PromDA: Prompt-based Data Augmentation for Low-Resource NLU Tasks

Yufei Wang[1]*, **Can Xu**[2], **Qingfeng Sun**[2], **Huang Hu**[2], **Chongyang Tao**[2]
Xiubo Geng[2] and **Daxin Jiang**[2]†
Macquarie University, Sydney, Australia[1]
Microsoft Corporation, Beijing, China[2]
yufei.wang@students.mq.edu.au
{caxu,qins,huahu,chongyang.tao,xigeng,djiang}@microsoft.com

Abstract

This paper focuses on the *Data Augmentation* for low-resource Natural Language Understanding (NLU) tasks. We propose **Prompt**-based **D**ata **A**ugmentation model (**PromDA**) which only trains small-scale *Soft Prompt* (i.e., a set of trainable vectors) in the frozen Pre-trained Language Models (PLMs). This avoids human effort in collecting *unlabeled in-domain data* and maintains the quality of generated synthetic data. In addition, **PromDA** generates synthetic data via two different views and filters out the low-quality data using NLU models. Experiments on four benchmarks show that synthetic data produced by **PromDA** successfully boost up the performance of NLU models which consistently outperform several competitive baseline models, including a state-of-the-art semi-supervised model using *unlabeled in-domain data*. The synthetic data from **PromDA** are also complementary with *unlabeled in-domain data*. The NLU models can be further improved when they are combined for training.

1 Introduction

Deep neural networks often require large-scale high-quality labeled training data to achieve state-of-the-art performance (Bowman et al., 2015). However, constructing labeled data could be challenging in many scenarios (Feng et al., 2021). In this paper, we study the low-resource Natural Language Understanding (NLU) tasks, including sentence classification and sequence labelling tasks, where only small labeled data is available. Previous works often produce extra "labeled data" for the NLU models to learn. Wang et al. (2021a) deploys the *self-training* framework to produce *pseudo labelled training data* from *unlabeled in-domain data* which could be expensive to obtain. Xu et al. (2021) has shown that extracting domain-specific unlabeled data from the general corpus is not trivial. Wei and Zou (2019); Dai and Adel (2020) expand the original small training data using automatic heuristic rules, such as randomly synonyms replacement, which effectively creates new training instances. However, these processes may distort the text, making the generated syntactic data grammatically and semantically incorrect.

To solve the above dilemma, many existing works (Ding et al., 2020; Yang et al., 2020; Anaby-Tavor et al., 2020) resort to applying Language Models (LMs) or Pre-trained Language Models (PLMs) for data augmentation in a low-resource setting. Given the labeled data, one can directly fine-tune PLMs to generate new synthetic data without additional human effort. However, we argue that, in the low-resource NLU tasks, directly fine-tuning all parameters of PLMs with small training data (especially when there are less than 100 samples) could result in over-fitting and PLMs simply *memorizes* the training instances. As a result, the generated synthetic data could be very similar to the original training instances and cannot provide new training signals to the NLU models. Recently, several works (Lester et al., 2021; Li and Liang, 2021) propose prompt tuning, which only back-propagates the error to *Soft Prompts* (i.e., a sequence of continuous vectors prepended to the input of PLMs) instead of the entire model. They show that prompt tuning is sufficient to be competitive with full model tuning while significantly reducing the amount of parameters to be tuned. Thus, the prompt tuning is quite suitable to tackle the above over-fitting issue in low-resource generative fine-tuning, which spawns more novel samples relative to the small labeled data under the premise of ensuring generation quality.

Motivated by this, we propose **Prompt**-based **D**ata **A**ugmentation model (**PromDA**). Specifically, we freeze the entire pre-trained model and only

*Work done during the internship at Microsoft STCA.
†Corresponding author

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 4242 - 4255

allow tuning the additional soft prompts during fine-tuning on the small labeled training data. In addition, we have observed that the initialization of soft prompts has a significant impact on fine-tuning, especially when the low-resource situation reaches an extreme extent. To better initialize the prompt parameters for the data augmentation tasks, we propose *task-agnostic Synonym Keyword to Sentence* pre-training task to directly pre-train the prompt parameters of PLMs on their pre-training corpora. This task simulates the process of generating entire training sample from partial fragment information (e.g., keywords). Similar to previous works (Ding et al., 2020; Yang et al., 2020; Anaby-Tavor et al., 2020), we could fine-tune PLMs to produce complete synthetic data conditioned on the output tags. We refer this as *Output View Generation*. To boost the diversity of the generated samples, we introduce another fine-tuning generative task named *Input View Generation*, which takes the extracted keywords from the sample as the input and the sample as the output. As NLG models trained from small training data still has a certain chance to generate low-quality samples, we leverage the NLU Consistency Filtering (Anaby-Tavor et al., 2020) to filter the generated samples.

We conduct experiments on four benchmarks: sequence labelling task CoNLL03 (Tjong Kim Sang and De Meulder, 2003) and Wikiann (Pan et al., 2017), sentence classification task SST-2 (Socher et al., 2013) and RT (Pang and Lee, 2005). Experiment results show that NLU models trained on synthetic data from **PromDA** consistently outperform several competitive baseline models, including a state-of-the-art semi-supervised NLU models MetaST (Wang et al., 2021a) on Sequence Labelling task. In addition, we find that the synthetic data from **PromDA** are also complementary with the *unlabeled in-domain data*. The performance of NLU models can be further improved when both of them are combined. Finally, we conduct diversity analysis and case study to further confirm the synthetic data quality from **PromDA**. Our source code is released at `https://github.com/GaryYufei/PromDA`.

2 Related Work

Prompt Learning The concept of prompt-based learning starts from the GPT3 model (Brown et al., 2020). Previous works design different prompts to query language models to extract knowledge triples (Petroni et al., 2019) or classify sentences into pre-defined categories (Schick and Schütze, 2021) in the few-shot setting. They construct various discrete prompts manually for these tasks. To reduce the human effort in this selection process, (Gao et al., 2021) proposes to expand prompts using pre-trained language models. However, the selection of discrete prompts is still an independent process and difficult to be optimized together with the downstream tasks in an end-to-end manner. Ben-David et al. (2021) proposes a complicated two-stage model to connect between prompt generation and downstream tasks. To solve this issue, (Lester et al., 2021; Li and Liang, 2021) propose to use soft prompts, which are sets of trainable vectors, in the frozen pre-trained language models. Unlike the hard prompts, these vectors do not correspond to any real words. It allows the optimization with the downstream tasks in an end-to-end manner. As shown in Li and Liang (2021), PLMs with *Soft Prompts* can often perform better in the low-resource setting.

