_{0.59}$	$56.59_{1.18}$	$68.21_{0.96}$	$42.65_{0.70}$	$53.55_{0.76}$	$62.19_{0.63}$	$72.00_{0.70}$	$44.68_{0.56}$	$57.54_{0.55}$
GraphFormers	$37.85_{0.32}$	$47.89_{0.69}$	$58.32_{1.22}$	$69.91_{1.19}$	$41.82_{0.65}$	$52.67_{0.76}$	$62.11_{0.87}$	$72.02_{0.73}$	$44.49_{0.71}$	$57.35_{0.50}$
SciBERT	$40.73_{0.50}$	$53.22_{0.51}$	$57.04_{1.05}$	$69.47_{0.92}$	$43.24_{0.79}$	$55.22_{0.67}$	-	-	-	-
SPECTER	$38.95_{0.67}$	$52.17_{0.71}$	$57.79_{0.69}$	$69.57_{0.46}$	$43.41_{1.10}$	$55.80_{1.02}$	-	-	-	-
SimCSE (unsup)	$32.34_{0.43}$	$42.59_{0.44}$	$49.60_{1.04}$	$61.51_{1.03}$	$36.37_{0.67}$	$47.18_{0.76}$	$57.03_{1.27}$	$68.16_{1.04}$	$43.29_{0.16}$	$55.41_{0.09}$
SimCSE (sup)	$34.85_{0.60}$	$44.76_{0.59}$	$48.07_{0.54}$	$59.79_{0.51}$	$37.01_{0.40}$	$48.05_{0.44}$	$52.74_{0.55}$	$64.28_{0.52}$	$42.00_{0.09}$	$53.92_{0.13}$
LinkBERT	$38.50_{1.15}$	$50.74_{1.12}$	$59.57_{0.96}$	$71.41_{0.93}$	$44.00_{1.12}$	$55.78_{0.95}$	$58.24_{1.93}$	$70.48_{1.58}$	$48.45_{1.02}$	$61.63_{1.01}$
BERT.MLM	$39.24_{0.47}$	$51.18_{0.35}$	$60.58_{0.29}$	$72.52_{0.28}$	$44.30_{0.68}$	$55.84_{0.69}$	$60.51_{0.31}$	$71.36_{0.28}$	$45.70_{4.49}$	$57.08_{4.60}$
SciBERT.MLM	$39.03_{0.48}$	$52.34_{0.39}$	$62.01_{0.55}$	$74.58_{0.47}$	$46.43_{0.21}$	$58.60_{0.21}$	-	-	-	-
SimCSE.in-domain	$40.37_{0.30}$	$53.80_{0.24}$	$61.13_{0.75}$	$73.89_{0.57}$	$45.27_{0.13}$	$58.33_{0.13}$	$64.81_{0.49}$	$75.77_{0.24}$	$50.05_{0.62}$	$62.56_{0.29}$
PATTON	$42.08_{0.17}$	$55.30_{0.17}$	$61.41_{0.62}$	$74.02_{0.49}$	$46.52_{0.53}$	$59.25_{0.44}$	$\mathbf{66.26}_{0.81}$	$\mathbf{77.01}_{0.55}$	$\mathbf{52.16}_{0.44}$	$64.96_{0.37}$
SciPATTON	$\mathbf{47.10}_{0.49}$	$\mathbf{60.86}_{0.55}$	$\mathbf{63.48}_{0.25}$	$\mathbf{75.86}_{0.18}$	$\mathbf{51.19}_{0.33}$	$\mathbf{63.86}_{0.34}$	-	-	-	-
w/o NMLM	$41.43_{0.16}$	$55.28_{0.21}$	$62.84_{1.79}$	$75.36_{1.43}$	$46.05_{2.04}$	$59.39_{1.91}$	$63.71_{1.11}$	$74.75_{0.81}$	$52.12_{0.13}$	$\mathbf{65.35}_{0.14}$
w/o MNP	$43.56_{0.53}$	$57.14_{0.52}$	$62.42_{0.47}$	$74.91_{0.40}$	$48.07_{0.30}$	$60.57_{0.32}$	$63.88_{0.47}$	$74.01_{0.36}$	$47.81_{0.56}$	$59.68_{0.54}$

4.6 Ablation Study

We perform ablation studies to validate the effectiveness of the two strategies in Tables 2-5. The full method is better than each ablation version in most cases, except R@100 on Economy retrieval, NDCG@10 on Sports reranking, and link prediction on Amazon datasets, which indicates the importance of both strategies.

