Towards Zero-Shot Functional Compositionality of Language Models

Anonymous ACL submission

Abstract

Large Pre-trained Language Models (PLM) have become the most desirable starting point in the field of NLP, as they have become remarkably good at solving many individual tasks. Despite such success, in this position paper, we argue (with a touch of empirical results) that current paradigms of working with PLMs are neglecting a critical aspect of modeling human intelligence. **Functional compositionality** – the ability to compose learned tasks – has been a long-standing challenge in the field of AI (and many other fields) as it is considered one of the hallmarks of human intelligence. An illustrative example of such is cross-lingual summarization, where a bilingual person (English-French) could directly summarize an English document into French sentences *without* having to translate the English document or summary into French explicitly. We discuss why this matter is an important open problem that requires further attention from the field. Then, through various experiments on composite tasks, we show how far we are currently from attaining such human-level generalizability. Finally, we suggest several research directions that could push the field towards *zero-shot* functional compositionality of language models.

1 Introduction

Recently developed large Pre-trained Language Models (PLM) (Devlin et al., 2019; Brown et al., 2020; Raffel et al., 2020) or Foundation Models (Bommasani et al., 2021) have not only achieved state-of-the-art performance through transfer learning in various benchmarks like GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019a), but have also shown dramatic improvements in few-shot and zero-shot learning (Alex et al., 2021; Liu et al., 2022).

It is clear that we have come a long way, but we are still far from achieving human-level generalizability. In particular, we argue that one reason for such is that there has not been enough focus

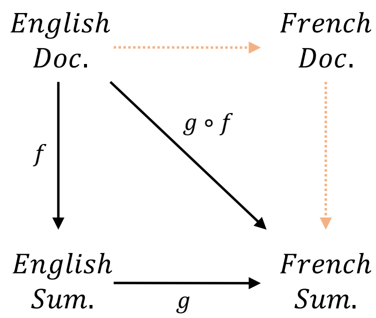

Figure 1: Function compositional representation of Cross-lingual Summarization. Dashed edges are not covered in this work. Sequentially conducting f (summarization) and g (translation) corresponds to the traditional pipeline architecture, while compositional models should follow the diagonal $g \circ f$ edge.

on how humans naturally compose tasks or functions that they learned (Singh, 1991; Li et al., 2020). In this position paper, inspired by composite functions from mathematics, we introduce a perspective called *functional compositionality*. This is a different concept from the traditional discussions about the semantic compositionality of human language, where atomic meanings are composed to create new semantics (Liang, 2013; Pasupat and Liang, 2015; Kim and Linzen, 2020)[1]. Instead, our scope of functional compositionality refers to end-to-end chaining of two different text-to-text transformations, just like function composition from mathematics. As many NLP tasks can be reformulated as text-to-text tasks (Raffel et al., 2020; Brown et al., 2020; Alex et al., 2021), we believe this is not a small scope.

The most illustrative example is Cross-Lingual Summarization (XLS) (Wang et al., 2022). As shown in Figure 1, bilingual people should naturally be able to compose their skills of summarization and translation in order to summarize an English document into a French sentence, *without*

[1] We will cover this more in Section 2.

requiring specialized training to do so. What we expect from large versatile PLMs is also similar. A model that can summarize English documents and translate English to French should be able to create French summary sentences or even summarize French documents *without explicit supervision* of such tasks[2].

However, as we will show later, this is not possible yet in an *end-to-end* fashion. As an alternative, we explore to teach PLMs how to compose tasks. Our key assumption is that the knowledge to compose tasks within a restricted set can be transferred to unseen combinations. This gives a potential direction toward human-level generalizability, but there is still a long way to go.

In this work, we attempt to answer *how far are current text-to-text PLMs from zero-shot functional compositionality*. Our findings can be summarized as such:

- Current PLMs have difficulty in composing text-to-text functions end-to-end by zero-shot.
- However, they were able to "Learn to Compose (L2C)" when explicitly trained to do so on StylePTB (Lyu et al., 2021).
- The L2C method also showed potential to work well with recent parameter-efficient fine-tuning methods, but struggled in transferring the learned task-composing skills to other more difficult benchmarks like WikiLingua (Ladhak et al., 2020).

Through this work, we aim to shed light to a new research direction for large PLMs that have been previously neglected in order to advance towards human-level generalizability.

2 Background and Related Work

Compositionality has been a long-standing challenge in AI and has been well-studied in other many fields, such as theory of computation, linguistics, philosophy, and mathematics. we first cover existing work on semantic compositions (or compositional semantics), then introduce the concept of functional compositions and its distinction from semantic compositions, and discuss its benefits and importance. Also, we discuss the scope of function we consider in this paper.

[2]We use the terms function and task interchangeably.

2.1 Semantic Compositions

The principle of compositionality (Pelletier, 1994) has been widely studied in many fields, In compositional semantics (Janssen and Partee, 1997), the meanings of words or phrases are determined by *combining* the meanings of their sub-words or sub-phrases, and this principle usually holds only when syntactic factors play in the increased complexity of a sentence (Szabó, 2004). As such, this field has often been studied in semantic parsing where complex syntactic rules play a major role in natural language understanding (Liang, 2013; Pasupat and Liang, 2015; Yin et al., 2021; Gupta et al., 2018; Oren et al., 2020; Kim and Linzen, 2020; Szpektor et al., 2020; Parthasarathi et al., 2020). Meanwhile, there was no consensus on whether neural networks are able to generalize compositionally. Hence, Hupkes et al. (2020) discusses this subject in depth by presenting a set of definitions and tests that is grounded on a vast amount of linguistic and philosophical literature, using probabilistic context-free grammar datasets. Another very good example can also be found in visual recognition (Misra et al., 2017; Wang et al., 2019b; Naeem et al., 2021; Purushwalkam et al., 2019; Logeswaran et al., 2021; Cohen et al., 2021; Nayak et al., 2022). Here, if a model understands the meaning of the phrases "grey elephant" and "blue bottle", they test if it also generalizes to new vision-language concepts like "blue elephant".

2.2 Functional Compositions

Inspired by *closed-form composite functions* from mathematics, we define a functional composition as the end-to-end chaining of any two tasks. Figure 1 illustrates this concept very well: instead of taking two side edges (like a pipeline) to conduct cross-lingual summarization, a functionally compositional model should take the diagonal edge. Just like a closed-form composite function, we should be able to compute only once while the output is the same as sequentially applying all functions.

This problem has been somewhat discussed in various kinds of literature. Task decomposition has been a big problem in reinforcement learning literature (Sahni et al., 2017; Devin et al., 2019; Li et al., 2020; Lee et al., 2018; Mendez et al., 2021). Zero-shot cross-lingual transfer is directly related to our definition of functional composition even though it was never really discussed in-depth (Conneau and Lample, 2019; Conneau et al., 2020; Zhao and Schütze, 2021; Ansell et al., 2021; Barbieri et al., 2021; Wu et al., 2022; Gritta et al., 2022). Recently, a compositional style transfer dataset has been released (Lyu et al., 2021). Finally, aggregation of entire network parameters (Madotto et al., 2020; Choshen et al., 2022) and adaptive integration of task-specific parameters (Pfeiffer et al., 2021; Zhang et al., 2022) can also be viewed as an instance of functional compositions.