Generative Data Augmentation Hou et al. (2018) generates diverse utterances to improve dialogue understanding models. Xia et al. (2019) uses a bilingual dictionary and an unsupervised machine translation model to expand low-resource machine translation training data. Wu et al. (2019); Kumar et al. (2020) make use of the masking mechanism in many PLM pre-training objective functions (e.g., BERT (Devlin et al., 2019), BART (Lewis et al., 2020)) and produce new synthetic data by masking randomly chosen words in the original training instances. Ding et al. (2020); Yang et al. (2020); Anaby-Tavor et al. (2020) apply LMs and PLMs to learn directly to generate new synthetic data for NLU tasks (i.e., sequence labeling and commonsense inference tasks after trained (fine-tuned) on the relatively large training data. These works often directly apply *off-the-shelf* LMs or PLMs to generate synthetic data. Wang et al. (2021b) proposes to use unlabelled data as *hard prompt* to generate synthetic data without any training, limiting its application in complicated NLP tasks. To best of our knowledge, **PromDA** is the first PLMs with *Soft Prompt* that are especially designed for the data augmentation task.

3 Prompt-based Data Augmentation

This section first formulates the data augmentation for low-resource NLU task. We then intro-

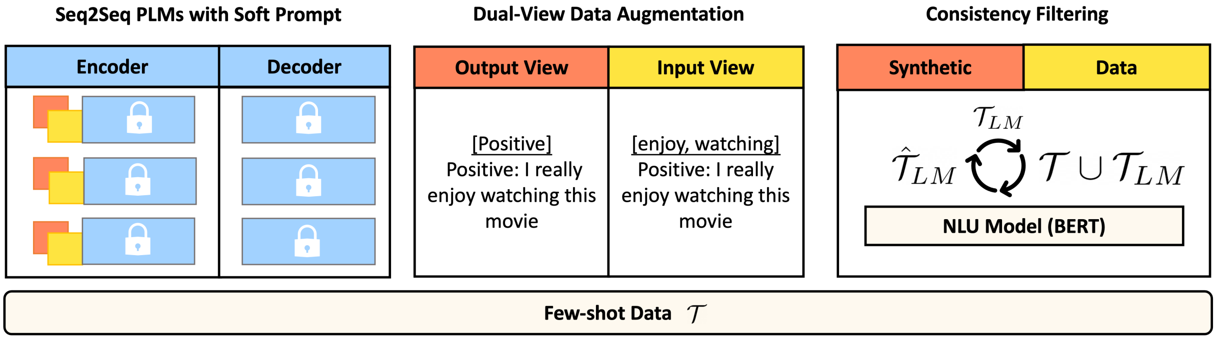

Figure 1: The Overall of **PromDA**. *Soft Prompt* prepend a sequence of trainable vector at each layer of the frozen PLMs. The white locker represents frozen parameters. We have separated sets of *Soft Prompt* to support *Daul-View Data Augmentation* where the *Output View* conditions on the output tags and *Input View* conditions on the keywords in the input sentences. Finally, we use the NLU models to iteratively filter out low-quality synthetic data and use the remaining synthetic data, combined with $\mathcal{T}$, to train stronger NLU models.

duce the three important components in Our proposed Prompt-based Data Augmentation method (**PromDA**), including *i)* prompt-based learning in pre-trained language models; *ii)* dual synthetic data generation view and *iii) Consistency Filtering*. Figure 1 shows the overall of **PromDA**.

3.1 Data Augmentation For NLU tasks

In the low-resource NLU tasks, only a set of labeled training data $\mathcal{T} = \{(x_1, y_1), \cdots, (x_n, y_n)\}$ is available where n is relatively small (i.e., less than a hundred). *Data Augmentation* generates synthetic labeled training data $\mathcal{T}_{LM} = \{(\hat{x}_1, \hat{y}_1), \cdots, (\hat{x}_n, \hat{y}_n)\}$ from the original labeled training data T using language models. The goal is that the NLU models trained using $\mathcal{T} \cup \mathcal{T}_{LM}$ outperform the NLU models only trained using $\mathcal{T}$.

3.2 Prompt-based learning

Fine-tuning is the prevalent way to adapt PLMs to specific down-stream tasks (Devlin et al., 2019). However, for low-resource data augmentation, we expect the generated synthetic training data $\mathcal{T}_{LM}$ to be different from $\mathcal{T}$ and to provide new information for NLU models to learn. A fine-tuned PLM, which is biased towards a small number of training instances, may not be an optimal solution.

Prompt-based learning, starting from the zero-shot instructions in GPT3 (Brown et al., 2020), keeps the whole PLMs parameters frozen and only prepends the discrete natural language task instructions (e.g. "translate to English") before the task inputs. Freezing the PLMs parameters might help generalization during training. However, finding suitable discrete task introductions cannot be easily optimized in an end-to-end fashion and requires extra human effort. In this paper, inspired by the recent work (Lester et al., 2021; Li and Liang, 2021), we replace the task introductions with *Soft Prompt* (i.e., a sequence of continuous and trainable vectors). During training, we only update the parameters of this *Soft Prompt* and fix all PLMs parameters. We mainly focus on generating synthetic training data using seq2seq Transformer-based PLMs.

Unlike Lester et al. (2021) which only prepends *Soft Prompt* at the input layer, inspired by Adaptor (Houlsby et al., 2019) which adds trainable Multi-layer Perceptron (MLP) at each transformer layer, we prepend a sequence of trainable vectors at each transformer layer. We denote $P^j = \{\boldsymbol{p}_1^j, \cdots, \boldsymbol{p}_k^j\}$ as the *Soft Prompt* at the j^{th} layer. The i^{th} hidden states at the j^{th} layer $\boldsymbol{h}_i^j$ in the Transformer model is defined as follows:

$$\boldsymbol{h}_i^j = \begin{cases} \boldsymbol{p}_i^j & i \le k \\ \boldsymbol{w}_i & i > k \wedge j = 0 \\ Trans(\boldsymbol{h}^{j-1})_i & \text{Otherwise} \end{cases} \quad (1)$$

where $Trans()$ is the forward function the Transformer layer and w_i is the fixed word embedding vector at the input layer. Compared to (Lester et al., 2021), this allows gradients to be updated at each layer and better complete the learning tasks.