4.7 Pretraining Step Study

We conduct an experiment on the Sports dataset to study how the pretrained checkpoint at different pretraining steps can perform on downstream tasks. The result is shown in Figure 3. From the figure, we can find that: 1) The downstream performance on retrieval, reranking, and link prediction generally improves as the pretraining step increases. This means that the pretrained language model can learn more knowledge, which can benefit these downstream tasks from the pretraining text-rich network as the pretraining step increases. 2) The downstream performance on classification increases and then decreases. The reason is that for downstream classification, when pretrained for too long, the pretrained language model may overfit the given text-rich network, which will hurt classification performance.

4.8 Scalability Study

We run an experiment on Sports to study the time complexity and memory complexity of the proposed pretraining strategies. The model is pretrained for 10 epochs on four Nvidia A6000 GPU devices with a total training batch size set as 512. We show the result in Table 6. From the result, we can find that: 1) Pretraining with the MNP strategy is faster and memory cheaper than pretraining with the NMLM strategy. 2) Combining the two strategies together will not increase the time complexity and memory complexity too much, compared with NMLM pretraining only.

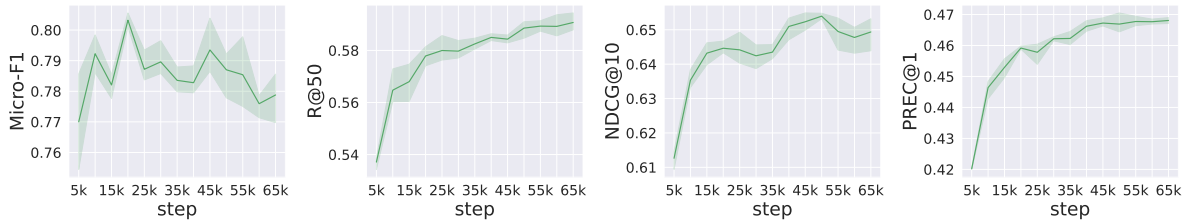
Further model studies on finetune data size can be found in Appendix F.

5 Attention Map Study

We conduct a case study by showing some attention maps of PATTON and the model without pretraining on four downstream tasks on Sports. We randomly pick a token from a random sample and plot the self-attention probability of how different tokens (x-

Table 5: Experiment results on Link Prediction. We show the mean_{std} of three runs for all the methods.

