(2021) first use multiple manually created sub-prompts for entity recognition and relation classification and then compose them into a complete prompt based on logic rules for relation extraction.

### 6.4 Prompt Decomposition

For tasks where multiple predictions should be performed for one sample (e.g., sequence labeling), directly defining a holistic prompt with regards to the entire input text $\boldsymbol{x}$ is challenging. One intuitive method to address this problem is to break down the holistic prompt into different sub-prompts, and then answer each sub-prompt separately. Fig.4-(d) illustrates this idea with an example from the named entity recognition task, which aims to identify all named entities in an input sentence. In this case, the input will first be converted into a set of text spans, and the model can then be prompted to predict the entity type (including "Not an Entity") for each span. It is not easy to predict all the span types at the same time due to the large number of spans, so different prompts for each span can be created and predicted separately. This sort of *prompt decomposition* for named entity recognition has been explored by Cui et al. (2021) where they apply the approach we discussed here.

## 7 Training Strategies for Prompting Methods

With the methods in the above sections, it is now clear how to obtain an appropriate prompt (or prompts) and corresponding answers. Now we discuss about methods that explicitly train models in concert with prompting methods, as outlined in the "Training Strategies" section of Fig.1.

### 7.1 Training Settings

In many cases, prompting methods can be used without *any* explicit training of the LM for the down-stream task, simply taking an LM that has been trained to predict the probability of text $P(\boldsymbol{x})$ and applying it as-is to fill the cloze or prefix prompts defined to specify the task. This is traditionally called the *zero-shot* setting, as there is zero training data for the task of interest.

However, there are also methods that use training data to train the model in concert with prompting methods. These consist of either *full-data learning*, where a reasonably large number of training examples are used to train the model, or *few-shot learning* where a very small number of examples are used to train the model. Prompting methods are particularly useful in the latter case, as there are generally not enough training examples to fully specify the desired behavior, and thus using a prompt to push the model in the right direction is particularly effective.

One thing to note is that for many of the prompt engineering methods described in §4, although annotated training samples are not explicitly used in the training of the downstream task model, they *are* often used in the construction or validation of the prompts that the downstream task will use. As noted by Perez et al. (2021), this is arguably not true zero-shot learning with respect to the downstream task.

### 7.2 Parameter Update Methods

In prompt-based downstream task learning, there are usually two types of parameters, namely those from (1) pre-trained models and (2) prompts. Which part of parameters should be updated is one important design decision, which can lead to different levels of applicability in different scenarios. We summarize five tuning strategies (as shown in Tab. 6) based on (i) whether the parameters of the underlying LM are tuned, (ii) whether there are additional prompt-related parameters, (iii) if there are additional prompt-related parameters, whether those parameters are tuned.

| Strategy | LM Params | Prompt Params | | Example |
|---|---|---|---|---|
| | | Additional | Tuned | |
| Promptless Fine-tuning | Tuned | - | | ELMo [130], BERT [32], BART [94] |
| Tuning-free Prompting | Frozen | ✗ | ✗ | GPT-3 [16], AutoPrompt [159], LAMA [133] |
| Fixed-LM Prompt Tuning | Frozen | ✓ | Tuned | Prefix-Tuning [96], Prompt-Tuning [91] |
| Fixed-prompt LM Tuning | Tuned | ✗ | ✗ | PET-TC [153], PET-Gen [152], LM-BFF [46] |
| Prompt+LM Fine-tuning | Tuned | ✓ | Tuned | PADA [8], P-Tuning [103], PTR [56] |

Table 6: Characteristics of different tuning strategies. "Additional" represents if there are additional parameters beyond LM parameters while "Tuned" denotes if parameters are updated.

### 7.2.1 Promptless Fine-tuning

As mentioned in the introduction, the *pre-train and fine-tune* strategy has been widely used in NLP since before the popularization of prompting methods. Here we refer to pre-training and fine-tuning *without* prompts as *promptless fine-tuning*, to contrast with the prompt-based learning methods introduced in the following sections. In this strategy, given a dataset of a task, all (or some (Howard and Ruder, 2018; Peters et al., 2019)) of the parameters of the pre-trained LM will be updated via gradients induced from downstream training samples. Typical examples of pre-trained models tuned in this way include BERT [32] and RoBERTa [105]. This is a simple, powerful, and widely-used method, but it may overfit or not learn stably on small datasets (Dodge et al., 2020). Models are also prone to *catastrophic forgetting*, where the LM loses its ability to do things that it was able to do before fine-tuning (McCloskey and Cohen, 1989).

- **Advantages**: Simplicity, no need for prompt design. Tuning all the LM parameters allows the model to fit to larger training datasets.
- **Disadvantages**: LMs may overfit or not learn stably on smaller datasets.

### 7.2.2 Tuning-free Prompting

*Tuning-free prompting* directly generates the answers without changing the parameters of the pre-trained LMs based only on a prompt, as described in the simplest incarnation of prompting in §2. These can be optionally augmenting input with answered prompts as described in §6.2, and this combination of tuning-free prompting and prompt augmentation is also referred to as *in-context learning* (Brown et al., 2020). Typical examples of tuning-free prompting include LAMA [133] and GPT-3 [16].

- **Advantages**: Efficiency, there is no parameter update process. No catastrophic forgetting, as LM parameters remain fixed. Applicable in zero-shot settings.
- **Disadvantages**: Because prompts are the only method that provide the task specification, heavy engineering is necessary to achieve high accuracy. In particular in the in-context learning setting, providing many answered prompts can be slow at test time, and thus cannot easily use large training datasets.

