On Transferability of Prompt Tuning for Natural Language Processing

Anonymous ACL submission

Abstract

Prompt tuning (PT) is a promising parameter-efficient method to utilize extremely large pre-trained language models (PLMs), which can achieve comparable performance to full-parameter fine-tuning by only tuning a few soft prompts. However, PT requires much more training time than fine-tuning. Intuitively, knowledge transfer can help to improve the efficiency. To explore whether we can improve PT via prompt transfer, we empirically investigate the transferability of soft prompts across different downstream tasks and PLMs in this work. We find that (1) in zero-shot setting, trained soft prompts can effectively transfer to similar tasks on the same PLM and also to other PLMs with a cross-model projector trained on similar tasks; (2) when used as initialization, trained soft prompts of similar tasks and projected prompts of other PLMs can significantly accelerate training and also improve the performance of PT. Moreover, to explore what decides prompt transferability, we investigate various transferability indicators and find that the overlapping rate of activated neurons strongly reflects the transferability, which suggests how the prompts stimulate PLMs is essential. Our findings show that prompt transfer is promising for improving PT, and further research shall focus more on prompts’ stimulation to PLMs. The source code will be publicly released.

1 Introduction

Pre-trained language models (PLMs), such as BERT (Devlin et al., 2019) and GPT (Radford et al., 2018) have achieved great performance on various natural language processing (NLP) tasks (Han et al., 2021). Recently, after the success of GPT-3 (Brown et al., 2020), people have found that extremely large PLMs can achieve remarkable improvements, and various large PLMs are continually developed (Raffel et al., 2020; Zhang et al., 2021; Zeng et al., 2021; Wei et al., 2021; Sun et al., 2021), which contain up to hundreds of billions of parameters.

Considering the extremely large scale of these state-of-the-art PLMs, conventional full-parameter fine-tuning methods become extremely expensive. Hence, various parameter-efficient tuning methods (Houlsby et al., 2019; Ben Zaken et al., 2021; Lester et al., 2021; Li and Liang, 2021; Liu et al., 2021) are explored, among which prompt tuning (PT) has attracted broad research attention. PT prepends some soft prompts, which are essentially learnable virtual tokens, into the input sequences and only train them while keeping all the PLM’s parameters fixed. The training objective is to generate desired outputs in the same way as the pre-training tasks. PT can match the downstream task performance of fine-tuning with only thousands of tunable parameters (Lester et al., 2021) when the PLM has billions of parameters.

Although PT is an effective approach to utilize extremely large PLMs, it requires much more training time than fine-tuning to reach the convergence as shown in Figure 2; hence, it is worthwhile to explore how to improve the efficiency of PT. In this work, we attempt to improve PT via prompt transfer across different tasks and models. Knowledge
transfer across tasks (Vu et al., 2020) and models (Qin et al., 2021) have been widely used to improve the efficiency and effectiveness of NLP systems. Intuitively, soft prompts are the only tuned parameters in PT and thus shall concentrate the knowledge required to solve tasks conditioned on PLMs. Hence only transferring the trained prompts is promising to accelerate PT.

As shown in Figure 1, we empirically analyze the transferability of prompts across different tasks (cross-task transfer setting) and PLMs (cross-model transfer setting) in this paper. The empirical analysis is conducted on 17 NLP tasks of 6 types and two representative PLM series: RoBERTa (Liu et al., 2019b) and T5 (Raffel et al., 2020). In cross-task transfer, the prompt transfer can be done by directly reusing the trained prompts of the source task on the target task. However, in cross-model transfer, directly reusing prompts is intractable since the semantic spaces of different PLMs are inconsistent; hence, we develop various prompt projectors to project the soft prompts trained on the source PLM to the semantic space of the target PLM. We conduct two lines of experiments: (1) We investigate the zero-shot transfer performance and find that the transferability of prompts is influenced by task types. In cross-task transfer, the soft prompts can directly transfer to same-type tasks and achieve non-trivial performance, but poorly transfer to different-type tasks requiring different language skills. In cross-model transfer, we can successfully train a prompt projector with PT on a task, but the trained projector also only well generalizes to the same-type tasks of the projector-training task. (2) To accelerate PT, we propose to transfer prompts with initialization. In cross-task transfer, we start PT with the trained soft prompts of similar tasks as initialization. While in cross-model transfer, the initialization is the projected prompts of the same task trained on the source PLM. The two methods are dubbed as TPT\textsubscript{TASK} and TPT\textsubscript{MODEL}, respectively. Experiments show that they can both significantly accelerate PT and also achieve a certain performance improvement.

Furthermore, we explore why can the prompts transfer and what decides their transferability. To this end, we design various prompt similarity metrics from different perspectives and examine how well they can serve as transferability indicators, i.e., how well they correlate with prompt transfer performance. Experiments find that the embedding distances of prompts do not well indicate prompt transferability but the overlapping rate of the prompts’ activated neurons in the feed-forward layers can better reflect prompt transferability. This suggests the prompts are essentially stimulating PLM’s inner ability distributing among neurons to do specific NLP tasks, and future prompt transfer works should focus more on how the PLMs respond to different prompts’ stimulation rather than the prompts’ embedding properties.

To summarize, our contributions are three-fold: (1) We thoroughly analyze the transferability of prompts across different tasks and models, and show that improving PT with prompt transfer is possible and promising. (2) We propose to transfer prompts with initialization, which enhances both PT’s efficiency and effectiveness. (3) We explore the effectiveness of various prompt similarity metrics serving as transferability indicators and demonstrate how the prompts stimulate PLMs to decide the transferability, which may facilitate further transferrable PT research.

2 Related Work

Prompt Tuning GPT-3 (Brown et al., 2020) demonstrates remarkable few-shot performance by prepending textual prompts before the inputs and thus help the PLM to generate desired outputs of NLP tasks directly. Motivated by this, many works have tried to improve various NLP tasks by creating manually-crafted (Schick and Schütze, 2021a,b; Mishra et al., 2021) or automatically-searched (Jiang et al., 2020; Shin et al., 2020; Gao et al., 2021) hard prompts, which are discrete tokens but not necessarily human-readable. Furthermore, soft prompts (Li and Liang, 2021; Hambardzumyan et al., 2021; Zhong et al., 2021; Liu et al., 2021) are proposed, which are tuneable embeddings rather than tokens in the vocabularies and can be directly trained with task-specific supervi-

![Figure 2: Validation accuracies against training time of fine-tuning and PT for RoBERTa\textsubscript{LARGE} on MNLI. PT takes much more training time.](image-url)
Knowledge Transfer Cross-task knowledge transfer (Ruder, 2017) has been a long-standing way to improve the effectiveness and efficiency of NLP systems. In the PLM era, some works propose to tune the PLMs on intermediate tasks (Phang et al., 2018; Pruksachatkun et al., 2020; Gururangan et al., 2020; Wang et al., 2019a; Vu et al., 2020; Poth et al., 2021) before fine-tuning on specific target tasks to achieve certain benefits. Vu et al. (2020) empirically analyze the transferability between tasks in this setting.

