ConsPrompt: Easily Exploiting Contrastive Samples for Few-shot Prompt Learning

Jinta Weng, Yue Hu
Institute of Information Engineering
Chinese Academy of Science
School of CyberSpace Security
wengjinta@iie.ac.cn

Heyan Huang
School of Computer Science
Beijing Institute of Technology
hhy63@bit.edu.cn

Abstract

Prompt learning recently become an effective linguistic tool to motivate the PLMs’ knowledge on few-shot-setting tasks. However, studies have shown the lack of robustness still exists in prompt learning, since suitable initialization of continuous prompt and expert-first manual prompt are essential in fine-tuning process. What is more, human also utilize their comparative ability to motivate their existing knowledge for distinguishing different examples. Motivated by this, we explore how to use contrastive samples to strengthen prompt learning. In detail, we first propose our model ConsPrompt combining with prompt encoding network, contrastive sampling module, and contrastive scoring module. Subsequently, two sampling strategies, similarity-based and label-based strategies, are introduced to realize differential contrastive learning. The effectiveness of proposed ConsPrompt is demonstrated in five different few-shot learning tasks and shown the similarity-based sampling strategy is more effective than label-based in combining contrastive learning. Our results also exhibit the state-of-the-art performance and robustness in different few-shot settings, which proves that the ConsPrompt could be assumed as a better knowledge probe to motivate PLMs. As far as we could reach, this is the first work exploring how to use contrastive learning approach and suitable contrastive samples to enhance prompt-based fine-tuning.

1 Introduction

With the exponential growth of Pre-trained Language Models (PLMs) parameters, PLMs now could serve as a huge knowledge base or linguistic tool for few-shot learning (Radford et al., 2019). Ideally, PLMs are similar with a huge linguistic memory of human brain, and thus PLMs are able to realize the fast classification or inference task via the finetuning of few-shot samples (Cao et al., 2021). However, most down-stream nature language tasks, though covering with huge training corpus, are still hard to realize more competitive result than humans. What is more, sufficient training corpus is often impractical in real scenarios. Therefore, how to carry out few-shot learning on PLMs and utilize more learning ability of humans have become a hotspot recently.

Roberts et al. found that using nature language query in fine-tuning process can achieve competitive results in knowledge probe (Roberts et al., 2020). Given simple task description or few in-context samples, GPT-3 is able to finish many NLP tasks efficiently with in-context learning (Liu et al., 2021). These researches shift current few-shot learning task from fine-tuning paradigm to prompt-based fine-tuning paradigm, which utilize different prompt strategies to activate the knowledge generation of PLMs. In different task-oriented applications of prompt strategies, Schick (Schick and Schütze, 2021; Schick et al., 2020) firstly introduces a PET model for few-shot text classification, which adds a cloze-question template in original input and then uses the embedding of masked token in label mappings to represent probability distribution of labels. Han et al. design some basic sub-prompts manually for text classification and apply some logical rules to combine these sub-prompts to construct a task-specific prompt (Han et al., 2021). Ding et al. (2021) construct an entity-oriented verbalizer and templates used for fine-grained entity tagging. Madotto et al. also use only few examples of conversation skills and prompt learning method to realize an end-to-end Bot for question answering (Madotto et al., 2021), and Zhong et al. (2021) put forward a OptiPrompt model for factual probing based on soft-prompt embedding (Zhong et al., 2021).

In general, these researches have certified the power of prompt-based fine-tuning in some degree. However, it has been found that most prompt-learning method always meet with issues of label-mapping robustness and template robustness (Web-
son and Pavlick, 2021), for example, only minor change of hard prompt or initialization of soft-prompt embedding leads task performance vary greatly, which reveals the limitation of the prompt-based motivating strategies in PLMs. As a matter of fact, based on internal memory of brain, human always learning with new samples by different knowledge processing operations, like knowledge prompt (e.g. prompt-based fine-tuning), knowledge comparison, knowledge inference, etc. We consider that utilizing the prompt strategies directly seems to ignore other complicated human’s knowledge processing methods. In detail, except prompt learning in PLMs to realize knowledge prompt, human also uses different learning rules to realize knowledge comparison or similarity learning. Therefore, in order to realize better few-shot prompt learning and empower its robustness of prompt learning, this paper try to exploit how to utilize the contrastive samples and similarity samples to realize the contrastive learning of prompt-based fine-tuning task.