### 7.2.3 Fixed-LM Prompt Tuning

In the scenario where additional prompt-relevant parameters are introduced besides parameters of the pre-trained model, *fixed-LM prompt tuning* updates only the prompts' parameters using the supervision signal obtained from the downstream training samples, while keeping the entire pre-trained LM unchanged. Typical examples are Prefix-Tuning [96] and WARP [55].

- **Advantages**: Similarly to tuning-free prompting, it can retain knowledge in LMs and is suitable in few-shot scenarios. Often superior accuracy to tuning-free prompting.
- **Disadvantages**: Not applicable in zero-shot scenarios. While effective in few-shot scenarios, representation power is limited in large-data settings. Prompt engineering through choice of hyperparameters or seed prompts is necessary. Prompts are usually not human-interpretable or manipulable.

### 7.2.4 Fixed-prompt LM Tuning

*Fixed-prompt LM tuning* tunes the parameters of the LM, as in the standard pre-train and fine-tune paradigm, but additionally uses prompts with fixed parameters to specify the model behavior. This potentially leads to improvements, particularly in few-shot scenarios.

The most natural way to do so is to provide a discrete textual template that is applied to every training and test example. Typical examples include PET-TC [153], PET-Gen [152], LM-BFF [46]. Logan IV et al. (2021) more recently observe that the prompt engineering can be reduced by allowing for a combination of answer engineering and partial LM fine-tuning. For example, they define a very simple template, *null prompt*, where the input and mask are directly concatenated "`[X][Z]`" without any template words, and find this achieves competitive accuracy.

- **Advantages**: Prompt or answer engineering more completely specify the task, allowing for more efficient learning, particularly in few-shot scenarios.
- **Disadvantages**: Prompt or answer engineering are still required, although perhaps not as much as without prompting. LMs fine-tuned on one downstream task may not be effective on another one.

#### 7.2.5 Prompt+LM Tuning

In this setting, there are prompt-relevant parameters, which can be fine-tuned together with the all or some of the parameters of the pre-trained models. Representative examples include PADA [8], P-Tuning [103]. Notably, this setting is very similar to the standard pre-train and fine-tune paradigm, but the addition of the prompt can provide additional bootstrapping at the start of model training.

- **Advantages**: This is the most expressive method, likely suitable for high-data settings.
- **Disadvantages**: Requires training and storing all parameters of the models. May overfit to small datasets.

## 8 Applications

In previous sections, we examined prompting methods from the point of view of the mechanism of the method itself. In this section, we rather organize prompting methods from the point of view of which applications they have been applied to. We list these applications in Tab. 7-8 and summarize them in the following sections.

### 8.1 Knowledge Probing

**Factual Probing** *Factual probing* (a.k.a. fact retrieval) is one of the earliest scenarios with respect to which prompting methods were applied. The motivation of exploring this task is to quantify how much factual knowledge the pre-trained LM's internal representations bear. In this task, parameters of pre-trained models are usually fixed, and knowledge is retrieved by transforming the original input into a cloze prompt as defined in §2.2, which can be manually crafted or automatically discovered. Relevant datasets including `LAMA` (Petroni et al., 2019) and `X-FACTR` (Jiang et al., 2020a). Since the answers are pre-defined, fact retrieval only focuses on finding effective templates and analyzing the results of different models using these templates. Both discrete template search (Petroni et al., 2019, 2020; Jiang et al., 2020c,a; Haviv et al., 2021; Shin et al., 2020; Perez et al., 2021) and continuous template learning (Qin and Eisner, 2021; Liu et al., 2021b; Zhong et al., 2021b) have been explored within this context, as well as prompt ensemble learning (Jiang et al., 2020c; Qin and Eisner, 2021).

**Linguistic Probing** Besides factual knowledge, large-scale pre-training also allows LMs to handle linguistic phenomena such as analogies (Brown et al., 2020), negations (Ettinger, 2020), semantic role sensitivity (Ettinger, 2020), semantic similarity (Sun et al., 2021), cant understanding (Sun et al., 2021), and rare word understanding (Schick and Schütze, 2020). The above knowledge can also be elicited by presenting *linguistic probing* tasks in the form of natural language sentences that are to be completed by the LM.

### 8.2 Classification-based Tasks

Prompt-based learning has been widely explored in classification-based tasks where prompt templates can be constructed relatively easily, such as text classification (Yin et al., 2019) and natural language inference (Schick and Schütze, 2021a). The key to prompting for classification-based tasks is reformulating it as an appropriate prompt. For example, Yin et al. (2019) use a prompt such as "the topic of this document is `[Z]`.", which is then fed into mask pre-trained LMs for slot filling.

**Text Classification** For *text classification* tasks, most previous work has used cloze prompts, and both prompt engineering (Gao et al., 2021; Hambardzumyan et al., 2021; Lester et al., 2021) and answer engineering (Schick and Schütze, 2021a; Schick et al., 2020; Gao et al., 2021) have been explored extensively. Most existing works explore the efficacy of prompt learning for text classification in the context of *few-shot* setting with "*fixed-prompt LM Tuning*" strategies (defined in §7.2.4).

**Natural Language Inference (NLI)** NLI aims to predict the relationship (e.g., `entailment`) of two given sentences. Similar to text classification tasks, for *natural language inference* tasks, cloze prompts are commonly used (Schick and Schütze, 2021a). Regarding prompt engineering, researchers mainly focus on the template search in the few-shot learning setting and the answer space $\mathcal{Z}$ is usually manually pre-selected from the vocabulary.