These explorations are all for fine-tuning. Considering the potential of PT, we believe the transferability and knowledge transfer methods for PT are worth exploring. As a prior attempt, Lester et al. (2021) demonstrate that PT’s cross-domain transferability is stronger than fine-tuning. Similar to our work, concurrent work (Vu et al., 2021) explores the cross-task transferability of PT and improves performance with transfer initialization. Differently, we attempt to improve the efficiency of PT and further analyze what decides the prompt transferability by exploring various transferability indicators. Additionally, we also attempt cross-model transfer, which is inspired by previous cross-model knowledge transfer works such as Net2Net (Chen et al., 2016), knowledge distillation (Hinton et al., 2015) and knowledge inheritance (Qin et al., 2021).

3 Preliminary

Here we introduce the basic knowledge about PT (§ 3.1) as well as the downstream tasks (§ 3.2) and models (§ 3.3) investigated in experiments.

3.1 Prompt Tuning

In this work, we study the PT method that is capable of tuning large PLMs (Li and Liang, 2021; Lester et al., 2021; Liu et al., 2021), i.e., we only explore the PT method freezing PLM parameters. PT prepends some virtual tokens, i.e., the soft prompts, into the inputs of the PLM to provide knowledge about downstream tasks. The soft prompts are essentially tunable embedding vectors, which are trained with the objective enforcing the PLM to generate desired outputs of the downstream task in the same way of the pre-training objective.

Formally, given an input sequence with n tokens \( X = \{x_1, x_2, \ldots, x_n\} \), we first prepend \( l \) randomly initialized soft prompts \( P = \{p_1, p_2, \ldots, p_l\} \) before them, where \( p_i \in \mathbb{R}_d \) is an embedding vector, and \( d \) is the input dimension of the PLM. The training objective is to maximize the likelihood of decoding the desired output \( y \):

\[
L = p(y|P, x_1, \ldots, x_n),
\]

where only \( P \) is learnable. For the language understanding tasks, \( y \) is the label token corresponding to the label of \( X \). For the conditional generation tasks, \( y \) is a sequence. Especially, for the models pre-trained with the masked language modeling objective like RoBERTa, we additionally prepend a special [MASK] token before the prompts and train the prompts to let the PLM fill \( y \) into it.

3.2 Investigated NLP Tasks

To comprehensively study the prompt transferability across various NLP tasks, we involve 17 diverse tasks, which can be divided into 6 types: (1) Sentiment Analysis (SA), including IMDb (Maas et al., 2011), SST-2 (Socher et al., 2013), laptop (Pontiki et al., 2014), restaurant (Pontiki et al., 2014), Movie Rationales (Movie) (Zaidan et al., 2008) and TweetEval (Tweet) (Barbieri et al., 2020); (2) Natural Language Inference (NLI), including MNLI (Williams et al., 2018), QNLI (Wang et al., 2019b) and SNLI (Bowman et al., 2015); (3) Ethical Judgement (EJ), including deontology (Hendrycks et al., 2021) and justice (Hendrycks et al., 2021); (4) Paraphrase Identification (PI), including QQP (Sharma et al., 2019) and MRPC (Dolan and Brockett, 2005); (5) Question Answering (QA), including SQuAD (Rajpurkar et al., 2016) and NQ–Open (Lee et al., 2019); (6) Summarization (SUM), including Multi-News (Fabbri et al., 2019) and SAMSum (Gliwa et al., 2019). Details for these tasks, evaluation metrics, label tokens, implementations are in appendix A.

3.3 Investigated Models

We investigate prompt transferability for two series of PLMs: RoBERTa (Liu et al., 2019b) and T5 (Raf-
We empirically study the cross-task transferability of soft prompts (§ 4.1) and try to improve the effectiveness and efficiency of PT with transfer (§ 4.2).

### 4.1 Zero-shot Transfer Performance

To study the cross-shot transferability, we first examine PT’s zero-shot transfer performance, i.e., we conduct PT on a source task, then directly reuse the trained prompts on other target tasks and evaluate their performance. The results are shown in Figure 3(a), from which we can observe that: (1) For the tasks within the same type, transferring soft prompts between them can generally perform well and may even outperform vanilla PT on the target task, especially when the source task has more data (the case of transferring from IMDb to Movie in Figure 3 (a) and transferring from restaurant to laptop in Figure 3 (b)), which demonstrates that it is promising to improve PT’s effectiveness and efficiency with knowledge transfer from similar tasks. (2) For the tasks of different types, the transferability of soft prompts among them is generally poor, and transferring soft prompts often achieve similar performance to randomly initialized prompts. (3) However, some tasks can transfer to different-type tasks to some extent, such as the QA and SUM tasks to SA tasks in Figure 3 (b). To understand this, it is worthwhile to explore what controls the transferability between prompts, and we do some preliminary study in § 6.

### 4.2 Transfer with Initialization

To improve the effectiveness and efficiency of PT with cross-task transfer, we explore a cross-task transferable prompt tuning (TPTTASK) method, which initializes soft prompts with well-trained prompts of the most similar task and then starts PT.

For a target task, we start TPTTASK with trained prompts of the source task achieving the best zero-shot transfer performance in Figure 3. From the results of the performance and training time com-

---

1. More results on other PLMs are left in appendix B.1.
5 Cross-Model Transfer

We further study the cross-model transferability of soft prompts. We investigate the feasibility of cross-model transfer on transferring from a source PLM (RoBERTaLARGE) to a larger and heterogeneous target PLM (T5XXL), which shall be the most difficult setting. Appendix C shows the experimental results of other settings. Directly reusing trained soft prompts between different PLMs is infeasible since their embedding spaces are different. Hence, we investigate how to do cross-model prompt projection (§5.1) and see the transfer performance (§5.2). Furthermore, we explore to improve PT with cross-model transfer initialization (§5.3).