We proposed our developing model —— ConsPrompt (Contrastive-sampling prompt), a contrastive prompt for few-shot prompt-based learning task. We first proposed two improved strategies for negative sampling, then we utilize the pre-trained language model SBERT (Reimers and Gurevych, 2019) to support the fast construction of ConsPrompt. What is more, we introduce the contrastive learning method SupContrast into the current prompt-based fine-tuning task. Finally, we verify the validity of the model ConsPrompt on different NLP tasks. Our result shows the effectiveness of utilize contrastive learning in prompt-based fine-tuning task. And the robustness of prompt-based learning is able to empower with suitable negative sampling strategies. As far as we could reach, this is the first paper to discuss a contrastive learning approach in prompt-based fine-tuning work. Our detailed contributions are as follows:

- Aimed at few-shot learning task, we proposed a model ConsPrompt combining with prompt-based fine-tuning and contrastive learning to empower model’s robustness.
- Two negative sampling strategies, similarity-based and labeled-based methods are proposed to enhance the attention for contrastive scoring module.
- We explore our proposed model ConsPrompt in few-shot setting and five different tasks, and the results demonstrate the state-of-the-art performance and show its effective on few-shot learning tasks.

![Figure 1: Using prompt learning method in emotion analysis task. The input would add with a fixed template T (e.g., some words or sentence) and a [MASK] token to combine the real input, and the task prediction would then transit into the prediction of specific label-mapping word given the hidden value of mask token.](image)

2 Theory Foundation

2.1 Prompt-based fine-tuning

Our main work is based on the design of hard prompt and prompt-based fine-tuning strategy in few-shot learning setting (Gao et al., 2020). Since the prompt-based fine-tuning method of different language tasks may be widely different, we take emotion classification as an example depicted in fig. 1 to explain the prompt-based fine-tuning way. We use D to represent the training set and \( D = (x_i, y_i) \). The task of emotion classification is divide \( x_i \) to specif emotional type \( y_i \), for example, positive or negative type in SST-2 task of emotion classification.

In prompt-based fine-tuning method, a template \( T \) consisting of continuous tokens and mask token would then be defined. Consequently, the original input \( x_i \) would then be transformed to \( x^i \). For example, if the template is adding \( It is [MASK] \) after origin input, the final input would be:

\[
\pi^i = T(x^i) = x^i. It is [MASK]
\]

, where the hidden value of [MASK] token is used to generate the word distribution over PLMs vocabulary consequently.

Only the distribution of special tokens would be chosen to represent the labels of current task. The detail formulation of label mapping is defined by:

\[
F(y): y_t \rightarrow v_t, y_t \in Y, v_t \in V
\]
Figure 2: The illustration of combining the contrastive learning and fine-tuning of emotion classification task.

, where $v$ is one of tokens in PLMs vocabulary, and $y_t$ is the labels of current task. And the final prediction would formulate in following equation:

$$p(y^t|x^i) = p(v^t|h_{[\text{MASK}]}) \Rightarrow p(v^t|W \cdot h_{[\text{MASK}]}) \Rightarrow p(v^t|h_{v_t}) = \ln \frac{\exp(h_{v_t})}{\sum_{k=1}^{|Y|} \exp(h_{v_k})}$$

(3)

, where the size of $W$ is $|h \times V|$, $v$ used to represent each token of the label mapping. The final loss of current task is formulated as:

$$L = -\frac{1}{N} \sum_{i} \sum_{t=1}^{|Y|} y_i \log p(y^t|x^i)$$

(4)

, where $i$ is the index of the training pair $(x_i, y_i)$.

With the help of defined prompts and label mappings, the label prediction is calculated by the label-mapping tokens of PLMs.

2.2 Prompt tuning with Contrastive samples

In order to learn higher-level features about the data, humans always comparing and recognizing “similar” and “different” pairs of samples by their distinction. Thus, comparative learning could seem as another fast learning method in the situation of few-shot samples and existing knowledge. Figuratively, we draw a diagram of how contrastive learning is used to transform the model learning process in Fig. 2.

The proposed diagram are based on the representation visualization of the PLMs. We consider the token-level representation are semantically organized in a internal token sphere, for example, queen-king and fruit-apple are located with similarity. The representation of training text are organized in a text sphere that restrictively decided by the related words of external token sphere. In fine-tuning process, like the red arrow depicted in Fig. 2, both token-level and text-level representation are dynamic adjusted to the special emotion surface to adapt the current training corpus. For example, during emotion classification task, the green text dot “I like movie” with positive emotion are moving to the positive aspect of blue emotion surface from timestamp $t$ to timestamp $t+1$. Although roughly plotted contrastive learning with the yellow row, it reveals the contrastive learning in similar and negative samples could not only contribute to the fine-tuning of current step, but also optimize the fine-tuning direction of text representation to the most related emotion surface of current text.

Therefore, the proposed manuscript put forward a model named as ConsPrompt to utilize the human ability of learning contrastive samples in prompt-based fine-tuning method, and thus increase the diversity and robustness of the learning process.