Method	Mathematics		Geology		Economics		Clothes		Sports	
	PREC@1	MRR	PREC@1	MRR	PREC@1	MRR	PREC@1	MRR	PREC@1	MRR
BERT	$6.60_{0.16}$	$12.96_{0.34}$	$6.24_{0.76}$	$12.96_{1.34}$	$4.12_{0.08}$	$9.23_{0.15}$	$24.17_{0.41}$	$34.20_{0.45}$	$16.48_{0.45}$	$25.35_{0.52}$
GraphFormers	$6.91_{0.29}$	$13.42_{0.34}$	$6.52_{1.17}$	$13.34_{1.81}$	$4.16_{0.21}$	$9.28_{0.28}$	$23.79_{0.69}$	$33.79_{0.66}$	$16.69_{0.36}$	$25.74_{0.48}$
SciBERT	$14.08_{0.11}$	$23.62_{0.10}$	$7.15_{0.26}$	$14.11_{0.39}$	$5.01_{1.04}$	$10.48_{1.79}$	-	-	-	-
SPECTER	$13.44_{0.5}$	$21.73_{0.65}$	$6.85_{0.22}$	$13.37_{0.34}$	$6.33_{0.29}$	$12.41_{0.33}$	-	-	-	-
SimCSE (unsup)	$9.85_{0.10}$	$16.28_{0.12}$	$7.47_{0.55}$	$14.24_{0.89}$	$5.72_{0.26}$	$11.02_{0.34}$	$30.51_{0.09}$	$40.40_{0.10}$	$22.99_{0.07}$	$32.47_{0.06}$
SimCSE (sup)	$10.35_{0.52}$	$17.01_{0.72}$	$10.10_{0.04}$	$17.80_{0.07}$	$5.72_{0.26}$	$11.02_{0.34}$	$35.42_{0.06}$	$46.07_{0.06}$	$27.07_{0.15}$	$37.44_{0.16}$
LinkBERT	$8.05_{0.14}$	$13.91_{0.09}$	$6.40_{0.14}$	$12.99_{0.17}$	$2.97_{0.08}$	$6.79_{0.15}$	$30.33_{0.56}$	$39.59_{0.64}$	$19.83_{0.09}$	$28.32_{0.04}$
BERT.MLM	$17.55_{0.25}$	$29.22_{0.26}$	$14.13_{0.19}$	$25.36_{0.20}$	$9.02_{0.09}$	$16.72_{0.15}$	$42.71_{0.31}$	$54.54_{0.35}$	$29.36_{0.09}$	$41.60_{0.05}$
SciBERT.MLM	$22.44_{0.08}$	$34.22_{0.05}$	$16.22_{0.03}$	$27.02_{0.07}$	$9.80_{0.00}$	$17.72_{0.01}$	-	-	-	-
SimCSE.in-domain	$33.55_{0.05}$	$46.07_{0.07}$	$24.56_{0.06}$	$36.89_{0.11}$	$16.77_{0.10}$	$26.93_{0.01}$	$\mathbf{60.41}_{0.03}$	$\mathbf{71.86}_{0.06}$	$49.17_{0.04}$	$63.48_{0.03}$
PATTON	$70.41_{0.11}$	$80.21_{0.04}$	$44.76_{0.05}$	$57.71_{0.04}$	$57.04_{0.05}$	$68.35_{0.04}$	$58.59_{0.12}$	$70.12_{0.12}$	$46.68_{0.09}$	$60.96_{0.23}$
SciPATTON	$\mathbf{71.22}_{0.17}$	$\mathbf{80.79}_{0.10}$	$\mathbf{44.95}_{0.24}$	$\mathbf{57.84}_{0.25}$	$\mathbf{57.36}_{0.26}$	$\mathbf{68.71}_{0.31}$	-	-	-	-
w/o NMLM	$71.04_{0.13}$	$80.60_{0.07}$	$44.33_{0.23}$	$57.29_{0.22}$	$56.64_{0.25}$	$68.12_{0.16}$	$60.30_{0.03}$	$71.67_{0.07}$	$\mathbf{49.72}_{0.06}$	$\mathbf{63.76}_{0.04}$
w/o MNP	$63.06_{0.23}$	$74.26_{0.11}$	$33.84_{0.60}$	$47.02_{0.65}$	$44.46_{0.03}$	$57.05_{0.04}$	$49.62_{0.06}$	$61.61_{0.01}$	$36.05_{0.20}$	$49.78_{0.25}$



(a) Classification (b) Retrieval (c) Reranking (d) Link Prediction

Figure 3: Pretrain step study on Amazon-Sports. The downstream performance on retrieval, reranking and link prediction generally improves when pretrained for longer, while the performance on classification improves and then drops.

Table 6: Time scalability and memory scalability study on Amazon-Sports.

Attribute	NMLM	MNP	NMLM+MNP
Time	15h 37min	14h 53min	15h 39min
Memory	32,363MB	30,313MB	32,365MB

axis), including neighbor virtual token ([n_CLS]) and the first eight original text tokens ([tk_x]), will contribute to the encoding of this random token in different layers (y-axis). The result is shown in Figure 4. From the result, we can find that the neighbor virtual token is more deactivated for the model without pretraining, which means that the information from neighbors is not fully utilized during encoding. However, the neighbor virtual token becomes more activated after pretraining, bringing more useful information from neighbors to enhance center node text encoding.

