Augmenting Low-Resource Text Classification with Graph-Grounded Pre-training and Prompting

Zhihao Wen
Singapore Management University
Singapore
zhwen.2019@smu.edu.sg

Yuan Fang
Singapore Management University
Singapore
yfang@smu.edu.sg

ABSTRACT

Text classification is a fundamental problem in information retrieval with many real-world applications, such as predicting the topics of online articles and the categories of e-commerce product descriptions. However, low-resource text classification, with few or no labeled samples, poses a serious concern for supervised learning. Meanwhile, many text data are inherently grounded on a network structure, such as a hyperlink/citation network for online articles, and a user-item purchase network for e-commerce products. These graph structures capture rich semantic relationships, which can potentially augment low-resource text classification. In this paper, we propose a novel model called Graph-Grounded Pre-training and Prompting (G2P2) to address low-resource text classification in a two-pronged approach. During pre-training, we propose three graph interaction-based contrastive strategies to jointly pre-train a graph-text model; during downstream classification, we explore prompting for the jointly pre-trained model to achieve low-resource classification. Extensive experiments on four real-world datasets demonstrate the strength of G2P2 in zero- and few-shot low-resource text classification tasks.

CCS CONCEPTS

• **Information systems** → **Content analysis and feature selection**; **Clustering and classification**.

KEYWORDS

Text classification, graph neural networks, low-resource learning, pre-training, prompt-tuning

ACM Reference Format:
Zhihao Wen and Yuan Fang. 2023. Augmenting Low-Resource Text Classification with Graph-Grounded Pre-training and Prompting. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3539618.3591641

SIGIR '23, July 23–27, 2023, Taipei, Taiwan

ACM ISBN 978-1-4503-9408-6/23/07…$15.00
https://doi.org/10.1145/3539618.3591641

1 INTRODUCTION

Text classification is a fundamental research problem with many important applications in information retrieval. For example, predicting the topics of online articles can help readers easily search and navigate within the website or portal [29], and classifying the category of e-commerce item descriptions enables businesses to structure their inventory efficiently and improve users' search experience [60]. Advances in supervised deep learning in the last decade have achieved remarkable success for text classification, especially when there are large-scale and high-quality labeled data. However, data labeling is often costly and time-consuming, making low-resource classification, in which no or few labeled samples are available, an appealing alternative.

To address low-resource text classification, one approach is to utilize pre-trained language models (PLMs) [16, 37], many of which are based on the transformer architecture [48] due to its powerful ability of encoding texts. A PLM can be adapted to different tasks by *fine-tuning* the model parameters to task-specific objectives. While the "pre-train, fine-tune" paradigm requires fewer labeled data than traditional supervised learning, it suffers from two drawbacks. First, state-of-the-art PLMs typically have a huge model size, *e.g.*, GPT-3 has 175 billion parameters [3], making fine-tuning prohibitively expensive [19]. Second, fine-tuning still needs a reasonable amount of labeled data due to the gap between pre-training and fine-tuning objectives, and thus struggles with low-resource scenarios including zero- and few-shot classification. To overcome the problem of pre-training and fine-tuning, *prompting* [3] has been proposed. It uses a natural language instruction or "prompt" to give a hint of the downstream task, whilst freezing the parameters of a large PLM. In other words, no fine-tuning or additional training is required at all for a new task. However, discrete natural language prompts can be difficult to design and may result in suboptimal performance compared to fine-tuning [18]. More recently, *prompt tuning* [18, 24] formulates a continuous prompt as a learnable embedding, which is optimized during task adaptation without updating the PLM.

Meanwhile, text data are frequently grounded on network structures, such as hyperlink or citation networks for online articles, and user-item interaction graphs for e-commerce. These graph structures expose valuable relationships between articles or items, which can be used to augment low-resource text classification. While existing PLMs and prompting do not exploit these relationships, graph neural networks (GNNs) [58] are designed to learn from graph structures based on a message-passing architecture. However, traditional end-to-end training of GNNs heavily relies on abundant task-specific labels, which motivates self-supervised GNNs [57] that employ well-designed pretext tasks on a label-free graph [14, 15, 50]. Unfortunately, the treatment of text features in

GNNs remains rudimentary. Typically, a simple bag-of-words representation [62] or aggregation of shallow word embedding vectors [30] is fed into GNNs as the initial node features, which are further propagated along graph structures. Hence, the modeling of texts in GNNs is coarse-grained, unable to fully capture the subtle semantic differences and similarities within texts.

Challenges and present work. To overcome the limitations of existing text- and graph-based solutions, we must address two open questions as follows.

Firstly, *how do we capture fine-grained textual semantics, while leveraging graph structure information jointly?* A naïve approach is to use a language model to generate features from raw texts as input, and then train a GNN. However, in this way, the texts and graph are only loosely coupled, lacking an explicit pairing to complement each other. In this paper, we propose graph-grounded contrastive pre-training, to maximize the alignment between text and graph representations based on three types of graph interaction, namely, text-node, text-summary, and node-summary interactions.

Secondly, *how do we augment low-resource text classification given a jointly pre-trained graph-text model?* We propose a novel approach of "prompting" a jointly pre-trained graph-text model instead of fine-tuning it. This allows us to leverage the most relevant structural and semantic information from the pre-trained model, making the process friendlier to low-resource scenarios. More specifically, we use handcrafted discrete prompts for zero-shot classification, and continuous prompts for few-shot settings based on automatic prompt-tuning. Due to the significantly fewer parameters involved, prompt-tuning is more label- and computation-efficient than fine-tuning the pre-trained model. Furthermore, we propose a context-based initialization for prompt-tuning that considers graph structures between texts to provide a more informative starting point.

Contributions. To summarize, we make the following contributions in this work. (1) This is the first attempt to pre-train text and graph encoders jointly for low-resource text classification. (2) We propose a novel model called Graph-Grounded Pre-training and Prompting (G2P2), with three graph interaction-based constrastive strategies in pre-training, and a prompting approach for the jointly pre-trained graph-text model in downstream tasks. (3) We conduct extensive experiments on four real-world datasets to demonstrate the strength of G2P2 in zero- and few-shot text classification.

2 RELATED WORK

Graph neural networks. Inspired by the success of convolutional networks in computer vision, GNNs have emerged to handle non-Euclidean relational data [58], ranging from early semi-supervised models such as GCN [17], GAT [49] and GIN [61], to the more recent self-supervised pre-training paradigm [14, 15, 27, 50]. Besides their widespread success in graph tasks, they have also been leveraged to improve text-based tasks through knowledge graphs [4] and heterogeneous graphs [20], or multi-modal learning [26]. However, these approaches either employ coarse-grained text treatment, or have decoupled graph and text encoders without fully exploiting the intrinsic relationship between them. Although a more recent approach called GLEM [68] integrates both the text and graph structure information by fusing language models and GNNs, it is not designed for low-resource learning.

Language pre-training and prompting. Pre-trained language models [10] have become the most popular backbone in natural langauge processing (NLP). While earlier PLMs such as GPT [37], BERT [16], XLNet [63] and RoBERTa [25] still have affordable model size, recent introductions such as T5 [38] and GPT-3 [3] produce massive models with billions of parameters. To avoid the high cost of fine-tuning on these large models, prompting [22] starts to receive more attention in the community. A prompt is a special template to pad the task input, with the goal of extracting useful knowledge from PLMs to flexibly adapt to downstream tasks. Fueled by the success of GPT-3, numerous prompting methods including discrete natural language prompt [8, 40, 42] and continuous prompt [18, 19, 24, 35, 69] have emerged. The strength of prompting has been validated in a wide range of NLP applications, including text classification [11, 13, 31, 45, 66], machine translation [46] and relation extraction [6, 39]. More recently, prompting has also been applied to GNNs for node classification [44].