3 Model

The core component of our proposed framework is the contrast-aware prompt learning module ConsPrompt depicted in fig. 3. The ConsPrompt...
are constituted by prompt-based Encoding Network, contrastive sampling module, and contrastive learning encoder. In prompt encoding network (§3.1), the original input would be encoded to prompting input, and then fine-tune in label-mapping tokens as (§2.1). What is more, in contrastive sampling module (§3.2), the prompting input would generate negative and positive samples by similarity-based or label-based strategies, while these samples are subsequently used to realize contrastive learning in contrastive learning encoder (§3.3). Finally, we apply a joint loss with respect to prompt encoding network and contrastive learning encoder to fine-tune the ConsPrompt.

3.1 Prompt Encoding Network

Owing to increasing attention in prompt-based learning domain, prompt-based fine-tuning method can also be divided into different categories. In the representation of prompt, it could divide into soft and hard prompt learning, which use different apparent or implicit embedding to activate the PLM’s power. In terms of training strategies, prompt learning could classify into the fine-tuning of prefix-prompt and fine-tuning of independent prompt encoder, such as P-tuning(Liu et al., 2021), Prefix-tuning(Li and Liang, 2021), Auto-prompt learning(Shin et al., 2020), and Prompt-tuning(Schick and Schütze, 2021). In order to preserve the semantics correlation and lower training parameters, we follow the setting of LMBBFF model (Gao et al., 2020) and choose hard prompt as the baseline of our prompt-based fine-tuning method2.1).

In this method, a template for input transformation and label mapping for each label should be predefined. Thus, the training data $D = (x_i, y_i)$ would transform to $\overline{D} = (\overline{x}_i, \overline{y}_i)$ with a prompting template $T$ and special $\lfloor MASK \rfloor$ token in $x_i$. What is more, each of labels is represented by a semantic-concerned token. The training loss is the model’s preference on generating which tokens of semantic-concerned token. The training loss is the representation of prompt, it could divide into soft and hard prompt learning, which use different apparent or implicit embedding to activate the PLM’s power. In terms of training strategies, prompt learning could classify into the fine-tuning of prefix-prompt and fine-tuning of independent prompt encoder, such as P-tuning(Liu et al., 2021), Prefix-tuning(Li and Liang, 2021), Auto-prompt learning(Shin et al., 2020), and Prompt-tuning(Schick and Schütze, 2021). In order to preserve the semantics correlation and lower training parameters, we follow the setting of LMBBFF model (Gao et al., 2020) and choose hard prompt as the baseline of our prompt-based fine-tuning method2.1).

$$L_{ce} = - \sum_{t=1}^{\lvert Y \rvert} y_t \log p(y_t|M(T(x^t))_{mask})$$ (5)

,where $y_t$ is the label-mapping token of label $t$, $M_{mask}$ is the the hidden value of PLM in $mask$ position, and $\lvert Y \rvert$ is the total number of labels.

3.2 Contrastive Sampling Module

The essence of contrastive learning is to learn the differences between positive and negative samples. Therefore, a suitable sampling strategies for selecting positive and negative samples is essential to develop contrastive learning. Most sampling strategies in NLP are based on the text augmentation, for example, word-level operation or label-based rewriting. Some methods also use randomly sampling strategy to collect positive and negative sample by their label consistency. However, the method of text augmentation could interfere the model since the text augmentation methods are not guaranteed to generate positive or negative samples every time. What is more, directly sampling based on label consistency may ignore its semantic similarity and hard to realize fast learning. Therefore, we introduce the sentence-Bert model SBERT to determine the negative and positive support sets. In detail, we utilize SBERT model as the representation of all samples in support set $S_q$, and then we calculate the similarity between the query instance $q$ and each sample of $S_q$ with cosine similarity algorithm. The final support set would be further ordered by the max similarity:

$$S^n_q = sim(LM_{ab}(q), LM_{ab}(S^n_q))$$

$$S_q = \text{Rank}_{max}(S_q)$$ (6)

, where $n$ is the number of supporting set $S_q$ used to generate contrastive samples, and $q$ is the query sample.

Note that the SBERT model is only use for generating the similarity-based order between the sentence representation of current input and its supporting set. After generating the similarity-based order, the representation of the positive samples and negative samples still use the same PLM of prompt encoding network.

$$s^n_q = PLM(S^n_q)$$ (7)

, where $s^n_q$ is the PLM representation of $S^n_q$ generating by the same pre-trained language model of prompt-based fine-tuning method.

3.3 Contrastive Scoring Module

Since contrast learning was proposed, it has been widely used in computer vision (Wu et al., 2018; Chen et al., 2020; Chen and He, 2021) and natural language processing (Yan et al., 2021; Gao et al., 2021; Wang et al., 2021) to learn the distinction
of positive and negative samples. The Contrastive Scoring Module is another core component that we used to learn the representation of above-mentioned distinction within support set. In detail, we only use one positive sample selecting by the highest similarity of support set, while negative samples are selected by the lower similarity among in other different labels of current input. After selecting one suitable positive sample and negative samples, we use same PLM of prompt encoding network to representing each samples and utilize the SimCLR model as our contrastive learning method (Chen et al., 2020). In this SimCLR model, the similarity of positive sample \( j \) and original input \( i \) is defined as:

\[
S(i,j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k \neq i} \exp(s_{i,k}/\tau)}
\]

, where \( k \) is the index of negative sample set, \( \tau \) is a controllable temperature parameter.