5.1 Cross-Model Prompt Projection

To project the trained soft prompts of a PLM to the semantic space of a different PLM, we train projectors with various objectives and examine their effectiveness. A good way to train the cross-model projectors may need some task-specific supervisions, but the trained projector shall generalize to different tasks so that the efficiency for learning the new tasks on the target model could be improved.

Formally, given the prompt of the source PLM \( P_s = \{p_1, \ldots, p_l\} \), we concatenate the \( l \) virtual tokens into a unified vector \( P_s \in \mathbb{R}^{ld_s} \). The projector \( \text{Proj}(\cdot) \) is to project it to \( \tilde{P}_s \in \mathbb{R}^{ld_t} \) in the semantic space of the target PLM, where \( d_s \) and \( d_t \)



\[ \tilde{P}_s = \text{Proj}(P_s) = W_2(\sigma(W_1 + b_1)) + b_2, \]

where \( W_1 \in \mathbb{R}^{d_h \times ld_s} \), \( W_2 \in \mathbb{R}^{d_h \times ld_t} \) are trainable matrices, \( b_1 \in \mathbb{R}^{d_h} \), \( b_2 \in \mathbb{R}^{d_t} \) are biases, \( \sigma \) is a non-linear activation function. We investigate two learning objectives to train the projector\(^3\):

- **Distance Minimizing** We firstly try to learn cross-model projectors by minimizing the distance between the projected prompt and the parallel prompt \( P^s \) originally trained on the target PLM with the same task, i.e., the training objective is to minimize their \( L_2 \)-distance \( \|\text{Proj}(P_s) − P^t\|_2 \).

- **Task Tuning** We then try to train the cross-model projector with task-specific supervision signals on the target PLM. Specifically, we directly tune the projector-learning methods are shown in Table 2\(^4\) (a). We can observe that: (1) Distance Minimizing works well to transfer the prompts of the projector-training task, but falls back to random performance on the other unseen tasks, which is not practically

\[ \text{Table 1: Performance on 17 NLP tasks of vanilla prompt tuning (PT) and prompt tuning with transferring initialization (TPT TASK) as well as the convergence speedup (the quotient of the training steps of PT by the training time of TPT TASK achieving comparable performance to PT). N/A represents the tasks that RoBERTaLARGE cannot conduct, or we fail to speed up training with TPT TASK.} \]

| Task Type | SA | NLI | EJ | PI | QA | SUM |
|-----------|----|-----|----|----|----|-----|
| Task      | IMDB, SST-2, laptop, restaurant, Tweet | MNLI, QNLI, SNLI | Ontology, juried | QQP, MRPC, QuAD | NQ, Open, Multi-News | SAMSsim |
| Metric    | Acc | Acc | Acc | Acc | Acc | Acc |
| Performance (PT) (%)      | 319 | 318 | 317 | 316 | 315 | 314 |
| Performance (TPT TASK) (%) | 92.2 | 96.1 | 76.4 | 83.7 | 84.9 | 76.1 |
| Convergence Speedup       | 1.7 | 1.1 | 1.0 | 1.9 | 1.2 | 0.9 |
| Comparable-result Speedup | 2.5 | 2.1 | 1.0 | 3.8 | 1.5 | 1.3 |

\(^2\)Training time comparisons are left in appendix B.3.
Table 2: Cross-model prompt transfer (RoBERTa\textsubscript{LARGE} to T5\textsubscript{XXL}) results, including non-transfer baselines (vanilla PT and randomly generated prompts), zero-shot transfer performance of various projectors, and TPT\textsubscript{MODEL} results (performance, convergence speedup, and comparable-result speedup similar to Table 1).

6 Exploring Transferability Indicator

Based on the positive results in cross-task and cross-model transfer, we explore why the soft prompts can transfer across tasks and what decides the transferability between them, which may shed light on the mechanisms behind PT and help to design transferable PT methods. We explore various prompt similarity metrics and examine how well do they align with the zero-shot transfer performance. If a similarity metric can well indicate transferability, it suggests the factors considered in designing this metric decide the prompt transferability. Moreover, the prompt similarity metrics can qualify task similarities used in the trained soft prompts as task embeddings and may help in developing cross-task transfer methods. As a straightforward example, if we build a prompt warehouse containing prompts of diverse tasks, we can retrieve prompts of similar tasks for a new task with a certain similarity metric and better improve PT with TPT\textsubscript{TASK}.

6.1 Prompt Similarity Metric

We explore the following two kinds of metrics:

Embedding Similarity We firstly regard the trained soft prompts as only embeddings in the vector space and calculate their Euclidean similarity and cosine similarity.

Given two groups of trained prompts containing $l$ virtual tokens: $P^{t_1} = \{p_{t_1}^1, \ldots, p_{t_1}^l\}$ and $P^{t_2} = \{p_{t_2}^1, \ldots, p_{t_2}^l\}$, which correspond to tasks $t_1$ and $t_2$. Firstly, we concatenate the $l$ virtual tokens for each group and get two concatenation em-
beddings \( \mathbf{P}^{t_1}, \mathbf{P}^{t_2} \in \mathbb{R}^{d_l} \), then we compute Euclidean similarity and cosine similarity of them:

\[
\begin{align*}
E_{\text{concat}}(\mathbf{P}^{t_1}, \mathbf{P}^{t_2}) &= \frac{1}{1 + \|\mathbf{P}^{t_1} - \mathbf{P}^{t_2}\|}, \\
C_{\text{concat}}(\mathbf{P}^{t_1}, \mathbf{P}^{t_2}) &= \frac{\mathbf{P}^{t_1} \cdot \mathbf{P}^{t_2}}{\|\mathbf{P}^{t_1}\| \|\mathbf{P}^{t_2}\|}.
\end{align*}
\]

We further explore a simple way to make the metrics invariant to token positions. We compute Euclidean distances and cosine similarities for every virtual token pairs in the two groups and use the averaged results in the final similarity metrics:

\[
\begin{align*}
E_{\text{average}}(\mathbf{P}^{t_1}, \mathbf{P}^{t_2}) &= \frac{1}{1 + \frac{1}{l^2} \sum_{i=1}^{l} \sum_{j=1}^{l} \|\mathbf{p}^{t_1}_i - \mathbf{p}^{t_2}_j\|}, \\
C_{\text{average}}(\mathbf{P}^{t_1}, \mathbf{P}^{t_2}) &= \frac{1}{l^2} \sum_{i=1}^{l} \sum_{j=1}^{l} \|\mathbf{p}^{t_1}_i \cdot \mathbf{p}^{t_2}_j\| \|\mathbf{p}^{t_2}_j\|.
\end{align*}
\]