Zero- or few-shot paradigms. Broadly speaking, our setting is also related to other learning paradigms. For example, in semi-supervised learning [5, 32, 59], each class may only have a few examples, but all classes must be seen in training and they cannot handle any novel class during testing. Meta-learning [1, 2, 7, 12, 52, 54, 55, 65, 70] is another popular paradigm that supports few-shot learning. However, large-scale labeled data are still required in a so-called "meta-training" phase, to support the few-shot learning of novel classes during "meta-testing". In contrast, we only need label-free data for pre-training, without requiring any meta-training phase that would consume large-scale labeled data. Separately, there also exists joint consideration of image and text data using a contrastive pre-training strategy for zero- or few-shot classification [36]. In our work, graph data are significantly different from images, which provide various types of interaction between texts. On graphs, zero-shot node classification has also been done [53]. It relies heavily on the availability of Wikipedia pages or other side information to generate class prototype embeddings. However, it is very labor intensive to find and curate the right side information, especially when there are a large number of classes and/or novel classes emerge frequently.

3 PROPOSED APPROACH

In this section, we introduce our approach G2P2 for low-resource text classification. We start with some preliminaries and an overview, and then present the details of the proposed approach.

3.1 Preliminaries

Graph-grounded text corpus. Consider a set of documents $\mathcal{D}$, which is grounded on a graph $\mathcal{G} = (\mathcal{D}, \mathcal{E}, \mathbf{X})$ such that each document $d_i \in \mathcal{D}$ is a node v_i in the graph. The documents are linked via edges in $\mathcal{E}$, which are formed based on the application (*e.g.*, if each document represents an article, the edges could be citations between articles). Each node v_i is also associated with a feature vector $\mathbf{x}_i$, given by the input feature matrix $\mathbf{X}$. Finally, each document/node[1] has a class label (*e.g.*, the topic of the article).

[1]We will use "node" and "document" interchangeably given their one-one correspondence in our context.

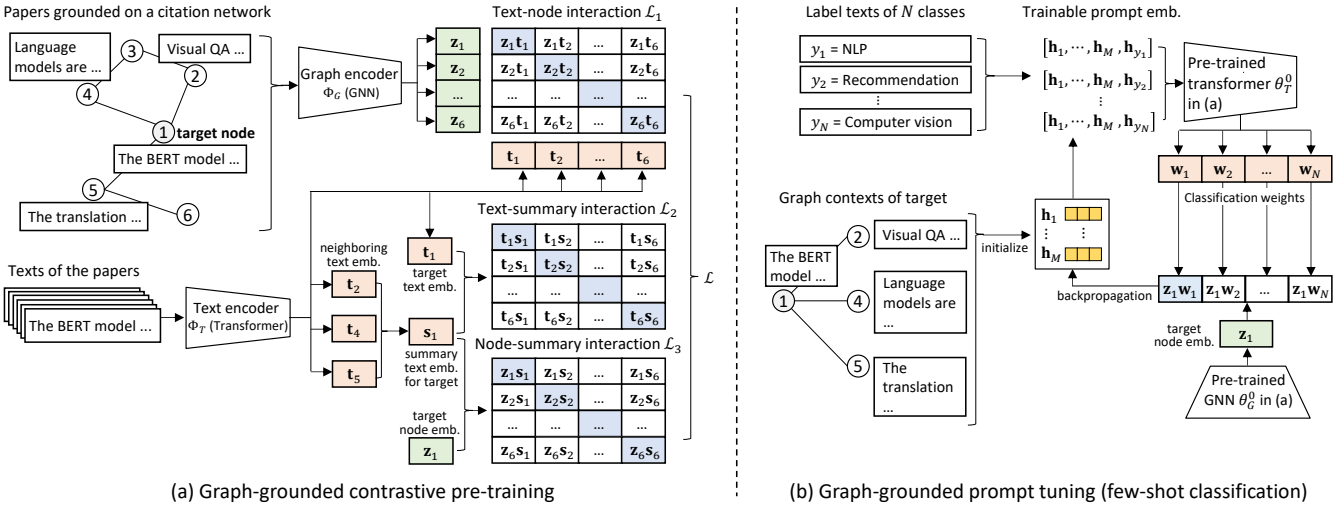


Figure 1: Overall framework of G2P2. (a) During pre-training, it jointly trains a text and a graph encoder through three contrastive strategies. (b) During testing, it performs prompt-assisted zero- or few-shot classification (the figure only shows prompt-tuning for few-shot classification, while zero-shot inference adopts a simplified scheme).

Low-resource classification. A low-resource task consists of a support set $\mathcal{S}$ and a query set $\mathcal{Q}$. The support set $\mathcal{S}$ contains N classes, and each class has K labeled examples where K is a small number (*e.g.*, 1 or 5), known as N-way K-shot classification. The query set $\mathcal{Q}$ contains one or more unlabeled instances belonging to the N classes in the support set. Our goal is to classify the instances in the query set based on the labeled examples in the support set. Unlike episodic few-shot meta-learning [7] which has both training tasks and testing tasks, we only have testing tasks; in the training stage, we perform self-supervised pre-training on label-free data only. As a special case, tasks with $K = 0$ are known as zero-shot classification, which means that there is no labeled example at all and we can only rely on class metadata (*e.g.*, class label text).

3.2 Overview of G2P2

As shown in Fig. 1, our model consists of two stages: (a) graph-grounded constrastive pre-training, and (b) graph-grounded prompt-tuning for low-resource classification.

During pre-training, we learn a dual-modal embedding space by jointly training a text encoder and graph encoder in a self-supervised fashion, since a document also exists as a node on the graph. More specifically, we use a transformer-based text encoder and a GNN-based graph encoder. The transformer takes the text on each node (*i.e.*, document) as the input, and outputs a text embedding vector $\mathbf{t}_i$ for node v_i. On the other hand, the GNN takes the graph and node features as input, and generates a node embedding vector $\mathbf{z}_i$ for node v_i. Subsequently, in the dual-modal embedding space, we align the text and graph representations on the same or related nodes through three contrastive strategies based on different types of interaction on the graph.

In downstream testing, we employ prompting on our jointly pre-trained graph-text model for zero- or few-shot classification. For zero-shot classification, we use handcrafted discrete prompts together with the label text. For few-shot classification, we use continuous prompts to pad the label text. In particular, for prompt-tuning, we initialize the continuous prompt embeddings based on graph contexts.

3.3 Graph-grounded contrastive pre-training

The graph-grounded pre-training learns a dual-modal embedding space by jointly training a text encoder and a graph encoder, based on three types of interaction on the underlying graph.

Dual encoders. The text encoder is a transformer [48], which we denote Φ_T. Given a document d_i, the text encoder[2] outputs the d-dimensional embedding vector of d_i, denoted $\mathbf{t}_i \in \mathbb{R}^d$:

$$\mathbf{t}_i = \Phi_T(d_i; \theta_T), \tag{1}$$

where θ_T represents the parameter set of the transformer. Correspondingly, let $\mathbf{T} \in \mathbb{R}^{|\mathcal{D}| \times d}$ represent the text embedding matrix for all documents.

At the same time, a document d_i is also a node v_i in the graph. We choose a classic GNN called graph convolutional network (GCN) [17] as the graph encoder, denoted Φ_Z. It similarly outputs an embedding vector $\mathbf{z}_i \in \mathbb{R}^d$ for a given node v_i:

$$\mathbf{z}_i = \Phi_Z(v_i; \theta_G), \tag{2}$$

where θ_G represents the parameter set of the GCN. Likewise, let $\mathbf{Z} \in \mathbb{R}^{|\mathcal{D}| \times d}$ represent the graph embedding matrix for all nodes.

Text-node interaction. Our graph-grounded texts naturally implies a bijection between nodes and texts, where each document d_i corresponds to the node v_i in the graph. Inspired by the pairing of image and its caption text [36] and the mapping of content and node sequences [21], we design a pre-training strategy to predict which text document matches which node in the graph.

Specifically, given n documents and the corresponding n nodes, there are n^2 possible document-node pairs $\{(d_i, v_j) \mid i, j = 1, \ldots, n\}$.

[2]Technically, the input to the text encoder is a sequence of continuous embeddings; the tokens in a document are first converted to word embeddings.