What is more, owing to the difference negative support set of the positive sample \( j \) and original input \( i \), we thus switch their position in \( S(i,j) \) to calculate another inverted loss. The final loss of a batch are thus defined as:

\[
L_{cot} = \frac{1}{N} \sum_{i} [S(i,j) + S(j,i)]
\]

, where \( N \) is the number of batch size.

Considering the loss of Contrastive Scoring Module and Prompt Encoding Network, the final loss of ConsPrompt is defined as:

\[
L = L_{ce} + t \cdot L_{cot}
\]

, where \( t \) is the hyper-parameter to determine the ratio of comparative learning loss.

4 Experiment

In this section, we introduce our using datasets, experimental setup, the main results of our experiments and its analyses.

4.1 Datasets

We evaluate our proposed model ConsPrompt on different tasks, including question classifications, emotion classification tasks, and text entailment tasks. In sentiment classification, we choose two representative SST Task, sst-2 and sst-5. The goal of SST dataset is given a movie review to predict its emotion. TREC task is a problem classification task, consisting of open domain, fact-based questions that are divided into broad semantic categories. The Stanford Natural Language Inference (SNLI) Corpus is a collection of English sentence pairs written by 570K people, manually tagged to balance classification, label meaning, contradiction, and neutrality. The QNLI is a natural language reasoning dataset automatically derived from the Stanford Questions dataset (SQuAD). The task is to determine whether the context sentence contains the answer to the question.

4.2 Experimental setup

We use RoBERTa-large as our pre-training language model. Our experiment are developed in NVIDIA V100 32GB (also could run in 1080ti with low batch size). In the training process, we
develop many experiments on different batch sizes $bs=4,8,16$, learning rate $lr=1e^{-5},2e^{-5},5e^{-5}$. For the reason that some study have shown the minor change on few-shot learning could lead differentiated results, we use five different sub-sets with the same small size. Also, in order to control the fairness and robustness of experiment, we use the mean score and variance of the prediction result on different subsets instead of the highest.

To satisfy the few-shot learning in PLMs, following the setting in Gao’s work, we pick five different K-shot sub-datasets and each sub-dataset is constructed by K=16 training pairs on each type of labels (Gao et al., 2020). For example, the SST-2 emotion classification task with two classes needs to construct five different training sets with the size of 32 and validation set with the same size of 32, while the testing size would use original size of the SST-2 task, without any other data setup. These sub-datasets are training and predicting singly. Their training process and prediction result would be recorded by different batch sizes and learning rates. All training pair would use consistent template transition in proposed prompting strategies to construct the prompting input. Finally, we choose the best result of each sub-dataset on different hyperparameters, and integrate the best result on the aspect of sub-dataset.

In the setting of contrastive learning, we select all instances except query instance q to combine the initial supporting set. In order to decree the calculation within initial supporting samples, we set the filtering ratio of the support set to 0.5. We set $\tau = 0.07$ to realize the smoothing of loss, and the ratio $t$ of the loss of contrastive learning is 0.5.

We chose the evaluation criteria in integrating the prediction of sub-datasets as the average accuracy and variance of different sub-dataset, since it reveal the overall performance and variation of current task.

### 4.3 Baselines

We compare with a number of baselines in related works and set our baselines with Majority method (merely select the majority class as prediction), fine-tuning method (Liu et al., 2019), prompt-based zero-shot learning, GPT3-in-context-learning (Brown et al., 2020) and LMBFF model (The method detail refers to Gao’s work (Gao et al., 2020)). Based on the sampling strategies of negative samples, our proposed baseline ConsPrompt is divide into two following categories:

- **ConsPrompt(-sim)**: the negative and positive samples are directly sampled based on the similarity between the support sets and query instance.

- **ConsPrompt(-label)**: the negative and positive samples must sampled from each label though considering their similarity.

### 4.4 Main Result

We compare our proposed model ConsPrompt with these baselines, and the main result are depicted in Tab. 1. Comparing other baselines, both of the ConsPrompt methods achieve the state-of-the-art result. It reveals the contrastive learning module indeed increases the model’s discrimination on negative samples.

What is more, the effectiveness of ConsPrompt(-sim) is better than ConsPrompt(-label) in TREC, SNLI and SST tasks. It seems that we need to tell the PLMs more information about the semantic distinction instead of the label distinction, since the sampling samples from ConsPrompt(-sim) use a similarity-first viewpoint on generating negative set from SBERT. In addition, the ConsPrompt model showed greater robustness than other traditional baselines. In detail, the variances on N different groups of K-shot datasets are generally lower than other baselines.