2.3 Why functional compositionality?

The most obvious benefits of functional compositions would be the behavioral stability of end-to-end models and more efficient inference during deployment than pipelines. More importantly, if a model can (functional) compositionally generalize, this means that collecting expensive datasets like WikiLingua (Ladhak et al., 2020) for XLS may no longer be necessary. Ideally, we can train a model only on the more abundant datasets of the decomposed tasks.

We believe the impact of this matter is very timely as our definition is not just limited to text sequences. The demand for multi-modal language models has been rapidly increasing in both the industry and research community, and there have already been many successful cases in various tasks: Dall-E 2 (Ramesh et al., 2022) and StableDiffusion (Rombach et al., 2022) for realistic text-to-image synthesis, and Make-A-Video (Singer et al., 2022) for text-to-video synthesis. However, such models often require a significantly large amount of multi-modal paired data (and model size) that often drastically exceeds academic budgets. Therefore, expanding these models to languages other than English would require a tremendous amount of data and model parameters. Furthermore, many multi-modal tasks that were solved through pipelines have recently been tackled with end-to-end models, such as Machine Translation directly on images (Jain et al., 2021) or on Speech (Jia et al., 2019) from Google. We believe creating models that generalize well to functional compositions will allow what is mentioned at a much lower cost.

2.4 Scope of Function

In this paper, we narrow down the scope of function to a text-to-text function with no side effects: the input is text and so is the output. Recent works (Raffel et al., 2020; Brown et al., 2020; Sanh et al., 2021) build unified learning frameworks by casting various NLP functions as a text-to-text functions. This would include most of the well-known text generation tasks like machine translation, text summarization, style transfer, conversation, etc. These text-to-text functions allow us using a consistent training objective for various NLP functions. As a future direction, we can also trivially extend this definition to *any sequence-to-sequence tasks* like Automatic Speech Recognition or text-to-image tasks or even Image Captioning – as we can consider an image as a sequence of patches (Dosovitskiy et al., 2020).

3 Methods

Firstly, we train the PLMs on the atomic tasks. After learning the atomic tasks, we explore their zero-shot functional compositionality without learning the target composite task.

Here, we mainly conduct our experiments on the prompt-based language models such as T5 (Raffel et al., 2020) and GPT (Brown et al., 2020) due to their recent successes. We finetune such language models with PROMPT-Tuning that was suggested in T5 (Raffel et al., 2020). As a variant of PROMPT-Tuning, we also compare the PREFIX-Tuning (Li and Liang, 2021) which is a parameter-efficient way of tuning only adaptable prefixes for each layer while freezing language model itself. Meanwhile, PIPELINE is the most straightforward implementation of functional composition with language models, which might be an upper-bound of the functional composition in ideal situation.

In this section, we first describe how each method (PROMPT, PREFIX, and PIPELINE) performs a composite task.

3.1 Prompt-based Fine-tuning (PROMPT)

In (Lester et al., 2021; Han et al., 2021), to specify which task the model should perform, a task-specific (text) prefix is added to the original input sequence before feeding it to the model. Suppose we have a language model parameterized by ϕ. Normally, prompting is done by pre-pending a series of tokens, Z, to the input X, such that the model maximizes the likelihood of the correct Y, or $Pr_\phi(Y|[Z;X])$. Specifically, we train PROMPT model on the atomic tasks[3], sharing the parameters across the tasks.

Prompt Composition A natural way of manipulating prompt-based PLMs to perform a se-

[3]The list of prompts are specified in Appendix (Table 10).

quence of tasks is to give a semantically composed prompt of target atomic tasks to the PLMs, hoping the model can perform the functional composition of the tasks corresponding to the composed prompts. Specifically, we automatically generate such prompts with template-based concatenations[4], such as "`{prompt1} then {prompt2}:`", "`{prompt2} after {prompt1}`", or "`{prompt1}+{prompt2}`".

3.2 Prefix Tuning (PREFIX)

One question that naturally comes up with the idea of PROMPT is whether we can learn compositionality in conjunction with recent parameter-efficient fine-tuning methods (Pfeiffer et al., 2021; Liu et al., 2022) of large language models. Prefix tuning (Li and Liang, 2021) is one of those successful methods. To learn a specific atomic task t, it keeps language model parameters ϕ frozen, but tunes a small continuous task-specific vector P_t (called prefix) and a multi-layer perceptron MLP_{θ_t} parameterized by θ_t. Then, the hidden representation h_i of i-th token at each layer, is computed as follows:

$$h_i = \begin{cases} \text{MLP}_{\theta_t}(P_t[i,:]), & \text{if } i \in \texttt{idx}_t, \\ \text{LM}_{\phi}(z_i, h_{<i}), & \text{otherwise}, \end{cases} \quad (1)$$

where $\texttt{idx}_t$ denotes the indices of prefix vectors in the given sequence, and z_i is i-th input token. Further details can be found in (Li and Liang, 2021).

Prefix Composition Inspired by AdapterFusion (Pfeiffer et al., 2021), we explore non-destructive compositions with task-specific parameters, by using a self-attention layer. Specifically, suppose there are two atomic tasks, t_1 and t_2, and corresponding prefix vectors P_{t_1} and P_{t_2}. Let $t_1 + t_2$ denote the new task, composition of t_1 and t_2. To get a new prefix vectors, we use self-attention (Vaswani et al., 2017) as illustrated in Appendix (Figure 4), *e.g.*, $P_{t_1+t_2} = \text{Attn}_{\eta}([P_{t_1}; P_{t_2}])$ where we have additional parameters η that will learn how to compose the tasks. Because they self-attention parameter η is randomly initialized, this type of composition cannot be done without training.

One modification from the original implementation is that we share a single MLP encoder for multiple atomic tasks by parameter sharing ($\theta_{t_1} = \theta_{t_2} = \theta$). Intuitively, it can be thought of as separating the roles of previous prefix tuning into learning how to perform a task (by P_t) and how to distribute the task vector to different transformer layers (by MLP_{θ}).

3.3 Pipeline (PIPELINE)

As a natural implementation of purely mathematical functional composition, we implement PIPELINE method of serving two different models sequentially, following a certain order. As pipeline requires no extra learning cost to mix various tasks, it has been preferred as strong baselines, still introduced in composite tasks such as TRANSLATE-TEST in XNLI (Conneau et al., 2018). However, its limitations are also clear: 1) calling multiple models in a sequence is computationally expensive, 2) the errors can be accumulated between the sub-tasks, and 3) further training on the target composite task cannot be performed in an end-to-end manner.