**Model Stimulation Similarity** In the second way, we depict their similarities based on how they stimulate the PLMs, i.e., we examine the similarities between the responses of PLMs to the two soft prompts. Motivated by Geva et al. (2021) and Dai et al. (2021), which both find that the activation of the neurons in the feed-forward layers of Transformers (Vaswani et al., 2017) corresponds to specific model behaviors, we propose to use the overlapping rate of activated neurons as a similarity metric of prompts. Specifically, the feed-forward network FFN(⋅) in a Transformer layer is:

\[
\text{FFN}(\mathbf{x}) = \max(\mathbf{x} \mathbf{W}^T_1 + \mathbf{b}_1, 0) \mathbf{W}_2 + \mathbf{b}_2,
\]

where \( \mathbf{x} \in \mathbb{R}^d \) is the input embedding, \( \mathbf{W}_1, \mathbf{W}_2 \in \mathbb{R}^{d \times d} \) are trainable matrices, and \( \mathbf{b}_1, \mathbf{b}_2 \) are bias vectors. The \( \max(\mathbf{x} \mathbf{W}^T_1 + \mathbf{b}_1, 0) \) can be regarded as the non-negative activation values for \( d_m \) hidden neurons (Geva et al., 2021). We then change all the positive elements of \( \max(\mathbf{x} \mathbf{W}^T_1 + \mathbf{b}_1, 0) \) to 1 and get the one-hot activation state vector \( \mathbf{s} \).

We feed an input sequence \( \{P, <s>\} \) into the PLMs, where \( <s> \) is the special token indicating the start of a sentence. For RoBERTa, a \([\text{MASK}]\) is additional prepended. This sequence is in the format of PT inputs but without specific input sentences. We use the activation states of the positions used to decode outputs, which shall be more task-specific. Specifically, for T5, we use the decoder module’s activation states at the first position. For RoBERTa, we use the activation states of \([\text{MASK}]\). Finally, we concatenate the activation states of PLM’s \( L \) layers to get the overall activation states:

\[
\text{AS}(P) = [s_1; s_2; \ldots; s_L].
\]

We can only retrieve the activation states of a part of layers in the similarity computation. In experiments, we find that the higher layers tend to be more task-specific, which is consistent with the probing results (Liu et al., 2019a). Hence we use the activation states of the top 3 layers\(^5\) in experiments below. We calculate the overlapping rate of activated neurons \( \text{ON}(\mathbf{P}^{t_1}, \mathbf{P}^{t_2}) \) between the trained soft prompts of task \( t_1 \) and \( t_2 \) with the cosine similarity:

\[
\text{ON}(\mathbf{P}^{t_1}, \mathbf{P}^{t_2}) = \frac{\text{AS}(\mathbf{P}^{t_1}) \cdot \text{AS}(\mathbf{P}^{t_2})}{\|\text{AS}(\mathbf{P}^{t_1})\| \|\text{AS}(\mathbf{P}^{t_2})\|}.
\]

\[\text{Table 4: The Spearman's rank correlation scores (%)}\]

| Model            | Metric | Same Task | Different Tasks |
|------------------|--------|-----------|-----------------|
| RoBERTa\_LARGE  | E\_concat | 9.4       | 6.8             |
|                  | E\_average | 41.6      | 37.6            |
|                  | C\_concat | 47.6      | 31.7            |
|                  | C\_average | 1.7       | 1.1             |
|                  | ON       | 39.4      | 21.4            |
| T5\_XXL         | E\_concat | 0.5       | 0.2             |
|                  | E\_average | 4.0       | 3.4             |
|                  | C\_concat | 29.4      | 3.4             |
|                  | C\_average | 4.0       | 2.1             |
|                  | ON       | 62.0      | 46.1            |

\[\text{Table 3: The average values (%)}\]

| Model            | Metric | RoBERTa\_LARGE | T5\_XXL |
|------------------|--------|----------------|---------|
|                  | E\_concat | 22.6        | 12.9    |
|                  | E\_average | 2.8         | -2.5    |
|                  | C\_concat | 24.8        | 31.6    |
|                  | C\_average | 44.7        | 33.5    |
|                  | ON       | 49.7        | 36.9    |

\[\text{Table 4: The Spearman’s rank correlation scores (%)}\]

\[\text{between various similarity metrics and cross-task zero-shot transfer performance of soft prompts.}\]

\[\text{5More results about the different layers’s performance are left in appendix D.4.}\]

\[\text{6.2 Experimental Results}\]

To evaluate the effectiveness of the above similarity metrics of soft prompts, we (i) test whether the similarity metrics can distinguish the trained prompts of the same tasks and different tasks, and (ii) examine whether these metrics align with the zero-shot transfer performance.
Regarding (i), we compare the similarities of the investigated metrics for two trained prompts within the same task (trained with different random seeds) and between different tasks in Table 3. From the results, we can observe that all the metrics work well to distinguish the prompts of the same task and different tasks. This suggests that the trained soft prompts of different tasks form distinguishable clusters in the embedding space and also stimulate different abilities within the PLM.

Moreover, to evaluate (ii), how well the similarity metrics align with the cross-task transfer performance, we quantify the correlations between the similarities and zero-shot transfer performance in Figure 3. Specifically, for each target task’s prompt, we rank various source tasks’ prompts with similarity scores and zero-shot transfer performance and then compute the Spearman’s rank correlation (Spearman, 1987) between the two ranks generated by these two ways. The overall results are shown in Table 4⁶. We can see that: (1) The overlapping rate of activated neurons (ON) metric works better than all the embedding similarities, which suggests that model stimulation is more important for prompt transferability than embedding distances. (2) ON works much worse on T5XXL (11B parameters) than on RoBERTaLARGE (330M parameters). We guess this is because larger PLMs have higher redundancy (Aghajanyan et al., 2021), which means prompts can activate different redundant neurons to do similar jobs and thus influence the sensitivity of ON metric. This is supported by the experiments showing that the Spearman’s correlation scores of ON drop with the increase of PLM scales (Figure 4). We encourage future work to explore how to overcome the PLM redundancy for better transferrable PT. As a preliminary trial, we find that by taking the intersection of activation states of 3 prompts trained with different random seeds, ON’s correlation score on T5XXL raises from 36.9% to 46.3%.