### 5 Analysis

Related studies have shown different settings of prompt-learning method and contrastive learning can make a big different. In this section, we explore following three setting to certificate the effectiveness and generalization capability.

#### 5.1 Different Ratio of Contrastive Learning Module.

In order to explore the effectiveness of the contrastive learning module, we choose SNLI task and set different ratio of $t$ to control the loss of contrastive module. In detail, we choose ConsPrompt(-sim) as the baseline and change the $t$ on 0.1, 0.5, 1.0 and 20. In order to get a more persuasive result, we first fix the max_step to 3000, and then conduct each t-setting experiment on five groups of k-shot dataset with different batch sizes $bs=2,4,8$ and different learning rate $lr=1e^{-5},2e^{-5}$ and $5e^{-5}$. What is more, only the best results of each k-shot group are
Table 1: Main result of The ConsPrompt. The results of all experiments are evaluated by selecting the mean and variance of accuracy on five different segmented training datasets and same testing dataset. The specific experiment setting is depicted in appendix.

| Baselines                              | TREC (acc) | SNIL (acc) | QNLI (acc) | SST-5 (acc) | SST-2 (acc) |
|----------------------------------------|------------|------------|------------|-------------|-------------|
| Majority                               | 18.8       | 33.8       | 49.5       | 50.9        | 23.1        |
| prompt-based zero-shot learning        | 32.0       | 49.5       | 50.8       | 35.0        | 83.6        |
| GPT3-in-context-learning               | 26.2(2.4)  | 47.1(0.6)  | 53.8(0.4)  | 30.6(0.9)   | 84.8        |
| fine-tuning                            | 26.2(2.4)  | 48.4(4.8)  | 60.2(6.5)  | 43.9(2.0)   | 81.4(3.8)   |
| LMBFF                                  | 84.8(5.1)  | 77.1(3.9)  | 64.5(4.2)  | 46.1(1.3)   | 92.1(1.1)   |
| ConsPrompt(-sim)                       | **87.5 (2.2)** | **77.3 (3.6)** | **72.0 (3.0)** | **47.2 (3.1)** | **95.0 (2.8)** |
| ConsPrompt(-label)                     | 86.8 (2.8) | 76.2 (3.8) | 71.9 (2.7) | **47.5 (2.4)** | 93.1 (1.4) |

Table 2: The comparative experiment using ratio of comparative learning module on SNLI task. The ‘t’ value determine the loss influence to the final prediction.

| t   | Average (acc.) | Variance (+std) | Median (acc.) |
|-----|----------------|-----------------|---------------|
| 0.1 | 75.5           | 2.7             | 76.0          |
| 0.5 | 75.7           | 2.4             | 76.0          |
| 1.0 | 75.5           | 3.2             | 76.1          |
| 20  | **77.3**       | 3.6             | **78.9**      |
| Ensemble | 77.0 | 2.9 | 77.7 |

Table 2: The comparative experiment using ratio of comparative learning module on SNLI task. The ‘t’ value determine the loss influence to the final prediction.

selected to represent the effectiveness of different t-value settings.

As the result shown in Tab. 2, with the increase of value $t$ in comparative learning module, the proposed ConsPrompt is able to receive more gain from the contrastive learning module. However, it does not make any big difference while setting the lower t values, while higher t value can empower the original prompt encoding network. In addition, we combine all the results of different t-value setting. And these results reveals the necessity of setting high t value and the effectiveness of comparative learning module are further proved.

5.2 The K-shot Robustness of ConsPrompt.

Since our experiments and proposed model are developing on prompt learning method and aiming for few-shot learning question, we should also consider the influence of different size of K-shot settings. Different K values mean distinct sampling numbers of each label for constructing the training set. Ideally, with the increase of the training corpus, the fine-tuning process is able to benefit from more task-related knowledge. Thus, we change different K value and set the value K to 8,16,32,64,128,160 on SST-5 tasks and follow all the settings in the section 5.1. We also fix the ratio t controlling the loss of contrastive learning module to 1.0, which means the ConsPrompt would regard the gain from prompt encoding network and contrastive learning module are consistent. Also, we explore the relation between the sampling strategies of contrastive learning module and k value setting.

As the result shown in Tab. 3, it reveals:

- The higher K-shot setting can bring more effectiveness on the accuracy and robustness.
- The ConsPrompt (Sim-based) are on smaller K(k=8) and higher K(K=32, 64,128,160) experiments, while label-based ConsPrompt is effective in 16-shot settings. It reveals the similarity-based sampling strategy is more efficient than label-based strategy.
- With the increase of K, the benefit from the training corpus is decreasing, since there is lesser gain between the high K-shot setting. It certificates both the promt encoding network and contrastive learning module are more suggestive for lower K-shot experiments.