3.3 Pre-training for Prompt Initialization

The parameter initialization of the *Soft Prompt* P has a significant impact on the generated synthetic data quality, especially in the low-resource *Data Augmentation* task. Lester et al. (2021) proposes to further pre-train the full PLMs parameters, without the prompt parameters, to enhance

Algorithm 1 Dual-View Data Augmentation: Given few-shot labeled dataset $\mathcal{T}$, the number of iteration N; return a trained NLU model M_{NLU}.

1: **procedure** DUALVIEWDA($\mathcal{D}, N$)
2: $\quad M_{LM} \leftarrow \text{TRAIN}(LM, \mathcal{T})$
3: $\quad \mathcal{T}_I^1 \leftarrow \text{GEN}(M_{LM}, \mathcal{T}, \text{I})$ $\triangleright$ Input
4: $\quad \mathcal{T}_O^1 \leftarrow \text{GEN}(M_{LM}, \mathcal{T}, \text{O})$ $\triangleright$ Output
5: $\quad \mathcal{T}_I^2 \leftarrow \text{GEN}(M_{LM}, \mathcal{T}_O^1, \text{I})$
6: $\quad \mathcal{T}_O^2 \leftarrow \text{GEN}(M_{LM}, \mathcal{T}_I^1, \text{O})$
7: $\quad \hat{\mathcal{T}}_{LM} \leftarrow \mathcal{T}_I^1 \cup \mathcal{T}_I^2 \cup \mathcal{T}_O^1 \cup \mathcal{T}_O^2$
8: $\quad M_{NLU}^0 \leftarrow \text{TRAIN}(NLU, \mathcal{T})$
9: $\quad$ **for** $r \in 1, \ldots, N$ **do**
10: $\quad\quad \mathcal{T}_{LM}^r \leftarrow \text{CONSIST}(M_{NLU}^{r-1}, \hat{\mathcal{T}}_{LM})$
11: $\quad\quad \mathcal{T}^r \leftarrow \mathcal{T}_{LM}^r \cup \mathcal{T}$
12: $\quad\quad M_{NLU}^r \leftarrow \text{TRAIN}(NLU, \mathcal{T}^r)$
13: $\quad M_{NLU} \leftarrow M_{NLU}^N$
14: $\quad$ **return** M_{NLU}

the prompt capability. However, this strategy (i.e., full PLM pre-training) introduces significant computation overhead and does not provide any insight about prompt initialization. Instead, we propose to directly pre-train the parameters of the *Soft Prompt* with the frozen PLMs. Given that data augmentation produces full syntactic data from partial information (e.g., output tags and keywords), we propose *Synonym Keywords to Sentence* pre-training task. Given a chunk of text, we extract keywords using unsupervised keyword extraction algorithm *Rake* (Rose et al., 2010). We randomly replace some of these extracted keywords with their synonyms, via WordNet (Fellbaum, 2010). Given these synonym keywords, the *Soft Prompt* is pre-trained to reconstruct the original text chunks. When applying this *Soft Prompt* for data augmentation, we only need to fine-tune the *Soft Prompt* with the few-shot labeled data $\mathcal{T}$. This pre-training process only happens once. We only use the task-agnostic general-purpose pre-training corpus.

3.4 Dual-View Data Augmentation

Previous works often restrict the encoder inputs to fixed keywords or limited labels, such as unconditional generation (Yang et al., 2020) and label-conditional generation (Anaby-Tavor et al., 2020). The relatively small input space could result in similar outputs. To enrich the input space, we propose *Dual-View* Data Augmentation that generates synthetic data from *Input View*, which is conditioned on the keywords in the input sentences, and *Output View*, which is conditioned on the output labels. Table 1 shows examples of these two views. As illustrated in Algorithm 1 (line 2 to 7), after fine-tuning the *Soft Prompt* in PLMs, **PromDA** first generates $\mathcal{T}_I^1$ and $\mathcal{T}_O^1$ from *Input View* and *Output View*, respectively. **PromDA** then extracts output labels from $\mathcal{T}_I^1$ and keywords from $\mathcal{T}_O^1$. These new output labels and keywords are fed into the *Output View* and *Input View* in M_{LM} to generate another two sets of new synthetic data $\mathcal{T}_O^2$ and $\mathcal{T}_I^2$. In this way, the resulting output text should maintain a higher level of diversity and include more novel words/phrases/knowledge.

Dual View via Prompt Ensemble Ensembles of different neural models can often achieve better performance (Hansen and Salamon, 1990). Prompt-based learning provides an efficient way to model ensemble. By training K sets of *Soft Prompt*, we create K models sharing the same frozen PLMs. In our case, after prompt pre-training, we treat *Input View* and *Output View* as two independent models and use the *Soft Prompt* parameters P to initialize the parameters of P_{input} and P_{output}. During the **PromDA** fine-tuning, the gradients from the *Input View* and *Output View* training instances are only applied to parameters P_{input} and P_{output}, respectively. This prompt ensemble allows the two views to generate synthetic data independently. As a result, the final output should include diverse real-world knowledge.

3.5 Consistency Filtering

As **PromDA** is trained from small training data, it is possible to generate low-quality samples. We leverage the NLU Consistency Filtering (Anaby-Tavor et al., 2020) to filter the generated samples. Specifically, given synthetic data with generated labels produced by **PromDA**, we use the NLU models to label these data again and only keep the instances with *consistent* outputs from **PromDA** and the NLU models. As shown in Algorithm 1 (line 8 to 12), M_{NLU}^r filters the raw synthetic data $\hat{\mathcal{T}}_{LM}$ into $\mathcal{T}_{LM}$ which are combined with few-shot labeled data $\mathcal{T}$ to train new NLU models M_{NLU}^{r+1}. As M_{NLU}^{r+1} is generally better than M_{NLU}^r, we iterate this process N times to obtain stronger NLU models.

4 Experiments

This section first introduces experimental setup in Sec 4.1, and then presents main experiment results in Sec 4.2. Sec 4.3 conducts ablation study. In

	Sequence Labelling
GT:	**[*Org* All Fishermen 's Association]** secretary **[*Per* N.J. Bose]** said the strike would continue indefinitely.
IV:	All Fishermen 's Association and N.J. Bose and strike and indefinitely
OV:	Organization and Person
	Sentence Classification
GT:	The story has its redundancies, and the young actors, not very experienced, are sometimes inexpressive. **Negative**
IV:	redundancies and young actors and experienced and inexpressive
OV:	Negative

Table 1: Examples of *Input View* (IV) and *Output View* (OV) in both tasks.

Sec 4.4, We compare **PromDA** and unlabeled data, present diversity analysis and a case study.

