Automatic Label Sequence Generation for Prompting Sequence-to-sequence Models

Zichun Yu[1] **Tianyu Gao**[2] **Zhengyan Zhang**[13] **Yankai Lin**[4]
Zhiyuan Liu[1356†] **Maosong Sun**[1356†] **Jie Zhou**[7]
[1]Dept. of Comp. Sci. & Tech., Institute for AI, Tsinghua University
[2]Princeton University
[3]Beijing National Research Center for Information Science and Technology
[4]Gaoling School of Artificial Intelligence, Renmin University of China
[5]International Innovation Center of Tsinghua University, Shanghai, China
[6]Beijing Academy of Artificial Intelligence
[7]Pattern Recognition Center, WeChat AI, Tencent Inc
yuzc19@mails.tsinghua.edu.cn {liuzy,sms}@tsinghua.edu.cn

Abstract

Prompting, which casts downstream applications as language modeling tasks, has shown to be sample efficient compared to standard fine-tuning with pre-trained models. However, one pitfall of prompting is the need of manually-designed patterns, whose outcome can be unintuitive and requires large validation sets to tune. To tackle the challenge, we propose **AutoSeq**, a fully automatic prompting method: (1) We adopt natural language prompts on sequence-to-sequence models, enabling free-form generation and larger label search space; (2) We propose label sequences – phrases with indefinite lengths to verbalize the labels – which eliminate the need of manual templates and are more expressive than single label words; (3) We use beam search to automatically generate a large amount of label sequence candidates and propose contrastive re-ranking to get the best combinations. AutoSeq significantly outperforms other no-manual-design methods, such as soft prompt tuning, adapter tuning, and automatic search on single label words; the generated label sequences are even better than curated manual ones on a variety of tasks. Our method reveals the potential of sequence-to-sequence models in few-shot learning and sheds light on a path to generic and automatic prompting. The source code of this paper can be obtained from https://github.com/thunlp/Seq2Seq-Prompt.

1 Introduction

Among ways of adapting pre-trained language models (Devlin et al., 2019; Raffel et al., 2020) to downstream applications, prompting, which uses a natural language prompt to reformulate tasks as cloze questions, has shown to be especially effective (Brown et al., 2020; Schick and Schütze, 2021a,b; Gao et al., 2021). For example, in sentiment classification, prompting appends a *template* "It was [MASK]" to the original input, and defines "great" and "terrible" as the *label words*, whose probabilities at [MASK] indicate the probabilities of the positive and negative sentiment labels. Prompting possesses better sample efficiency and performs significantly better than standard fine-tuning in the low resource case.

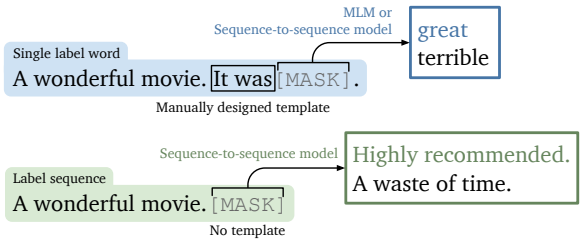

Figure 1: Single label words vs label sequences. Label sequences are more expressive and eliminate the need of manually-designed templates.

However, the prompting performance is highly sensitive to the prompt choice, whose effectiveness needs abundant validation data to evaluate and is difficult to predict by intuition (Gao et al., 2021; Perez et al., 2021). Even though there exist methods that explore automatic prompt search (Schick et al., 2020; Gao et al., 2021), they still require substantial human efforts, for the algorithms start from either manual templates or label words.

We propose AutoSeq, a prompting method that is fully automatic and requires no human input. AutoSeq has three innovations: (1) AutoSeq adopts *sequence-to-sequence models* like T5 (Raffel et al.,

Part of the work was done while Yankai Lin was working at Tencent.

† Corresponding authors

Proceedings of the 29th International Conference on Computational Linguistics, pages 4965–4975
October 12–17, 2022.