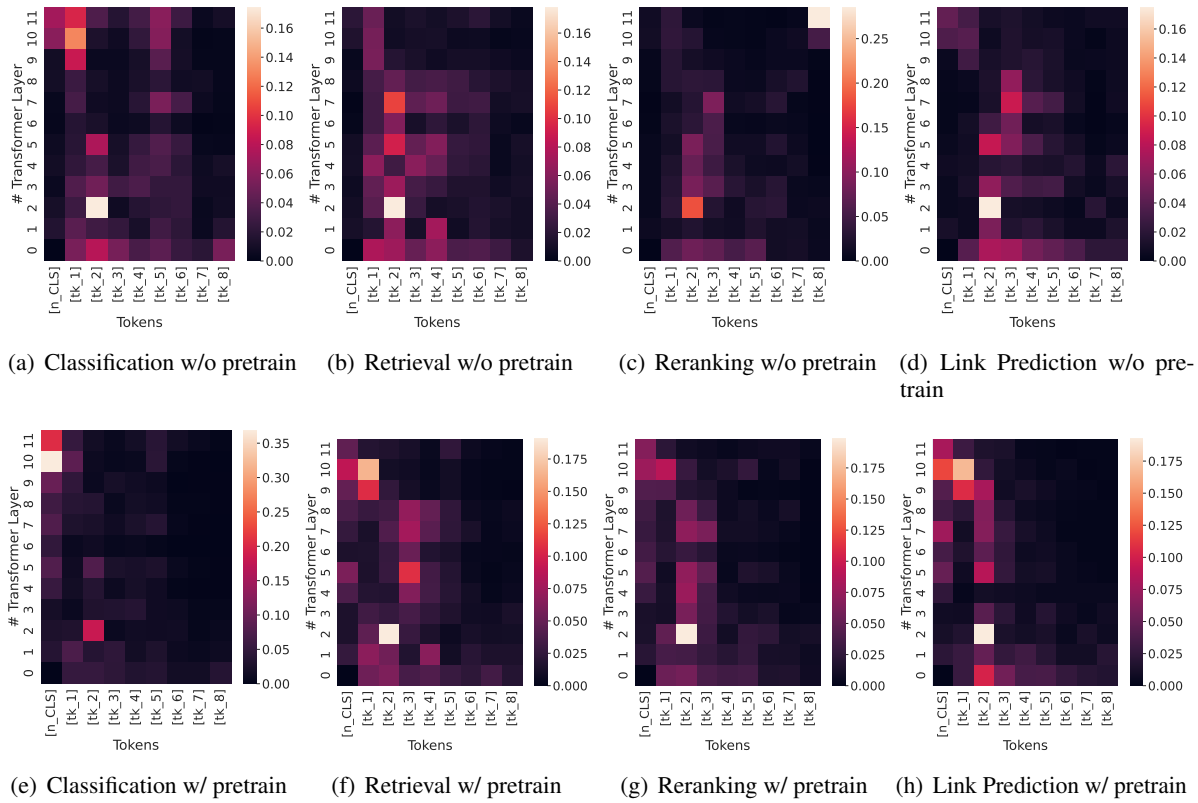
6 Related Work

6.1 Pretrained Language Models

Pretrained language models have been very successful in natural language processing since they were introduced (Peters et al., 2018; Devlin et al., 2019). Follow-up research has made them stronger by scaling them up from having millions of parameters (Yang et al., 2019; Lewis et al., 2020; Clark et al., 2020) to even trillions (Radford et al., 2019; Raffel et al., 2020; Brown et al., 2020). Another way that these models have been improved is by using different training objectives, including masked language modeling (Devlin et al., 2019), auto-regressive causal language modeling (Brown et al., 2020), permutation language modeling (Yang et al., 2019), discriminative language modeling (Clark et al., 2020), correcting and contrasting (Meng et al., 2021) and document relation modeling (Yasunaga et al., 2022). However, most of them are designed for modeling texts exclusively, and do not consider the inter-document structures. In this paper, we innovatively design strategies to capture the semantic hints hidden inside the complex document networks.