6 Related work

The effectiveness of two-step PLMs mostly depend on its mutual motivation between huge open-domain datasets and small task-oriented data. On the one hand, huge pre-training corpus offer a data foundation and interface for recording and searching external knowledge by the memory parameter. On the other hand, task-oriented data could retain or restrain adjusted parameters to motivate local similar knowledge. Nowadays, a consensus
Table 3: The comparative experiment using Different k-shot setting. The K-shot means we select how many sample of each labels to constitute the supporting set.

| K-shot | Sim-based Average (+std) | Medianc | Label-based Acc (+std) | Medianc |
|--------|--------------------------|---------|------------------------|---------|
| 8      | 47.1 (1.6)               | 47.3    | 44.5 (2.1)             | 44.4    |
| 16     | 47.2 (3.1)               | 46.9    | 47.5 (2.4)             | 46.5    |
| 32     | 48.8 (1.6)               | 48.9    | 48.3 (1.3)             | 48.6    |
| 64     | 51.1 (1.0)               | 50.9    | 50.8 (0.9)             | 50.6    |
| 128    | 51.8 (2.0)               | 52.4    | 51.5 (1.1)             | 51.4    |
| 160    | 51.8 (1.3)               | 51.9    | 51.7 (0.8)             | 51.8    |

is PLMs could learn the language structure and knowledge existing in large-scale corpus (Petroni et al., 2019). At the same time, what knowledge PLMs have learned also attract many attentions (Jiang et al., 2020). Lin et al. found that BERT encodes the location information about the word markers well at its lower level, but switches to the hierarchy-oriented encoding at the higher level (Lin et al., 2019). Jiang et al. propose mining and paraphrase based approaches to automatically generate high-quality and diverse prompts to estimate the knowledge contained in LM. While learning language knowledge, these PLMs models may also store relational knowledge that exists in training data. In addition, the performance of PLMs was largely due to biased hints about which data sets had been fitted, and the prediction is improved mainly through entity guidance and golden answer leakage (Cao et al., 2021). Therefore, with suitable prompting strategies, PLMs are able to seem as a searching database (Radford et al., 2019).

7 Conclusion

Since the prompt-based fine-tuning network use some representative tokens to construct a instructive knowledge probe for PLMs, it could not consider the distinction of different samples’ ability on prompting ability. With the combination of Prompt-based fine-tuning method and contrastive learning method, we put forward a more generative and robust prompt tuning network ConsPrompt to motivate the PLMs with more humanism and robust way. In detail, the ConsPrompt is combined by a prompting encoding network, Contrastive sampling module and contrastive scoring module. Also, two sampling strategies of contrastive learning are also proposed to realize the model’s attentions on construing the support set of positive and negative. We explore the effectiveness of ConsPrompt on few-shot learning and five representative tasks, which shows its state-of-the-art performance on empowering the prompt-based fine-tuning methods. What is more, the robustness and effectiveness of ConsPrompt are also certificate in different k-shot setting.

However, it still exists some limitations in our proposed model: (a) The prompt encoding network still use the manual template and label mapping, while more strategies should be concerned on auto prompt design and soft prompt tuning. (b) As the result shown in k-shot experiments, the contrastive learning module could not give more knowledge gain with the increase of the sampling value K, while it is also the limitation of prompt-based fine-tuning networks.

References

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 632–642.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. ArXiv, abs/2005.14165.

Boxi Cao, Hongyu Lin, Xianpei Han, Le Sun, Lingyong Yan, Meng Liao, Tong Xue, and Jin Xu. 2021. Knowledgeable or educated guess? revisiting language models as knowledge bases. In Proceedings of the 59th Annual Meeting of the Association
for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 1860–1874, Online. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In International conference on machine learning, pages 1597–1607. PMLR.

Xinlei Chen and Kaiming He. 2021. Exploring simple siamese representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 15750–15758.

Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Hai-Tao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. 2021. Prompt-learning for fine-grained entity typing. arXiv preprint arXiv:2108.10604.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. arXiv preprint arXiv:2012.15723.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 6894–6910.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. Prt: Prompt tuning with rules for text classification. arXiv preprint arXiv:2105.11259.

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

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 4582–4597, Online. Association for Computational Linguistics.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019. Open sesame: Getting inside bert’s linguistic knowledge.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too. arXiv preprint arXiv:2103.10385.

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, abs/1907.11692.

Andrea Madotto, Zhaojiangen Lin, Genta Indra Winata, and Pascale Fung. 2021. Few-shot bot: Prompt-based learning for dialogue systems. CoRR, abs/2110.08118.