Furthermore, it is noteworthy that pipelines are sensitive to the order of sub-tasks. For instance, from StylePTB data (Table 6), consider doing a composition of PPR (removing prepositional phrases) and PTA (voice switch from passive to active) styles to a sentence "1,214 cars were sold last year *by luxury automakers in the U.S.*". Then, a pipeline (PPR → PTA) of first deleting the prepositional phrase "by luxury automakers in the U.S." before voice change can be problematic as the resulting sentence is missing the subject, such that it cannot be rewritten into active voice. On the other hand, the other pipeline of reverse order (PTA → PPR) can easily lead to the proper sentence "Luxury automakers sold 1,214 cars last year.". In some cases, the order can be even restricted because some of component tasks do not exist: We can summarize-then-translate, but cannot translate-then-summarize (Figure 1), as document-level translation is very challenging. We will further explore such order sensitivity of PIPELINE in later (Section 5.1).

4 Experiment Setting

4.1 Dataset

We first evaluate functional compositionality of PLMs on the recently released compositional style-

[4]We also explore a manual writing of the prompts, like "`remove all prepositional phrases and change to future tense`" for style transfer and "`summarize into French:`" for cross-lingual summarization. However, we empirically found that the template-based concatenations outperformed the manual writings. We posit that such counter-intuitive behavior stems from the large diversity of natural language instructions, making it harder to focus on learning how to compose the tasks.

Target: A+B	Strategy	Description	Seen Tasks
Zero-Shot	Two Atomics	the minimal subset of atomic tasks	**A**, **B**
	All Atomics	all atomic tasks	+ [**C**, **D**, **E**, ...]
Zero-Shot (L2C)	Unseen Both	all compositions that does not include any component of the target	+ [**C+D**, **C+E**, **D+E** ...]
	Unseen One (A)	all compositions that does not include one component of the target, **A**	+ [**B+C**, **D+B**, ...]
	Hold-1-Out	all compositions other than the target	+ [**E+A**, **A+D**, ...]
Full-Shot	Full	all compositions	+ [**A+B**]

Table 1: Training strategies regarding data usage with descriptions. There are totally six options, and each row stands for one option. As shown in the last column, the set of seen tasks is accumulated from the top to the bottom. Therefore, the set of training data strictly increases as the row goes down.

transfer dataset, StylePTB[5] (Lyu et al., 2021) which is built upon Penn TreeBank (Marcinkiewicz, 1994). As illustrated in Table 6, each task in StylePTB is either a syntactic or semantic style transfer of a single sentence such as changing the tense or removing certain phrases. It is noteworthy that StylePTB serves composite tasks: For example, given two atomic tasks of changing styles, TFU (to future tense) and PTA (to active voice), a model is tasked to change the two styles at once, TFU+PTA (to future tense in active voice). For our experiments, we use the `Compositional Datasets` partition of StylePTB. It consists of all composite tasks and their atomic components, excluding every atomic task that is not composed. As a result, we use 9 **atomic** tasks, and 22 valid **composite** tasks.

We also experiment with cross-lingual abstractive summarization on the WikiLingua (Ladhak et al., 2020)[6], which gathered multi-lingual guides and their summary from the WikiHow website. We introduce this task for special purpose: to verify whether learned task-composing skill within StylePTB is generalizable to a combination of more realistic and difficult tasks, rather than doing itself better. Out of 10 language pairs in WikiLingua, we only use two that the basic T5 can already translate: English to French (en-fr) and English to German (en-de) (Raffel et al., 2020)[7].

4.2 Training Strategies

One of the most important considerations is that, how many and which atomic/composite tasks are required to learn how to compose the tasks, or how a *training strategy* affects the transferability of functional compositionality. Here, as illustrated in Table 1, for a target composite task $(A + B)$, we design several training strategies, in increasing order of the number of seen composite tasks, which can highlight the effects of dataset construction:

- Two Atomics shows only the two atomic tasks, thus the most realistic settings in our experiments. The model is evaluated on a unique composition of the two atomic tasks.
- All Atomics shows all atomic tasks but without any composite tasks. In comparison with Two Atomics, this strategy will highlight the impact of the number of seen atomic tasks.
- Unseen Both provides all atomic tasks and some composite tasks, where composite tasks that share any atomic tasks with the target composition are excluded.
- Unseen One (A) is similar to Unseen Both, but only excludes the composite tasks that include the atomic task A of target composition.
- Hold-1-Out includes all composite tasks except only the target composite task. By comparing with Unseen Both and Unseen One, we can check the impact of knowing how the atomic tasks are used in other composite tasks during training.
- Full includes all atomic tasks and all composite tasks.

We divide the strategies into three big categories: 1) Zero-Shot, 2) Zero-Shot (L2C), and 3) Full-Shot, where Zero-Shot doesn't allow any composite tasks in training, while Zero-Shot (L2C) allows

[5]https://github.com/lvyiwei1/StylePTB
[6]https://github.com/esdurmus/Wikilingua
[7]We use the official data splits for StylePTB dataset. However, for the WikiLingua dataset, we randomly divide the dataset with an 8:1:1 ratio, using them for train, valid, and test splits respectively because the data splits are not provided publicly for French and German.

some composite tasks except the target composite task. Full-Shot provides the target composite task in training, which can be used as an upper bound performance. Each composition methods (PROMPT, PREFIX, and PIPELINE) can be trained with the training strategies. However, as mentioned in Section 3, PREFIX cannot apply Zero-Shot, and PIPELINE cannot apply Zero-Shot (L2C) and Full-Shot.

5 Results and Discussion

We perform intensive experiments to answer five research questions (RQ), where each of them is a title of following subsections.

5.1 RQ1: Can PLMs compose tasks?

We first evaluate whether T5 can compose the already acquired functions on StylePTB dataset, where the results are presented in Table 2. Overall, we empirically confirmed that T5 struggles to compose already acquired functions, where the Zero-shot PROMPT fails drastically in some cases, which is consistent with the results in Table 4. Though there are some successful cases of showing comparable performance with Full-shot models, it gives only a partial answer to our first research question of asking functional compositionality to language models.

On the other hand, it is noteworthy that PIPELINE shows the second-best score among the methods, which drops only 0.01 points from PROMPT of full-shot training on average, even outperforming in some tasks like “ARR+PFB” task. It demonstrates that PIPELINE is the strongest zero-shot baseline as mentioned above. However, it is manually composed by humans and the models still cannot how to compose such tasks.

5.2 RQ2: Can PLMs learn how to compose?

Zero-shot (L2C) results show that a language model *can learn how to compose* tasks, by training a some number of compositions and then generalize the mixing mechanism to unseen combinations of atomic tasks. Compared to Zero-shot, the **Zero-shot (L2C)** PROMPT performance improves over 100%, and drops around 10% compared to Full-shot PROMPT setting. It is noteworthy that the Zero-shot (L2C) setting does not provide any training data for the target task. We can also see that the same approach considerably well works for GPT2, but not as drastic.

