Instance-aware Prompt Learning for Language Understanding and Generation

FEIHU JIN, Institute of Automation, Chinese Academy of Sciences (CAS), School of Artificial Intelligence, University of Chinese Academy of Sciences, China
JINLIANG LU, Institute of Automation, Chinese Academy of Sciences (CAS), the School of Artificial Intelligence, University of Chinese Academy of Sciences, China
JIAJUN ZHANG, Institute of Automation, Chinese Academy of Sciences (CAS), the School of Artificial Intelligence, University of Chinese Academy of Sciences, China
CHENGQING ZONG, Institute of Automation, Chinese Academy of Sciences (CAS), the School of Artificial Intelligence, University of Chinese Academy of Sciences, China

Prompt learning has emerged as a new paradigm for leveraging pre-trained language models (PLMs) and has shown promising results in downstream tasks with only a slight increase in parameters. However, the current usage of fixed prompts, whether discrete or continuous, assumes that all samples within a task share the same prompt. This assumption may not hold for tasks with diverse samples that require different prompt information. To address this issue, we propose an instance-aware prompt learning method that learns a different prompt for each instance. Specifically, we suppose that each learnable prompt token has a different contribution to different instances, and we learn the contribution by calculating the relevance score between an instance and each prompt token. The contribution weighted prompt would be instance aware. We apply our method to both unidirectional and bidirectional PLMs on both language understanding and generation tasks. Extensive experiments demonstrate that our method achieves comparable results using as few as 1.5% of the parameters of PLMs tuned and obtains considerable improvements compared to strong baselines. In particular, our method achieves state-of-the-art results using ALBERT-xxlarge-v2 on the SuperGLUE few-shot learning benchmark[1].

CCS Concepts: • **Computing methodologies → Natural language understanding, Natural language generation**.

Additional Key Words and Phrases: Pre-trained language model, Prompt learning, Parameter-efficient tuning, Few-shot learning

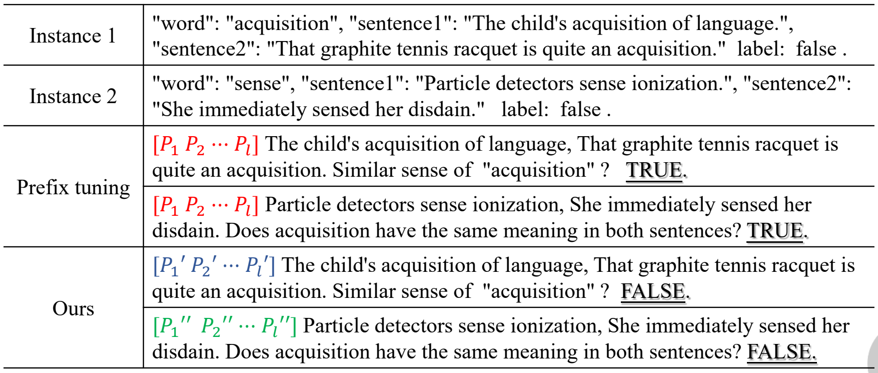
1 INTRODUCTION

Prompt learning aims to design or learn appropriate prompts which can induce the capacity of pre-trained language models (PLMs) to perform specific tasks, and it becomes a new paradigm to use PLMs due to its

[1] We have released our code in: https://github.com/jinfeihu-stan/IPL

Authors' addresses: Feihu Jin, jinfeihu2020@ia.ac.cn, Institute of Automation, Chinese Academy of Sciences (CAS), School of Artificial Intelligence, University of Chinese Academy of Sciences, No. 95 Zhongguancun east road, Beijing, China, 100089; Jinliang Lu, Institute of Automation, Chinese Academy of Sciences (CAS), the School of Artificial Intelligence, University of Chinese Academy of Sciences, No. 95 Zhongguancun east road, Beijing, China, 100089, lujinliang2019@ia.ac.cn; Jiajun Zhang, Institute of Automation, Chinese Academy of Sciences (CAS), the School of Artificial Intelligence, University of Chinese Academy of Sciences, No. 95 Zhongguancun east road, Beijing, China, 100089, jjzhang@nlpr.ia.ac.cn; Chengqing Zong, Institute of Automation, Chinese Academy of Sciences (CAS), the School of Artificial Intelligence, University of Chinese Academy of Sciences, No. 95 Zhongguancun east road, Beijing, China, 100089, cqzong@nlpr.ia.ac.cn.

2375-4699/2023/6-ART $15.00
https://doi.org/10.1145/3604613



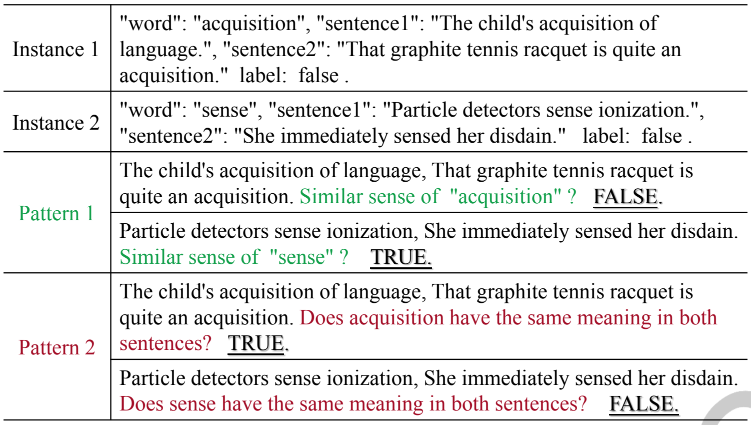
Instance 1	"word": "acquisition", "sentence1": "The child's acquisition of language.", "sentence2": "That graphite tennis racquet is quite an acquisition." label: false .
Instance 2	"word": "sense", "sentence1": "Particle detectors sense ionization.", "sentence2": "She immediately sensed her disdain." label: false .
Prefix tuning	$[P_1\ P_2 \cdots P_l]$ The child's acquisition of language, That graphite tennis racquet is quite an acquisition. Similar sense of "acquisition" ? TRUE.
	$[P_1\ P_2 \cdots P_l]$ Particle detectors sense ionization, She immediately sensed her disdain. Does acquisition have the same meaning in both sentences? TRUE.
Ours	$[P_1'\ P_2' \cdots P_l']$ The child's acquisition of language, That graphite tennis racquet is quite an acquisition. Similar sense of "acquisition" ? FALSE.
	$[P_1''\ P_2'' \cdots P_l'']$ Particle detectors sense ionization, She immediately sensed her disdain. Does acquisition have the same meaning in both sentences? FALSE.

Fig. 1. The example is chosen from the WiC dataset in SuperGLUE. Prefix tuning uses the same prompt for all samples, while our method learns a special prompt for each instance, yields the correct answer.

flexibility and fewer extra parameters. There are two types of prompts: discrete prompts and continuous prompts. Discrete prompts, such as those used in GPT-3 [Brown et al. 2020], use task instructions and task-related instances as prompts for zero-shot and few-shot learning, respectively. PET/iPET [Schick and Schütze 2021a,b] utilizes the manually-designed prompts to reformulate many tasks as cloze questions (e.g., by appending phrases such as "Similar sense of two sentences?") and performs gradient-based fine-tuning with smaller PLMs. However, designing discrete prompts manually can be time-consuming and labor-intensive, and therefore, several efforts have focused on searching for proper discrete prompts automatically [Gao et al. 2021; Shin et al. 2020; Zhong et al. 2021].

Although discrete prompts can reflect rationality from the perspective of humans, it may not be necessarily suitable for PLMs. To tackle this problem, a lot of studies begin to focus on continuous prompts. The continuous prompt is another form of prompt learning method, which mainly includes two methods: prompt tuning [Lester et al. 2021] and prefix tuning [Li and Liang 2021]. Lester et al. [2021] propose prompt tuning and concatenate the fixed continuous prompts with instances in the embedding layer of PLMs. When using small PLMs, the performance of prompt tuning has a clear gap with fine-tuning. Li and Liang [2021] propose prefix tuning and show comparable results with fine-tuning on generation tasks. Prefix tuning prepends learnable prefix vectors to the key ($\boldsymbol{K}$) and the value ($\boldsymbol{V}$) of the multi-head attention at each layer of the transformer and only optimizes 0.1%~3% parameters of the model. However, the current usage of discrete and continuous prompts assumes that all samples in one task share the same prompt, and does not consider the diversity of the instances, which require different prompt information. In Figure 1, we show that prefix tuning uses the same prompt for all samples that give wrong answers, while our method learns a unique prompt for each instance, yields the correct answer. Therefore, it is desirable to learn a special prompt for each instance.