Among them, only n pairs with $i = j$ are true matching, whereas the remaining $n^2 - n$ pairs are false matching. As our first contrastive strategy, we exploit the bijective interaction between texts and nodes on the graph, to maximize the cosine similarity of the n matching pairs, while minimizing the cosine similarity of the $n^2 - n$ unmatching pairs. To compute the cosine similarity for the n^2 pairs, we first perform a row-wise L2 normalization on embedding matrices $\mathbf{T}$ and $\mathbf{Z}$ to obtain $\tilde{\mathbf{T}}$ and $\tilde{\mathbf{Z}}$, respectively. We then compute a node-text similarity matrix $\mathbf{\Lambda}_1 \in \mathbb{R}^{n \times n}$ to capture pairwise cosine similarity, as follows.

$$\mathbf{\Lambda}_1 = \left(\tilde{\mathbf{Z}}\tilde{\mathbf{T}}^\top\right) \cdot \exp(\tau), \tag{3}$$

where $\tau \in \mathbb{R}$ is a trainable temperature parameter to scale the similarity values [36].

Remark. Although $\mathbf{\Lambda}_1 \in \mathbb{R}^{n \times n}$ is a dense matrix, it is constructed batchwise for practical implementation. That is, n is not the total number of documents, but the relatively small batch size, and thus the overhead is negligible. $\mathbf{\Lambda}_2$ and $\mathbf{\Lambda}_3$ will be introduced later following the same treatment. □

To formulate the contrastive loss based on the text-node bijective interaction, we adapt the *multi-class N-pair loss* [43, 67], by considering both the row-wise and column-wise cross entropy loss w.r.t. the row or column index. For example, the i-th row of $\mathbf{\Lambda}_1$ represents the similarity scores between node v_i and every document, in which the row index i indicates the ground truth document d_i that matches v_i.

$$\mathcal{L}_1 = \tfrac{1}{2}\left(\mathrm{CE}(\mathbf{\Lambda}_1, \mathbf{y}) + \mathrm{CE}(\mathbf{\Lambda}_1^\top, \mathbf{y})\right), \tag{4}$$

where $\mathbf{y} = (1, 2, \ldots, n)^\top$ is the label vector for contrastive training, and CE denotes the cross entropy loss applied to the input matrix $\mathbf{\Lambda}_1$ or $\mathbf{\Lambda}_1^\top$ in a row-wise manner.

Text-summary interaction. Apart from the bijective text-node interaction, we further exploit higher-order interactions on the graph. In particular, each document has a set of neighboring documents defined by graph topology. The neighboring documents can be understood as a summary of the target document given the semantic relatedness between them. For example, on an e-commerce network, the products purchased by a user naturally portray a summary of the user and vice versa. Without loss of generality, we employ a simple mean pooling to generate the summary embedding $\mathbf{s}_i \in \mathbb{R}^d$ as follows.

$$\mathbf{s}_i = \tfrac{1}{|\mathcal{N}_i|} \textstyle\sum_{j \in \mathcal{N}_i} \mathbf{t}_j. \tag{5}$$

For efficiency, we only sample a fixed number of neighboring documents to generate the summary. Then, let $\mathbf{S} \in \mathbb{R}^{n \times d}$ denote the summary text embedding matrix for all documents.

Hence, as our second contrastive strategy, we seek to align the text embedding of each document and its corresponding summary text embedding, based on the text-summary interaction derived from graph neighborhood. In other words, we maximize the cosine similarity of the n matching pairs of document and its neighborhood-based summary, while minimizing the cosine similarity of the $n^2 - n$ unmatching pairs. Specifically, we first follow Eq. (3) to construct a text-summary similarity matrix $\mathbf{\Lambda}_2 \in \mathbb{R}^{n \times n}$:

$$\mathbf{\Lambda}_2 = \left(\tilde{\mathbf{T}}\tilde{\mathbf{S}}^\top\right) \cdot \exp(\tau). \tag{6}$$

Algorithm 1 Pre-training Procedure of G2P2

Require: A graph-grounded text corpus $\mathcal{G} = (\mathcal{D}, \mathcal{E}, \mathbf{X})$.
Ensure: Pre-trained weights of text encoder θ_T^0, graph encoder θ_G^0.

1: $\theta_T^0, \theta_G^0 \leftarrow$ parameters initialization;
2: **while** not converged **do**
3: sample batches of documents from $\mathcal{D}$;
4: **for** each batch **do**
5: **for** each node v_i/document d_i in the batch **do**
6: calculate d_i's text embedding $\mathbf{t}_i$; ▷ Eq. (1)
7: calculate v_i's node embedding $\mathbf{z}_i$; ▷ Eq. (2)
8: calculate v_i's summary embedding $\mathbf{s}_i$; ▷ Eq. (5)
9: **end for**
10: calculate the similatity matrices $\mathbf{\Lambda}_1, \mathbf{\Lambda}_2, \mathbf{\Lambda}_3$; ▷ Eqs. (3), (6), (8)
11: calculate the contrastive losses $\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{L}_3$; ▷ Eqs. (4), (7), (9)
12: update the overall loss $\mathcal{L}$; ▷ Eq. (10)
13: $\theta_T^0, \theta_G^0 \leftarrow$ update via backpropagation
14: **end for**
15: **end while**
16: **return** θ_T^0, θ_G^0.

Subsequently, we apply the same contrastive loss following Eq. (4), as follows.

$$\mathcal{L}_2 = \tfrac{1}{2}\left(\mathrm{CE}(\mathbf{\Lambda}_2, \mathbf{y}) + \mathrm{CE}(\mathbf{\Lambda}_2^\top, \mathbf{y})\right) \tag{7}$$

Node-summary interaction. The neighborhood-based summary for document d_i also serves as a semantic description of node v_i. Mirroring the text-summary interaction, as our third contrastive strategy, we seek to align the node embedding and its neighborhood-based summary text embedding. In the following, we similarly compute a node-summary similarity matrix $\mathbf{\Lambda}_3 \in \mathbb{R}^{n \times n}$, and formulate the corresponding contrastive loss $\mathcal{L}_3$.

$$\mathbf{\Lambda}_3 = \left(\tilde{\mathbf{Z}}\tilde{\mathbf{S}}^\top\right) \cdot \exp(\tau), \tag{8}$$

$$\mathcal{L}_3 = \tfrac{1}{2}\left(\mathrm{CE}(\mathbf{\Lambda}_3, \mathbf{y}) + \mathrm{CE}(\mathbf{\Lambda}_3^\top, \mathbf{y})\right). \tag{9}$$

Overall pre-training objective. Finally, we integrate the three contrastive losses based on the text-node, text-summary and node-summary interactions. We obtain a pre-trained model $\theta^0 = (\theta_T^0, \theta_G^0)$ consisting of the parameters of the dual encoders, given by

$$\theta^0 = \arg\min_{\theta_T, \theta_G} \mathcal{L}_1 + \lambda(\mathcal{L}_2 + \mathcal{L}_3), \tag{10}$$

where $\lambda \in \mathbb{R}^+$ is a hyperparameter to balance the contribution from summary-based interactions.

The pre-training procedure is outlined in Algorithm 1, which has the following complexity per epoch. Let $|\mathcal{D}|$ be the number of documents, η be the number of neighbors sampled to generate the summary embedding in Eq. (5), and β be the batch size. First, the cost of generating the three types of embeddings (lines 5–8) per epoch is $O(|\mathcal{D}|\eta)$, given that calculating the summary embedding needs go through η neighbors. Second, the cost of calculating the three similarity matrices in each batch is $O(\beta^2)$, and the total cost per epoch is $O\left(\frac{|\mathcal{D}|}{\beta}\beta^2\right) = O(|\mathcal{D}|\beta)$ given $\frac{|\mathcal{D}|}{\beta}$ batches in an epoch. Thus, the overall complexity is $O(|\mathcal{D}|(\eta + \beta))$, which is linear in the number of documents, since η and β are small constants. In our implementation, we set $\eta = 3$ and $\beta = 64$.