Finally, **Zero-shot (L2C)** PREFIX shows that this observation is also valid for such a parameter-efficient model architecture. However, there is a significant performance drop compared to PROMPT in general. Another observation in Figure 2 is that PROMPT converges faster than PREFIX. One possible explanation is that learning to compose is difficult enough to require full power of large PLMs.

5.3 RQ3: Important factors for L2C?

Number of seen composite tasks As mentioned in Section 5.2, language models can learn how to compose if it is trained with an adequate set of atomic tasks and their combinations. However, it is infeasible to train all combinations, which is exponentially many, so there comes up with the question on how many is enough.

We provide extra detail for the experiment to evaluate the effect of the number of composite tasks on Zero-shot (L2C) performance. We first randomly shuffle the list of 22 composite tasks in StylePTB. Cutting until the first $n = 0, 2, 4, \ldots$ elements of the list, we get a sequence of increasing pool of composite tasks, $S_0 \leq S_2 \leq \ldots \leq S_{20}$. For each n, we basically train the model with S_n and evaluate tasks in S_n. However, for demonstration, we bound n by 14 and show evaluation results on the complement set of S_{14}, containing 8 tasks, to see the trend[8].

Figures 2 and **??** indicate that increasing the number of composite tasks for L2C significantly increases the performance as we expected. We gradually increase the number of trained compositions from 0 to 14 as described above. Figure **??** has individual results per task while Figure 2 shows averaged results among 8 unseen composite tasks.

Choice of seen composite tasks We observed that more seen composite tasks in training data increase the ability to generalize to unseen composite tasks. However, the scenario of adding more tasks totally depends on the permutation of the task sequence. Assuming that not only the number of seen composite tasks but also the **choice** matters, we conduct an ablation study. We adopt more logical data restriction strategies described in Section 4.2. Following the rules, for each target composition out of the 22, an increasing sequence of training datasets is built. Then, models are tuned differently depending on those strategies and evaluated on the target task. The general effect of each strategy on

[8]See Appendix for the full results

	Model	Target Composition (number of samples)								Avg.
		PPR+PTA (959)	TPR+PBF (162)	TFU+PPR (4492)	PPR+ATP (1330)	ARR+PFB (178)	TFU+PTA (2967)	TFU+ATP (2455)	TFU+PFB (233)	
Full-shot	PROMPT	0.9625	0.9375	0.8912	0.8440	0.6471	0.8880	0.8340	0.8261	0.8759
	PREFIX	0.8750	0.9375	0.8796	0.7660	0.4706	0.8533	0.7992	0.8261	0.8399
Zero-shot	PIPELINE	0.9750	0.9375	0.8750	0.8156	0.8824	0.8687	0.8263	0.8261	0.8655
	PROMPT	0.0375	0.7500	0.7593	0.0142	0.2353	0.0695	0.4054	0.8261	0.3931
Zero-shot (L2C)	PROMPT	0.9500	0.9375	0.8912	0.1206	0.7059	0.8610	0.8378	0.8696	0.7957
	PROMPT (GPT-2)	0.5000	0.8750	0.5532	0.3333	0.1176	0.5753	0.4324	0.6957	0.5089
	PREFIX	0.6250	0.8750	0.8519	0.2695	0.4706	0.7066	0.6564	0.8696	0.6982

Table 2: The exact match (EM) scores in StylePTB. **Full-shot** models are trained with both all atomic tasks and all composite tasks. **Zero-shot** models learn all atomic tasks only. **Zero-shot (L2C)** models learn all atomic tasks and all composite tasks, except the target composite task (HOLD-1-OUT). Scores are weighted by test sample size of each task to take average. **Zero-shot (L2C)** models achieve better performance than **Zero-shot** models, showing the possibility of learning to compose tasks. We evaluate the exact match (EM) scores for each task and take average across tasks using test sample sizes as weights. See appendix for the full report including 22 composite tasks.

Zero-shot composition ability is evaluated by averaging out the result through all target tasks.

The result is shown in Table 3. Most of the cases, the EM score increases with the level of composite task disclosure. Such monotonicity is clearer in the average EM score. Note that the mean score of UNSEEN ONE (FIRST) and UNSEEN ONE (SECOND) is still lower than the score of HOLD-1-OUT [9]. We observe same trend even with the controlled training data size. Details are found in appendix.

5.4 RQ4: Can learned task-composing skills be transferred to other difficult benchmarks?

One may ask whether the functional compositionality can be transferred to other benchmarks. If the model truly learns how to compose, it can compose any unseen combination of atomic tasks even from different domain. In our setting, this general question is reduced as whether a T5 model that additionally learned Zero-shot (L2C) from StylePTB can compose two pre-trained tasks, summarization and translation.

Table 4 shows that for that case Zero-shot (L2C) performance is almost same with Zero-shot. This result indicates that learned task-composing skills is transferable to a limited set of compositions. 5.3 supports this observation more. This limitation motivates a new research direction for large PLMs to achieve human-level generalizability.

[9] For those tasks where full-shot is worse than zero-shot, dataset errors made during synthetic generation might let additional data not beneficial beyond certain amount.

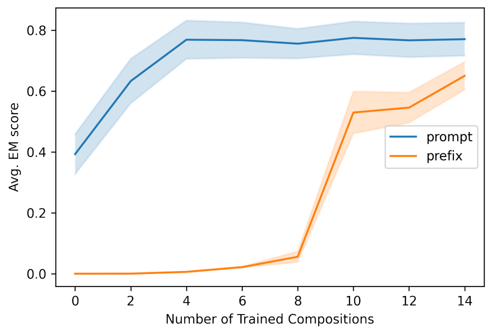

Figure 2: Zero-shot (L2C) average EM scores with respect to number of seen composite tasks. We add two new composite tasks at once and evaluate performance of two models, PROMPT and PREFIX, on a fixed set of 8 unseen tasks.

5.5 RQ5: Do larger LMs have more functional compositionality?

In our preliminary experiments, we observe a very slight chance of GPT-3 (Brown et al., 2020) performing functional compositions in a zero-shot manner. For example, when we give a manually written prompt "`What is the one-sentence French translation of {text}? Please answer in one sentence:`", GPT-3 outputs the French summary of the given text. However, such observations require a bunch of manual prompt tuning. Furthermore, they cannot generalize to other instances, showing just broken results of performing one of the atomic tasks, yielding an English summary or French translation. It is thus recommended to further explore the ability of recent extremely large language models, from GPT-3 (Brown et al., 2020) to Megatron-Turing (Smith et al., 2022).