In this paper, we propose an **I**nstance-aware **P**rompt **L**earning method (abbreviated as IPL) which learns a unique prompt for each instance. As shown in Figure 2, two manually-designed patterns (e.g., different colors mean different patterns) are used to formalize the instances into cloze-style questions and fed into the pre-trained language model (PLM). As we can see, using different prompts can lead to different answers for each instance, where pattern 1 may be suitable for instance 1 while pattern 2 may fit instance 2, indicating that each instance requires a specific prompt. However, it is challenging to dynamically identify a suitable discrete prompt for each instance. To address this issue, we propose using a look-up module to obtain a dynamic continuous prompt for each instance. Specifically, we treat each learnable prompt token as a query and calculate its contribution to



Instance 1	"word": "acquisition", "sentence1": "The child's acquisition of language.", "sentence2": "That graphite tennis racquet is quite an acquisition." label: false .
Instance 2	"word": "sense", "sentence1": "Particle detectors sense ionization.", "sentence2": "She immediately sensed her disdain." label: false .
Pattern 1	The child's acquisition of language, That graphite tennis racquet is quite an acquisition. Similar sense of "acquisition" ? FALSE.
	Particle detectors sense ionization, She immediately sensed her disdain. Similar sense of "sense" ? TRUE.
Pattern 2	The child's acquisition of language, That graphite tennis racquet is quite an acquisition. Does acquisition have the same meaning in both sentences? TRUE.
	Particle detectors sense ionization, She immediately sensed her disdain. Does sense have the same meaning in both sentences? FALSE.

Fig. 2. The example is chosen from the WiC dataset in SuperGLUE. The color words indicate the manually-designed patterns which are used to formalize the instance into close-style questions.

each instance through the look-up module. This way, each learnable prompt token has a different impact on the instance, and the weighted continuous prompts can guide the pre-trained language models to perform the downstream task in an instance-aware manner.

Our approach is evaluated on both natural language understanding (NLU) and generation (NLG) tasks. For NLU tasks, we conduct experiments on SuperGLUE [Wang et al. 2019] with RoBERTa [Delobelle et al. 2020]. For NLG tasks, we conduct experiments on summarization using GPT-2 [Radford et al. 2019]. The experimental results on various tasks demonstrate that our method, with as few as 1.5% of the parameters of PLMs tuned, achieves comparable results with fine-tuning and obtains significant improvements over strong baselines. Notably, our method achieves a new state-of-the-art on the SuperGLUE few-shot learning benchmark using ALBERT-xxlarge-v2 [Lan et al. 2020]. In summary, our key contributions can be listed as follows:

- We propose an instance-aware prompt learning method that can learn a unique prompt for each instance.
- Extensive experiments on both language understanding and generation tasks under both unidirectional and bidirectional PLMs verify the effectiveness of our method.
- Detailed analyses verify that IPL can indeed dynamically learn appropriate continuous prompts for each instance.

2 APPROACH

In this section, we present the details of our instance-aware prompt learning method IPL. Previous studies have shown the promise of prompt learning for downstream tasks. However, using fixed prompts (e.g., discrete prompts like "convert the table into a sentence" or continuous prompts after optimization) for diverse instances in one task ignores the peculiarity of different instances. To address this problem, our IPL model is designed to learn a special prompt for each specific instance. We first introduce prompt learning and then present the details of our IPL model.

2.1 Prompt Learning

For the standard paradigm of pre-training and fine-tuning, there is a gap (e.g., inconsistent objective function) between the pre-training stage and the fine-tuning stage. Fortunately, prompt learning bridges this gap by formalizing the downstream tasks into the form of a conditional language model or masked language model. The discrete prompt is an important method in prompt learning. For example, given a masked language model $\mathcal{M}$, we can use a prompt to formulate a question-and-answer instance x (e.g., [passage] Can you have too much oxygen in your body? where [passage] represents the context information.) as follows:

$$\hat{x} = x \text{ the answer is [MASK].}$$

Then $\hat{x}$ is fed into $\mathcal{M}$, and let $\mathcal{M}$ determine whether "Yes" or "No" is more appropriate to replace [MASK] [Gao et al. 2021].

Continuous prompt is an alternative approach in prompt learning. As shown in Figure 3(a), $\boldsymbol{P_k}, \boldsymbol{P_v} \in \mathbb{R}^{l \times d}$ are two sets of prefix vectors, $\boldsymbol{K}, \boldsymbol{V} \in \mathbb{R}^{n \times d}$ denote key and value that are the projection of the input $\{\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h_n}\}$. Prefix tuning [Li and Liang 2021] prepends learnable prefix vectors $\boldsymbol{P_k}, \boldsymbol{P_v}$ to $\boldsymbol{K}$ and $\boldsymbol{V}$ of the multi-head attention at each layer of the Transformer, which can be formalized as follows:

$$\text{Head} = \text{Attn}(\boldsymbol{Q}, Con(\boldsymbol{P_k}, \boldsymbol{K}), Con(\boldsymbol{P_v}, \boldsymbol{V}))$$

where Con means concatenation.

In this paper, we combine the advantages of continuous prompts with discrete prompts and propose an instance-aware prompt learning method that learns a unique prompt for each instance. Next, we detail our proposed IPL mdoel.

2.2 Instance-aware Prompt Learning

We denote $\boldsymbol{H} = \{\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h_n}\} \in \mathbb{R}^{n \times d}$ as the input in each layer of the Transformer, where n is the length of input $\boldsymbol{H}$ and d is the the dimension of the embedding space. Following the Prefix tuning [Li and Liang 2021], we use a reparameterization encoder such as an MLP to generate learnable prefix vectors $\boldsymbol{P} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p_m}\} \in \mathbb{R}^{m \times d}$, where m is the length of prefix vectors $\boldsymbol{P}$. As shown in Figure 3(b), we project the input $\boldsymbol{H}$ and the prefix vectors $\boldsymbol{P}$ to a lower-dimensional space using the projection matrix $\boldsymbol{N_k}, \boldsymbol{N_v} \in \mathbb{R}^{d \times d_l}$ and $\boldsymbol{M_k}, \boldsymbol{M_v} \in \mathbb{R}^{d \times d_l}$, where d_l is the dimension of the projection space, leading to the form:

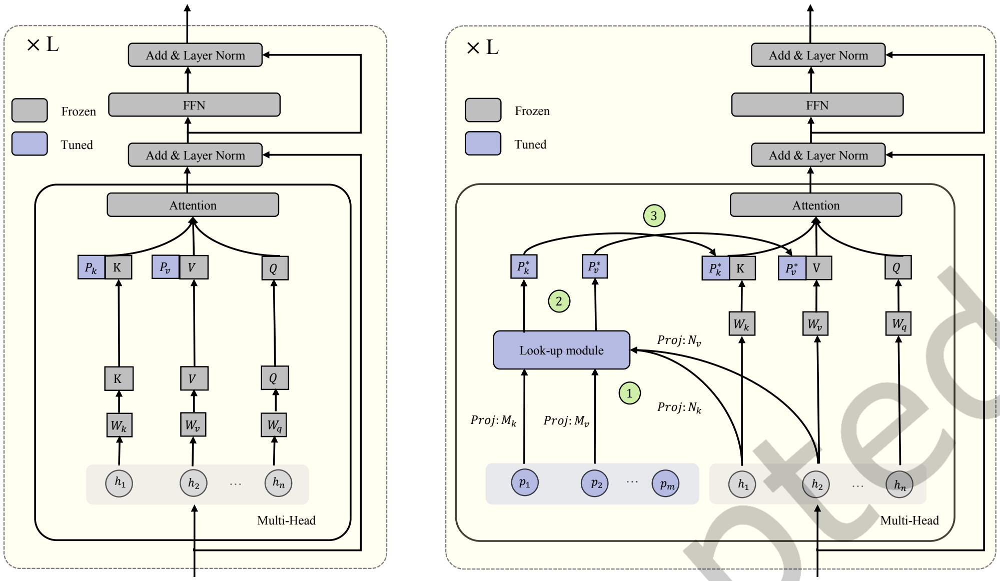
$$\begin{aligned} \boldsymbol{H^k} &= \boldsymbol{HN_k} \\ \boldsymbol{H^v} &= \boldsymbol{HN_v} \end{aligned} \tag{1}$$

and

$$\begin{aligned} \boldsymbol{P^k} &= \boldsymbol{PM_k} \\ \boldsymbol{P^v} &= \boldsymbol{PM_v} \end{aligned} \tag{2}$$

where $\boldsymbol{H^k}, \boldsymbol{H^v} \in \mathbb{R}^{n \times d_l}$ and $\boldsymbol{P^k}, \boldsymbol{P^v} \in \mathbb{R}^{m \times d_l}$