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

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

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 2383–2392.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 3982–3992.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 5418–5426, Online. Association for Computational Linguistics.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. In The 28th International Conference on Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, pages 255–269, Online. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. arXiv preprint arXiv:2010.15980.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment
Ellen M Voorhees and Dawn M Tice. 2000. Building a question answering test collection. In Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, pages 200–207.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pages 353–355.

Haobo Wang, Ruixuan Xiao, Sharon Li, Lei Feng, Gang Niu, Gang Chen, and Junbo Zhao. 2021. Contrastive label disambiguation for partial label learning. In International Conference on Learning Representations.

Albert Webson and Ellie Pavlick. 2021. Do prompt-based models really understand the meaning of their prompts? ArXiv, abs/2109.01247.

Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. 2018. Unsupervised feature learning via non-parametric instance-level discrimination. ArXiv, abs/1805.01978.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. Consert: A contrastive framework for self-supervised sentence representation transfer. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 5065–5075.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2020. Revisiting few-sample bert fine-tuning. In International Conference on Learning Representations.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [mask]: Learning vs. learning to recall.
A Data

For SST5 (Socher et al., 2013) and TREC (Voorhees and Tice, 2000), we use their official test sets. For SNLI (Bowman et al., 2015) and datasets from GLUE (Wang et al., 2018), including SST2 (Socher et al., 2013), MNLI (Williams et al., 2018), QNLI (Rajpurkar et al., 2016), we follow Zhang et al. (2020) and use their original development sets for testing. In sentiment classification, we choose two representative SST Tasks, sst-2 and sst-5. The goal of SST dataset is given a movie review to predict its emotion. The label of sst-2 task is constitute of positive and negative tags, while in sst-5 task are tags of ‘very positive’, ‘positive’, ’netural’, ‘negative’, and ‘very negative’.

TREC datasets are data sets for problem classification, consisting of open domain, fact-based questions that are divided into broad semantic categories. It has 5,452 training examples and 500 test examples.

The Stanford Natural Language Inference (SNLI) Corpus is a collection of English sentence pairs written by 570K people, manually tagged to balance classification, label meaning, contradiction, and neutrality. Its goal is to make it both a benchmark for evaluating textual presentation systems, especially including those induced by presentation learning methods, and a resource for developing NLP models of any kind.

The QNLI is a natural language reasoning dataset automatically derived from the Stanford Questions dataset (SQuAD). SQuAD consists of pairs of question paragraphs, in which a sentence (from Wikipedia) in the paragraph contains the answer to the corresponding question. The dataset is converted to sentence pair classification by forming a pair between each question and each sentence in the corresponding context, and filtering out pairs with low lexical overlap between question and context sentences. The task is to determine whether the context sentence contains the answer to the question.

All of our utilizing dataset statistics are depicted in Tab. A.1. To satisfy the few-shot learning in PLMs, following the setting in Gao’s work, we pick five different K-shot sub-datasets and each sub-dataset is constructed by 16*|Y| training pairs on each type of labels (Gao et al., 2020). For example, the SST-2 emotion classification task with two classes needs to construct five different training sets with the size of 32 and validation set with the same size of 32, while the testing size would use original size of the SST-2 task, without any other data setup. These sub-datasets are training and predicting singly and then combining the final prediction.

B Setting

The proposed model is combining with prompt encoding network, Contrastive Sampling Module, and Contrastive Scoring Module. In this section we will describe the setup of each module in detail.

B.1 Prompt Encoding Network

Since our proposed method is based on the prompt-based fine-tuning, the label mapping and prompt template are predefined. We are based on the principle of minimum improvement and follow the formal Gao et al. (2020) setting. Our final label mappings and templates used in Dep-prompt section are depicted in Tab. B.1. The template is composed of special tokens and variables (surrounded by ‘*’) and text (e.g., ‘It was’, where space is replaced by ‘_’). Special tokens and variables contain:

- ‘*cls*’, ‘*sep*’, ‘*sep+*’ and ‘*mask*’: Special tokens of CLS, SEP and MASK (different for different pre-trained models and tokenizers). ‘*sep+*’ means the contents before and after this token have different segment embeddings (only for BERT).
- ‘*sent_i*’: The i-th sentence.
- ‘*sent-_i*’: The i-th sentence, discarding the last character.
- ‘*sentl_i*’: The i-th sentence, lower-casing the first letter.
- ‘*sentl-_i*’: The i-th sentence, discarding the last character and lower-casing the first letter.
- ‘*+sent_i*’: The i-th sentence, adding an extra space at the beginning.
- ‘*+sentl_i*’: The i-th sentence, adding an extra space at the beginning and lower-casing the first letter.

For more detail, please referring (Gao et al., 2020).