Training Strategy	Target Composition (number of samples)								Avg.
	PPR+PTA (959)	TPR+PBF (162)	TFU+PPR (4492)	PPR+ATP (1330)	ARR+PFB (178)	TFU+PTA (2967)	TFU+ATP (2455)	TFU+PFB (233)	
TWO ATOMICS	0.0125	0.0625	0.5394	0.0071	0.0000	0.0000	0.0425	0.7391	0.2137
ALL ATOMICS	0.0375	0.7500	0.7593	0.0142	0.2353	0.0695	0.4054	0.8261	0.3931
UNSEEN BOTH	0.4250	0.9375	0.8565	0.2199	0.1765	0.7838	0.7220	0.8261	0.7061
UNSEEN ONE (FIRST)	0.7875	0.8750	0.9028	0.0142	0.0000	0.8340	0.8533	0.8261	0.7618
UNSEEN ONE (SECOND)	0.9000	0.9375	0.8773	0.5603	0.8824	0.7838	0.7490	0.8696	0.8003
HOLD-1-OUT	0.9500	0.9375	0.8912	0.1206	0.7059	0.8610	0.8378	0.8696	0.7957
FULL	0.9625	0.9375	0.8912	0.8440	0.6471	0.8880	0.8340	0.8261	0.8759

Table 3: The exact match (EM) scores in StylePTB, especially focused on comparing training strategies while model is fixed with PROMPT. The results for all composite tasks are in Appendix Figure 3. Rows are sorted in strictly increasing order in terms of training data. Average score is weighted by test sample size of each task.

Model	XLS (En-De)		XLS (En-Fr)	
	ROUGE-4	ROUGE-L	ROUGE-4	ROUGE-L
Fine-tune	0.0314	0.3263	0.0445	0.3556
Pipeline	0.0320	0.3235	0.0390	0.3368
Zero-shot	0.0043	0.1705	0.0110	0.2232
Zero-shot (L2C)	0.0043	0.1698	0.0113	0.2243

Table 4: Cross-lingual summarization results in English-to-French & English-to-German WikiLingua XLS (Ladhak et al., 2020). We trained `t5-base` (Raffel et al., 2020) on English Summarization and the above translations with prompts in a multi-task learning manner. Note that the "Zero-shot" and "Pipeline" are trained only with the atomic tasks (translation and summarization), while "Fine-tune" model is also further trained with direct cross-lingual summaries. Details about training strategies are listed in the Table 1.

6 Future Directions

Pre-training with Pipeline We see great potential for future work utilizing pipeline-based pseudo-labels in the context of functional compositionality. Given the positive results we have observed in terms of noisy few-shot training, we are interested in pre-training language models that can learn how to compose seen tasks. As recent language models have achieved better and better performances on various single (or, component) tasks, pre-training will benefit from pipeline systems.

Decomposition in Pre-training As studied (Lyu et al., 2021), even a well-defined task can be decomposed into multiple sub-tasks. For example, reading comprehension requires recognizing named entities or events in the text, resolving coreferences of them, and selecting an answer among them. However, recent pre-training strategies, specifically T5, treat it as an atomic task, simply forming an input text as "`question: {question} context: {context}`". In this paper, we argue that giving procedural information of each task in T5-style pre-training, like "`entity recognition, coreference resolution, and answer ranking for answering the question: {question} context: {context}`", would be helpful to equip language models with functional compositionality and explainability (Kojima et al., 2022).

7 Conclusion

In this paper, we explore whether PLMs can compose the functions that they already learned. Our empirical results suggest that 1) PLMS cannot compose as it is, 2) but it can be partially learned (L2C), and 3) the learned task-composing skill is not transferable to other benchmark, from style transfer to cross-lingual summarization. From the results, we suggest several future research directions to explore further generalization of its ability to other tasks (*e.g.*, bias-free generation and cross-lingual classification) and training stages (*e.g.*, pre-training and few-shot fine-tuning).

8 Limitations

A recent extremely large language models, such as GPT-3 and Megatron, are not thoroughly covered in this paper due to limitations in resources. For simplicity, we limited our work to compositions of "pure functions" meaning that there are no side-effects generated by the functions. Thus, it is difficult to immediately apply our approach to all NLP pipelines (e.g. Task-oriented Dialogue Systems, classical NLP pipelines, etc.). Furthermore, we limited our experiments to "text-to-text" models so that it is easier to define compositions as the input and output types are equivalent. Considering these jointly restricted our scope of work to a certain set of problems.

References

Neel Alex, Eli Lifland, Lewis Tunstall, Abhishek Thakur, Pegah Maham, C Jess Riedel, Emmie Hine, Carolyn Ashurst, Paul Sedille, Alexis Carlier, et al. 2021. Raft: A real-world few-shot text classification benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Alan Ansell, Edoardo Maria Ponti, Anna Korhonen, and Ivan Vulić. 2021. Composable sparse fine-tuning for cross-lingual transfer. *arXiv preprint arXiv:2110.07560*.

Francesco Barbieri, Luis Espinosa Anke, and Jose Camacho-Collados. 2021. Xlm-t: A multilingual language model toolkit for twitter. *arXiv preprint arXiv:2104.12250*.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Zewen Chi, Li Dong, Shuming Ma, Shaohan Huang Xian-Ling Mao, Heyan Huang, and Furu Wei. 2021. mt6: Multilingual pretrained text-to-text transformer with translation pairs. *arXiv preprint arXiv:2104.08692*.

Leshem Choshen, Elad Venezian, Noam Slonim, and Yoav Katz. 2022. Fusing finetuned models for better pretraining. *arXiv preprint arXiv:2204.03044*.

Vanya Cohen, Geraud Nangue Tasse, Nakul Gopalan, Steven James, Matthew Gombolay, and Benjamin Rosman. 2021. Learning to follow language instructions with compositional policies. *arXiv preprint arXiv:2110.04647*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. *Advances in neural information processing systems*, 32.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485.

Coline Devin, Daniel Geng, Pieter Abbeel, Trevor Darrell, and Sergey Levine. 2019. Plan arithmetic: compositional plan vectors for multi-task control. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 14989–15000.

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

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Milan Gritta, Ruoyu Hu, and Ignacio Iacobacci. 2022. Crossaligner & co: Zero-shot transfer methods for task-oriented cross-lingual natural language understanding. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 4048–4061.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2787–2792.

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

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. 2020. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795.

Puneet Jain, Orhan Firat, Qi Ge, and Sihang Liang. 2021. Image translation network.

Theo MV Janssen and Barbara H Partee. 1997. Compositionality. In *Handbook of logic and language*, pages 417–473. Elsevier.

Ye Jia, Ron J Weiss, Fadi Biadsy, Wolfgang Macherey, Melvin Johnson, Zhifeng Chen, and Yonghui Wu. 2019. Direct speech-to-speech translation with a sequence-to-sequence model. *Proc. Interspeech 2019*, pages 1123–1127.

Najoung Kim and Tal Linzen. 2020. Cogs: A compositional generalization challenge based on semantic interpretation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9087–9105.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*.

Faisal Ladhak, Esin Durmus, Claire Cardie, and Kathleen McKeown. 2020. Wikilingua: A new benchmark dataset for cross-lingual abstractive summarization. *arXiv preprint arXiv:2010.03093*.