We suppose that each learnable prefix vector has a different contribution to different instances and firstly we learn the contribution scores by calculating the relevance score between matrix $\boldsymbol{P^k} = \{\boldsymbol{p}_1^k, \boldsymbol{p}_2^k, \cdots, \boldsymbol{p}_m^k\}$ and $\boldsymbol{H^k} = \{\boldsymbol{h}_1^k, \boldsymbol{h}_2^k, \cdots, \boldsymbol{h}_n^k\}$, $\boldsymbol{P^v} = \{\boldsymbol{p}_1^v, \boldsymbol{p}_2^v, \cdots, \boldsymbol{p}_m^v\}$ and $\boldsymbol{H^v} = \{\boldsymbol{h}_1^v, \boldsymbol{h}_2^v, \cdots, \boldsymbol{h}_n^v\}$. After getting the relevance score, we



(a) Prefix tuning

(b) Our method: IPL

Fig. 3. Illustration of prefix tuning (Left) and our method: IPL (Right). $\{\boldsymbol{h_1}, \boldsymbol{h_2}, \cdots, \boldsymbol{h_n}\}$ represents the hidden states in each layer of Transformer. $\{\boldsymbol{p_1}, \boldsymbol{p_2}, \cdots, \boldsymbol{p_m}\}$ represents the tunable prefix vectors. $\boldsymbol{N_k}, \boldsymbol{N_v}, \boldsymbol{M_k}, \boldsymbol{M_v}$ are the projection (Proj) matrix. Blue blocks refer to the trainable parameters and gray blocks refer to the frozen parameters of PLMs.

pass the score to the look-up module.

$$s_j^k = \sigma(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{p}_j^{\boldsymbol{k}} \cdot (\boldsymbol{h}_i^{\boldsymbol{k}})^T) \tag{3}$$

$$s_j^v = \sigma(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{p}_j^{\boldsymbol{v}} \cdot (\boldsymbol{h}_i^{\boldsymbol{v}})^T) \tag{4}$$

Secondly, in the look-up module, we adopt a method of mean operation and apply a sigmoid function σ to obtain how much does each learnable prefix vector contribute to the input $\boldsymbol{H}$.

$$\boldsymbol{p^{\star}}_j^{\boldsymbol{k}} = s_j^k \cdot \boldsymbol{p}_j^{\boldsymbol{k}} \tag{5}$$

$$\boldsymbol{p^{\star}}_j^{\boldsymbol{v}} = s_j^v \cdot \boldsymbol{p}_j^{\boldsymbol{v}} \tag{6}$$

where s_j^k and s_j^v are the contribution scores of the j-th prefix vector after applying a sigmoid funtion σ, and $\boldsymbol{p^{\star}}_j^{\boldsymbol{k}}$ and $\boldsymbol{p^{\star}}_j^{\boldsymbol{v}}$ are the j-th weighted representation for the input. After doing such a calculation for all prefix vectors, we get the instance-aware prompt as $\boldsymbol{P_k^{\star}} = \{\boldsymbol{p^{\star}}_1^{\boldsymbol{k}}, \boldsymbol{p^{\star}}_2^{\boldsymbol{k}}, \cdots, \boldsymbol{p^{\star}}_{\boldsymbol{m}}^{\boldsymbol{k}}\}$ and $\boldsymbol{P_v^{\star}} = \{\boldsymbol{p^{\star}}_1^{\boldsymbol{v}}, \boldsymbol{p^{\star}}_2^{\boldsymbol{v}}, \cdots, \boldsymbol{p^{\star}}_{\boldsymbol{m}}^{\boldsymbol{v}}\}$.

Thirdly, we concatenate our instance-aware prefix vectors with the original key $\boldsymbol{K}$ and value $\boldsymbol{V}$.

$$\text{Head} = \text{Attn}(\boldsymbol{Q}, Con(\boldsymbol{P_k^{\star}}, \boldsymbol{K}), Con(\boldsymbol{P_v^{\star}}, \boldsymbol{V}))$$

2.3 Optimization

We optimize the parameters of our proposed IPL model in two ways: parameter-efficient tuning[Houlsby et al. 2019] and vanilla fine-tuning[Delobelle et al. 2020]. For parameter-efficient tuning, we introduce an instance-aware module within each layer of the Transformer and only optimize the parameters of the instance-aware module while keeping the parameters of the original model frozen. We also apply our method to the embedding layer, following the approach of prompt tuning [Lester et al. 2021], and only optimize the parameters of the instance-aware module. For vanilla fine-tuning, we aim to reduce the number of trainable parameters by only applying our method to the embedding layer and optimizing all parameters of the instance-aware module and original pre-trained language model.

Corpus	|Train|	|Dev|	|Test|	Task	Metrics
SuperGLUE					
BoolQ [Clark et al. 2019]	9427	3270	3245	QA	acc.
CB [De Marneffe et al. 2019]	250	57	250	NLI	acc.
MultiRC [Khashabi et al. 2018]	5100	953	1800	QA	$F1_a$
RTE [Bar-Haim et al. 2014]	2500	278	300	NLI	acc.
WiC [Pilehvar and Camacho-Collados 2019]	6000	638	1400	WSD	acc.
COPA [Roemmele et al. 2011]	400	100	500	QA	acc.
WSC [Levesque et al. 2012]	554	104	146	coref.	acc.
ReCoRD [Zhang et al. 2018]	101k	10k	10k	QA	F1
FewGLUE					
BoolQ [Clark et al. 2019]	32	3270	3245	QA	acc.
CB [De Marneffe et al. 2019]	32	57	250	NLI	acc.
MultiRC [Khashabi et al. 2018]	32	953	1800	QA	$F1_a$
RTE [Bar-Haim et al. 2014]	32	278	300	NLI	acc.
WiC [Pilehvar and Camacho-Collados 2019]	32	638	1400	WSD	acc.
COPA [Roemmele et al. 2011]	32	100	500	QA	acc.
WSC [Levesque et al. 2012]	32	104	146	coref.	acc.
ReCoRD [Zhang et al. 2018]	32	10k	10k	QA	F1
Summarization					
SAMSum [Gliwa et al. 2019]	14733	818	819	Dialog Summarization	Rouge-L
DialogSum [Chen et al. 2021]	12473	501	500	Dialog Summarization	Rouge-L

Table 1. The NLU and NLG datasets evaluated in our work. We report the accuracy or F1 score for each dataset.

3 EXPERIMENTS

3.1 Experimental Settings

Datasets: We conduct extensive experiments on several downstream tasks, including NLU and NLG tasks, using various datasets. Specifically, we evaluate our method on the following datasets: (1) SuperGLUE[2] [Wang et al. 2019]: A benchmark containing eight NLU tasks. (2) SAMsum [Gliwa et al. 2019] and Dialogsum [Chen et al. 2021]: Two English dialogue summarization tasks where models predict a short summary given a dialogue. (3)

[2] https://supergluebenchmark.com/

Corpus	Epoch	Batch size	Learning rate	Prompt length	Weight Decay
Fully-supervised					
BoolQ	40	32	2e-4	20	0.1
CB	40	8	5e-4	16	0.1
MultiRC	30	32	3e-4	20	0.1
RTE	40	32	2e-4	16	0.1
WiC	40	32	3e-4	25	0.1
COPA	40	8	3e-4	30	0.1
WSC	40	32	1e-3	16	0.1
ReCoRD	20	32	3e-4	16	0.1
SAMSum	15	32	5e-5	20	0.1
DialogSum	15	32	4e-5	20	0.1
Few-shot					
BoolQ	30	8	3e-5	16	0.1
CB	30	8	3e-5	16	0.1
MultiRC	30	8	2e-5	10	0.1
RTE	30	4	3e-5	16	0.1
WiC	30	4	3e-5	16	0.1
COPA	30	16	1e-5	16	0.1
WSC	30	8	3e-5	16	0.1
ReCoRD	30	8	2e-5	10	0.1

Table 2. Hyperparameter settings for our method IPL in different models and different tasks.