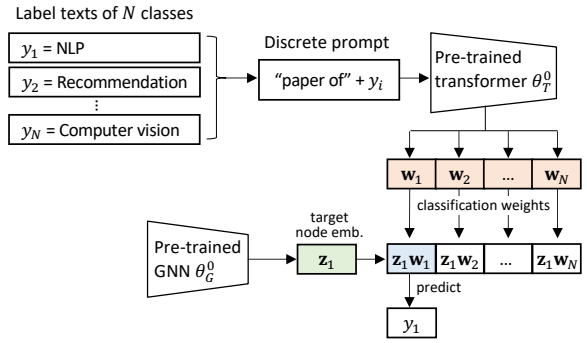


Figure 2: Schematic diagram for zero-shot classification. The pre-trained models θ_G^0 and θ_T^0 are obtained from Fig. 1(a).

3.4 Prompting joint graph-text model

After pre-training our graph-text model, it is non-trivial to apply it to low-resource classification. To narrow the gap between pre-training and downstream tasks, the traditional "pre-train, fine-tune" paradigm typically introduces a new projection head for the downstream task, which will be fine-tuned together with the whole pre-trained model. However, in a low-resource setting, it is neither effective nor efficient to update the entire model with a huge number of parameters. Without updating massive PLMs, prompting has recently emerged as a powerful alternative to fine-tuning in NLP [22]. However, prompting has not been explored for graph-text models, where structural and textual information have been jointly pre-trained. In the following, we elaborate on our prompting strategies for zero- and few-shot classification.

Zero-shot classification. In the zero-shot setting, we can only use handcrafted discrete prompts, as the absence of labeled data in zero-shot tasks cannot support learnable prompts.

In N-way zero-shot classification, out of N classes, we predict the class which has the highest similarity to the given node. As illustrated by the diagram in Fig. 2, the classification weights can be generated by the text encoder based on the class label texts [51], without requiring any labeled sample for the classification task. Specifically, the weight vector $\mathbf{w}_y$ for class $y \in \{1, 2, \ldots, N\}$ is the output of the pre-trained text encoder, *i.e.*,

$$\mathbf{w}_y = \phi_T(\text{"}\texttt{prompt [CLASS]}\text{"}; \theta_T^0). \tag{11}$$

Here "`prompt [CLASS]`" is a prompt template, where `[CLASS]` refers to the label text of the target class y (*e.g.*, "`NLP`" for paper area classification), and `prompt` is a manually engineered sequence of natural language tokens to signal the relevance of the label text (*e.g.*, "`paper of NLP`" helps focus on the topic of the paper). In the simplest case, "`prompt`" can be an empty string so that we only rely on the label text. Then, the class distribution given node representation $\mathbf{z}_i$ is predicted as

$$p(y \mid \mathbf{z}_i) = \frac{\exp\left(\langle \mathbf{z}_i, \mathbf{w}_y \rangle\right)}{\sum_{y=1}^{N} \exp\left(\langle \mathbf{z}_i, \mathbf{w}_y \rangle\right)}, \tag{12}$$

where $\langle \cdot, \cdot \rangle$ is the cosine similarity.

Few-shot classification. The problem with discrete prompts is that they are difficult to optimize, given that PLMs are intrinsically continuous. Substituting discrete natural language prompts with learnable continuous prompts, prompt tuning [18, 23, 24] can automate the optimization of prompts when some labeled data are available. Hence, in the few-shot setting, we explore prompt tuning to cue in the relevant structural and semantic information from our jointly pre-trained graph-text model.

Specicifally, instead of a sequence of discrete tokens, we take a sequence of continuous embeddings $[\mathbf{h}_1, \cdots, \mathbf{h}_M, \mathbf{h}_{\texttt{CLASS}}]$ as the prompt, where M is a hyperparameter indicating the number of context tokens, each $\mathbf{h}_m$ ($m \leq M$) is a trainable vector, and $\mathbf{h}_{\texttt{CLASS}}$ is the word embedding sequence of the target class label. The continuous prompt is fed as input to the text encoder to generate the classification weights for each class y:

$$\mathbf{w}_y = \phi_T([\mathbf{h}_1, \cdots, \mathbf{h}_M, \mathbf{h}_{\texttt{CLASS}}]; \theta_T^0), \tag{13}$$

where each $\mathbf{h}_m$ ($m \leq M$) has the same dimension as the input word embeddings to the text encoder.

Using the same softmax layer in Eq. (12), we further update the continuous prompt embeddings using the labeled support set of the few-shot task by minimizing a cross entropy loss, whilst freezing the parameters of the dual encoders. This prompt tuning process is both data- and computation-efficient, given the small number of learnable parameters in the prompt.

Furthermore, existing prompt tuning methods either initialize the prompt embeddings randomly [18, 23] or using the word embeddings of handcrafted discrete prompts [71]. While random initialization is non-informative and more prone to local optimum, it is still difficult to pick the right discrete prompts for initialization. Therefore, we take the advantage of graph structures to initialize the prompt embeddings.

Specifically, given a node v_i, we define its *graph contexts* as its neighbor set $\{v_j \mid j \in \mathcal{N}_i\}$. Due to the underlying semantic relatedness, the graph contexts of the few-shot examples carry strong signals about the task, which can be exploited to improve the initialization. For each document/node v_i in the task support set, we sample η nodes from its graph contexts. For v_i itself and each context node sampled, we truncate its corresponding document to M words, and convert it to a sequence of M word embedding vectors, each having the same dimension as the vector $\mathbf{h}_m$ ($m \leq M$) in our continuous prompt. Hence, for each support node, we would obtain $\eta+1$ such sequences; in an N-way K-shot task, there is a total of $NK(\eta+1)$ sequences. We take the average of these embedding sequences to initialize the learnable prompt vectors $\mathbf{h}_1, \ldots, \mathbf{h}_M$, which is derived from graph contexts and thus could provide a more informative starting point than random initialization.

4 EXPERIMENTS

We conduct extensive experiments to evaluate our proposed approach G2P2, with comparison to state-of-the-art baselines and model analyses.

4.1 Experimental setup

Datasets. Four public graph-grounded text corpora are used, as summarized in Tab. 1.

- **Cora**: Known as the "Cora Research Paper Classification" dataset [28], it is a collection of research papers that are linked to each

Table 1: Statistics of datasets.

Dataset	Cora	Art	Industrial	M.I.
# Documents	25,120	1,615,902	1,260,053	905,453
# Links	182,280	4,898,218	3,101,670	2,692,734
# Avg. doc length	141.26	54.23	52.15	84.66
# Avg. node deg	7.26	3.03	2.46	2.97
# Total classes	70	3,347	2,462	1,191

other through citations. The abstract of a paper is deemed a text document. The papers are classified into a topic hierarchy with 73 leaves. After removing papers with no content or label, the resulting hierarchy has 70 leaf topics. Note that we are using a more comprehensive version of the Cora dataset, which is larger and has more classes than the version used elsewhere [17].

- **Art**, **Industrial** and **Music Instruments (M.I.)** are three Amazon review datasets [33], respectively from three broad areas, namely, arts, crafts and sewing (Art), industrial and scientific (Industrial), and musical instruments (M.I.). The description of each product is deemed a text document, whereas the reviews of a user are combined into one document to reflect the user's preferences. If a user has reviewed a product, a link is constructed between them. The product subcategories within a broad area represent the classes, which are fine-grained and may involve thousands of classes with subtle differences. The classification is only performed on product descriptions, whereas the user reviews only serve to enrich the text semantics.

For all datasets, we employ the word2vec algorithm [30] to obtain the 128-dimensional word embeddings of each word in the text documents. Then, for each node, we average the word embedding vectors of all the words in its document, and the averaged vector is used as the node's input features for the GNN-based methods.