2020). Compared to masked language models (MLM) like BERT (Devlin et al., 2019), it allows free-form generation, enables more types of tasks, and extends the label space for prompting. (2) We propose *label sequences*, which are indefinite-length phrases or sentences that represent each label. They are more expressive than previous single label words and eliminate the need for a manual template (Figure 1). (3) We design an automatic label sequence search pipeline, which first generates a large amount of candidates by T5, then re-ranks them by *contrastive probability*.

Our main experiment results on natural language understanding datasets show that AutoSeq performs significantly better than automatic prompt search using single label words as well as no-prompt methods like soft prompt tuning and adapter tuning. AutoSeq also outperforms hand-crafted prompts on a variety of tasks. We hope our work enlightens automatic prompting and building a universal prompt-based fine-tuning framework.

2 Related Work

Prompting. Schick and Schütze (2021a,b); Gao et al. (2021) introduced prompting into MLM. Though showing remarkable few-shot performance, those models are constrained by the single `[MASK]` token and are limited to classification tasks; they also require manually-designed prompts. In parallel, soft prompt tuning (Lester et al., 2021) and adapter tuning (Houlsby et al., 2019; Zaken et al., 2022; Hu et al., 2022) do not require manual design, but they lag behind prompting in few-shot performance (Gu et al., 2022). Recent work (Zhang et al., 2022) tries to mitigate the gap, but it still requires the help of manual prompts and thus falls out of the scope of our discussion.

Automatic prompt search. There have been plenty of attempts for automatic prompt search – yet all of them require to start from either human-designed label words or templates (Davison et al., 2019; Jiang et al., 2020; Shin et al., 2020; Schick et al., 2020; Gao et al., 2021; Yuan et al., 2021; Haviv et al., 2021). In contrast, our AutoSeq is a general-purpose, fully automatic search method that depends only on few-shot annotations.

3 AutoSeq

3.1 Prompts for sequence-to-sequence models

We introduce the sequence-to-sequence version of *prompt-based fine-tuning*, bringing in *label sequences* that are more expressive than one token. Using sentiment classification as an example, and given the input sentence as x, the model input can be formulated as "x `[MASK]`". We define the label sequences for the positive class as "*Highly recommended.*" and that for the negative class as "*Not for me.*". Then the probability of each class is tied with that of the T5 model generating "*Highly recommended.*" and "*Not for me.*" at position `[MASK]`. As we compare the MLM single label words to our label sequences (Figure 1), we see that label sequences encode richer semantic meaning and get rid of sophisticated templates, since label sequences themselves can be standalone sentences.

In natural language inference (NLI) tasks[1] with two input sentences, our model input changes to "x_1? `[MASK]`, x_2" and label sequences can be "*I mean*" (entailment), "*For example*" (neutral), and "*However*" (contradiction).

Formally, we have a task-specific template $\mathcal{T}$[2] and a task-specific mapping $\mathcal{M}\colon \mathcal{Y} \to \mathcal{V}^+$ from the task label space $\mathcal{Y}$ to the label sequence space ($\mathcal{V}$ is the vocabulary of the model $\mathcal{L}$). Then, for a formulated example $\mathcal{T}(x)$ and its corresponding label sequences, we use the cross-entropy loss (the same way how T5 is trained)[3] to fine-tune the model. In inference, we compute the score of each class $y \in \mathcal{Y}$ as the auto-regressive log-probability of the corresponding label sequence:

$$q(\mathcal{M}(y) \mid \mathcal{T}(x)) = \sum_{j=1}^{|\mathcal{M}(y)|} \log P_{\mathcal{L}}\big(t_j \mid t_{1:j-1}, \mathcal{T}(x)\big), \quad (1)$$

where $P_{\mathcal{L}}$ denotes the output probability of the sequence-to-sequence model, $\mathcal{M}(y) = (t_1, \ldots, t_{|\mathcal{M}(y)|})$ is the corresponding label sequence tokens, and $t_{1:j-1}$ is $t_1, ..., t_{j-1}$.

3.2 Automatic label sequence generation

Thanks to the introduction of label sequences, manually-designed templates are no longer needed, and the goal of automatic prompt search is simply to construct a label sequence mapping $\mathcal{M}$ that performs well. Our proposed automatic label sequence

[1] We have details for all tasks in Appendix B.2.

[2] Unlike in the MLM case, the template here is simply the way to concatenate the input and the mask.