FewGLUE[3] [Schick and Schütze 2021b]: A low-resource task that contains only 32 labeled examples per task for training from the SuperGLUE dataset. Table 1 shows the detailed information for the NLU and NLG datasets used in our work.

Architectures: In NLU tasks, we use RoBERTa-large [Delobelle et al. 2020] as the base PLMs for SuperGLUE in fully-supervised learning. For FewGLUE, which involves few-shot learning, we use ALBERT-xxlarge-v2 [Lan et al. 2020] as the base PLMs, as it was also used in other baselines for few-shot learning. For summarization, we use GPT2-large [Radford et al. 2019] as the base PLMs.

Evaluation and Hyperparameters: For SuperGLUE, we report accuracy and F1-score metrics. The model is trained for 40 epochs with a default setting, using RoBERTa-large [Delobelle et al. 2020] as the underlying PLMs, a learning rate of 3e-4, a batch size of 32, and a prompt length of 16. For FewGLUE, considering the limited labeled examples, we do not freeze the parameters of PLMs and use ALBERT-xxlarge-v2 [Lan et al. 2020] as the underlying PLMs. The model is trained for 20 epochs with a learning rate of 1e-5, a batch size of 8, and a prompt length of 16. For the summarization tasks SAMsum and Dialogsum, we report ROUGE-1, ROUGE-2, and ROUGE-L [Lin 2004] metrics. The model is trained for 10 epochs with GPT2-large [Radford et al. 2019] as the underlying PLMs, a learning rate of 4e-5, a batch size of 32, and a prompt length of 20. The few-shot learning models are trained on Tesla V100, and the SuperGLUE and summarization models are trained on NVIDIA DGX-A100. In Table 2, we provide detailed hyperparameters used to train the models in our experiments.

[3] https://github.com/timoschick/fewglue

Method	**#Params**	BoolQ	CB	MultiRC	RTE	WiC	COPA	WSC	ReCoRD
		Acc.	Acc.	F1a	Acc.	Acc.	Acc.	Acc.	F1
Fully fine tuning [Schick and Schütze 2021b]	100%	**85.5**	**99.1**	**83.4**	87.1	70.9	**87.0**	**88.5**	91.9
Prompt tuning [Lester et al. 2021][†]	0.01%	62.3	71.4	59.9	58.8	56.9	63.0	64.4	90.6
Prefix tuning [Li and Liang 2021]	2%	84.8	98.7	81.3	86.7	70.0	86.0	86.5	91.4
IPL (Prompt tuning)	0.02%	72.2	75.0	60.0	63.9	57.4	73.0	66.4	91.3
IPL (Prefix tuning)	1.5%	85.1	**99.1**	81.9	**87.4**	**71.6**	**87.0**	87.5	**92.1**

Table 3. Results on SuperGLUE validation set with RoBERTa-large. IPL (Prompt tuning) means the implementation of our method in the embedding layer. IPL (Prefix tuning) means the implementation of our method in each layer of the Transformer. † indicates the results reported in [Liu et al. 2021a]. We report the average tunable parameters for SuperGLUE tasks.

Method	**#Params**	SAMSum			DialogSum								
		R_1	R_2	R_L	R_1	R_2	R_L	R_1	R_2	R_L	R_1	R_2	R_L
Fully fine-tuning [Schick and Schütze 2021b]	100%	47.2	22.2	42.8	43.4	18.7	39.0	41.8	16.4	36.7	43.0	17.7	37.8
Prompt tuning [Lester et al. 2021]	0.01%	10.4	3.9	9.5	11.9	2.5	10.8	11.7	2.2	10.3	12.0	2.5	10.6
Prefix tuning [Li and Liang 2021]	3%	45.8	21	41.2	40.7	16.2	36.6	39.9	14.6	35.0	40.5	16.0	35.5
IPL (Prompt tuning)	0.02%	12.3	4.7	11,4	12.9	2.7	11.7	12.6	2.3	11.0	12.8	2.7	11.3
IPL (Prefix tuning)	3.6%	**46.6**	**21.2**	**42.2**	**41.2**	**16.7**	**37.2**	**40.1**	**14.8**	**35.2**	**41.0**	**16.3**	**36.1**

Table 4. Results on SAMSum [Gliwa et al. 2019] and DialogSum [Chen et al. 2021] test set with GPT2-large. IPL (Prompt tuning) means the implementation of our method in the embedding layer. IPL (Prefix tuning) means the implementation of our method in each layer of the Transformer. R_1, R_2, and R_L refer to the ROUGE-1, ROUGE-2, and ROUGE-L respectively. We report the average tunable parameters for two dialogue summarization datasets. The **bold** font means the best in parameter-efficient tuning.

3.2 Experiments on NLU and NLG Tasks

The IPL implementation for NLU and NLG tasks is based on PET[4] for NLU and HuggingFace [Wolf et al. 2020] for NLG. The experimental results include NLU and NLG task results.

Table 3 presents the performance of IPL on RoBERTa-large. We implement the method of prefix tuning on NLU tasks as P-tuning v2 [Liu et al. 2021a] does. The P-tuning v2 uses a randomly-initialized classification head on top of the tokens as in BERT [Devlin et al. 2019]. Different from P-tuning v2, our implementation uses the verbalizer with LM head as in PET[Schick and Schütze 2021a]. The results show that IPL with few parameters tuned matches the performance of fully fine-tuning in all tasks on SuperGLUE. Even in textual entailment task CB & RTE, co-reference resolution task WiC, causal reasoning task COPA, and reading comprehension task ReCoRD, IPL is equal to or better than fully fine-tuning. Compared to prefix tuning, IPL can lead to better results in all the tasks on SuperGLUE. We also implement our method in the embedding layer as in prompt tuning [Lester et al. 2021; Liu et al. 2021b], and the results in Table 3 demonstrate that our method can improve the performance of pre-trained language models.

[4] https://github.com/timoschick/pet

Method	BoolQ	CB	MultiRC	RTE	WiC	BoolQ	CB	MultiRC	RTE	WiC
	Acc.	Acc./F1	F1a/EM	Acc.	Acc.	Acc.	Acc./F1	F1a/EM	Acc.	Acc.
	`GPT2-base`					`GPT2-large`				
PET	74.6	94.1/95.6	70.4/**22.9**	67.1	65.7	80.4	92.9/94.8	75.8/32.4	78.7	**70.2**
PT (Fully fine-tuning)	74.2	92.9/94.7	69.8/21.2	67.0	64.2	79.7	96.4/97.4	75.8/**34.1**	75.5	69.1
IPL (Fully fine-tuning)	**74.9**	**94.6/96.0**	**70.5**/22.4	**69.7**	**66.7**	**80.8**	**98.2/98.7**	**76.0**/33.3	**80.1**	69.6

Table 5. Fully-supervised learning on SuperGLUE validation set with unidirectional pre-trained language models. PET means PET fine-tuning with a single pattern, and PT refers to prompt-tuning. For a fair comparison, we use the same pattern for all models.

Method	BoolQ	CB	MultiRC	RTE	WiC	BoolQ	CB	MultiRC	RTE	WiC
	Acc.	Acc./F1	F1a/EM	Acc.	Acc.	Acc.	Acc./F1	F1a/EM	Acc.	Acc.
	`RoBERTa-base`					`RoBERTa-large`				
PET	80.0	**96.4/95.6**	76.1/**35.1**	82.7	69.3	85.5	98.8/99.1	**83.4/51.1**	87.1	70.9
PT (Fully fine-tuning)	80.3	96.4/94.8	76.1/33.4	80.1	68.9	85.4	98.8/99.1	83.2/50.8	87.0	72.1
IPL (Fully fine-tuning)	**80.6**	**96.4/95.6**	**76.2**/34.6	**82.9**	**70.9**	**85.7**	**99.4/99.6**	**83.4**/50.9	**87.5**	**73.5**

Table 6. Fully-supervised learning on SuperGLUE validation set with bidirectional pre-trained language models. PET means PET fine-tuning with a single pattern, and PT refers to prompt tuning. For a fair comparison, we use the same pattern for all the models.

Table 4 presents the performance of IPL on GPT2-large. For the two dialogue summarization tasks, IPL achieves comparable performance to fully fine-tuning with a smaller number of parameters compared to prefix tuning. Even with fine-tuning only the parameters in the embedding layer, our method outperforms prompt tuning [Lester et al. 2021] by 0.3 points on ROUGE-2.