6.2 Domain Adaptation in NLP

Large language models have demonstrated their power in various NLP tasks. However, their performance under domain shift is quite constrained (Ramponi and Plank, 2020). To overcome the negative effect caused by domain shift, continuous pretraining is proposed in recent works (Gururangan et al., 2020), which can be further categorized into domain-adaptive pretraining (Han and Eisenstein, 2019) and task-specific pretraining (Howard and Ruder, 2018). However, existing works mainly focus on continuous pretraining based on textual in-



(a) Classification w/o pretrain (b) Retrieval w/o pretrain (c) Reranking w/o pretrain (d) Link Prediction w/o pretrain

(e) Classification w/ pretrain (f) Retrieval w/ pretrain (g) Reranking w/ pretrain (h) Link Prediction w/ pretrain

Figure 4: Attention map study on Amazon-Sports. [n_CLS] refers to network neighbor virtual token and [tk_x]s refer to word tokens. [n_CLS] is more activated after pretraining (PATTON), which means that more useful information from network neighbors is extracted to enhance center node text encoding.

formation, while our work tries to conduct pretraining utilizing textual signal and network structure signal simultaneously.

6.3 Pretraining on Graphs

Inspired by the recent success of pretrained language models, researchers are starting to explore pretraining strategies for graph neural networks (Hu et al., 2020b; Qiu et al., 2020; Hu et al., 2020a). Famous strategies include graph autoregressive modeling (Hu et al., 2020b), masked component modeling (Hu et al., 2020a), graph context prediction (Hu et al., 2020a) and constrastive pretraining (Qiu et al., 2020; Velickovic et al., 2019; Sun et al., 2020). These works conduct pretraining for graph neural network utilizing network structure information and do not consider the associated rich textual signal. However, our work proposes to pretrain the language model, adopting both textual information and network structure information.

7 Conclusions

In this work, we introduce PATTON, a method to pretrain language models on text-rich networks. PATTON consists of two objectives: (1) a network-contextualized MLM pretraining objective and (2) a masked node prediction objective, to capture the rich semantics information hidden inside the complex network structure. We conduct experiment on four downstream tasks and five datasets from two different domains, where PATTON outperforms baselines significantly and consistently.

Acknowledgments

We thank anonymous reviewers for their valuable and insightful feedback. Research was supported in part by US DARPA KAIROS Program No. FA8750-19-2-1004 and INCAS Program No. HR001121C0165, National Science Foundation IIS-19-56151, IIS-17-41317, and IIS 17-04532, and the Molecule Maker Lab Institute: An AI Research Institutes program supported by NSF under Award No. 2019897, and the Institute for Geospatial Understanding through an Integrative Discovery Environment (I-GUIDE) by NSF under Award No. 2118329. Any opinions, findings, and conclusions or recommendations expressed herein are those of the authors and do not necessarily represent the views, either expressed or implied, of DARPA or the U.S. Government. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

Limitations

In this work, we mainly focus on language model pretraining on homogeneous text-rich networks and explore how pretraining can benefit classification, retrieval, reranking, and link prediction. Interesting future studies include 1) researching how to conduct pretraining on heterogeneous text-rich networks and how to characterize the edges of different semantics; 2) exploring how pretraining can benefit broader task spaces including summarization and question answering.

Ethics Statement

While it has been shown that PLMs are powerful in language understanding (Devlin et al., 2019; Lewis et al., 2020; Raffel et al., 2020), there are studies highlighting their drawbacks such as the presence of social bias (Liang et al., 2021) and misinformation (Abid et al., 2021). In our work, we focus on pretraining PLMs with information from the inter-document structures, which could be a way to mitigate bias and eliminate the contained misinformation.

References

Abubakar Abid, Maheen Farooqi, and James Zou. 2021. Persistent anti-muslim bias in large language models. In *AIES*.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: Pretrained language model for scientific text. In *EMNLP*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *NeurIPS*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *ICLR*.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S. Weld. 2020. SPECTER: Document-level Representation Learning using Citation-informed Transformers. In *ACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *NAACL*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *EMNLP*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: adapt language models to domains and tasks. *ACL*.

Xiaochuang Han and Jacob Eisenstein. 2019. Unsupervised domain adaptation of contextualized embeddings for sequence labeling. *EMNLP*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *ACL*.

Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. 2020a. Strategies for pre-training graph neural networks. *ICLR*.

Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. 2020b. Gpt-gnn: Generative pre-training of graph neural networks. In *KDD*.