4.1 Experimental Setup

We conduct experiments on Sentence Classification tasks SST2 (Socher et al., 2013) and RT (Pang and Lee, 2005) and Sequence Labeling tasks CoNLL03 (Tjong Kim Sang and De Meulder, 2003) and Wikiann (Pan et al., 2017). For each benchmark, we conduct shot-10, 20, 50, 100 experiment. In Shot-K, we sample K labeled instances for each output tag from the full training data. We repeatedly experiments 5 times and report the averaged micro-F1. The **Baseline** model is *BERT-BASE* model only trained with few-shot training data $\mathcal{T}$. Given the newly generated synthetic data $\mathcal{T}_{LM}$, we train the same *BERT-BASE* model using the same set of hyper-parameters. In sequence labeling tasks, we use rule-based data augmentation method **SDANER** (Dai and Adel, 2020) and **MetaST** (Wang et al., 2021a), a state-of-the-art self-training method, requiring additional *unlabeled in-domain data*. For sentence classification tasks, rule-based **EDA** (Wei and Zou, 2019), Back-Translation (**BackT.**) and bert-based **CBERT** methods are used. We adapt **LAMBADA** (Anaby-Tavor et al., 2020) as a PLM-based method for all tasks.

Implementation Details **PromDA** is built on the top of the T5-Large model (Raffel et al., 2020). **PromDA** requires Prompt Pre-training and fine-tuning with down-stream tasks. In both stages, we use Adafactor optimizer (Shazeer and Stern, 2018) with learning rate 1e-3 and weight decay 1e-5 to train the *Soft Prompt* parameters. For pre-training, we use the *realnewslike* split in the T5 pre-training corpus C4 as the input. The pre-training batch size is 72 and we pre-train **PromDA** for 100k steps. We split the *realnewslike* dataset into train and development split (i.e., 10000 pages). We will check the PPL on the development split every 5,000 steps. We save the model with lowest PPL. When fine-tuning on the few-shot data $\mathcal{T}$, we set the batch size 32 and we train **PromDA** for 1,000 steps. We only upgrade the fine-tuning step to 5,000 on the shot-50 and shot-100 for Wikiann and CoNLL03. More experiment setup see Section A in the Appendix.

4.2 Main Results

Sequence Labeling Tasks Table 2 summarizes the experiment results in shot-10 and shot-50. In both settings, the performance of NLU models trained with the synthetic data from **PromDA** are boosted up by a large margin (i.e., 4.8% and 7.5% for CoNLL03 and Wikiann, respectively). **PromDA** also outperforms rule-based **SDANER** and fully fine-tuned PLM **LAMBADA** methods. In general, PLM-based approaches produce better synthetic data than **SDANER** does. Surprisingly, the NLU models supported by **PromDA** achieve slightly better performance than **MetaST** which uses *unlabeled in-domain data*. This shows that **PromDA** could potentially reduce extra human effort in collecting *unlabeled in-domain data* for the low-resource NLU tasks. Figure 2 shows the performance in the shot-{10, 20, 50, 100} settings. The NLU models supported by **PromDA** consistently outperform other systems in all settings. Compared to Wikiann, the improvement margin in CoNLL03 is smaller. This could because the performance of CoNLL03 baseline is relatively high.

DataSet	C03		Wiki	
Shot	10	50	10	50
Baseline	72.7	82.9	50.8	65.4
SDANER♠	72.9	82.8	51.7	65.8
LAMBADA	75.0	83.7	<u>52.9</u>	<u>66.4</u>
MetaST♣	76.7	83.6	56.6	69.2
PromDA	<u>77.5</u>	<u>84.1</u>	<u>58.3</u>	<u>70.1</u>

Table 2: Experiment Results of the Sequence Labeling Tasks. ♣ results taken from (Wang et al., 2021a). ♠ we run Dai and Adel (2020)'s source code. C03 refers to CoNLL03 and Wiki refers to Wikiann. Underline are the significant results compared to the **Baseline** model (paired student's t-test, $p < 0.05$).

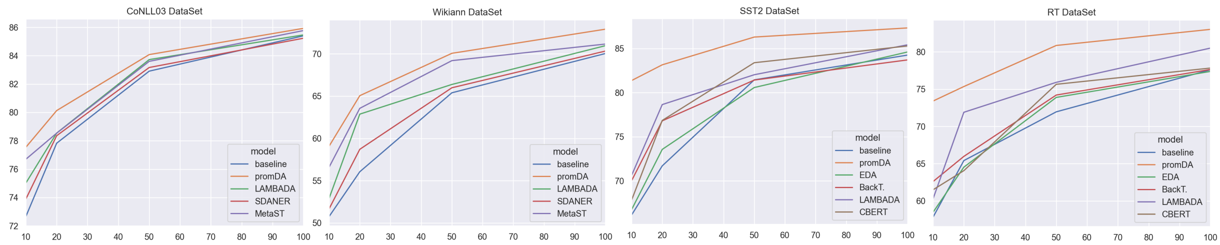

Figure 2: Experiment results under the Shot-{10, 20, 50, 100} settings.

Sentence Classification Tasks Table 3 shows the experiment results in shot-10 and shot-50. Similar to the results in the sequence labeling tasks, adding the synthetic data from **PromDA** significantly boosts up the performance of NLU models (more than 10% in both benchmarks in shot-10). **PromDA** also outperforms various competitive methods, including **BackT.**, **CBERT** and **LAMBADA**. Although **LAMBADA** has higher level of flexibility and generates synthetic data from output tags, it only performs similar to **CBERT**. This could be because of the over-fitting issues when fine-tuning with small training data. Prompt-empowered **PromDA** successfully avoids this issue and produce high-quality synthetic data to support the NLU model training. Figure 2 shows the performance in the shot-{10, 20, 50, 100} settings. NLU models supported by **PromDA** consistently outperform all other systems in all setups.

DataSet	SST2		RT	
Shot	10	50	10	50
Baseline	66.1	81.5	57.8	72.0
EDA♠	66.7	80.4	58.5	73.9
Back T.	70.0	81.4	62.6	74.2
CBERT♣	67.8	83.4	61.5	75.3
LAMBADA	70.6	82.0	60.3	75.9
PromDA	81.4	86.3	73.4	80.9

Table 3: Experiment Results of the Sentence Classification Tasks. ♠ we run Wei and Zou (2019)'s source code. ♣ we run Wu et al. (2019)'s source code. Underline are the significant results compared to the **Baseline** model (paired student's t-test, $p < 0.05$).