Youngwoon Lee, Shao-Hua Sun, Sriram Somasundaram, Edward S Hu, and Joseph J Lim. 2018. Composing complex skills by learning transition policies. In *International Conference on Learning Representations*.

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.

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

Yunfei Li, Yilin Wu, Huazhe Xu, Xiaolong Wang, and Yi Wu. 2020. Solving compositional reinforcement learning problems via task reduction. In *International Conference on Learning Representations*.

Percy Liang. 2013. Lambda dependency-based compositional semantics. *arXiv preprint arXiv:1309.4408*.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *arXiv preprint arXiv:2205.05638*.

Lajanugen Logeswaran, Wilka Torrico Carvalho, and Honglak Lee. 2021. Learning compositional tasks from language instructions. In *Deep RL Workshop NeurIPS 2021*.

Yiwei Lyu, Paul Pu Liang, Hai Pham, Eduard Hovy, Barnabás Póczos, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2021. Styleptb: A compositional benchmark for fine-grained controllable text style transfer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2116–2138.

Andrea Madotto, Zhaojiang Lin, Chien-Sheng Wu, Jamin Shin, and Pascale Fung. 2020. Attention over parameters for dialogue systems. *arXiv preprint arXiv:2001.01871*.

Mary Ann Marcinkiewicz. 1994. Building a large annotated corpus of english: The penn treebank. *Using Large Corpora*, 273.

Jorge A Mendez, Harm van Seijen, and Eric Eaton. 2021. Modular lifelong reinforcement learning via neural composition. In *International Conference on Learning Representations*.

Ishan Misra, Abhinav Gupta, and Martial Hebert. 2017. From red wine to red tomato: Composition with context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1792–1801.

Muhammad Ferjad Naeem, Yongqin Xian, Federico Tombari, and Zeynep Akata. 2021. Learning graph embeddings for compositional zero-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 953–962.

Nihal V Nayak, Peilin Yu, and Stephen H Bach. 2022. Learning to compose soft prompts for compositional zero-shot learning. *arXiv preprint arXiv:2204.03574*.

Inbar Oren, Jonathan Herzig, Nitish Gupta, Matt Gardner, and Jonathan Berant. 2020. Improving compositional generalization in semantic parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2482–2495.

Prasanna Parthasarathi, Sharan Narang, and Arvind Neelakantan. 2020. On task-level dialogue composition of generative transformer model. In *Proceedings of the First Workshop on Insights from Negative Results in NLP*, pages 41–47.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480.

Francis Jeffry Pelletier. 1994. The principle of semantic compositionality. *Topoi*, 13(1):11–24.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503.

Senthil Purushwalkam, Maximilian Nickel, Abhinav Gupta, and Marc'Aurelio Ranzato. 2019. Task-driven modular networks for zero-shot compositional learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3593–3602.

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.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695.

Himanshu Sahni, Saurabh Kumar, Farhan Tejani, and Charles Isbell. 2017. Learning to compose skills. *arXiv preprint arXiv:1711.11289*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. 2022. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*.

Satinder P Singh. 1991. Transfer of learning across compositions of sequential tasks. In *Machine Learning Proceedings 1991*, pages 348–352. Elsevier.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Zoltán Gendler Szabó. 2004. Compositionality.

Idan Szpektor, Deborah Cohen, Gal Elidan, Michael Fink, Avinatan Hassidim, Orgad Keller, Sayali Kulkarni, Eran Ofek, Sagie Pudinsky, Asaf Revach, et al. 2020. Dynamic composition for conversational domain exploration. In *Proceedings of The Web Conference 2020*, pages 872–883.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

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

Jiaan Wang, Fandong Meng, Duo Zheng, Yunlong Liang, Zhixu Li, Jianfeng Qu, and Jie Zhou. 2022. A survey on cross-lingual summarization. *arXiv preprint arXiv:2203.12515*.

Xin Wang, Fisher Yu, Trevor Darrell, and Joseph E Gonzalez. 2019b. Task-aware feature generation for zero-shot compositional learning. *arXiv preprint arXiv:1906.04854*.

Linjuan Wu, Shaojuan Wu, Xiaowang Zhang, Deyi Xiong, Shizhan Chen, Zhiqiang Zhuang, and Zhiyong Feng. 2022. Learning disentangled semantic representations for zero-shot cross-lingual transfer in multilingual machine reading comprehension. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 991–1000.

Pengcheng Yin, Hao Fang, Graham Neubig, Adam Pauls, Emmanouil Antonios Platanios, Yu Su, Sam Thomson, and Jacob Andreas. 2021. Compositional generalization for neural semantic parsing via span-level supervised attention. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2810–2823.

Yanzhe Zhang, Xuezhi Wang, and Diyi Yang. 2022. Continual sequence generation with adaptive compositional modules. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3653–3667.

Mengjie Zhao and Hinrich Schütze. 2021. Discrete and soft prompting for multilingual models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8547–8555.

A Training Details

For experiments, we follow the hyper-parameters from huggingface T5 [10]. Specifically, we train `t5-base` with a batch size of 16 for StylePTB dataset. We train the model with a learning rate of $5e-5$ using the AdamW optimizer until convergence. For learning objectives, we cast all the tasks into a "text-to-text" format and train them with a maximum likelihood objective:

$$\max_{\phi} \log Pr_{\phi}(Y|X), \tag{2}$$

where X and Y denote the input and output token sequences, and ϕ is the set of model parameters. To avoid catastrophic forgetting of atomic tasks, the training is done in a multi-task manner with a mixed-task batch. The average time for training is 1 hour.

For the WikiLingua dataset, we follow the hyper-parameter settings from (Chi et al., 2021). For our experiments, we start training on `t5-base` with a batch size of 32. The average time for training is 24 hours.

We use GTX 2080ti $\times$ 4 for training our models. For PREFIX, we additionally train approximately 48M parameters with T5-base.

We do a single run for evaluation/training.

B Result with Controlled training data size

We observe same trend even with the controlled training data size. Table 5 shows the result. All training strategies that belong to Zero-shot (L2C) are compared, while a randomly sampled subset of fixed size is used as a training dataset for each option. We can confirm that the EM score still increases following the level of composite task disclosure.

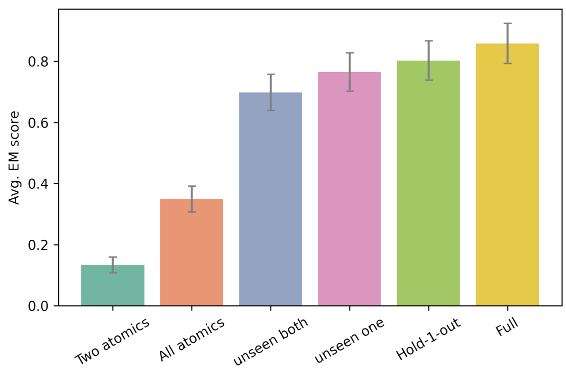

Figure 3: Average EM scores for variants of training strategies with PROMPT methods.