3.3 Instance-aware FT vs. FT

We conduct experiments on the SuperGLUE benchmark, specifically on BoolQ, MultiRC, RTE, CB, and WiC tasks, to explore the performance of the instance-aware method when fine-tuning all parameters in different models. We used both unidirectional PLM GPT-2 and bidirectional PLM RoBERTa as the baseline models and compared the performance of fine-tuning with PET [Schick and Schütze 2021a], prompt tuning [Lester et al. 2021], and our method IPL. The experiments were conducted using a default setting of 20 epochs, a learning rate of 2e-5, a batch size of 32, and a prompt length of 16.

Table 5 and Table 6 present our main results on GPT-2 and RoBERTa, respectively. For unidirectional PLMs like GPT2-base and GPT2-large, IPL outperforms PET fine-tuning and prompt tuning on all 5 tasks with GPT2-base and 4 out of 5 tasks on GPT2-large. For bidirectional PLMs like RoBERTa-base and RoBERTa-large, IPL outperforms all other RoBERTa-based models on all 5 tasks. These results demonstrate that IPL can lead to significant improvements on both GPT-2 and RoBERTa models.

For NLG tasks, we compare IPL on GPT2-base and GPT2-large with two baseline methods: standard fine-tuning and prompt tuning, where we do not freeze the model parameters as IPL does. We choose the dialog summarization task: SAMsum and report ROUGE-1, ROUGE-2, and ROUGE-L. The hyperparameters we tune include the number of epochs, batch size, learning rate, and prefix length. We set the batch size as 32, prefix

Method	R-1	R-2	R-L	R-1	R-2	R-L
	GPT2-base			GPT2-large		
FT	42.6	18.9	38.5	47.2	22.2	42.8
PT (Fully fine-tuning)	46.5	21.4	41.8	49.3	24.5	44.8
IPL (Fully fine-tuning)	**46.6**	**21.7**	**42.0**	**49.7**	**24.8**	**45.0**

Table 7. Results for summarization on SAMSum using GPT-2 models. The FT refers to fine-tuning. PT (Full) refers to prompt tuning with all parameters tuned. IPL (Full) refers to our method with all parameters tuned.

	E2E				WebNLG									DART			
	BLUE	NIST	R_L	CIDEr	BLUE			MET			TER ↓			BLUE	MET	TER ↓	BERT
					Seen	Unseen	All	Seen	Unseen	All	Seen	Unseen	All				
	GPT2-base																
FT	69.55	8.79	71.52	**2.49**	56.01	26.46	41.70	39.33	25.04	32.46	46.40	81.80	62.63	42.08	35.17	52.58	94.12
PT	69.78	8.81	71.55	**2.49**	60.55	28.03	45.51	43.25	28.82	36.30	38.03	74.17	54.60	**45.27**	**37.62**	49.83	94.76
IPL	**69.82**	**8.82**	**71.65**	**2.49**	**60.93**	**29.94**	**46.46**	**43.27**	**29.15**	**36.50**	**37.76**	**72.23**	**53.56**	42.98	35.62	**48.50**	**95.43**
	GPT2-large																
FT	**69.32**	**8.76**	**71.25**	2.48	62.11	43.52	53.61	44.56	37.39	41.21	37.06	53.62	44.65	47.16	38.24	47.35	94.43
PT	68.32	8.65	71.04	2.49	**64.18**	46.04	55.85	**45.30**	38.62	**42.17**	**34.81**	50.92	**42.19**	**48.57**	39.04	**46.12**	94.90
IPL	68.53	8.68	71.2	**2.51**	64.06	**46.12**	**55.90**	45.24	**38.64**	42.12	35.28	**50.55**	42.28	48.38	**39.15**	46.17	**95.47**

Table 8. The best score is in bold for both GPT2-base and GPT2-large. The FT refers to fine-tuning. PT refers to prompt tuning. For the metrics, the higher the better except for TER.

length as 100, and the number of epochs as 10 for both GPT2-base and GPT2-large, in addition to setting the learning rate as 5e-5 for GPT-base and 5e-6 for GPT-large. As shown in Table 7, the results show IPL performs better than fine-tuning and prompt tuning on both GPT2-base and GPT2-large models, suggesting it has the potential to scale to even larger models.

We also choose three table-to-text tasks to analyze the effectiveness of our method. E2E [Novikova et al. 2017], WebNLG [Gardent et al. 2017], DART [Radev et al. 2021] are three table-to-text generation tasks where models generate a text given a table. On E2E, we use the official evaluation script, which reports BLUE [Papineni et al. 2002], NIST [Belz and Reiter 2006], ROUGE-L [Lin 2004], and CIDEr [Vedantam et al. 2015]. On WebNLG, we use the official evaluation script, which reports BLEU, METEOR [Lavie and Agarwal 2007], and TER [Snover et al. 2006]. On DART, we use the official evaluation script and report BLEU, METEOR, TER, and BERTScore [Zhang et al. 2020]. The hyperparameters we tune include the number of epochs, batch size, learning rate, and prefix length. We set batch size as 32, prefix length as 10, the number of epochs as 10 for both GPT2-base and GPT2-large, in addition to the learning rate as 5e-5 for GPT-base, 5e-6 for GPT-large.

As shown in Table 8, on GPT2-base, IPL performs better than fine-tuning and prompt tuning on E2E and WebNLG, while on DART, which is an open domain table-to-text dataset, IPL slightly underperforms prompt tuning. On GPT2-large, IPL outperforms fine-tuning and can be comparable or better than prompt tuning.

Method	BoolQ Acc.	CB Acc./F1	MultiRC EM/F1a	RTE Acc.	WiC Acc.	COPA Acc.	WSC Acc.	ReCoRD Acc./F1	Avg.
GPT-3 [Brown et al. 2020]†	77.5	82.1/57.2	32.5/74.8	72.9	55.3	92.0	75.0	**89.0**/90.1	73.2
PET [Schick and Schütze 2021b]†	79.4	85.1/59.4	37.9/77.3	69.8	52.4	**95.0**	80.1	86.0/86.5	74.1
iPET [Schick and Schütze 2021b]†	**80.6**	**92.9**/92.4	33.0/74.0	74.0	52.2	**95.0**	80.1	86.0/86.5	76.8
ADAPET [Tam et al. 2021]‡	80.3	89.3/86.8	**39.2/80.1**	**76.5**	54.4	89.0	81.7	85.4/92.1	77.3
IPL	79.2	**92.9/94.8**	38.5/76.8	76.2	**64.6**	91.0	**84.8**	83.6/**91.1**	**79.3**

Table 9. Few-shot learning (32 examples) on SuperGLUE validation set with ALBERT-xxlarge-v2. † indicates the results reported in [Schick and Schütze 2021b], and ‡ indicates the results reported in [Tam et al. 2021].

Method	BoolQ Acc.	CB Acc./F1	MultiRC EM/F1a	RTE Acc.	WiC Acc.	COPA Acc.	WSC Acc.	ReCoRD Acc./F1	Avg.
GPT-3 [Brown et al. 2020]†	76.4	75.6/52.0	30.5/75.4	69.0	49.4	**92.0**	80.1	**90.2/90.1**	71.8
PET [Schick and Schütze 2021b]†	79.1	87.2/60.2	**36.4/76.6**	67.2	50.7	90.8	**88.4**	85.4/85.9	74.0
iPET [Schick and Schütze 2021b]†	**81.2**	88.8/79.9	31.7/74.1	70.8	49.3	90.8	**88.4**	85.4/85.9	75.4
ADAPET [Tam et al. 2021]‡	80.0	**92.0**/82.3	35.7/76.2	**75.0**	53.5	85.4	85.6	85.5/86.1	76.0
IPL	78.4	**92.0/85.9**	35.1/75.9	74.9	**60.9**	85.6	84.9	83.5/84.3	**76.6**

Table 10. Few-shot learning (32 examples) on SuperGLUE test set with ALBERT-xxlarge-v2. † indicates the results reported in [Schick and Schütze 2021b], and ‡ indicates the results reported in [Tam et al. 2021]

Additionally, IPL obtains better performance on WebNLG unseen domains suggesting that IPL can generalize to other domains better.

3.4 Few-shot Learning Results

Considering the limited data in few-shot learning and our desire to introduce fewer parameters, we apply our method only in the embedding layer and do not freeze the PLMs. For a fair comparison, we choose ALBERT-xxlarge-v2 [Lan et al. 2020] for experiments and use the same data split as in PET [Schick and Schütze 2021b], which consists of 32 labeled examples for each task.