Discussion **LAMBADA** performs consistently worse than **PromDA** (e.g., more than 10% F1 score gap in the SST2 and RT experiment). This is because fully fine-tuned PLMs can easily *memorize* the limited labeled training data and produce similar synthetic data. In contrast, the prompt-based learning allows **PromDA** to maintain high generalization ability and provide new training signals to the NLU models. The results from **PromDA** are all statistical significant, compared to the **Baseline** model (paired student's t-test, $p < 0.05$).

4.3 Ablation Study

We conduct ablation study for the components *Prompt Pre-training*, *Dual-View Data Augmentation* and *Consistency Filtering* on the CoNLL03 and SST2 Benchmark under the shot-10 setting.

Prompt Pre-Training In **No PT**, we directly fine-tune two separated PLMs to learn the *Input View* and *Output View*. In **No PT Pre-Training**, we remove the Prompt Pre-training Task (*Synonym Keywords to Sentence*). In **Full Pre-Training**, we apply the Prompt Pre-training Task to fine-tune the whole PLMs parameters. Finally, in **LM Adaptation**: we replace **PromDA** with solution in Lester et al. (2021). As shown in Table 4, the fully fine-tuned PLMs (**No PT**) performs worse than our proposed **PromDA** method (4.6% F1 score lower), showing the positive contribution of *Soft Prompt* for low-resource NLU Data Augmentation. Further, removing PT Pre-training (**No PT Pre-Training**) or applying PT Pre-training to fine-tune all PLMs parameters (**Full Pre-Training**) also delegate the PT Pre-training performance by 3.1% and 6.0% F1 score, respectively, showing the importance of using PT Pre-training to learn a reasonable prompt initialization. Similarly, **LM Adaptation** also fine-tunes the whole PLMs and achieves similar performance as **Full Pre-Training**. It is recommended to directly train the prompt parameters.

Dual-View Data Augmentation Next, we show the effect of Dual-View Data Augmentation in **PromDA**. **Input Only** and **Output Only** only generate synthetic data via the *Input View* and *Output view*, respectively. These two *Single-View* models generate the same number of synthetic data as the **PromDA** does. As shown in Table 4, the synthetic data from these two *Single-View* models

DataSet	C03	SST2	Ave.
Few-shot NLU Baseline	72.7	66.1	69.4
PromDA	77.5	81.4	79.5
Ablation for PT Pre-Training			
No PT	75.2	74.5	74.9
No PT Pre-Training	74.0	78.2	76.1
Full Pre-Training	75.0	72.0	73.5
LM Adaptation	75.4	73.3	74.4
Ablation for Dual-View DA			
Output Only	75.6	81.0	78.0
Input Only	74.4	70.6	72.5
Single Prompt	76.7	79.5	78.1

Table 4: Ablation Study for Prompt Pre-Training and Dual-View Data Augmentation for CoNLL03 and SST2 Benchmark under shot-10 settings.

successfully boost up the NLU model performance. However, their corresponding NLU models perform worse than the ones supported by **PromDA**. This shows that synthetic data from different views provide meaningful and different training signals to the NLU models. Interestingly, NLU models trained on the *Output view* perform better than the ones trained on the *Input View*, indicating that output tags are more expressive signals to guide PLMs to generate high-quality synthetic data. Finally, instead of training two views on the separated prompt parameters, we train two views on the same prompt parameters in **Single Prompt**. The NLU models trained on **Single Prompt** synthetic data perform worse than the NLU models supported by **PromDA**, showing the importance of *Prompt Ensemble* for *Dual-View Data Augmentation*.

Setup	w/o Filtering	Iter-1	Iter-2	Iter-3
C03	72.0	76.7	77.6	77.5
SST2	69.2	77.5	79.7	81.4

Table 5: Ablation Study For Iteration-based NLU Consistency Filtering.

Consistency Filtering Finally, we examine the effect of *Consistency Filtering* in **PromDA**. In table 5, we show the NLU model performance without any filtering (**w/o Filtering**) and with k iteration (**Iter-1**, **Iter-2** and **Iter-3**). The filtering has an important effect on the NLU performance. Without removing low-quality synthetic data, the performance gap almost disappears. The iteration filtering also has a positive effect on the NLU performance. In particular, in the SST2 Benchmark, the NLU model performance increases ~4% F1 score after three iterations.

Dataset	C03	Wiki	SST2	RT	Δ
Baseline	72.7	50.8	66.1	57.8	-
w/ UID	76.2	55.2	70.2	59.7	+3.5
w/ UND	71.5	51.3	69.3	59.4	+1.0
w/ UGD	64.6	44.8	66.4	58.7	-3.2
PromDA	77.5	58.3	81.4	73.4	+10.8
w/ UID	80.0	61.7	83.0	73.9	+12.8

Table 6: Experiment Results for **PromDA** and *Unlabeled Data* under the shot-10 setting.

4.4 Discussion

PromDA with T5-Base We verify whether **PromDA** could work with different pre-trained language models. We replace the T5-Large model with the T5-base model. The new **PromDA** can also improve the few-shot baseline models by a large margin. On the SST2 shot-10 setup, the NLU model is improved from 66.1 to 76.3 F1 score, which also beats other models presented in Table 3.

PromDA in the high-resource setting To show the advantages of **PromDA** in the high-resource setting, We replace the few-shot training data with the full training data. We find that **PromDA** can still improve the baseline model performance. In SST2, after adding syntactic data, the NLU performance is improved from 90.8 to 92.3 F1 score.

Improvement Margin Difference As shown in Table 2 and 3, the improvement margins in the sentence classification tasks (i.e., more than 15% F1 score) are generally larger than the ones in the sequence labelling tasks (i.e., less than 10% F1 score). This could because *i)* the sequence labelling task is a more fine-grained and knowledge-intensive task than the sentence classification task; *ii)* the synthetic data for the sequence labelling tasks includes entity type and boundary, which is more challenging for PLMs to generate, in particular for low-resource settings, compared to the sentence classification task.