We further explore whether the prompt similarity metrics also work in the cross-model transfer setting by testing whether they work between the projected prompts and original prompts of the same task. In Table 5, we show the similarities of prompts projected with Task Tuning projectors by the two best metrics \( C_{\text{average}} \) and ON. We can see: (1) ON metric shows that the projected prompts are highly similar to the original prompts within the same type of projector-training tasks but are not so similar to different-type tasks, which is quite consistent with the cross-model zero-shot transfer performance in Table 2. (2) However, \( C_{\text{average}} \) cannot reflect this phenomena, which shows that the perspective of model stimulation is more promising for understanding transferability again.

### 7 Conclusion

We empirically investigate the transferability of prompts in this paper. In the cross-task setting, we find that soft prompts can transfer to similar tasks without training. In the cross-model setting, we successfully project prompts into the space of other PLMs. Further, we utilize trained prompts of other tasks or other PLMs as initialization to significantly accelerate training and improve effectiveness. Moreover, we explore various prompt transferability indicators and show that how the prompts stimulate PLMs are important to transferability. We hope the empirical analyses and the model stimulation idea can facilitate further research on transferrable and efficient PT.

---

⁶The detailed results by task types are left in appendix D.2.

### Table 5: Similarities (%) between the prompts projected with Task Tuning projector and the original prompts trained on T5XXL.

| Projector | Task         | \( C_{\text{average}} \) | ON |
|-----------|--------------|---------------------------|-----|
| Task Tuning | Laptop       | 3.8                       | 52.4|
|           | Same-Type Tasks | 4.1                   | 51.0|
|           | Different-Type Tasks | 3.4          | 46.0|
|           | MNLI         | 2.7                       | 70.7|
|           | Same-Type Tasks | 2.7                   | 56.7|
|           | Different-Type Tasks | 4.1          | 53.4|
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.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In Proceedings of the Third International Workshop on Paraphrasing (IWP2005).

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1074–1084, Florence, Italy. 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.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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, pages 70–79, Hong Kong, China. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don’t stop pretraining: Adapt language models to domains and tasks. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 8342–8360, Online. Association for Computational Linguistics.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. WARP: Word-level Adversarial ReProgramming. 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 4921–4933, Online. Association for Computational Linguistics.
Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? Transactions of the Association for Computational Linguistics, 8:423–438.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 6086–6096, Florence, Italy. 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.

Xiang 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.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In Text Summarization Branches Out, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual 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), pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. RoBERTa: A robustly optimized bert pretraining approach. In Proceedings of ICLR.

Ilia Loschchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In International Conference on Learning Representations.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, Yejin Choi, and Hannaneh Hajishirzi. 2021. Reframing instructional prompts to gpt’s language. ArXiv preprint, abs/2109.07830.

Jason Phang, Thibault Févry, and Samuel R. Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. ArXiv preprint, abs/1811.01088.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Clifton Poth, Jonas Pfeiffer, Andreas Rücklé, and Iryna Gurevych. 2021. What to pre-train on? Efficient intermediate task selection. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 10585–10605, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yada Pruskachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. Intermediate-task transfer learning with pretrained language models: When and why does it work? In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 5231–5247, Online. Association for Computational Linguistics.

Yujia Qin, Yankai Lin, Jing Yi, Jiajie Zhang, Xu Han, Zhengyan Zhang, Yusheng Su, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Knowledge inheritance for pre-trained language models. ArXiv preprint, abs/2105.13880.
Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. arXiv e-prints.

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, 21:1–67.

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

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. ArXiv preprint, abs/1706.05098.

Timo Schick and Hinrich Schütze. 2021a. 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.

Timo Schick and Hinrich Schütze. 2021b. It’s not just size that matters: Small language models are also few-shot learners. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 2339–2352, Online. Association for Computational Linguistics.

Lakshay Sharma, Laura Graesser, Nikita Nangia, and Utku Evci. 2019. Natural language understanding with the quaora question pairs dataset. ArXiv preprint, abs/1907.01041.

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 Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 4222–4235, Online. Association for Computational Linguistics.

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.

Charles Spearman. 1987. The proof and measurement of association between two things. In Proceedings of The American journal of psychology, volume 100, pages 441–471.

Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Iiixiang Liu, Xiuyi Chen, Yanbin Zhao, Yuxiang Lu, Weixin Liu, Zhihua Wu, Weibao Gong, Jianzhong Liang, Zhizhou Shang, Peng Sun, Wei Liu, Xuan Ouyang, Dianhai Yu, Hao Tian, Hua Wu, and Haiyong Wang. 2021. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. ArXiv preprint, abs/2107.02137.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pages 5998–6008.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. 2021. Spot: Better frozen model adaptation through soft prompt transfer. ArXiv preprint, abs/2110.07904.

Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. 2020. Exploring and predicting transferability across NLP tasks. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 7882–7926, Online. Association for Computational Linguistics.

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R. Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, Shuning Jin, Berlin Chen, Benjamin Van Durme, Edouard Grave, Ellie Pavlick, and Samuel R. Bowman. 2019a. Can you tell me how to get past sesame street? sentence-level pretraining beyond language modeling. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4465–4476, Florence, Italy. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. ArXiv preprint, abs/2109.01652.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 1112–1122, New Orleans,
Louisiana. Association for Computational Linguistics.

Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. 2015. Empirical evaluation of rectified activations in convolutional network. ArXiv preprint, abs/1505.00853.

Omar F. Zaidan, Jason Eisner, and Christine Piatko. 2008. Machine learning with annotator rationales to reduce annotation cost. In Proceedings of NeurIPS (Workshop).

Wei Zeng, Xiaozhe Ren, Teng Su, Hui Wang, Yi Liao, Zhiwei Wang, Xin Jiang, ZhenZhang Yang, Kaisheng Wang, Xiaoda Zhang, et al. 2021. Pangu-α: Large-scale autoregressive pretrained chinese language models with auto-parallel computation. ArXiv preprint, abs/2104.12369.