Our main results on the validation and test sets on SuperGLUE are shown in Table 9 and Table 10. We compare against GPT-3, PET/iPET and ADAPET [Tam et al. 2021]. Initially, ADAPET does not use the unlabeled data and achieves the state-of-the-art in small language models[5] on SuperGLUE few-shot learning tasks compared to PET/iPET which uses the unlabeled data. As for IPL, we train IPL with a single pattern and do not use the unlabeled data.

Table 9 demonstrates that, on average, IPL outperforms GPT-3 by 6 points and PET's iterative variant, iPET, by 2.5 points. Moreover, IPL even outperforms the previous state-of-the-art model ADAPET by 2 points on the validation set. Specifically, compared to iPET and GPT-3, IPL exhibits improvements in 5 out of 8 tasks and 6 out of 8 tasks, respectively, highlighting the effectiveness of our method in few-shot natural language understanding

[5] PaLM Chowdhery et al. [2022] with 540 billion parameters is currently the state-of-the-art model for few-shot learning on SuperGLUE.

Method	Dataset	0	4	8	16	20	30	40
PT (Fully fine-tuning)	WiC	55.3	59.5	62.4	62.0	60.5	59.1	58.6
IPL (Fully fine-tuning)	WiC	55.3	62.5	63.3	64.6	61.9	60.3	60.0
PT (Fully fine-tuning)	WSC	80.1	80.4	81.0	83.6	85.2	80.1	77.6
IPL (Fully fine-tuning)	WSC	80.1	80.8	82.1	84.8	86.5	80.8	78.9
PT (Fully fine-tuning)	CB	89.3	90.2	89.3	91.0	88.7	88.3	88.4
IPL (Fully fine-tuning)	CB	89.3	91.1	89.3	92.9	89.3	89.3	91.1

Table 11. Few-shot learning (32 examples) on validation set of WiC, WSC, and CB with ALBERT-xxlarge-v2. We analyze the performance of task-specific and instance-aware prompts under different prompt lengths (e.g., 0, 4, 8, 16, 20, 30, 40). PT (Fully fine-tuning) refers to prompt tuning with all parameters tuned. IPL (Fully fine-tuning) refers to our method with all parameters tuned.

Method	0	5	10	20	50	100	0	5	10	20	50	100
	GPT2-base						GPT2-large					
PT (Fully fine-tuning)	38.5	40.6	41.2	41.4	41.4	41.8	42.8	43.4	44.0	43.8	44.2	44.8
IPL (Fully fine-tuning)	38.5	41.3	41.7	41.8	41.7	42.0	42.8	44.6	44.7	44.5	45.2	45.0

Table 12. Results for summarization on SAMSum with GPT-2 models. We analyze the performance of task-specific and instance-aware prompts under different prompt lengths (e.g., 0, 5, 10, 20, 50, 100). PT (Fully fine-tuning) refers to prompt tuning with all parameters tuned. IPL (Fully fine-tuning) refers to our method with all parameters tuned.

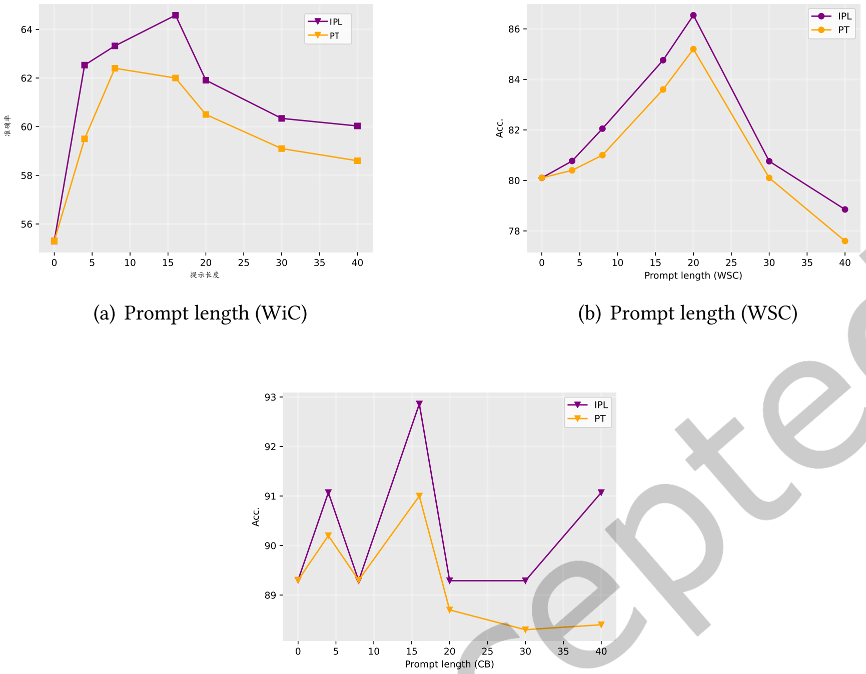
tasks. Table 10 presents our test set results on SuperGLUE, where IPL outperforms GPT-3 by 4.8 points, PET by 2.6 points, iPET by 1.2 points, and ADAPET by 0.6 points. Our approach achieves a new state-of-the-art in small language models for few-shot learning on SuperGLUE, demonstrating its effectiveness in improving the performance of natural language understanding models in low-resource settings.

4 ANALYSIS

We conduct detailed analyses on IPL. In section 4.1, we study the effect of the prompt length on the performance of NLU tasks and NLG tasks. In section 4.2, we visualize the attention matrix of similar instances and dissimilar instances to verify the effectiveness of our approach. In section 4.3, we show the average Euclidean distance between the prompt and instances for different methods and present two case studies.

4.1 Prompt Length

We present a visualization of the relationship between performance and varying prompt lengths, while keeping other settings fixed, and utilizing different prompt methods. For NLU tasks, we conduct experiments on three tasks from the SuperGLUE benchmark, namely CB, WSC, and WiC. We utilize the ALBERT-xxlarge-v2 model for these experiments and the results are presented in Table 11. Figure 4(a), 4(b) and 4(c) show that performance



(a) Prompt length (WiC)

(b) Prompt length (WSC)

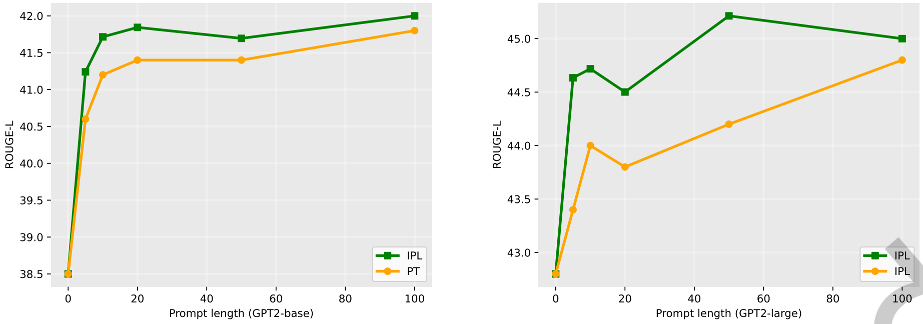
(c) Prompt length (CB)

Fig. 4. In the few-shot setting of SuperGLUE, which includes WiC, WSC, and CB, the performance on the validation set varies as the prompt length changes. Specifically, the prompt length can take on values of 0, 4, 8, 16, 20, 30, or 40.

increases as the prompt length increases up to a threshold (16 for CB and WiC, 20 for WSC), and then the performance slightly drops. For NLG tasks, We conduct experiments on the SAMSum dataset with GPT2-base and GPT2-large, and the obtained results are presented in Table 12. Figure 5(a) and 5(b) illustrate the impact of prompt length on the performance of NLG models with different sizes, evaluated on the SAMSum dataset. The results indicate that the performance of the models consistently improves until the prompt length reaches 50. Further increasing the prompt length does not result in significant improvements. Furthermore, we observe that our proposed instance-aware prompt learning method outperforms the task-specific prompt learning method across different prompt lengths, indicating the effectiveness of our approach.

4.2 Visualization of Instance-aware Prompt

In Table 6, we select similar and dissimilar cases from WSC [Levesque et al. 2012] and analyze them using IPL. Figure 7 displays the analysis results for IPL on both sets of cases, with the attention matrix between the case and prompt visualized in the figure. Figure 7(a) and Figure 7(b) show that the attention matrices between similar cases are similar. This result indicates that IPL can generate comparable prompts for similar cases. In contrast, when comparing the attention matrices between dissimilar cases (Figure 7(a) and Figure 7(c), or Figure 7(b) and Figure 7(c)), we observe that the matrices are dissimilar. This finding suggests that IPL can generate distinct



(a) Prompt length (GPT2-base)

(b) Prompt length (GPT2-large)

Fig. 5. The performance on the test set of SAMSum varies as the prompt length changes. Specifically, the prompt length can take on values of 0, 5, 10, 20, 50, or 100.