[3] For regression tasks like STS-B, we use the same method as Gao et al. (2021) to compute the loss instead.

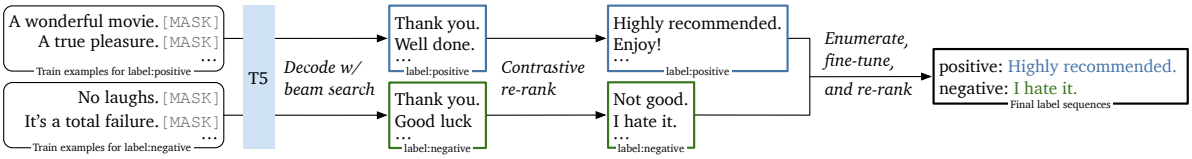

Figure 2: Illustration of AutoSeq. We first use T5 to generate label sequence candidates given each label's training instances; we then use *contrastive re-ranking* to get label sequences that are more label-specific; in the end we enumerate all the combinations and re-rank by the fine-tuning performance.

generation pipeline contains three steps (Figure 2): (1) candidate generation by using T5 and beam search; (2) re-ranking by contrastive probability; (3) enumerating label sequence combinations and re-ranking by fine-tuning performance.

We first use the T5 model and beam search to generate multiple sequence label candidates $\mathcal{S}^y \subset \mathcal{V}^+$ for each class y. Denote $\mathcal{D}^y_{\text{train}} \subset \mathcal{D}_{\text{train}}$ be the subset of all few-shot training data of class y, we find s^y that has the top scores by this equation:

$$\sum_{(x,y)\in\mathcal{D}^y_{\text{train}}} q(s^y \mid \mathcal{T}(x)), \tag{2}$$

where $q(\cdot)$ is defined as Eq. (1). Since the search space is too large, we decompose it to an auto-regressive decoding following Gao et al. (2021):

$$\sum_{j=1}^{|s^y|} \sum_{(x,y)\in\mathcal{D}^y_{\text{train}}} \log P_{\mathcal{L}}(s^y_j | s^y_{1:j-1}, \mathcal{T}(x)). \tag{3}$$

By using beam search, we can generate a large amount of label sequence candidates by just one decoding pass. However, we notice that it tends to generate similar generic label sequences across different classes, while we expect the label sequences to be distinguishable for each class. For example, in sentiment classification, both classes will get a generic candidate of "Thank you", which is coherent to be put at the mask but does not help with the classification (more discussion in Appendix F).

To eliminate the problem, we introduce the second step of our automatic pipeline, which re-ranks all the candidates based on the *contrastive probability* $\tilde{q}(s^y)$ of $s^y \in \mathcal{S}^y$:

$$\frac{\sum_{(x,y)\in\mathcal{D}^y_{\text{train}}} q(s^y \mid \mathcal{T}(x))}{|\mathcal{D}^y_{\text{train}}|} - \frac{\sum_{(x,y')\in\overline{\mathcal{D}}^y_{\text{train}}} q(s^y \mid \mathcal{T}(x))}{|\overline{\mathcal{D}}^y_{\text{train}}|}, \tag{4}$$

where $\overline{\mathcal{D}}^y_{\text{train}} = \mathcal{D}_{\text{train}} \backslash \mathcal{D}^y_{\text{train}}$.

Then, we define the score of a label mapping as the sum of corresponding $\tilde{q}(s^y)$ for each class y. To shorten the time for further re-ranking, we only select the top n mappings with the highest scores. Finally, we fine-tune the model over the top n label mapping candidates, and re-rank them to find the best one based on the few-shot development set, which has been proved critical in the label mapping selection (Gao et al., 2021).

4 Experiments

4.1 Main results

We use a `T5-base` v1.1 (Shazeer, 2020)[4] model and set the number of training examples per class as 16 in our experiments. Datasets and experiments details can be found in Appendix A and B. To make our results convincing, we compare to the following baselines in our few-shot setting: (1) parameter-efficient tuning – soft prompt tuning (Lester et al., 2021) and adapter tuning (Houlsby et al., 2019; Karimi Mahabadi et al., 2022) – which fixes the pre-trained model parameters and only tunes the soft prompt or adapter part; (2) standard fine-tuning; (3) manual prompts (Table D.1) proposed in Logan IV et al. (2021); (4) automatic label word search (AutoWord), which has the same setting as AutoSeq except that it is limited to only using one single token as a label word. This can be seen as an approximation of Auto-L in Gao et al. (2021). We also include the results from standard fine-tuning based on the full training set.