Zhengyan Zhang, Yuxian Gu, Xu Han, Shengqi Chen, Chaojun Xiao, Zhenbo Sun, Yuan Yao, Fanchao Qi, Jian Guan, Pei Ke, et al. 2021. Cpm-2: Large-scale cost-effective pre-trained language models. ArXiv preprint, abs/2106.10715.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [MASK]: Learning vs. learning to recall. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5017–5033, Online. Association for Computational Linguistics.
A Basic Setup for Various Tasks

A.1 Dataset and Task

Sentiment Analysis (SA) Given a sentence, a PLM will identify the opinions in this sentence. We use IMDB (Maas et al., 2011), SST-2 (Socher et al., 2013), SemEval/laptop (Ponti-iki et al., 2014), SemEval/restaurant (Pontiki et al., 2014), Movie Rationales (Movie) (Zaidan et al., 2008), and TweetEval (Tweet) (Barbieri et al., 2020) to analyze.

Natural Language Inference (NLI) Given a premise and hypothesis pair, a PLM determines whether the hypothesis is entailed, contradict, or undetermined by the premise. We use MNLI (Williams et al., 2018), QNLI (Wang et al., 2019b), and SNLI (Bowman et al., 2015) to analyze.

Ethical Judgement (EJ) Given a sentence, a PLM judges whether it is ethically acceptable. We choose Ethics/deontology (Hendrycks et al., 2021) and Ethics/justice (Hendrycks et al., 2021) to analyze.

Paraphrase Identification (PI) Given a pair of sentences, a PLM judges whether they are semantically identical. We choose QQP (Sharma et al., 2019) and MRPC (Dolan and Brockett, 2005) to analyze.

Question Answering (QA) Given a question, a PLM answers the question. We choose SQuAD (Rajpurkar et al., 2016) and NQ-Open (Lee et al., 2019) to analyze. For SQuAD, a PLM finds the answer from the content. As for NQ-Open, a PLM directly generates the answer without the content.

Summarization (SUM) Given an article, a PLM summarizes it. We choose Multi-News (Fabbri et al., 2019), and SAMSum (Gliwa et al., 2019) to analyze.

A.2 Evaluation Metrics

For SA, NLI, EJ, and PI tasks, we choose accuracy (Acc.) as their evaluation metric in the experiments. For QA and SUM tasks, we utilize F1 and ROUGEL (Lin, 2004), respectively.

A.3 Prompt Tuning Setting

In the experiments, for all the investigated tasks, we use AdamW (Loshchilov and Hutter, 2019) as the optimizer and set the learning rate as 0.001. We set the length of soft prompts $l$ as 100. All the soft prompts are randomly initialized and optimized with Equation 1. In the inference stage, RoBERTa predicts the label tokens at the $[\text{MASK}]$ position and T5 directly uses its decoder to do generation.

A.4 Label Tokens

The used label tokens for the classification tasks (SA, NLI, EJ, PI) are shown in Table 6. For generation tasks (QA, SUM), the desired output is just the annotated answers.

| Task       | Label Tokens                                                                 |
|------------|-------------------------------------------------------------------------------|
| Sentiment Analysis (SA) | positive, negative |
| IMDB       | positive, negative               |
| SST-2      | positive, negative               |
| laptop     | positive, moderate, negative     |
| restaurant | positive, moderate, negative     |
| Movie      | positive, negative               |
| Tweet      | positive, moderate, negative     |
| MNLI       | yes, neutral, no                 |
| QNLI       | yes, no                          |
| SNLI       | yes, neutral, no                 |
| deontology | acceptable, un                   |
| justice    | acceptable, un                   |
| QQP        | true, false                      |
| MRPC       | true, false                      |

Table 6: Label tokens of classification tasks.

B Cross-Task Transfer

B.1 More Zero-shot transfer performance

In § 4.1, we report the zero-shot transfer performance (relative performance) on RoBERTa-LARGE and T5-XXL. Here, we investigate the zero-shot transfer performance on other sizes of RoBERTa and T5, which are shown in Figure 5. According to these results, we can find that the transferability of soft prompts between the tasks of different types is generally poor, which is consistent with the conclusion in § 4.1.

B.2 Unifying Label Tokens

We hypothesize that the poor transferability between different task types may result from the fact that different-type tasks usually use different label tokens, e.g., yes and no are for NLI tasks while positive and negative are for SA tasks. To verify whether this factor influences the transferability, we unify the label tokens of different tasks into the same set of numbers (1, 2, ...). and choose...
Figure 5: Relative performance (transferring zero-shot performance / original PT performance) (%) on the target tasks (columns) of the soft prompts trained on the source tasks (rows), both of which demonstrate the relative performance for zero-shot transfer of prompts of RoBERTa and T5. Colors of the tasks names indicate the task types. **Blue**: sentiment analysis (SA). **Green**: natural language inference (NLI). **Brown**: ethical judgement (EJ). **Orange**: paraphrase identification (PI). **Purple**: question answering (QA). **Gray**: summarization (SUM). **Random Prompt** of the last row means the soft prompts are randomly generated without any training.
(b) Unifying the label tokens (RoBERTaBASE)

Figure 6: To exclude the poor transferability, which may result from the fact that different-type tasks use different label tokens, we unify the label tokens of different tasks into the same set of numbers (Figure 6).

In this paper, we compute convergence speedup and comparable-result speedup as follows:

Convergence Speedup(x) = \frac{PT \text{ convergence time}}{TPT \text{ convergence time}}.

Comparable-result Speedup(x) = \frac{PT \text{ convergence time}}{\text{time of TPT achieving comparable result to } PT}.

We calculate the training loss and the evaluation score per 100 steps during the training. When the training loss stops dropping and the evaluation score stops increasing for 300 steps, we set the point as the convergence point. For the convergence speedup in Equation 8, the PT convergence time is divided by the TPT convergence time. As for the comparable-result speedup in Equation 8, the PT convergence time are divided by the time of TPT achieving comparable performance to PT.