Instance 1	{"text": "Billy cried because Toby wouldn't share his toy.", "target": {"span2_index": 6, "span1_index": 0, "span1_text": "Billy", "span2_text": "his"}, "idx": 50, "label": false}
Instance 2	{"text": "Bill passed the gameboy to John because his turn was over.", "target": {"span2_index": 7, "span1_index": 5, "span1_text": "John", "span2_text": "his"}, "idx": 202, "label": false}
Instance 3	{"text": "Carol believed that Rebecca regretted that she had stolen the watch.", "target": {"span2_index": 6, "span1_index": 3, "span1_text": "Rebecca", "span2_text": "she"}, "idx": 57, "label": true}

Fig. 6. The instances are chosen from WSC dataset in SuperGLUE. Red represents similar examples, while green represents dissimilar ones.

prompts for dissimilar cases. As a result, our approach learns a unique prompt for each instance and can identify the critical information within that instance.

4.3 Prompt Investigation

We show the average Euclidean distance between the continuous prompt and instances for IPL and Prefix tuning, where the instances are from the validation set of five tasks on SuperGLUE. As shown in Figure 8, the dynamic prompts learned by IPL are closer to the instances and can acquire more knowledge from the instances. In Figure 9, we show two case studies. For indistinguishable instances, PET utilizes a fixed discrete prompt and makes a wrong judgment on the meaning of the word ‘put’ and ‘department’. Prefix tuning prepends the fixed continuous prompt with the two instances also gives wrong answers. In contrast, our method IPL learns a unique prompt for each instance and contains much information of the instance yielding the correct answer.

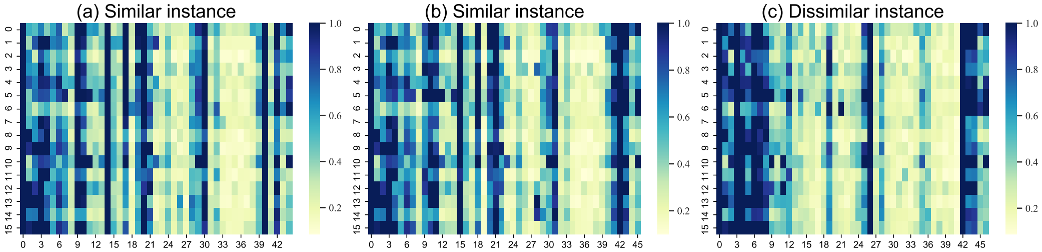


Fig. 7. Attention visualization of different instances. (a) and (b) are similar instances, (a) and (c) or (b) and (c) are dissimilar instances. The X-axis represents the input sequence without prompt, and Y-axis represents the prompt sequence.

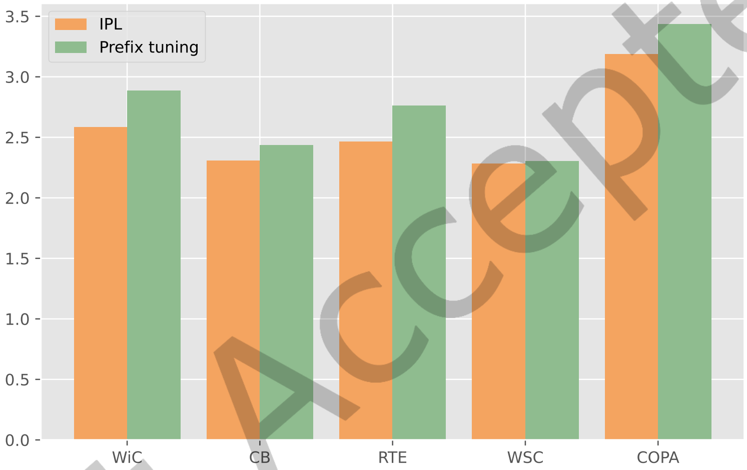
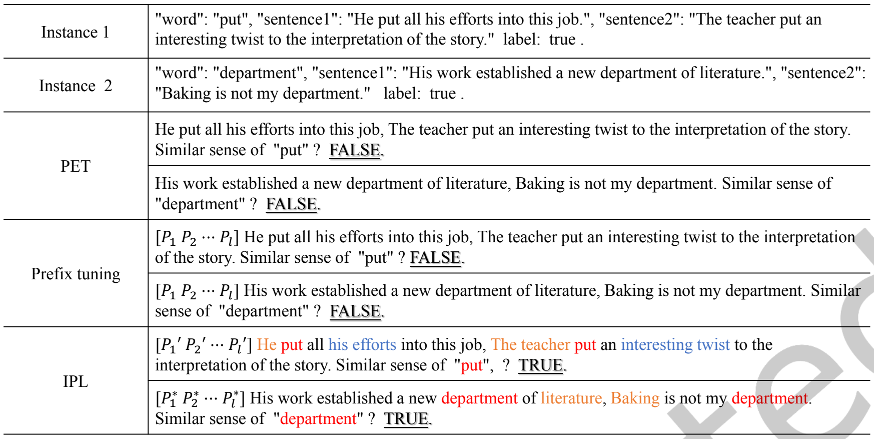


Fig. 8. The average Euclidean distance between the continuous prompt and instances for IPL and Prefix tuning. The instances are from the validation set of five tasks on SuperGLUE.

5 RELATED WORK

GPT-3 [Brown et al. 2020], which uses the task description and several typical examples as prompt to guide the generation, indicates the language models are few-shot learners and leads to the waves of prompt learning. Recently, PET/iPET [Schick and Schütze 2021b] utilizes the manually-designed prompts to reformulate natural language understanding tasks as cloze-style questions with gradient-based fine-tuning. There are also a lot of studies that utilize the manually-designed prompt to mine the knowledge from the PLMs [Jiang et al. 2020; Trinh and Le 2018]. Since manual-designed prompt is time-consuming and the search space is huge, researchers focus on automatic prompt search [Gao et al. 2021; Shin et al. 2020; Zhong et al. 2021].

However, handcrafted prompts can only reflect human perspectives on rationality, leading to a surge in research into continuous prompt learning methods. Li and Liang [2021] proposes prefix tuning, which concatenates learnable prefix vectors at each layer of the Transformer and only optimizes the prefix parameters. In contrast,



Instance 1	"word": "put", "sentence1": "He put all his efforts into this job.", "sentence2": "The teacher put an interesting twist to the interpretation of the story." label: true .
Instance 2	"word": "department", "sentence1": "His work established a new department of literature.", "sentence2": "Baking is not my department." label: true .
PET	He put all his efforts into this job, The teacher put an interesting twist to the interpretation of the story. Similar sense of "put" ? FALSE.
	His work established a new department of literature, Baking is not my department. Similar sense of "department" ? FALSE.
Prefix tuning	$[P_1\ P_2 \cdots P_l]$ He put all his efforts into this job, The teacher put an interesting twist to the interpretation of the story. Similar sense of "put" ? FALSE.
	$[P_1\ P_2 \cdots P_l]$ His work established a new department of literature, Baking is not my department. Similar sense of "department" ? FALSE.
IPL	$[P_1'\ P_2' \cdots P_l']$ He put all his efforts into this job, The teacher put an interesting twist to the interpretation of the story. Similar sense of "put", ? TRUE.
	$[P_1^*\ P_2^* \cdots P_l^*]$ His work established a new department of literature, Baking is not my department. Similar sense of "department" ? TRUE.

Fig. 9. The instances are chosen from WiC dataset in SuperGLUE. The manually-designed patterns are used from PET. The colored words indicate that our approach is aware of the critical information in the instance through the attention matrix.

prompt tuning [Lester et al. 2021] concatenates learnable prompt only in the embedding layer and optimizes the prompt parameters in the embedding layer. Although Lester et al. [2021] demonstrate the effectiveness of light-weight prompt-tuning, the gap with fully parameter fine-tuning still exists especially when the PLM is small.