Task construction. We perform zero- or few-shot text classification. We adopt a *5-way* setting, *i.e.*, we sample five classes from all the classes to construct a task. In each task, we construct a K-shot support set by further sampling K examples from each class for $K \in \{0, 1, \ldots, 5\}$, and a validation set of the same size as the support set. The remaining examples form the query set. Note that the support set is labeled and serves as task training data, whereas the query set is unlabeled and used for evaluation. Note that in our experiments, all the classes are used—it is only that each task involves five classes, and we have multiple tasks during testing to cover all the classes. This is a typical task setup [7], which allows for a comprehensive evaluation under different class combinations. The reported results are averaged over all the tasks on each dataset.

Baselines for few-shot classification. We consider competitive baselines from four categories.

(1) *End-to-end GNNs*, which are graph neural networks trained in a supervised, end-to-end manner from random initialization.

- GCN [17]: an extension of the convolutional neural network that operates on the graph.
- SAGE_{sup} [9]: the supervised version of GraphSAGE, an inductive GNN that generates node embeddings by sampling and aggregating features from a node's local neighborhood.
- TextGCN [64]: a GCN-based model on a text graph constructed from word co-occurrence and document-word relations, which jointly learns the embeddings of both words and documents.

(2) *Pre-trained/self-supervised GNNs*, these GNNs are pre-trained using pretext tasks without labeled data, followed by fine-tuning or fitting a classification head while freezing the model parameters.

- GPT-GNN [15]: a GNN pre-training approach by a self-supervised graph generation task, including node attribute generation and edge generation. It follows the "pre-train, fine-tune" paradigm.
- DGI [50]: a GNN pre-training approach that maximizes the mutual information between local and global representations. As an unsupervised method, it also freezes the model parameters and fits a simple logistic regression model for the downstream few-shot classification, after pre-training.
- $\text{SAGE}_{\text{self}}$ [9]: the self-supervised version of GraphSAGE, encouraging similar embeddings for neighboring nodes and distinct embeddings for non-adjacent nodes. After pre-training, it follows the same approach of DGI for the downstream classification.

(3) *Pre-trained transformers*, which are pre-trained using masked language modeling [16], and then fine-tuned together with a randomly initialized classification head (*e.g.*, a fully connected layer), for the downstream few-shot classification task.

- BERT [16]: a bidirectionally trained transformer using masked language modeling, which learns from unlabeled text by being jointly conditioned on both left and right contexts in all layers.
- RoBERTa [25]: a replication of BERT that carefully measures the impact of many key hyperparameters and training data size during training.
- BERT^* and RoBERTa^*: variants of BERT and RoBERTa, which are obtained by fine-tuning the pre-trained BERT and RoBERTa, respectively, using masked language modeling on our datasets, to mitigate the domain gap between our datasets and the datasets used for pre-training BERT and RoBERTa.

(4) *Prompt tuning*: P-Tuning v2 [23], is a version of prefix-tuning [19] optimized and adapted for natural language. It uses deep prompt tuning, which applies continuous prompts for every layer of the pre-trained language model.

Note that our setting is different from few-shot learning under the meta-learning paradigm [7], since there are no few-shot tasks for the meta-training phase. Hence, we cannot use state-of-the-art meta-learning models for comparison. Besides, two of the baselines we compared, DGI and $\text{SAGE}_{\text{self}}$, have adopted a form of linear probe which is known to be a strong few-shot learner [47].

Baselines for zero-shot classification. We only compare with PLMs, as all other methods require at least one shot to work. For each method, we use the discrete prompt `[CLASS]` (*i.e.*, the label text alone). We also evaluate handcrafted prompts "`prompt [CLASS]`", where `prompt` is a sequence of tokens found by prompt engineering, and annotate the model name with "+d". Essentially, we compute the similarity between the target document and the label text of each class (with or without additional tokens), and predict the most similar class following Fig. 2.

Settings of G2P2 and baselines. For G2P2, the text encoder is a transformer [48]. Following CLIP [36], we use a 63M-parameter, 12-layer 512-wide model with 8 attention heads. It operates on a

Table 2: *Five-shot* classification performance (percent) with 95% confidence intervals.

In each column, the best result among all methods is **bolded** and the best among the baselines is underlined. Improvement by G2P2 is calculated relative to the best baseline. * indicates that our model significantly outperforms the best baseline based on the two-tail t-test ($p < 0.05$).

	Cora		Art		Industrial		M.I.	
	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
GCN	41.15±2.41	34.50±2.23	22.47±1.78	15.45±1.14	21.08±0.45	15.23±0.29	22.54±0.82	16.26±0.72
$SAGE_{sup}$	41.42±2.90	35.14±2.14	22.60±0.56	16.01±0.28	20.74±0.91	15.31±0.37	22.14±0.80	16.69±0.62
TextGCN	59.78±1.88	55.85±1.50	43.47±1.02	32.20±1.30	53.60±0.70	45.97±0.49	46.26±0.91	38.75±0.78
GPT-GNN	76.72±2.02	72.23±1.17	65.15±1.37	52.79±0.83	62.13±0.65	54.47±0.67	67.97±2.49	59.89±2.51
DGI	78.42±1.39	74.58±1.24	65.41±0.86	53.57±0.75	52.29±0.66	45.26±0.51	68.06±0.73	60.64±0.61
$SAGE_{self}$	77.59±1.71	73.47±1.53	76.13±0.94	65.25±0.31	71.87±0.61	65.09±0.47	77.70±0.48	70.87±0.59
BERT	37.86±5.31	32.78±5.01	46.39±1.05	37.07± 0.68	54.00±0.20	47.57±0.50	50.14±0.68	42.96±1.02
BERT*	27.22±1.22	23.34±1.11	45.31±0.96	36.28±0.71	49.60±0.27	43.36±0.27	40.19±0.74	33.69±0.72
RoBERTa	62.10±2.77	57.21±2.51	72.95±1.75	62.25±1.33	76.35±0.65	70.49±0.59	70.67±0.87	63.50±1.11
RoBERTa*	67.42±4.35	62.72±3.02	74.47±1.00	63.35±1.09	77.08±1.02	71.44±0.87	74.61±1.08	67.78±0.95
P-Tuning v2	71.00±2.03	66.76±1.95	76.86±0.59	66.89±1.14	79.65±0.38	74.33±0.37	72.08±0.51	65.44±0.63
G2P2-p	79.16±1.23	74.99±1.35	79.59±0.31	68.26±0.43	80.86±0.40	74.44±0.29	81.26±0.36	74.82±0.45
G2P2	**80.08*** ±1.33	**75.91*** ±1.39	**81.03*** ±0.43	**69.86*** ±0.67	**82.46*** ±0.29	**76.36*** ±0.25	**82.77*** ±0.32	**76.48*** ±0.52
(improv.)	(+2.12%)	(+1.78%)	(+5.43%)	(+4.44%)	(+3.53%)	(+2.7%)	(+6.53%)	(+7.92%)

lower-cased byte pair encoding (BPE) representation of the texts with a 49,152 vocabulary size [41]. The maximum sequence length is capped at 128. The graph encoder employs a GCN [17], using two layers [9] with LeakyReLU activation, each with 128 dimensions [34]. The pre-training of our model starts from scratch without initializing the graph and text encoders with previously pre-trained weights. λ in Eq. (10) is set to 0.1 on Cora, and set to 10 on the three Amazon review datasets, which were chosen from {0.01, 0.1, 1, 10, 100} according to the accuracy on the validation data. The number of learnable prompt tokens, M in Eq. (13), is set to 4, which was chosen from {2, 4, 8, 16, 32} based on the validation data. We use the Adam optimizer with the learning rate 2×10^{-5} with 2 training epochs, and a batch size of 64 in pre-training, referring to Hugging Face's [56] example settings. The text embedding size is 128, the same as the output of the graph encoder. To generate the summary embedding and the context-based prompt initialization, the number of neighboring nodes sampled is 3. For prompt tuning, we set the learning rate as 0.01, which was chosen from {0.0001,0.001,0.01,0.1} according to the accuracy on validation data.