PromDA and Unlabeled Data The above experiments are based on the assumption that no *unlabeled data* is available. In this section, we explore the connection between **PromDA** and *unlabeled data*. To incorporate *unlabeled data* into our NLU

Sequence Labeling	
GT:	It quoted an [*Org* **Interior Ministry**] statement as saying [*Per* **Shabir Ahmad Muhammad Jalil**] was executed in [*Loc* **Mecca**].
SDANER:	It , an [*Org* **Interior Ministry**] statement as. [*Per* **Mike Lloyd Muhammad Jalil**] was , in [*Loc* **Berlin**]
LAMBADA:	It quoted an [*Misc* **German**] newspaper as saying [*Per* **Klaus Kinkel**] was **killed in** [*Loc* **Berlin**] **plane crash.**
PromDA:	The [*Org* **AP**] quoted the head of [*Org* **the Back of New Zealand**] as saying **"**[*Per* **Noel Davis**] **said the bank would cut its ties with** [*Org* **Bancorp**] **and** [*Per* **Peter Johnson**] **would be the next to take over."**
GT:	[*Loc* **U.S.**] embassy officials in [*Loc* **Colombo**] were not immediately available to comment on the report.
SDANER:	[*Loc* **Mahala**] embassy jail in [*Loc* **China**] were not immediately available to v on the board.
LAMBADA:	[*Loc* **U.S.**] embassy officials in [*Loc* **Saudi Arabia**] were not immediately available to comment on the report.
PromDA:	The [*Loc* **United States**] Embassy in [*Loc* **Germany**] **has not commented on the latest developments**
GT:	[*Org* **All Fishermen 's Association**] secretary [*Per* **N.J. Bose**] said the strike would continue indefinitely and the fishermen would block road and rail traffic if their demands were not met.
LAMBADA:	[*Org* **All Fishermen's Association**] secretary [*Per* **N. Fairbrother**] said the strike would continue indefinitely and the fishermen would block road and rail traffic if their demands were not met.
PromDA:	[*Org* **FIA**] secretary [*Per* **David Davidson**] said the [*Org* **World Bank**] had offered them about 60 million euros.
GT:	[*Org* **Araneta Coliseum**], [*Loc* **Quezon City**]; [*Org* **University of Akron**], [*Loc* **Akron**]
PromDA:	[*Org* **Memorial Theatre**], [*Loc* **Houston, Texas**]; [*Org* **University of Louisville**], [*Loc* **Louisville**]
Sentence Classification	
GT1:	The story has its redundancies, and the young actors, not very experienced, are sometimes inexpressive. **Negative**
GT2:	Full frontal, which opens today nationwide, could almost be classified as a movie-industry satire, but it lacks the generous inclusiveness that is the genre 's definitive, if disingenuous, feature. **Negative**
LAMBADA:	The story has its redundancies, the bullying that are the genre's definitive, if disingenuous, episodes. **Negative**
PromDA:	Despite their experience, the young actors are sometimes inexpressive because of the redundancies in the story, **which may have caused them to feel rushed and out of place**. **Negative**
GT:	I could just feel the screenwriter at every moment "tap , tap , tap , tap , tapping away" on this screenplay. **Negative**
PromDA:	**Mr. Mcsweeney**, the screenwriter, has done a masterful job with **a dynamic plot, full of suspense, wit, and humor**. **Positive**
GT:	The saigon of 1952 is an uneasy mix of sensual delights and simmering violence , and the quiet american brings us right into the center of that world . **Positive**
LAMBADA:	many of the ladies in blue are very much in love with the saigon of 1952, and many of them are very much in love with the country. **Positive**
PromDA:	The **saigon of the '70s** is the antithesis of that, with a sardonic tone and well conceived plot that builds toward a great end. **Positive**

Table 7: Generated synthetic data from our proposed **PromDA** and other baseline methods. Text chunks in Red are duplicated with the few-shot training data. Text chunks in **Blue** are the novel words/phrases.

models, we apply the classic *self-training* framework (Scudder, 1965) to the NLU models. Specifically, for each unlabeled instance, we use the NLU models to label it and record the output tags and corresponding likelihood score. The low likelihood score means predictions with less confidence. We rank all unlabeled instances based on the likelihood score and remove instances at the bottom 20%. Table 6 shows the experiment result of four benchmarks under the shot-10 setting.

The Effect of Unlabeled Data Domain We design three settings: *Unlabeled In-domain Data* (**UID**), *Unlabeled Near-domain Data* (**UND**) and *Unlabeled General-domain Data* (**UGD**) where the unlabeled data come from *exactly same*, *similar* and *general-purpose* domains. We exchange the training data between CoNLL03 and Wikiann, and between SST2 and RT to simulate *similar* domains. We randomly sample sentences from PLM pre-training corpus to simulate the *general-purpose* domain. We note that unlabeled data domain has a great impact of the *self-training* performance. Even a slight domain shift (i.e., **UND**) delegates the NLU performance by 2.5%. The performance of NLU models trained with unlabeled data from general-purpose corpus are even 3.2% lower than the NLU baseline models only trained with few-shot labeled data $\mathcal{T}$. Both sequence labeling tasks and sentence classification tasks follow this trend, but sequence labeling tasks is more sensitive to the unlabeled data domain. Extra human effort is still required, for semi-supervised learning, to select suitable domains to collect unlabeled data.

Combining *Unlabeled In-domain Data* with `PromDA` We apply the above *self-training* algorithm to the final NLU models (**`PromDA`**) supported by **`PromDA`** with *unlabeled in-domain data*. The resulting NLU models are further improved, on average, by 2.0% (**w/ UID** in the last row). More sophisticated semi-supervised learning algorithms may introduce more improvement. This shows that *a)* synthetic data from **`PromDA`** and *unlabeled in-domain data* provide different information to the NLU models; *b)* **`PromDA`** successfully extracts the embedded knowledge in the PLMs and presents them in the generated synthetic data.

Diversity Analysis In Table 8, we show the diversity of the generated synthetic data from **`PromDA`** and other baseline models. We sample 10 new synthetic data from each training instance. We use **Novel Mention** (number of entity mentions or keywords not appearing in the training data) and Self-BLEU score (Zhu et al., 2018) to measure the diversity. In general, simple generative data augmentation approaches (i.e, **BackT.** and **CBERT**) can easily produce Novel Mentions, but their generated synthetic data lacks diversity (relatively low self-BLEU score). The prompt-based learning helps **`PromDA`** to produce the most diverse synthetic data with the most Novel Mentions in both benchmarks. Due to the over-fitting issues, **LAMBADA** produces synthetic data that are less or equal diverse than other baseline approaches. Interestingly, the NLU models trained on these synthetic data achieve the second best performance. This could because **LAMBADA** coherently generate the whole synthetic sentences, while others reply on the random and/or heuristic rules.