There are also a lot of works that interleave the prompt throughout the input layer. Hambardzumyan et al. [2021] propose WARP, initializing the prompt parameters either with word embeddings of [MASK] or similar to the vectors from the word embedding layer. Their work is based on a series of masked language models [Delobelle et al. 2020; Lan et al. 2020] and uses a learnable output layer to project the mask to class logits, which restricts the model and only produces a single output. Liu et al. [2021b] propose P-tuning and use the patterns based on human design and replace unimportant words with continuous prompts in the embedding layer. When optimizing the model, P-tuning jointly updates both the prompt and model parameters.

However, the above usage of the discrete and continuous prompts assumes that the prompt is fixed for a specific task and all samples in the task share the same prompt. Different from the previous method, our method IPL takes each learnable prompt token as a query and calculates its contribution to each instance through the look-up module, and then learns a unique prompt for each instance.

Very recently, several contemporaneous works present other instance-dependent prompt approaches. Gu et al. [2021] propose the dialog prompt and learn continuous prompt embeddings optimized for dialogue contexts. Wu et al. [2022] study only masked language model on only NLU tasks with two forward passes of the PLMs during inference time. In contrast, our IPL model is simple and effective for both unidirectional and bidirectional PLMs on both NLU and NLG tasks without increasing the inference time.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose an instance-aware prompt learning method named IPL, which learns a unique prompt for each instance. We find that IPL has the potential to be applied to both unidirectional and bidirectional PLMs on both language understanding and generation tasks. In the few-shot learning SuperGLUE benchmark, IPL outperforms all other methods and obtains the new state-of-the-art using ALBERT-xxlarge-v2. The detailed analysis demonstrates that our IPL model can indeed dynamically learn appropriate prompts for various instances.

In the future, we would explore how to learn prompts with both instance-aware and task-specific information in few-shot learning scenarios.

7 ACKNOWLEDGEMENT

This work is supported by National Key R&D Program of China 2022ZD0160602 and the Natural Science Foundation of China 62122088.

REFERENCES

Roy Bar-Haim, Ido Dagan, and Idan Szpektor. 2014. Benchmarking Applied Semantic Inference: The PASCAL Recognising Textual Entailment Challenges. In *Language, Culture, Computation. Computing - Theory and Technology - Essays Dedicated to Yaacov Choueka on the Occasion of His 75th Birthday, Part I (Lecture Notes in Computer Science, Vol. 8001)*, Nachum Dershowitz and Ephraim Nissan (Eds.). Springer, 409–424. https://doi.org/10.1007/978-3-642-45321-2_19

Anja Belz and Ehud Reiter. 2006. Comparing Automatic and Human Evaluation of NLG Systems. In *EACL*.

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 *NeurIPS*. https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

Yulong Chen, Yang Liu, Liang Chen, and Yue Zhang. 2021. DialogSum: A Real-Life Scenario Dialogue Summarization Dataset. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Association for Computational Linguistics, Online, 5062–5074. https://doi.org/10.18653/v1/2021.findings-acl.449

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. PaLM: Scaling Language Modeling with Pathways. *CoRR* abs/2204.02311 (2022). https://doi.org/10.48550/arXiv.2204.02311 arXiv:2204.02311

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. 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)*. Association for Computational Linguistics, Minneapolis, Minnesota, 2924–2936. https://doi.org/10.18653/v1/N19-1300

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The CommitmentBank: Investigating projection in naturally occurring discourse. *Proceedings of Sinn und Bedeutung* 23, 2 (Jul. 2019), 107–124. https://doi.org/10.18148/sub/2019.v23i2.601

Pieter Delobelle, Thomas Winters, and Bettina Berendt. 2020. RobBERT: a Dutch RoBERTa-based Language Model. In *Findings of EMNLP*. https://doi.org/10.18653/v1/2020.findings-emnlp.292

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *ArXiv* abs/1810.04805 (2019).

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making Pre-trained Language Models Better Few-shot Learners. In *ACL*. https://doi.org/10.18653/v1/2021.acl-long.295

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG Challenge: Generating Text from RDF Data. In *ICNLG*. https://doi.org/10.18653/v1/W17-3518

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. SAMSum Corpus: A Human-annotated Dialogue Dataset for Abstractive Summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*. https://doi.org/10.18653/v1/D19-5409

Xiaodong Gu, Kang Min Yoo, and Sang-Woo Lee. 2021. Response Generation with Context-Aware Prompt Learning. *CoRR* abs/2111.02643 (2021). arXiv:2111.02643 https://arxiv.org/abs/2111.02643

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. WARP: Word-level Adversarial ReProgramming. In *ACL*. https://doi.org/10.18653/v1/2021.acl-long.381

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 *ICML*.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How Can We Know What Language Models Know? *TACL* (2020). https://doi.org/10.1162/tacl_a_00324

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking Beyond the Surface: A Challenge Set for Reading Comprehension over Multiple Sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, 252–262. https://doi.org/10.18653/v1/N18-1023

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *ICLR*. https://openreview.net/forum?id=H1eA7AEtvS

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*. https://aclanthology.org/W07-0734

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning. In *EMNLP*. https://doi.org/10.18653/v1/2021.emnlp-main.243

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *International Conference on the Principles of Knowledge Representation and Reasoning*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *ACL*. https://doi.org/10.18653/v1/2021.acl-long.353

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *ACL*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021a. P-Tuning v2: Prompt Tuning Can Be Comparable to Fine-tuning Universally Across Scales and Tasks. *CoRR* abs/2110.07602 (2021). arXiv:2110.07602 https://arxiv.org/abs/2110.07602

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. GPT Understands, Too. *ArXiv* abs/2103.10385 (2021).

Jekaterina Novikova, Ondrej Dusek, and Verena Rieser. 2017. The E2E Dataset: New Challenges For End-to-End Generation. In *SIGDIAL*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *ACL*.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the Word-in-Context Dataset for Evaluating Context-Sensitive Meaning Representations. 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)*. Association for Computational Linguistics, Minneapolis, Minnesota, 1267–1273. https://doi.org/10.18653/v1/N19-1128

Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chia-Hsuan Hsieh, Nazneen Rajani, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, Yangxiaokang Liu, Nadia Irwanto, Jessica Pan, Faiaz Rahman, Ahmad Zaidi, Murori Mutuma, Yasin Tarabar, Ankit Gupta, Tao Yu, Yi Chern Tan, Xi Victoria Lin, Caiming Xiong, and Richard Socher. 2021. DART: Open-Domain Structured Data Record to Text Generation. In *NAACL-HLT*.

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 (2019).

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. 2011. Choice of Plausible Alternatives: An Evaluation of Commonsense Causal Reasoning. In *Logical Formalizations of Commonsense Reasoning, Papers from the 2011 AAAI Spring Symposium, Technical Report SS-11-06, Stanford, California, USA, March 21-23, 2011*. AAAI. http://www.aaai.org/ocs/index.php/SSS/SSS11/paper/view/2418

Timo Schick and Hinrich Schütze. 2021a. Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference. In *EACL*. https://doi.org/10.18653/v1/2021.eacl-main.20

Timo Schick and Hinrich Schütze. 2021b. It's Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners. In *NAACL-HLT*. https://doi.org/10.18653/v1/2021.naacl-main.185

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *EMNLP*.

Matthew Snover, Bonnie Dorr, Rich Schwartz, Linnea Micciulla, and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*. https://aclanthology.org/2006.amta-papers.25

Derek Tam, Rakesh R. Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. 2021. Improving and Simplifying Pattern Exploiting Training. In *EMNLP*. https://doi.org/10.18653/v1/2021.emnlp-main.407

Trieu H. Trinh and Quoc V. Le. 2018. A Simple Method for Commonsense Reasoning. *ArXiv* abs/1806.02847 (2018).

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. CIDEr: Consensus-based image description evaluation. In *CVPR*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *NeurIPS*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2020. Transformers: State-of-the-Art Natural Language Processing. In *EMNLP*.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Rui Hou, Yuxiao Dong, V. G. Vinod Vydiswaran, and Hao Ma. 2022. IDPG: An Instance-Dependent Prompt Generation Method. *CoRR* abs/2204.04497 (2022). https://doi.org/10.48550/arXiv.2204.04497 arXiv:2204.04497

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. ReCoRD: Bridging the Gap between Human and Machine Commonsense Reading Comprehension. *CoRR* abs/1810.12885 (2018). arXiv:1810.12885 http://arxiv.org/abs/1810.12885

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating Text Generation with BERT. In *ICLR*. https://openreview.net/forum?id=SkeHuCVFDr

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual Probing Is [MASK]: Learning vs. Learning to Recall. In *NAACL-HLT*. https://doi.org/10.18653/v1/2021.naacl-main.398