For all the GNN methods, including the GNN component in G2P2, we use the 128-dimensional word2vec embeddings [30] averaged over the words in the raw texts as the input node features. We use a two-layer architecture, and set the hidden dimension to be 128, except for GCN and $SAGE_{sup}$ whose hidden dimension is set to 32 [17] which gives better empirical performance. For all GNN pre-training baselines, we use 0.01 as the learning rate. For BERT, RoBERTa and G2P2, we adopt 0.00002 as the learning rate. Our implementations of BERT, RoBERTa and their masked language modeling are based on Hugging Face's transformers [56]. For both BERT and RoBERTa, we use their base versions, given that our model G2P2 uses just a small 63M-parameter model, following previous work [36]. For P-Tuning v2, we use the original code on the RoBERTa backbone, and take the recommended 0.005 as the learning rate for prompt tuning. For G2P2, the learning rate for prompt tuning is set to 0.01.

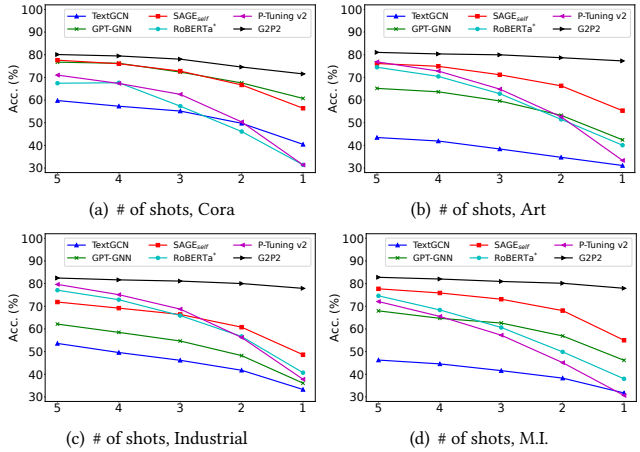
We conduct all experiments on a server with 4 units of GeForce RTX 3090 GPU. Pre-training G2P2 takes about 0.5/6/9/10 hours on Cora/M.I./Industrial/Art, respectively, on a single GPU. The inference (with prompt-tuning) is carried out with five different splits generated from five random seeds {1, 2, 4, 8, 16}.

4.2 Performance of low-resource classification

We evaluate the classification performance under various shots.

Five shots. In Tab. 2, we first compare the performance of G2P2 with baselines under the *5-shot* setting. G2P2 emerges as the winner consistently, outperforming the best baseline by around 2–8% with statistical significance.

We also make a few more observations. Firstly, among the GNNs, pre-trained/self-supervised models tend to perform better than the end-to-end approaches, since the latter heavily rely on labeled data. Among the former, DGI and $SAGE_{self}$ perform better as they are a form of linear probe, known to be a strong few-shot learner [47]. Note that, instead of using word2vec embeddings [30] of raw texts as node features, we also tired using the pre-trained RoBERTa [25] to generate the node features for DGI and $SAGE_{self}$. However, doing so does not bring any improvement, showing that it is ineffective to simply combine a language model and GNN in a decoupled manner. In contrast, our proposed model jointly learns the text and graph encoders through three graph-grounded contrastive strategies. Secondly, PLMs are generally superior to GNNs, illustrating the importance of leveraging texts in a fine-grained way. Additionally, RoBERTa outperforms BERT owing to an improved pre-training procedure [25]. However, further fine-tuning PLMs on our text data gives mixed results: RoBERTa* slightly outperforms RoBERTa but BERT* is much worse than BERT. In other words, it is not straightforward to mitigate the domain gap by simply



(a) # of shots, Cora (b) # of shots, Art

(c) # of shots, Industrial (d) # of shots, M.I.

Figure 3: Performance on different shots.

fine-tuning with the domain texts. Thirdly, the continuous prompt approach P-Tuning v2 achieves competitive results compared to fine-tuning, while having the advantage of being much cheaper than fine-tuning. However, our model G2P2 still significantly outperforms it. Furthermore, G2P2-p without prompt tuning is inferior to G2P2, showing the benefit of continuous prompts.

Fewer shots. In addition to the 5-shot setting, in Fig. 3 we also study the impact of fewer shots on G2P2 and several representative baselines. G2P2 generally performs the best across different shots. In general, the performance of all approaches decreases as the number of shots is reduced. However, the baselines suffer significantly under extreme low-resource (*e.g.*, 1- or 2-shot) settings. In contrast, G2P2 remains robust, reporting a relatively small decrease in performance even with just 1 or 2 shots.

The results demonstrate the practical value of our proposed model especially when labeled data are difficult or costly to obtain in time. On the other hand, traditional approaches constantly face the challenge of the inability to keep up with the rapid growth of emerging classes in dynamic and open environments [53]. For example, labeling a large volume of texts for novel topics in online articles, or new product categories in open-ended e-commerce platforms, can suffer a substantial time lag.

Zero shot. Finally, we report the zero-shot performance in Tab. 3, where our models G2P2 and G2P2+d significantly outperform the baselines. The results particularly demonstrate the effectiveness of our graph-grounded contrastive pre-training in the absence of labeled data, which is crucial to handling evolving classes without any labeled sample in many real-world scenarios. Moreover, handcrafted discrete prompts (*i.e.*, BERT*+d and G2P2+d) can be superior to using label text only (*i.e.*, BERT* and G2P2), showing the effectiveness of additional prompt tokens.

However, finding the optimal discrete prompts often requires significant engineering work. Specifically, for the three approaches with discrete prompts, namely, RoBERTa*+d, BERT*+d and G2P2+d, we explored more than 10 handcrafted prompt templates on each dataset, which are typically relevant to the corresponding dataset and require some domain knowledge to devise. While discrete

Table 3: *Zero-shot* classification accuracy (percent).

See Table 2 for explanations on entry styles.

	Cora	Art	Industrial	M.I.
RoBERTa	30.46±2.01	42.80±0.94	42.89±0.97	36.40±1.20
RoBERTa*	39.58±1.26	34.77±0.65	37.78±0.32	32.17±0.68
RoBERTa*+d	45.53±1.33	36.11±0.66	39.40±1.22	37.65±0.33
BERT	23.58±1.88	35.88±1.44	37.32±0.85	37.42±0.80
BERT*	23.38±1.96	54.27±1.85	56.02±1.22	50.19±0.72
BERT*+d	26.65±1.71	56.61±1.76	55.93±0.96	52.13±0.88
G2P2	63.52±2.89	76.52±0.59	76.66±0.31	74.60±0.62
G2P2+d	**65.28*** ±3.12	**76.99*** ±0.60	**77.43*** ±0.27	**75.86*** ±0.69
(improv.)	(+45.38%)	(+36.00%)	(+38.22%)	(+45.52%)

prompts are generally helpful to zero-shot classification, their effectiveness varies. In Tab. 3, we simply report the performance of the best handcrafted template for each approach and each dataset. It is also worth noting that the same prompt can sometimes generate opposite results on different models. For instance, in Cora dataset, while "`a model of [CLASS]`" is the best prompt for RoBERTa*+d, it is a bad choice for G2P2+d. Moreover, some prompts without any semantic meaning, like "`a [CLASS]`", can be the best choice sometimes. The observations imply that prompt engineering involves labor-intensive work, and the outcomes contain much uncertainty on what the optimal discrete prompt would be. Therefore, using only the label text is still a reasonably good choice.

4.3 Model analyses

We conduct more in-depth studies on G2P2. Unless otherwise stated, we report the classification *accuracy* under the *5-shot* setting.