C Cross-Model Transfer

C.1 Implementation Details of Projector

As mentioned in § 5.1, we give the prompt of the source PLM, \( P^s = \{p_1^s, \ldots, p_n^s\} \), and concatenate its \( l \) virtual tokens into a unified vector \( \mathbf{P}^s \in \mathbb{R}^{d_s} \), where \( d_s \) is the hidden size of the source PLM. To transfer \( \mathbf{P}^s \) to the target PLM whose hidden size is \( d_t \), we design a projection function \( \text{Proj}(\cdot) \) parameterized by a two-layer perceptron as follows:

\[
\mathbf{P}^s = \text{Proj}(\mathbf{P}^s) = \mathbf{W}_2(\sigma(\mathbf{P}^s\mathbf{W}_1 + \mathbf{b}_1)) + \mathbf{b}_2, \tag{9}
\]

where \( \mathbf{W}_1 \in \mathbb{R}^{d_s \times d_l}, \mathbf{W}_2 \in \mathbb{R}^{d_t \times d_h} \) are trainable matrices, \( \mathbf{b}_1 \in \mathbb{R}^{d_l}, \mathbf{b}_2 \in \mathbb{R}^{d_h} \) are biases, \( \sigma \) is a non-linear activation function. We set the inner hidden size \( d_t \) to 768. In this paper, we investigate cross-model transfer among various PLMs including BERTBASE, RoBERTaBASE, RoBERTaLARGE, T5SMALL, T5BASE, and T5XXL, whose hidden sizes are 768, 768, 1024, 512, 768, and 1024, respectively. Besides, for non-linear activation functions, we have tried tanh and LeakyReLU (Xu et al., 2015), and find their performance on various PLMs are similar. The reported results are based on the LeakyReLU activation.
C.2 More Zero-shot Transfer Performance

In § 5.2, we have introduced the zero-shot transfer performance of various projector-learning methods in the setting of transferring from RoBERTaBase to T5XXL. We explore more cross-model transfer settings here, which are transferring between various PLMs in different scales and heterogeneous frameworks, including from BERTBase to RoBERTaBase, from RoBERTaBase to RoBERTaLarge, and from T5Base to T5XXL.

Table 7 shows the experimental results. We can see the phenomena and conclusions are all consistent with § 5.2.

C.3 Technical Details of TPTModel (Transfer with Initialization)

In § 5.3, we demonstrate cross-model transferrable prompt tuning (TPTModel) can well improve performance and reduce training time.

However, when we apply TPTModel to more PLMs, we find that the projected prompts may have quite different $L_2$ norm values with the original prompts, especially for the small-scale PLMs (e.g., from BERTBase to RoBERTaBase). Specifically, we obtain the projected prompts with the trained Task Tuning projector, and find that the projected prompts are hard to optimize in some tasks as shown in Figure 7 [Without LayerNorm].

Thus, we attempt to add the layer normalization operation (Ba et al., 2016) LayerNorm into the projectors to regularize the norm of the projected prompt as follows:

$$\tilde{P}^* = \text{LayerNorm}(\text{Proj}(P^*)) \quad (10)$$

By the LayerNorm, the projected prompts can work well on TPTModel and achieve better performance and speedup as shown in Figure 7 [With LayerNorm]. Interestingly, although prompts projected by the projectors [Without LayerNorm] are hard to be trained in TPTModel, they can achieve similar zero-shot transfer performance with the prompts projected by the projectors [With LayerNorm] in Table 8.

D Transferability Indicator

D.1 Effectiveness of Similarity Metrics

We categorize all prompts into three groups: same (e.g., from BERT to RoBERTa), same-type tasks (prompts trained with different seeds on the same dataset), same-type tasks, and heterogeneous frameworks, including from RoBERTa to T5, B to RoBERTa, and T5 to RoBERTa. We obtain the projected prompts with the trained Task Tuning projector, and find that the projected prompts are hard to optimize in some tasks as shown in Figure 7 [Without LayerNorm].

Thus, we attempt to add the layer normalization operation (Ba et al., 2016) LayerNorm into the projectors to regularize the norm of the projected prompt as follows:

$$\tilde{P}^* = \text{LayerNorm}(\text{Proj}(P^*)) \quad (10)$$

By the LayerNorm, the projected prompts can work well on TPTModel and achieve better performance and speedup as shown in Figure 7 [With LayerNorm]. Interestingly, although prompts projected by the projectors [Without LayerNorm] are hard to be trained in TPTModel, they can achieve similar zero-shot transfer performance with the prompts projected by the projectors [With LayerNorm] in Table 8.
Here, we further show Spearman’s rank correlation scores grouped by the task types on more PLMs. The results are shown in Table 10 and Table 11.
Table 9: The average values (%) of the 5 similarity metrics for prompt pairs within the same task (trained with 3 different random seeds) and between different tasks (of the same type and different types) on RoBERTa\textsubscript{LARGE} and T5\textsubscript{XXL}.

| Metric | Same Tasks | Same-type Tasks | Different-type Tasks |
|--------|------------|-----------------|---------------------|
| E\textsubscript{concat} | 9.4 | 9.4 | 6.8 |
| E\textsubscript{average} | 41.6 | 41.4 | 37.6 |
| C\textsubscript{concat} | 47.6 | 45.3 | 31.7 |
| C\textsubscript{average} | 1.7 | 1.3 | 1.1 |
| ON (Bottom 3) | 42.8 | 43.3 | 39.1 |
| ON (Top 3) | 39.4 | 28.2 | 21.4 |
| ON (All 24) | 40.0 | 35.8 | 29.6 |

T5\textsubscript{XXL} (Decoder Module)

| Metric | Same Tasks | Same-type Tasks | Different-type Tasks |
|--------|------------|-----------------|---------------------|
| E\textsubscript{concat} | 0.5 | 0.5 | 0.3 |
| E\textsubscript{average} | 4.0 | 5.1 | 3.4 |
| C\textsubscript{concat} | 29.4 | 2.8 | 2.4 |
| C\textsubscript{average} | 4.0 | 2.6 | 2.1 |
| ON (Bottom 3) | 80.3 | 75.4 | 76.3 |
| ON (Top 3) | 62.0 | 52.7 | 46.1 |
| ON (All 24) | 60.8 | 54.0 | 49.2 |

Table 10: Spearman’s rank correlation scores (%) between various similarity metrics and zero-shot transfer performance of soft prompts for various scales of RoBERTa.