Synthetic Data Case Study Table 7 shows representative examples generated by our proposed **`PromDA`** and methods. In the Sequence Labelling example, the rule-based **SDANER** shuffles the original word order and creates low-quality text. The **LAMBADA** model generates a new synthetic instance by modifying three text spans in the original training instance (e.g., changing "statement" to "newspaper"). In contrast, Our **`PromDA`** method generates a completely new and reasonable event in a bank, as well as correct and novel geographical locations in the generated synthetic data. Similarly, in the sentence classification tasks, **LAMBADA** naively combines text chunks from two training instances in the second example. **`PromDA`** mentions some keywords in the training data, but adds more information into the output. In another example, **`PromDA`** comments on a screenwriter (not appearing in the training data) with a sequence of coherent words. Finally, **`PromDA`** successfully moves the topic from the film "The Saigon of 1952" to the Saigon in 70s. In summary, **`PromDA`** can extract the embedded real-world knowledge from the PLMs and introduces these knowledge into a relatively long sentence in a fluent way.

Model	NM↑	Self-B↓	F1↑
CoNLL03			
SDANER	141.4	0.770	72.9
LAMBADA	107.6	0.761	75.0
`PromDA`	**351**	**0.259**	**77.5**
SST2			
EDA	59.6	0.889	66.7
BackT.	101.8	0.826	70.0
CBERT	127	0.900	67.8
LAMBADA	51.8	0.926	70.6
`PromDA`	**276**	**0.578**	**81.4**

Table 8: Diversity Analysis for the generated synthetic data in CoNLL03 and SST2 under the shot-10 settings. NM refers to Novel Mentions.

5 Conclusion and Future Work

In this paper, we present the first prompt-based pre-trained language model **`PromDA`** for low-resource NLU data augmentation. Experiments on four benchmarks show the effectiveness of our proposed **`PromDA`** method. In the future, we plan to expand **`PromDA`** to other NLP tasks, including question answering, machine reading comprehension and text generation tasks.

Acknowledgement

We thank anonymous reviewers for their insightful suggestions to improve this paper. Yufei Wang, Can Xu, Qingfeng Sun, Huang Hu, Chongyang Tao, Xiubo Geng and Daxin Jiang are supported by Microsoft Software Technology Center at Asia (STCA). Yufei Wang also receives a MQ Research Excellence Scholarship and a CSIRO's DATA61 Top-up Scholarship.

References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do not have enough data? deep learning to the rescue! *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7383–7390.

Eyal Ben-David, Nadav Oved, and Roi Reichart. 2021. PADA: A prompt-based autoregressive approach for adaptation to unseen domains. *CoRR*, abs/2102.12206.

Samuel R. 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, Lisbon, Portugal. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris 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. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Xiang Dai and Heike Adel. 2020. An analysis of simple data augmentation for named entity recognition. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3861–3867, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bosheng Ding, Linlin Liu, Lidong Bing, Canasai Kruengkrai, Thien Hai Nguyen, Shafiq Joty, Luo Si, and Chunyan Miao. 2020. DAGA: Data augmentation with a generation approach for low-resource tagging tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6045–6057, Online. Association for Computational Linguistics.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. 2021. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48.

Christiane Fellbaum. 2010. Wordnet. In *Theory and applications of ontology: computer applications*, pages 231–243. Springer.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. 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 3816–3830, Online. Association for Computational Linguistics.

Lars Kai Hansen and Peter Salamon. 1990. Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12(10):993–1001.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Yutai Hou, Yijia Liu, Wanxiang Che, and Ting Liu. 2018. Sequence-to-sequence data augmentation for dialogue language understanding. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1234–1245, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26, Suzhou, China. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

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.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124, Ann Arbor, Michigan. Association for Computational Linguistics.

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.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text mining: applications and theory*, 1:1–20.

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.

H. Scudder. 1965. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11(3):363–371.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Yaqing Wang, Subhabrata (Subho) Mukherjee, Haoda Chu, Yuancheng Tu, Ming Wu, Jing Gao, and Ahmed H. Awadallah. 2021a. Meta self-training for few-shot neural sequence labeling. In *SIGKDD 2021 (Research Track)*.

Zirui Wang, Adams Wei Yu, Orhan Firat, and Yuan Cao. 2021b. Towards zero-label language learning. *CoRR*, abs/2109.09193.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. 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 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In *International Conference on Computational Science*, pages 84–95. Springer.

Mengzhou Xia, Xiang Kong, Antonios Anastasopoulos, and Graham Neubig. 2019. Generalized data augmentation for low-resource translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5786–5796, Florence, Italy. Association for Computational Linguistics.

Xinnuo Xu, Guoyin Wang, Young-Bum Kim, and Sungjin Lee. 2021. AugNLG: Few-shot natural language generation using self-trained data augmentation. 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 1183–1195, Online. Association for Computational Linguistics.

Yiben Yang, Chaitanya Malaviya, Jared Fernandez, Swabha Swayamdipta, Ronan Le Bras, Ji-Ping Wang, Chandra Bhagavatula, Yejin Choi, and Doug Downey. 2020. Generative data augmentation for commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1008–1025, Online. Association for Computational Linguistics.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100.

A Experiment Details

A.1 Implementation Details for NLU model

We use *BERT-BASE* as our NLU models. The **Baseline** model is only trained with the few-shot training data $\mathcal{T}$. Given the newly generated synthetic data, we will train the same NLU model with the same set of hyper-parameters. The only difference between the two NLU models is the training data. To train the *BERT-BASE* model, we use the Adam optimizer to train the model with learning rate 5e-5 and weight decay 5e-6. We train all NLU models with 4,000 steps and check the validation performance every 400 steps. We use batch size 8.

A.2 Implementation Details for Compared Models

EDA [1] and **SDANER** [2] are rule-based data augmentation methods. They modify the available training instances via simple rules, including word order shuffle, synonym replace, etc. Since they have released their source code on GitHub, we directly run their source code, without any modification, for our experiments. **BackT.** first translates the input sentence in language A to language B, and then translates back to language A, which may create new linguistic expressions in the back-translated sentences. We directly use the *M2M100* model (Fan et al., 2021), without any fine-tuning, to translate the sentence from English to French and backwards. **CBERT** (Wu et al., 2019) uses BERT model to replace words in the input sentences. Compared to EDA, the decision is made based on the context information, which should be more accurate. We use the suggested parameters and code released by the authors [3]. We Implement the **LAMBADA** model based on its original paper (Anaby-Tavor et al., 2020). The only difference is that, to allow a fair comparison with our proposed **PromDA** method, we replace its PLMs (i.e., GPT2) with T5-Large model. For **LM adaptation**, we follow the fine-tuning configuration in its original paper (Lester et al., 2021).