Bowen Jin, Yu Zhang, Yu Meng, and Jiawei Han. 2023a. Edgeformers: Graph-empowered transformers for representation learning on textual-edge networks. In *ICLR*.

Bowen Jin, Yu Zhang, Qi Zhu, and Jiawei Han. 2023b. Heterformer: Transformer-based deep node representation learning on heterogeneous text-rich networks. *KDD*.

Bharath Kandimalla, Shaurya Rohatgi, Jian Wu, and C Lee Giles. 2021. Large scale subject category classification of scholarly papers with deep attentive neural networks. *Frontiers in research metrics and analytics*, 5:600382.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *EMNLP*.

Onur Küçüktunç, Erik Saule, Kamer Kaya, and Ümit V Çatalyürek. 2012. Recommendation on academic networks using direction aware citation analysis. *arXiv preprint arXiv:1205.1143*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *ACL*.

Paul Pu Liang, Chiyu Wu, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2021. Towards understanding and mitigating social biases in language models. In *ICML*.

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*.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *SIGIR*.

Yu Meng, Chenyan Xiong, Payal Bajaj, Paul Bennett, Jiawei Han, Xia Song, et al. 2021. Coco-lm: Correcting and contrasting text sequences for language model pretraining. *NeurIPS*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL*.

Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. Gcc: Graph contrastive coding for graph neural network pre-training. In *KDD*.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*.

Alan Ramponi and Barbara Plank. 2020. Neural unsupervised domain adaptation in nlp—a survey. *COLING*.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*.

Christian Schulz, Amin Mazloumian, Alexander M Petersen, Orion Penner, and Dirk Helbing. 2014. Exploiting citation networks for large-scale author name disambiguation. *EPJ Data Science*.

Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June Hsu, and Kuansan Wang. 2015. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th international conference on world wide web*, pages 243–246.

Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. 2020. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. *ICLR*.

Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep graph infomax. *ICLR*.

Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *WWW*.

Junhan Yang, Zheng Liu, Shitao Xiao, Chaozhuo Li, Defu Lian, Sanjay Agrawal, Amit Singh, Guangzhong Sun, and Xing Xie. 2021. Graphformers: Gnn-nested transformers for representation learning on textual graph. In *NeurIPS*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *NeurIPS*.

Michihiro Yasunaga, Jure Leskovec, and Percy Liang. 2022. Linkbert: Pretraining language models with document links. *ACL*.

Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. 2019. Heterogeneous graph neural network. In *KDD*.

A Pretrain Settings

To facilitate the reproduction of our pretraining experiment, we provide the hyperparameter configuration in Table 7. All reported continuous pretraining and in-domain pretraining methods use exactly the same set of hyperparameters for pretraining for a fair comparison. All GraphFormers (Yang et al., 2021) involved methods have the neighbor sampling number set as 5. Paper titles and item titles are used as text associated with the nodes in the two kinds of networks, respectively. (For some items, we concatenate the item title and description together since the title is too short.) Since most paper titles (88%) and item titles (97%) are within 32 tokens, we set the max length of the input sequence to be 32. The models are trained for 5/10/30 epochs (depending on the size of the network) on 4 Nvidia A6000 GPUs with a total batch size of 512. The total time cost is around 24 hours for each network. Code is available at https://github.com/PeterGriffinJin/Patton.

B Classification

Task. The coarse-grained category names for academic networks and e-commerce networks are the first-level category names in the network-associated category taxonomy. We train all the methods in the 8-shot setting (8 labeled training samples and 8 labeled validation samples for each class) and test the models with hundreds of thousands of new query nodes (220,681, 215,148, 85,346, 477,700, and 129,669 for Mathematics, Geology, Economics, Clothes, and Sports respectively). Detailed information on all category names can be found in Table 8-12.

Finetuning Settings. All reported methods use exactly the same set of hyperparameters for finetuning for a fair comparison. The median results of three runs with the same set of three different random seeds are reported. For all the methods, we finetune the model for 500 epochs in total. The peak learning rate is 1e-5, with the first 10% steps as warm-up steps. The training batch size and the validation batch size are both 256. During training, we validate the model every 25 steps and the best checkpoint is utilized to perform prediction on the test set. The experiments are carried out on one Nvidia A6000 GPU.