[10] https://huggingface.co/docs/transformers/model_doc/t5

Training Strategy	Target Composition (number of samples)								Avg.
	PPR+PTA (959)	TPR+PBF (162)	TFU+PPR (4492)	PPR+ATP (1330)	ARR+PFB (178)	TFU+PTA (2967)	TFU+ATP (2455)	TFU+PFB (233)	
UNSEEN BOTH	0.4250	0.9375	0.8565	0.2199	0.1765	0.7838	0.7220	0.8261	0.7061
UNSEEN ONE (FIRST)	0.8875	0.9375	0.8750	0.0922	0.0588	0.8533	0.7992	0.8261	0.6662
UNSEEN ONE (SECOND)	0.9000	0.9375	0.8796	0.4823	0.7059	0.8224	0.7722	0.8696	0.8040
UNSEEN ONE (AVG)	0.8937	0.9375	0.8773	0.2872	0.3824	0.8378	0.7857	0.8478	0.7837
HOLD-1-OUT	0.7875	1.0000	0.9005	0.3050	0.7059	0.8340	0.8147	0.8261	0.7953

Table 5: The exact match (EM) scores in StylePTB, especially focused on comparing training strategies while the number of training samples is fixed. The model is fixed with PROMPT. Rows are sorted in strictly increasing order in terms of training data. Average score is weighted by test sample size of each task.

Category	Change	Abbreviation	Description	# of samples (train/valid/test)
Syntax	Tense	TFU	To future tense	9279 / 1013 / 1006
		TPR	To present tense	5564 / 645 / 643
		TPA	To past tense	4684 / 511 / 502
	Voice	ATP	Active to passive	2533 / 278 / 284
		PTA	Passive to Active	2533 / 278 / 284
	PP Front Back	PFB	PP front to back	426 / 23 / 26
		PBF	PP back to front	426 / 23 / 27
Semantic	ADJ/ADV Removal	ARR	ADJ or ADV Removal	4639 / 273 / 276
	PP Removal	PPR	PP Removal	14123 / 986 / 1013

Table 6: StylePTB dataset distribution.

	Model	Target Composition (number of samples)							
		PPR+PTA (959)	TPR+PBF (162)	TFU+PPR (4492)	PPR+ATP (1330)	ARR+PFB (178)	TFU+PTA (2967)	TFU+ATP (2455)	TFU+PFB (233)
Full-shot	PROMPT	0.9625	0.9375	0.8912	0.8440	0.6471	0.8880	0.8340	0.8261
	PREFIX	0.8750	0.9375	0.8796	0.7660	0.4706	0.8533	0.7992	0.8261
Zero-shot	PIPELINE	0.9750	0.9375	0.8750	0.8156	0.8824	0.8687	0.8263	0.8261
	PROMPT	0.0375	0.7500	0.7593	0.0142	0.2353	0.0695	0.4054	0.8261
Zero-shot (L2C)	PROMPT	0.9500	0.9375	0.8912	0.1206	0.7059	0.8610	0.8378	0.8696
	PROMPT (GPT-2)	0.5000	0.8750	0.5532	0.3333	0.1176	0.5753	0.4324	0.6957
	PREFIX	0.6250	0.8750	0.8519	0.2695	0.4706	0.7066	0.6564	0.8696
		TPR+ATP (1561)	TPA+PBF (61)	ARR+PBF (178)	TFU+PBF (245)	TPR+PFB (171)	TFU+ARR (2166)	TPR+PTA (2163)	TPA+ARR (1444)
Full-shot	PROMPT	0.8333	1.0000	0.6471	0.8333	0.9412	0.7904	0.8830	0.7500
	PREFIX	0.8086	1.0000	0.7647	0.9167	0.7647	0.6419	0.8511	0.7285
Zero-shot	PIPELINE	0.8333	1.0000	0.9412	0.8750	0.8824	0.7773	0.8936	0.7881
	PROMPT	0.3210	0.6667	0.1765	0.5000	0.9412	0.0393	0.1064	0.0464
Zero-shot (L2C)	PROMPT	0.8457	1.0000	0.7647	0.8333	0.8824	0.7511	0.7926	0.8146
	PROMPT (GPT-2)	0.4568	0.8333	0.4706	0.8333	0.5882	0.3231	0.6383	0.6755
	PREFIX	0.7407	1.0000	0.4706	0.8750	0.8824	0.6157	0.6702	0.6556
		TPA+PFB (70)	TPA+PTA (1617)	TPA+PPR (658)	TPA+PPR (1926)	TPR+PPR (3054)	TPR+ARR (1260)	Avg (29350)	
Full-shot	PROMPT	1.0000	0.9357	0.7692	0.9135	0.8733	0.7500	0.8585	
	PREFIX	1.0000	0.8714	0.6923	0.8757	0.8288	0.6364	0.8103	
Zero-shot	PIPELINE	1.0000	0.8571	0.6769	0.8973	0.8527	0.7727	0.8488	
	PROMPT	1.0000	0.0571	0.0769	0.7622	0.7397	0.0379	0.3492	
Zero-shot (L2C)	PROMPT	1.0000	0.8286	0.4615	0.9081	0.8630	0.7197	0.8027	
	PROMPT (GPT-2)	0.7143	0.5500	0.2308	0.7027	0.6610	0.5227	0.5394	
	PREFIX	1.0000	0.6429	0.3077	0.8378	0.8151	0.6515	0.7002	

Table 7: The exact match (EM) scores in StylePTB. **Full-shot** models are trained with both all atomic tasks and all composite tasks. **Zero-shot** models learn all atomic tasks only. **Zero-shot (L2C)** models learn all atomic tasks and all composite tasks, except the target composite task. Scores are weighted by test sample size of each task to take average. We evaluate the exact match (EM) scores for each task and take average across tasks using test sample sizes as weights.