We use RoBERTa-large as our pre-training language model developed in 2080ti GPU device.
| Category          | Task   | |Y| | L | #Train | #Test | Type                                      | Labels (Classification tasks) |
|-------------------|--------|-----|-----|-----|--------|------|-------------------------------------------|--------------------------------|
| Single Sentence   | SST-2  | 2   | 19  | 6,920 | 872    | Sentiment                    | positive, negative          |
|                   | SST-5  | 5   | 18  | 8,544 | 2,210  | sentiment                    | v. pos., positive, neutral, negative, v. neg. |
|                   | TREC   | 6   | 10  | 5,452 | 500    | Question cls.                | abbr., entity, description, human, loc., num |
| Sentence Pair     | QNLI   | 3   | 11-30 | 104,743 | 5,463  | Nature Language Interference | entailment, not entailmen    |
|                   | SNLI   | 2   | 14-8 | 549,367 | 9,842  | Nature Language Interference | entailment, neutral, contradiction |

Table 4: The datasets evaluated in our work. |Y|: classes Number for classification tasks. L: average words in input sentence(s). In our few-shot experiments, we also sample $D_{\text{train}}$ and $D_{\text{dev}}$ of K × |Y| examples from the original training set. We use the dev set with the same number of samples in train set, while the test sets are chosen from the official datasets without any change.

In the training process, we set the batch size to $bs=2,4,8$ and learning rate $lr=1e-5,2e-5,5e-5$. We still use the average accuracy and variance value to evaluate the overall performance in different settings.

### B.2 Contrastive Sampling Module

In this module, we try to explore how to construct a supporting set to generate the positive and negative samples for each input. In the training process, the input is the prompting samples of each batch size that have been manual transformed. For each prompting input, we construct the primary supporting set from 16*|Y| training pairs except current input. Then, we set the filtering ratio as 0.5 to decrease the number of supporting set, which means only half of the primary supporting set will remain. Subsequently, we would calculate the similarity between current input and all the samples in filtering supporting by SBET encoder (Reimers and Gurevych, 2019) and Sentence Transformers Toolkit. Finally, a similarity list of each query example (eg. the prompting input) would be formulated based the value of their cosine similarity.

### B.3 Contrastive Scoring Module

In the selection of negative samples and positive samples, we must considering two setting, the number of comparative samples and the sampling strategy. In the number of comparative samples, we introduce first parameter cr_nums to control comparative number of negative samples and positive samples. If we set cr_nums = 1, it means only one positive sample and cr_nums * |Y| negative samples would be selected. What is more, the user should select which kind of sampling strategies to construct contrastive samples. In similarity-based strategy, the negative samples and positive samples are all selected from the similarity order of current input. In label-based strategy, the negative sample should be constructed by the least similar cr_nums samples of different labels. In the proposed paper, we set cr_nums=1. After selecting the positive samples and negative samples, we use their prompting representation generated by prompting encoding network and the SimCLR contrastive scoring method (Chen et al., 2020) to calculate final contrastive loss.

Other hyper-parameters settings are listed in B.2.
Table B.1: The label mapping and template in the Dep-prompt. |Y|: classes Number for classification tasks. L: average words in input sentence(s). In our few-shot experiments, we also sample $D_{train}$ and $D_{dev}$ of $K \times |Y|$ examples from the original training set.

| Task   | Label Mapping                                      | Template                                                                 |
|--------|-----------------------------------------------------|--------------------------------------------------------------------------|
| SST-2  | {‘0’: ‘good’, ‘1’: ‘bad’}                           | *cls*+*sent_0*+_It_was*mask_*+*sep*+                                    |
| SST-5  | {‘contradiction’: ‘No’, ‘entailment’: ‘Yes’, ‘neutral’: ‘Maybe’} | *cls*+*sent_0*+_This_movie_was*mask_*+*sep*+                              |
| TREC   | {0: ‘Description’, 1: ‘Entity’, 2: ‘Expression’, 3: ‘Human’, 4: ‘Location’, 5: ‘Number’} | *cls*+*mask*+*+*sent_0*+*sep*+                                          |
| QNLI   | {‘not_entailment’: ‘No’, ‘entailment’: ‘Yes’}       | *cls*+*sent_0*+*mask*+*+*sent_1*+*sep*+                                 |
| SNLI   | {‘contradiction’: ‘No’, ‘entailment’: ‘Yes’, ‘neutral’: ‘Maybe’} | *cls*+*sent_0*+*mask*+*+*sent_1*+*sep*+                                 |

Table B.2: The hyper-parameters setting

| Comparative_filter_rate | Eval_steps | Fix_layers | Max_seq_length | Num_k   |
|-------------------------|------------|------------|----------------|---------|
| 0.5                     | 100        | 0          | 128            | 16      |
| Demo_filter_model       | First_sent_limit | Other_sent_limit | Max_steps | lr       |
| sbert-roberta-large      | 110        | 20         | 3000           | 1e-5, 2e-5, 5e-5 |