Table 1 shows our main results. First, **prompt-based fine-tuning can significantly beat standard fine-tuning**, either using manual prompts or generated ones, let alone parameter-efficient tuning. Our method AutoSeq achieves a 9.4% gain on average compared to standard fine-tuning.

Second, **AutoSeq achieves a 3.2% improvement on average compared to the manual prompts**, and performs significantly better in NLI tasks. However, for most of the sentiment classification tasks, though without engineering, the manual prompts can still outperform AutoSeq. We attribute it to the simplicity of these tasks, making the manual design of prompts more intuitive.

[4]The released original T5 models are also fine-tuned on downstream tasks while T5 v1.1 models exclude those tasks.

	SST-2 (acc)	SST-5 (acc)	MR (acc)	CR (acc)	MPQA (acc)	Subj (acc)	TREC (acc)	CoLA (Matt.)
Prompt tuning	51.4 (0.0)	24.9 (0.0)	50.6 (0.0)	50.7 (0.0)	50.0 (0.0)	59.9 (0.0)	22.6 (0.0)	-4.0 (0.0)
Adapter tuning	84.7 (3.2)	27.7 (4.8)	73.6 (3.9)	86.9 (1.4)	78.6 (3.6)	84.7 (3.3)	27.6 (3.6)	4.8 (4.8)
Fine-tuning	80.9 (2.0)	36.0 (2.2)	70.1 (7.1)	76.7 (6.8)	80.9 (2.6)	84.3 (4.4)	68.3 (14.7)	0.5 (6.3)
Prompt-based FT (Manual)	**91.2** (0.7)	**45.2** (1.5)	**85.4** (1.5)	**89.8** (1.5)	**85.1** (2.9)	89.1 (1.1)	80.0 (2.5)	0.7 (5.3)
Prompt-based FT (AutoWord)	87.6 (2.0)	40.4 (4.1)	82.1 (2.7)	87.0 (4.7)	75.1 (4.5)	87.4 (5.0)	82.4 (3.7)	**8.5** (4.5)
Prompt-based FT (AutoSeq)	89.8 (1.1)	42.3 (3.4)	83.9 (1.3)	87.2 (2.5)	82.5 (2.7)	**91.6** (1.9)	**85.2** (4.3)	7.6 (9.9)
Fine-tuning (Full train set)	93.3	56.1	89.3	86.9	89.0	96.2	97.0	30.1
	MNLI (acc)	**MNLI-mm (acc)**	**SNLI (acc)**	**QNLI (acc)**	**RTE (acc)**	**MRPC (F1)**	**QQP (F1)**	**STS-B (Pear.)**
Prompt tuning	34.6 (0.0)	34.2 (0.0)	34.1 (0.0)	54.2 (0.0)	47.3 (0.0)	**81.2** (0.0)	53.8 (0.0)	10.7 (0.0)
Adapter tuning	33.5 (1.4)	33.9 (1.8)	34.7 (1.3)	55.4 (2.5)	50.2 (2.0)	77.4 (2.3)	50.7 (5.2)	6.8 (2.8)
Fine-tuning	36.1 (2.3)	36.4 (2.6)	36.0 (3.0)	58.6 (2.5)	51.8 (2.7)	74.9 (5.2)	57.0 (3.5)	11.9 (2.8)
Prompt-based FT (Manual)	41.9 (3.4)	43.0 (3.6)	40.8 (1.6)	55.5 (3.1)	53.3 (3.1)	75.6 (7.0)	55.4 (1.8)	17.3 (9.5)
Prompt-based FT (AutoWord)	49.0 (4.7)	51.3 (4.6)	56.2 (8.4)	59.9 (4.7)	48.7 (2.4)	73.5 (6.3)	60.6 (4.3)	**30.0** (8.4)
Prompt-based FT (AutoSeq)	**51.8** (1.8)	**53.9** (2.0)	**62.7** (3.7)	**61.3** (4.0)	**55.3** (4.9)	72.3 (4.9)	**66.2** (2.6)	17.8 (13.4)
Fine-tuning (Full train set)	86.9	87.1	91.6	91.0	59.6	84.0	87.9	86.1
	BoolQ (acc)	**CB (F1)**	**COPA (acc)**	**MultiRC (F1)**	**ReCoRD (F1)**	**WiC (acc)**	**WSC (acc)**	**Average**
Prompt tuning	**59.5** (0.0)	36.4 (0.0)	47.0 (0.0)	54.4 (0.0)	16.3 (0.0)	50.0 (0.0)	**65.4** (0.0)	42.8
Adapter tuning	45.3 (1.5)	55.3 (9.0)	47.2 (3.7)	**59.1** (0.0)	23.2 (5.2)	51.7 (1.9)	60.2 (2.2)	50.1
Fine-tuning	48.1 (6.2)	66.4 (14.1)	47.4 (7.2)	**59.1** (0.0)	18.1 (2.4)	50.3 (2.8)	60.4 (5.1)	52.6
Prompt-based FT (Manual)	48.3 (5.5)	75.5 (8.6)	51.6 (1.5)	56.0 (3.3)	**56.6** (3.5)	52.5 (3.5)	63.5 (2.7)	58.8
Prompt-based FT (AutoWord)	50.1 (3.9)	66.1 (15.8)	49.8 (3.2)	57.9 (1.5)	**56.6** (3.5)	**53.1** (3.6)	62.1 (1.4)	59.8
Prompt-based FT (AutoSeq)	55.4 (8.1)	**76.6** (10.7)	**52.0** (6.8)	58.2 (0.9)	**56.6** (3.5)	52.6 (2.9)	62.1 (1.4)	**62.0**
Fine-tuning (Full train set)	64.1	92.1	52.0	59.1	76.0	59.1	66.3	77.4