Training Strategy	Target Composition (number of samples)							
	PPR+PTA (959)	TPR+PBF (162)	TFU+PPR (4492)	PPR+ATP (1330)	ARR+PFB (178)	TFU+PTA (2967)	TFU+ATP (2455)	TFU+PFB (233)
TWO ATOMICS	0.0125	0.0625	0.5394	0.0071	0.0000	0.0000	0.0425	0.7391
ALL ATOMICS	0.0375	0.7500	0.7593	0.0142	0.2353	0.0695	0.4054	0.8261
UNSEEN BOTH	0.4250	0.9375	0.8565	0.2199	0.1765	0.7838	0.7220	0.8261
UNSEEN ONE (FIRST)	0.7875	0.8750	0.9028	0.0142	0.0000	0.8340	0.8533	0.8261
UNSEEN ONE (SECOND)	0.9000	0.9375	0.8773	0.5603	0.8824	0.7838	0.7490	0.8696
HOLD-1-OUT	0.9500	0.9375	0.8912	0.1206	0.7059	0.8610	0.8378	0.8696
FULL	0.9625	0.9375	0.8912	0.8440	0.6471	0.8880	0.8340	0.8261

Training Strategy	Target Composition (number of samples)							
	TPR+ATP (1561)	TPA+PBF (61)	ARR+PBF (178)	TFU+PBF (245)	TPR+PFB (171)	TFU+ARR (2166)	TPR+PTA (2163)	PTA+ARR (1444)
TWO ATOMICS	0.2901	0.0000	0.1176	0.5833	0.8824	0.0044	0.0000	0.0066
ALL ATOMICS	0.3210	0.6667	0.1765	0.5000	0.9412	0.0393	0.1064	0.0464
UNSEEN BOTH	0.7901	1.0000	0.1765	0.8333	1.0000	0.4454	0.7074	0.5497
UNSEEN ONE (FIRST)	0.8395	1.0000	0.1176	0.8750	0.9412	0.7555	0.7447	0.6821
UNSEEN ONE (SECOND)	0.7840	1.0000	0.5294	0.7917	0.9412	0.5852	0.7287	0.5894
HOLD-1-OUT	0.8457	1.0000	0.7647	0.8333	0.8824	0.7511	0.7926	0.8146
FULL	0.8333	1.0000	0.6471	0.8333	0.9412	0.7904	0.8830	0.7500

Training Strategy	Target Composition (number of samples)						Avg.
	TPA+PFB (70)	TPA+PTA (1617)	TPA+ATP (658)	TPA+PPR (1926)	TPR+PPR (3054)	TPR+ARR (1260)	
TWO ATOMICS	0.7143	0.0000	0.0154	0.0919	0.2466	0.0076	0.1539
ALL ATOMICS	0.0375	0.7500	0.7593	0.0142	0.2353	0.0695	0.3492
UNSEEN BOTH	1.0000	0.7357	0.4154	0.8630	0.8459	0.5530	0.6980
UNSEEN ONE (FIRST)	1.0000	0.8429	0.5077	0.9189	0.8733	0.7500	0.7796
UNSEEN ONE (SECOND)	1.0000	0.7500	0.4154	0.8865	0.8356	0.4394	0.7506
HOLD-1-OUT	1.0000	0.8286	0.4615	0.9081	0.8630	0.7197	0.8028
FULL	1.0000	0.9357	0.7692	0.9135	0.8733	0.7500	0.8585

Table 8: The exact match (EM) scores in StylePTB, especially focused on comparing training strategies while model is fixed with PROMPT. The results for all composite tasks are in Appendix Figure 3. Rows are sorted in strictly increasing order in terms of training data. Average score is weighted by test sample size of each task.

	Target Composition (number of samples)								Avg.
	PPR+PTA (959)	PPR+ATP (1330)	TFU+PTA (2967)	TFU+ATP (2455)	TPR+PTA (1561)	TPR+ATP (2163)	TPA+PTA (1617)	TPA+ATP (658)	
VOICE FIRST	0.9750	0.8156	0.8263	0.8687	0.8333	0.8936	0.8571	0.6769	0.8527
VOICE LATER	0.0250	0.0142	0.7799	0.7954	0.7963	0.5904	0.3000	0.4462	0.5555

Table 9: The exact match (EM) scores of PIPELINE with different order of computation. 8 target tasks in this table is the set of all compositions that includes a component task from *Voice* category, PTA or ATP. Two annotations VOICE FIRST or VOICE LATER specify the order of components to be applied. For example, VOICE FIRST option with a target task PPR+PTA means we perform PTA first, and then do PPR later.

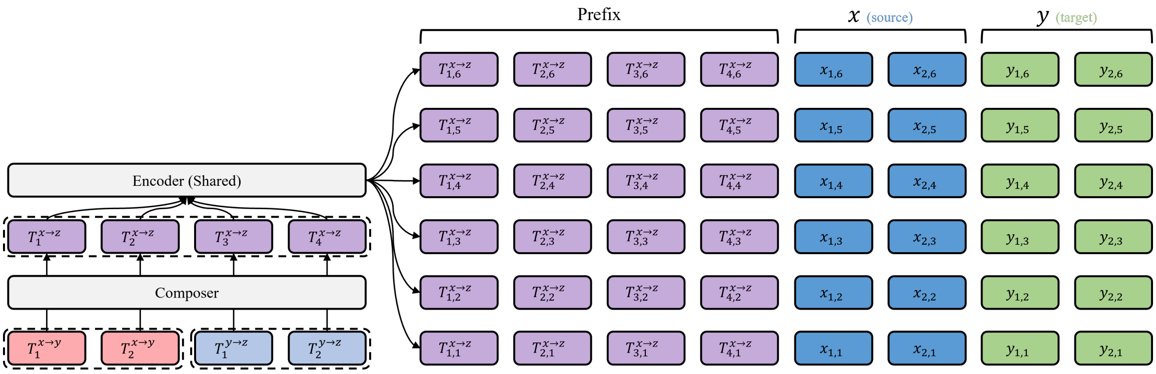

Figure 4: The overall architecture of prefix composition.

Dataset	Type	Task	Prompt
StylePTB	Atomic	PPR	`PPR:`
		PTA	`PTA:`
		ATP	`ATP:`
		TFU	`TFU:`
		TPR	`TPR:`
		TPA	`TPA:`
		ARR	`ARR:`
		PBF	`PBF:`
		PFB	`PFB:`
	Composition	PPR+ATP	`PPR + ATP:`
		PPR+PTA	`PPR + PTA:`
		TFU+ATP	`TFU + ATP:`
		TFU+PTA	`TFU + PTA:`
		TPR+ATP	`TPR + ATP:`
		TPR+PTA	`TPR + PTA:`
		TPA+ATP	`TPA + ATP:`
		TPA+PTA	`TPA + PTA:`
		TFU+PPR	`TFU + PPR:`
		TPR+PPR	`TPR + PPR:`
		TPA+PPR	`TPA + PPR:`
		ARR+PFB	`ARR + PFB:`
		ARR+PBF	`ARR + PBF:`
		TFU+ARR	`TFU + ARR:`
		TPA+ARR	`TPA + ARR:`
		TPR+ARR	`TPR + ARR:`
		TFU+PBF	`TFU + PBF:`
		TFU+PFB	`TFU + PFB:`
		TPA+PFB	`TPA + PFB:`
		TPA+PBF	`TPA + PBF:`
		TPR+PBF	`TPR + PBF:`
		TPR+PFB	`TPR + PFB:`
WikiLingua	Atomic	Summarization	`summarize:`
		Translation (en-fr)	`translate_en_fr:`
		Translation (en-de)	`translate_en_de:`
	Composition	XLS (en-fr)	`summarize + translate_en_fr:`
		XLS (en-de)	`summarize + translate_en_de:`

Table 10: Prompt for language model.