C Retrieval

Task. The retrieval task corresponds to fine-grained category retrieval. Given a node in the network, we aim to retrieve its fine-grained labels from a large label space. We train all the compared methods in the 16-shot setting (16 labeled queries in total) and test the models with tens of thousands of new query nodes (38,006, 33,440, 14,577, 95,731, and 34,979 for Mathematics, Geology, Economics, Clothes, and Sports, respectively). The fine-grained label spaces for both academic networks and e-commerce networks are constructed from all the labels in the network-associated taxonomy [5] [6]. The statistics of the label space for all networks can be found in Table 1.

Finetuning Settings. We finetune the models with the widely-used DPR pipeline (Karpukhin et al., 2020). All reported methods use exactly the same set of hyperparameters for finetuning for a fair comparison. The median results of three runs with the same set of three different random seeds are reported. For all the methods, we finetune the model for 1,000 epochs with the training data. The peak learning rate is 1e-5, with the first 10% steps as warm-up steps. The training batch size is 128. The number of hard BM25 negative samples[7] is set as 4. We utilize the faiss library [8] to perform an approximate search for nearest neighbors. The experiments are carried out on one Nvidia A6000 GPU.

D Reranking

Task. The reranking task corresponds to fine-grained category reranking. Given a retrieved category list for the query node, we aim to rerank all categories within the list. We train all the methods in the 32-shot setting (32 training queries and 32 validation queries) and test the models with 10,000 new query nodes and candidate list pairs. The category space in reranking is the same as that in retrieval. In our experiment, the retrieved category list is constructed with BM25 and exact matching of category names.

Finetuning Settings. All reported methods use exactly the same set of hyperparameters for fine-

[5] https://www.microsoft.com/en-us/research/project/academic/articles/visualizing-the-topic-hierarchy-on-microsoft-academic/

[6] http://jmcauley.ucsd.edu/data/amazon/links.html

[7] https://github.com/dorianbrown/rank_bm25

[8] https://github.com/facebookresearch/faiss

Table 7: Pretraining hyper-parameter configuration.

Parameter	Mathematics	Geology	Economics	Clothes	Sports
Max Epochs	30	10	30	5	10
Peak Learning Rate	1e-5	1e-5	1e-5	1e-5	1e-5
Batch Size	512	512	512	512	512
Warm-Up Epochs	3	1	3	0.5	1
Sequence Length	32	32	32	32	32
Adam ϵ	1e-8	1e-8	1e-8	1e-8	1e-8
Adam (β_1, β_2)	(0.9, 0.999)	(0.9, 0.999)	(0.9, 0.999)	(0.9, 0.999)	(0.9, 0.999)
Clip Norm	1.0	1.0	1.0	1.0	1.0
Dropout	0.1	0.1	0.1	0.1	0.1

Table 8: Class names of MAG-Mathematics.

0	mathematical optimization	5	econometrics	10	control theory	15	computational science
1	mathematical analysis	6	mathematical physics	11	geometry	16	mathematics education
2	combinatorics	7	statistics	12	applied mathematics	17	arithmetic
3	algorithm	8	pure mathematics	13	operations research		
4	algebra	9	discrete mathematics	14	mathematical economics		

tuning for a fair comparison. The median results of three runs with the same set of three different random seeds are reported. For all the methods, we finetune the model for 1,000 epochs in total with the training data. The peak learning rate is 1e-5, with the first 10% steps as warm-up steps. The training batch size and validation batch size are 128 and 256, respectively. During training, the model is validated every 1,000 steps and the best checkpoint is utilized to conduct inference on the test set. The experiments are carried out on one Nvidia A6000 GPU.

E Link Prediction

Task. The task aims to predict if there should exist an edge with specific semantics between two nodes. It is worth noting that the semantics of the edge here is different from the semantics of the edge in the pretraining text-rich network. In academic networks, the edge semantics in the pretraining network is "citation", while the edge semantics in downstream link prediction is "author overlap" [9]. In e-commerce networks, the edge semantics in the pretraining network is "co-viewed", while the edge semantics in the prediction of the downstream link is "co-purchased". We train all the methods in the 32-shot setting (32 training labeled pairs and 32 validation labeled pairs) and test the models with 10,000 new node pairs. We utilize in-batch samples as negative samples in training to finetune the model and in testing to evaluate the performance of the methods.