A.3 Trainable Parameters

PromDA adds 5 trainable vectors at each encoder layer of the frozen T5-Large model. The total trainable parameters in **PromDA** is 2 * 5 * 24 * 1024 = 245760 (2 for two sets of *Soft Prompt* for *Input View* and *Output View*). This parameter scale is very closed to the **LM Adaptation** approach which has 2 * 100 * 1024 = 204800 trainable parameters.

A.4 Dual-View Data Augmentation

As shown in Alg. 1, we train M_{LM} using few-shot data $\mathcal{T}$. We then feed the keywords in $\mathcal{T}$ to the *Input View* and the output label sequence to the *Output View*. We duplicate each instance in $\mathcal{T}$ 40 times before feeding them into **PromDA** for generation. We use the standard nucleus sampling (Holtzman et al., 2020) with top_p = 0.9. For each input sequence, we sample 5 output sequences. Finally, we duplicate each instance in $\mathcal{T}$ 100 times, then combine them with $\mathcal{T}_{LM}^{T}$. For iteration-based NLU Consistency Filtering, we find that iterating 3 times is a powerful filtering strategy.

A.5 Computing Infrastructure and Running Time

We use Nvidia A100 and V100 for our experiment. A single A100 or V100 is capable to handle the T5-Large model. In general, it takes around 6-8 hours to generate synthetic data for few-shot training data $\mathcal{T}$ with 300 - 400 instances.

A.6 Evaluation Metrics

We report averaged Micro-F1 (short for micro-averaged F1 score), which assesses the quality of multi-label binary problems by measuring the F1-score of the aggregated contributions of all classes, for the 5 times for each of our experiment. We also conduct statistical test using the paired t-student test between the baseline model results and **PromDA** method. We use the implementation of scipy [4] to calculate p values. All of **PromDA** result are statistical significant ($p < 0.05$).

B Dataset

B.1 Evaluation Source

As for the evaluation benchmarks, the CoNLL03 and Wikiann dataset are from the repository of MetaST (Wang et al., 2021a) [5]. CoNLL03 and Wikiann are public benchmarks for Named Entity Recognition. CoNLL03 is a collection of news wire articles from the Reuters Corpus with manual annotations, whereas Wikiann comprises of extractions

[1] https://github.com/jasonwei20/eda_nlp

[2] https://github.com/boschresearch/data-augmentation-coling2020

[3] https://github.com/1024er/cbert_aug

[4] https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest_rel.html

[5] https://github.com/microsoft/MetaST

from Wikipedia. The SST2 (Stanford Sentiment Tree-bank) and RT (a movie review corpus from Rotten Tomatoes) dataset are from the repository of CBERT (Wu et al., 2019) [6].

B.2 Training data for different Few-shot Settings

Table 9 shows the number of training data in different few-shot settings.

Shot	10	20	50	100
CoNLL03	40	80	200	400
Wikiann	30	60	150	300
SST2	20	40	100	200
RT	20	40	100	200

Table 9: The new of training data instances for each benchmark under different shot-k settings.

C Experiment Analysis

C.1 Shot-20 and Shot-100 Results

Table 10 and 11 show the concrete performance of **PromDA** and other baseline models under the shot-20 and shot-100 settings. It is interesting to note that **F.LMs** often outperforms other baseline models in the shot-100 setting. This could because **F.LMs** avoids over-fitting and starts to learn to generate novel mentions when the few-shot training data becomes larger.

DataSet	C03		Wiki	
Shot	20	100	20	100
Baseline	77.8	85.4	56.1	70.0
SDANER♠	78.4	85.2	<u>58.7</u>	70.3
F.LMs	78.6	85.5	<u>62.9</u>	<u>71.0</u>
MetaST♣	78.5	85.8	63.6	71.2
PromDA	<u>80.1</u>	85.9	<u>65.1</u>	<u>72.9</u>

Table 10: Experiment Results of the Sequence Labelling Tasks. ♣ results taken from (Wang et al., 2021a). ♠ we run Dai and Adel (2020)'s source code. C03 refers to CoNLL03 and Wiki refers to Wikiann. <u>Underline</u> are the significant results compared to the **Baseline** model (paired student's t-test, $p < 0.05$).

C.2 Unlabeled Data Domain

In Sec 4.4, we analysis three types of unlabeled data: *Unlabeled In-domain Data* (**UID**), *Unlabeled Near-domain Data* (**UND**) and *Unlabeled General-domain Data* (**UGD**). We will give details on how these three types of unlabeled data are constructed. The *Unlabeled In-domain Data* are the training instances in the original full training data but not included in the current few-shot training set $\mathcal{T}$. When used as unlabeled data, we ignore their supervised labels. Those training instances are from the exactly same source and therefore, they are guaranteed to be in the *same* domain. We exchange the training data between CoNLL03 and Wikiann, and between SST2 and RT as *Unlabeled Near-domain Data* to simulate *similar* domains. This is because that *1)* both CoNLL03 and Wikiann have *Person*, *Organization* and *Location*; *2)* both SST2 and RT are reviews in daily life. Finally, we randomly sample 10,000 sentences from the T5 pre-training corpus to simulate the *general-purpose* domain.

[6] https://github.com/1024er/cbert_aug

DataSet	SST2		RT	
Shot	20	100	20	100
Baseline	71.7	84.3	65.4	77.6
EDA♠	73.6	84.6	64.5	77.4
BackT.	<u>76.8</u>	83.7	66.0	77.6
CBERT♣	<u>76.9</u>	85.3	64.1	77.8
F.LMs	<u>78.7</u>	85.4	<u>71.9</u>	80.5
PromDA	<u>83.2</u>	<u>87.3</u>	<u>75.4</u>	<u>83.0</u>

Table 11: Experiment Results of the Sentence Classification Tasks. ♠ we run Wei and Zou (2019)'s source code. ♣ we run Wu et al. (2019)'s source code. <u>Underline</u> are the significant results compared to the **Baseline** model (paired student's t-test, $p < 0.05$).

C.3 Diversity Metrics

In Sec 4.4, we use two metrics, **Novel Mention** and **Self-Bleu**, to measure the diversity of generated synthetic data. **Novel Mention** is defined as the entity mention or keywords that do not appearing in the training data. For the sequence labelling tasks, we directly extract the named entity mentions from each instance as the *Mentions*. For the sentence classification tasks, we extract top-3 keywords from the input sentence using the unsupervised keyword extract *Rake* (Rose et al., 2010) as the *Mentions*. The higher **Novel Mention** is, the better. **Self-Bleu** evaluates how one sentence resembles the rest in a generated collection. The lower **Self-Bleu** is, the better.