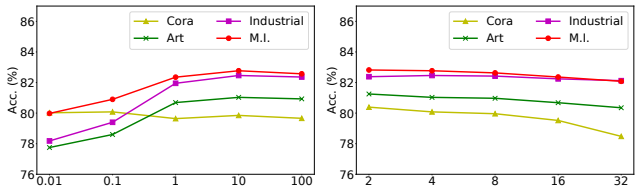
Ablation study. We first evaluate the contribution from each of the three graph interaction-based contrastive strategies, by employing different combinations of the proposed loss terms $\mathcal{L}_1$, $\mathcal{L}_2$ and $\mathcal{L}_3$. As shown in Tab. 4, strategies without $\mathcal{L}_1$ have performed quite poorly, demonstrating that the bijective text-node interaction is the fundamental component of our pre-training. That being said, when further adding $\mathcal{L}_2$ or $\mathcal{L}_3$ to $\mathcal{L}_1$, we still observe a noticeable performance improvement, showing the benefit of incorporating additional graph-based interactions for text data. Lastly, G2P2 with all three loss terms outperforms all 1- or 2-combinations of the losses, demonstrating that the three contrastive strategies are all useful and they are well integrated. Overall, the results reveal that graph information is vital to low-resource text classification, since graph structures reveal rich relationships between documents.

Next, we evaluate the contribution from our prompt-tuning approach. Specifically, we compare G2P2 with two ablated variants: using label text only without trainable prompt vectors, and randomly initializing the prompt vectors. As reported in Tab. 4, only using label text clearly degrades classification performance, implying the importance of learning continuous prompts through prompt-tuning. Furthermore, our approach G2P2 with context-based initialization for prompt vectors shows a small but consistent advantage over random initialization, which implies the usefulness of considering graph structures in prompt-tuning.

Hyperparameter study. We first investigate the impact of the interaction coefficient λ in Fig. 4(a), which balances the high-order

Table 4: Ablation study.

	Cora	Art	Industrial	M.I.
Only $\mathcal{L}_3$	74.66±1.80	52.56±1.09	45.97±0.81	49.05±0.54
Only $\mathcal{L}_2$	77.01±1.30	58.90±0.55	52.99±0.46	59.41±0.85
Only $\mathcal{L}_1$	79.50±1.19	77.37±0.72	78.10±0.34	79.70±0.56
$\mathcal{L}_2$+$\mathcal{L}_3$	70.04±2.89	49.91±1.57	50.07±0.50	56.14±1.01
$\mathcal{L}_1$+$\mathcal{L}_3$	79.73±0.89	78.60±0.40	79.97±0.43	80.42±0.45
$\mathcal{L}_1$+$\mathcal{L}_2$	79.42±1.04	80.55±0.52	81.06±0.33	82.39±0.41
Only label text	79.16±1.23	79.59±0.31	80.86±0.40	81.26±0.36
Random init.	80.03±0.99	80.85±0.43	82.43±0.33	82.64±0.21
G2P2	**80.08**±1.33	**81.03**±0.43	**82.46**±0.35	**82.77**±0.32



(a) Interaction coefficient, λ (b) Prompt length, M

Figure 4: Hyperparameter study.

contrastive losses ($\mathcal{L}_2$, $\mathcal{L}_3$). The performance is generally better and stable when λ is slightly bigger (*e.g.*, $\geq$ 10), indicating the significance of the high-order text-summary and node-summary interactions. Next, we study the prompt length M in Fig. 4(b), which refers to the number of trainable prompt vectors in Sect. 3.4. The performance is relatively unaffected by the prompt length, and thus it is robust to choose a small M (*e.g.*, 4) for efficiency.

Efficiency of prompt tuning. In our work, the continuous prompts are optimized by prompt tuning [23, 71] without updating the pre-trained model. In this experiment, we investigate the efficiency of prompt-tuning in G2P2 compared to the efficiency of traditional fine-tuning. As G2P2 has a transformer component, we compare it with four transformer based models, all of which follow the classical "pre-train, fine-tune" paradigm [16].

As shown in Tab. 5, "Tuning time per task" refers to the average time required per task for prompt-tuning in G2P2 or fine-tuning in the baselines, while "Param. size" refers to the number of parameters that require updating. The results demonstrate that prompt tuning in G2P2 is much more efficient than fine-tuning in the baselines, achieving 2.1~18.8x speedups. The reason is that prompt tuning updates far fewer parameters. In G2P2, we used 4 trainable 512-dimensional prompt vectors, totaling 2048 parameters only, while fine-tuning in the baselines needs to update the whole pre-trained model with more than 100M parameters. Note that the speedup is not linear w.r.t. the parameter size, due to the overhead in the data loader and the optimizer. Overall, our prompt tuning is not only effective under low-resource settings, but also parameter- and computation-efficient.

Generalization study. Our previous experiments can be considered "transductive" as both the pre-training of the text encoder and the downstream classification are conducted on the whole corpus. To further evaluate the generalization ability of our model, we adopt an "inductive" setting, whereby we pre-train the text encoder only on a subset of the corpus and perform downstream classification on a disjoint subset. Particularly, in the three Amazon datasets, since user texts have no labels and item texts have labels, it is natural for us to pre-train with only user texts and classify only item texts downstream. We also employ masked language modeling on only the user texts for BERT and RoBERTa, to get BERT* and RoBERTa*. As shown in Tab. 6, G2P2 still performs very well in the inductive setting, illustrating the strong generalization ability of our pre-trained model.

Table 5: Tuning time and parameter size.

	Tuning time per task (in seconds)				Param.
	Cora	Art	Industrial	M.I.	size
RoBERTa	45.47±2.38	64.22±3.62	43.46±2.99	44.99±2.58	123 M
RoBERTa*	39.38±2.01	59.56±3.55	35.10±2.75	38.84±2.39	123 M
BERT	32.23±1.71	51.77±2.00	31.72±1.77	33.55±2.39	110 M
BERT*	34.82±1.68	55.16±2.32	31.11±1.74	29.00±2.23	110 M
G2P2	**2.42**±0.41	**22.03**±1.39	**14.63**±1.26	**12.72**±1.17	**2048**

Table 6: Inductive performance on text classification.

	Art	Industrial	M.I.
BERT*	43.66±0.90	48.35±0.25	39.24±0.88
RoBERTa*	69.55±1.14	73.65±0.86	71.96±1.44
G2P2	**79.81**±0.22	**81.29**±0.32	**81.85**±0.33

5 CONCLUSION

In this paper, we studied the problem of low-resource text classification. Given that many text documents are related through an underlying network, we proposed a novel model called Graph-Grounded Pre-training and Prompting (G2P2). It consists of three graph interaction-based contrastive strategies in pre-training, and a prompting mechanism for the jointly pre-trained graph-text model in downstream classification. We conducted extensive experiments and showed the advantages of G2P2 in zero- and few-shot text classification.

A limitation of this work is the need of a graph to complement the texts. Although graphs are ubiquitous in information retrieval applications, in the case that an organic graph is unavailable, a potential solution is to construct synthetic graphs based on word co-occurrences or other relations, *e.g.*, linking up news articles in close time periods and locations. We leave further explorations to future work.

ACKNOWLEDGMENTS

This research / project is supported by the Ministry of Education, Singapore, under its Academic Research Fund Tier 2 (Proposal ID: T2EP20122-0041). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the Ministry of Education, Singapore.

REFERENCES

[1] Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. 2020. Self-Supervised Meta-Learning for Few-Shot Natural Language Classification Tasks. In *EMNLP*. 522–534.

[2] Yujia Bao, Menghua Wu, Shiyu Chang, and Regina Barzilay. 2020. Few-shot Text Classification with Distributional Signatures. In *ICLR*.

[3] 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* 33 (2020), 1877–1901.

[4] Xianshuai Cao, Yuliang Shi, Han Yu, Jihu Wang, Xinjun Wang, Zhongmin Yan, and Zhiyong Chen. 2021. DEKR: description enhanced knowledge graph for machine learning method recommendation. In *SIGIR*. 203–212.

[5] Jiaao Chen, Zichao Yang, and Diyi Yang. 2020. MixText: Linguistically-Informed Interpolation of Hidden Space for Semi-Supervised Text Classification. In *ACL*. 2147–2157.

[6] Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2022. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction. In *Proceedings of the ACM Web Conference 2022*. 2778–2788.

[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*. 1126–1135.

[8] Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making Pre-trained Language Models Better Few-shot Learners. In *ACL*. 3816–3830.

[9] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NeurIPS*.