Finetuning Settings. All reported methods use exactly the same set of hyperparameters for finetuning for a fair comparison. The median results of three runs with the same set of three different random seeds are reported. For all the methods, we finetune the model for 200 epochs in total. The peak learning rate is 1e-5, with the first 10% step as warm-up steps. The training batch size and validation batch size are 128 and 256, respectively. During training, we validate the model in 20 steps and use the best checkpoint to perform the prediction on the test set. The experiments are carried out on one Nvidia A6000 GPU.

F Finetuning Data Size Study

We conduct a parameter study to explore how beneficial our pretraining method is to downstream tasks with different amounts of finetuning data on the four tasks on Sports. The results are shown in Figure 5, where we can find that: 1) As finetuning data increases, the performance of both PATTON and the model without pretraining (GraphFormers) improves. 2) The performance gap between PATTON and the model without pretraining (GraphFormers) becomes smaller as finetuning data increases, but PATTON is consistently better than the model without pretraining (GraphFormers).

[9] Two papers have at least one same author.

Table 9: Class names of MAG-Geology.

0	geomorphology	5	paleontology	10	petrology	15	mining engineering
1	seismology	6	climatology	11	geotechnical engineering	16	petroleum engineering
2	geochemistry	7	atmospheric sciences	12	soil science		
3	mineralogy	8	geodesy	13	earth science		
4	geophysics	9	oceanography	14	remote sensing		

Table 10: Class names of the MAG-Economics

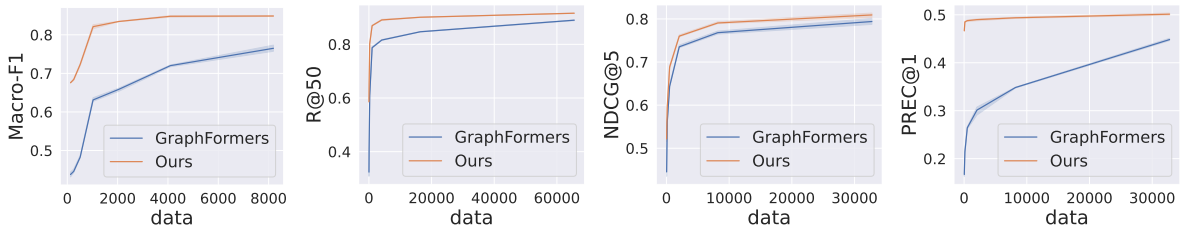
0	mathematical economics	10	economy	20	development economics	30	economic policy
1	labour economics	11	monetary economics	21	international trade	31	market economy
2	finance	12	operations management	22	keynesian economics	32	environmental economics
3	econometrics	13	actuarial science	23	positive economics	33	classical economics
4	macroeconomics	14	industrial organization	24	agricultural economics	34	management science
5	microeconomics	15	political economy	25	international economics	35	management
6	economic growth	16	commerce	26	demographic economics	36	welfare economics
7	financial economics	17	socioeconomics	27	neoclassical economics	37	economic system
8	public economics	18	financial system	28	natural resource economics	38	environmental resource management
9	law and economics	19	accounting	29	economic geography	39	economic history

Table 11: Class names of Amazon-Clothes.

0	girls	3	luggage	5	fashion watches	7	boys
1	men	4	baby	6	shoes	8	adidas
2	novelty						

Table 12: Class name of Amazon-Sports.

0	accessories	4	cycling	8	golf	12	paintball & airsoft
1	action sports	5	baby	9	hunting & fishing & game room	13	racquet sports
2	boating & water sports	6	exercise & leisure sports	10	outdoor gear	14	snow sports
3	clothing	7	fan shop	11	fitness	15	team sports



(a) Classification (b) Retrieval (c) Reranking (d) Link Prediction

Figure 5: Finetuning data size study on Amazon-Sports. As finetuning data size increases, both the performance of our proposed PATTON and the model without pretraining (GraphFormers) improves. PATTON consistently outperforms the language model without pretraining (GraphFormers).