| Metric | SA | NLI | EJ | PI | QA | SUM | All |
|--------|----|-----|----|----|----|-----|-----|
| T5\textsubscript{BASE} (Decoder Module) | E\textsubscript{concat} | 10.1 | 19.6 | 31.3 | 53.7 | 27.3 | 38.0 | 21.9 |
| E\textsubscript{average} | -6.8 | -28.0 | 18.7 | -2.6 | 29.1 | 42.9 | 8.9 |
| C\textsubscript{concat} | 34.6 | 63.6 | 26.6 | 19.3 | -2.1 | 12.5 | 25.7 |
| C\textsubscript{average} | 64.3 | 65.1 | 30.7 | 15.7 | 27.7 | 19.2 | 37.1 |
| ON (Bottom 3) | 32.9 | 72.6 | 14.8 | 14.2 | 45.5 | 52.8 | 43.3 |
| ON (Top 3) | 50.6 | 74.8 | 51.4 | 2.6 | 60.3 | 78.8 | 52.5 |
| ON (All 24) | 44.8 | 79.7 | 44.5 | 6.3 | 59.7 | 67.9 | 50.5 |

| Metric | Same Tasks | Same-type Tasks | Different-type Tasks |
|--------|------------|-----------------|---------------------|
| E\textsubscript{concat} | 55.2 | -17.0 | 10.2 | 21.5 | 5.9 | -1.1 | 20.8 |
| E\textsubscript{average} | 53.4 | -42.3 | -10.7 | 7.5 | -27.7 | -10.8 | 9.0 |
| C\textsubscript{concat} | 57.2 | 25.2 | 35.1 | 37.0 | 30.2 | -20.5 | 28.4 |
| C\textsubscript{average} | 47.6 | 70.0 | 30.4 | 48.0 | 34.9 | 16.8 | 42.4 |
| ON (Bottom 3) | 34.7 | 29.8 | 40.8 | 16.9 | 24.2 | 72.2 | 36.0 |
| ON (Top 3) | 53.8 | 24.3 | 50.6 | 46.1 | 54.7 | 79.1 | 49.1 |
| ON (All 24) | 46.1 | 25.0 | 42.6 | 39.7 | 56.7 | 72.3 | 43.4 |

| Metric | Same Tasks | Same-type Tasks | Different-type Tasks |
|--------|------------|-----------------|---------------------|
| E\textsubscript{concat} | 40.8 | -13.4 | 19.3 | 11.4 | -4.3 | -19.5 | 12.9 |
| E\textsubscript{average} | 32.2 | -42.6 | 9.7 | -2.0 | -27.7 | -34.0 | 2.5 |
| C\textsubscript{concat} | 21.4 | 40.9 | 42.6 | 24.6 | 30.2 | 45.6 | 31.6 |
| C\textsubscript{average} | 23.3 | 44.8 | 33.3 | 29.3 | 34.9 | 49.9 | 33.5 |
| ON (Bottom 3) | 9.1 | 20.7 | 14.8 | 18.3 | 24.2 | -9.9 | 12.4 |
| ON (Top 3) | 42.7 | 33.6 | 39.1 | 30.3 | 54.7 | 11.1 | 36.9 |
| ON (All 24) | 31.0 | 23.6 | 37.7 | 34.2 | 56.7 | 15.4 | 32.0 |

| ON\textsubscript{T} (Bottom 3) | --- | --- | --- | --- | --- | --- | 25.3 |
| ON\textsubscript{T} (Top 3) | --- | --- | --- | --- | --- | --- | 46.3 |
| ON\textsubscript{T} (All 24) | --- | --- | --- | --- | --- | --- | 40.0 |

Table 11: Spearman’s rank correlation scores (%) between various similarity metrics and zero-shot transfer performance of soft prompts for various scales of RoBERTa.

D.3 PLMs’ Redundancy Influence Indicators

From Table 10, we find that the correlation between prompt transferability and prompt similarity will drop with the increase of PLM size.

We guess that this phenomena may result from PLMs’ high redundancy (Aghajanyan et al., 2021). To try to overcome this, we simultaneously utilize the prompts trained with three random seeds on the same dataset and take their intersection of activation states as the activated neurons into the similarity (ON) computation. This similarity is called ON\textsubscript{N}. By using it, the correlation score of ON can significantly raise as shown in Table 10.

D.4 Overlapping Rate of Activated Neurons in Different Layers

To further understand model stimulus in PLMs, we investigate ON in different layers of PLMs. Specifically, on RoBERTa\textsubscript{BASE}, we measure the similarity between different prompts with activation states of from 1 to 3 layers (Figure 8), from 4 to 6 layers (Figure 9), from 7 to 9 layers (Figure 10), from 10 to 12 layers (Figure 11), and all 12 layers (Figure 12), respectively.

We find that the activated neurons are common in the bottom layers but tend to be more task-specific in top layers, which is consistent with the findings of previous works (Liu et al., 2019a).
Figure 8: ON in 1 - 3 layers of RoBERTaBase.

Figure 9: ON in 4 - 6 layers of RoBERTaBase.

Figure 10: ON in 7 - 9 layers of RoBERTaBase.
### Overlapping Percentage of Activated Parameters 10 - 12 Layers

| Prompt       | IMDB(SA) | SST-2(SA) | laptop(SA) | restaurant(SA) | Movie(SA) | Tweet(SA) | MNLI(NLI) | QNLI(NLI) | SNLI(NLI) | deontology(EJ) | justice(EJ) | QQP(PI) | MRPC(PI) |
|--------------|----------|-----------|------------|----------------|-----------|-----------|-----------|-----------|-----------|----------------|-------------|----------|----------|
| Low          | 0.16     | 0.14      | 0.21       | 0.22           | 0.16      | 0.15      | 0.20      | 0.31      | 0.38      | 0.21          | 0.22        | 0.23     | 0.16     |
| High         | 0.34     | 0.27      | 0.32       | 0.34           | 0.32      | 0.26      | 0.33      | 0.36      | 0.28      | 0.28          | 0.29        | 0.33     | 0.30     |

**Figure 11:** ON in 10 - 12 layers of RoBERTaBASE.

### Overlapping Percentage of Activated Parameters (All 12 Layers)

| Prompt       | IMDB(SA) | SST-2(SA) | laptop(SA) | restaurant(SA) | Movie(SA) | Tweet(SA) | MNLI(NLI) | QNLI(NLI) | SNLI(NLI) | deontology(EJ) | justice(EJ) | QQP(PI) | MRPC(PI) |
|--------------|----------|-----------|------------|----------------|-----------|-----------|-----------|-----------|-----------|----------------|-------------|----------|----------|
| Low          | 0.33     | 0.30      | 0.32       | 0.31           | 0.31      | 0.29      | 0.34      | 0.31      | 0.34      | 0.21          | 0.24        | 0.28     | 0.33     |
| High         | 0.50     | 0.46      | 0.50       | 0.45           | 0.45      | 0.41      | 0.45      | 0.42      | 0.40      | 0.28          | 0.29        | 0.35     | 0.38     |

**Figure 12:** ON in all 12 layers of RoBERTaBASE.