[10] Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Yuan Yao, Ao Zhang, Liang Zhang, et al. 2021. Pre-trained models: Past, present and future. *AI Open* 2 (2021), 225–250.

[11] Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. Ptr: Prompt tuning with rules for text classification. *arXiv preprint arXiv:2105.11259* (2021).

[12] Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel: A Large-Scale Supervised Few-Shot Relation Classification Dataset with State-of-the-Art Evaluation. In *EMNLP*. 4803–4809.

[13] Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Jingang Wang, Juanzi Li, Wei Wu, and Maosong Sun. 2022. Knowledgeable Prompt-tuning: Incorporating Knowledge into Prompt Verbalizer for Text Classification. In *ACL*. 2225–2240.

[14] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. 2019. Strategies for Pre-training Graph Neural Networks. In *ICLR*.

[15] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. 2020. Gpt-gnn: Generative pre-training of graph neural networks. In *KDD*. 1857–1867.

[16] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*. 4171–4186.

[17] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*. OpenReview.net.

[18] Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning. In *EMNLP*. 3045–3059.

[19] Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *ACL*. 4582–4597.

[20] Hu Linmei, Tianchi Yang, Chuan Shi, Houye Ji, and Xiaoli Li. 2019. Heterogeneous graph attention networks for semi-supervised short text classification. In *EMNLP*. 4821–4830.

[21] Jie Liu, Zhicheng He, Lai Wei, and Yalou Huang. 2018. Content to node: Self-translation network embedding. In *KDD*. 1794–1802.

[22] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586* (2021).

[23] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-Tuning: Prompt Tuning Can Be Comparable to Fine-tuning Across Scales and Tasks. In *ACL*. 61–68.

[24] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. GPT understands, too. *arXiv preprint arXiv:2103.10385* (2021).

[25] 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* (2019).

[26] Yong Liu, Susen Yang, Chenyi Lei, Guoxin Wang, Haihong Tang, Juyong Zhang, Aixin Sun, and Chunyan Miao. 2021b. Pre-training graph transformer with multimodal side information for recommendation. In *ACMMM*. 2853–2861.

[27] Yuanfu Lu, Xunqiang Jiang, Yuan Fang, and Chuan Shi. 2021. Learning to pre-train graph neural networks. In *AAAI*, Vol. 35. 4276–4284.

[28] Andrew McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. 2000. Automating the Construction of Internet Portals with Machine Learning. *Information Retrieval Journal* 3 (2000), 127–163. www.research.whizbang.com/data.

[29] Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. 2000. Automating the construction of internet portals with machine learning. *Information Retrieval* 3, 2 (2000), 127–163.

[30] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[31] Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Noisy Channel Language Model Prompting for Few-Shot Text Classification. In *ACL*. 5316–5330.

[32] Takeru Miyato, Andrew M. Dai, and Ian J. Goodfellow. 2017. Adversarial Training Methods for Semi-Supervised Text Classification. In *ICLR*. OpenReview.net.

[33] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *EMNLP*. 188–197.

[34] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *KDD*. 701–710.

[35] Guanghui Qin and Jason Eisner. 2021. Learning How to Ask: Querying LMs with Mixtures of Soft Prompts. In *NAACL*. 5203–5212.

[36] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *ICML*. PMLR, 8748–8763.

[37] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. (2018).

[38] 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. *J. Mach. Learn. Res.* 21, 140 (2020), 1–67.

[39] Oscar Sainz, Oier Lopez de Lacalle, Gorka Labaka, Ander Barrena, and Eneko Agirre. 2021. Label Verbalization and Entailment for Effective Zero and Few-Shot Relation Extraction. In *EMNLP*. 1199–1212.

[40] Timo Schick and Hinrich Schütze. 2021. Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference. In *EACL*. 255–269.

[41] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *ACL*. Association for Computational Linguistics (ACL), 1715–1725.

[42] 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 *EMNLP*. 4222–4235.

[43] Kihyuk Sohn. 2016. Improved deep metric learning with multi-class n-pair loss objective. *NeurIPS* 29 (2016).

[44] Mingchen Sun, Kaixiong Zhou, Xin He, Ying Wang, and Xin Wang. 2022. Gppt: Graph pre-training and prompt tuning to generalize graph neural networks. In *KDD*. 1717–1727.

[45] Yi Sun, Yu Zheng, Chao Hao, and Hangping Qiu. 2021. NSP-BERT: A Prompt-based Zero-Shot Learner Through an Original Pre-training Task–Next Sentence Prediction. *arXiv preprint arXiv:2109.03564* (2021).

[46] Zhixing Tan, Xiangwen Zhang, Shuo Wang, and Yang Liu. 2022. MSP: Multi-Stage Prompting for Making Pre-trained Language Models Better Translators. In *ACL*. 6131–6142.

[47] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. 2020. Rethinking few-shot image classification: a good embedding is all you need?. In *ECCV*. Springer, 266–282.

[48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *NeurIPS* 30 (2017).

[49] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.

[50] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep Graph Infomax. *ICLR* 2, 3 (2019), 4.

[51] Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. 2018. Joint Embedding of Words and Labels for Text Classification. In *ACL*. 2321–2331.

[52] Ning Wang, Minnan Luo, Kaize Ding, Lingling Zhang, Jundong Li, and Qinghua Zheng. 2020. Graph few-shot learning with attribute matching. In *CIKM*. 1545–1554.

[53] Zheng Wang, Jialong Wang, Yuchen Guo, and Zhiguo Gong. 2021. Zero-shot node classification with decomposed graph prototype network. In *KDD*. 1769–1779.

[54] Zhihao Wen. 2023. Generalizing graph neural network across graphs and time. In *WSDM*. 1214–1215.

[55] Zhihao Wen, Yuan Fang, and Zemin Liu. 2021. Meta-inductive node classification across graphs. In *SIGIR*. 1219–1228.

[56] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *EMNLP*. 38–45.

[57] Lirong Wu, Haitao Lin, Cheng Tan, Zhangyang Gao, and Stan Z Li. 2021. Self-supervised learning on graphs: Contrastive, generative, or predictive. *TKDE* (2021).

[58] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *TNNLS* 32, 1 (2020), 4–24.
[59] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *NeurIPS* 33 (2020), 6256–6268.
[60] Hu Xu, Bing Liu, Lei Shu, and P Yu. 2019. Open-world learning and application to product classification. In *The World Wide Web Conference*. 3413–3419.
[61] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks?. In *ICLR*. OpenReview.net.
[62] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. 2016. Revisiting semi-supervised learning with graph embeddings. In *ICML*. PMLR, 40–48.
[63] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *NeurIPS* 32 (2019).
[64] Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *AAAI*, Vol. 33. 7370–7377.
[65] Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar Yu Cheng Gerald Tesauro, Haoyu Wang Bowen Zhou, and AI Foundations-Learning. 2018. Diverse Few-Shot Text Classification with Multiple Metrics. In *NAACL*. 1206–1215.
[66] Wenxuan Zhang, Yang Deng, Xin Li, Yifei Yuan, Lidong Bing, and Wai Lam. 2021. Aspect Sentiment Quad Prediction as Paraphrase Generation. In *EMNLP*. 9209–9219.
[67] Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D Manning, and Curtis P Langlotz. 2020. Contrastive learning of medical visual representations from paired images and text. *arXiv preprint arXiv:2010.00747* (2020).
[68] Jianan Zhao, Meng Qu, Chaozhuo Li, Hao Yan, Qian Liu, Rui Li, Xing Xie, and Jian Tang. 2023. Learning on Large-scale Text-attributed Graphs via Variational Inference. In *ICLR*. https://openreview.net/forum?id=q0nmYciuuZN
[69] Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual Probing Is [MASK]: Learning vs. Learning to Recall. In *NAACL*. 5017–5033.
[70] Fan Zhou, Chengtai Cao, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Ji Geng. 2019. Meta-gnn: On few-shot node classification in graph meta-learning. In *CIKM*. 2357–2360.
[71] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. 2022. Learning to prompt for vision-language models. *IJCV* 130, 9 (2022), 2337–2348.