Table 1: Our main results using `T5-base` (16 training examples per class). We report mean (and standard deviation) performance over 5 different splits. FT: fine-tuning; Manual: human-designed prompts (Table D.1); AutoWord: automatically searched single label words. The score marked as **bold** means the best performance in few-shot. The score marked with an underline means the best performance among automatic search methods.

	SST-2	SNLI	QQP	MultiRC
Manual with eng.	**90.8**	**64.1**	56.1	57.5
AutoSeq	89.8	62.7	**66.2**	**58.2**

Table 2: Manual prompts with engineering on large validation sets vs AutoSeq (Full results in Table E.2).

	SNLI	QQP	ReCoRD	WSC
RoBERTa-PET	43.3	53.4	42.7	55.0
T5-AutoSeq	**62.7**	**66.2**	**56.6**	**62.1**

Table 3: Sequence-to-sequence vs MLM prompting.

Third, **using AutoSeq leads to steady gains in a majority of tasks compared to AutoWord**, indicating that label sequences, which is only enabled by using sequence-to-sequence models, are more expressive than single label words.

The results indicate that automatic prompt generation, especially with template-free format and label sequences, is a promising path for prompt-based fine-tuning in low resource scenarios.

4.2 Analysis of prompt engineering

Table 2 compares manual prompts with considerable engineering efforts (Table E.1) to AutoSeq. In general, AutoSeq achieves on par performance with models using manual prompts across various types of tasks, illustrating the effectiveness of our method, especially when trial-and-error with large validation sets is impossible.

4.3 Analysis of different pre-trained models

To highlight the advantages of using sequence-to-sequence models, we also report the PET[5] results with `RoBERTa-base` in Table 3. We see that T5 performs better than RoBERTa by a large margin. Although the comparison is not fair[6] given T5 and RoBERTa are pre-trained with different corpora, we highlight the importance to have sequence-to-

[5]Without unlabeled corpora and ensemble.

[6]Surprisingly, `T5-base` has a lower GLUE average (84.67) than `RoBERTa-base` (86.35) with full-dataset.

sequence models in the world of prompt-based fine-tuning. Furthermore, for tasks like ReCoRD and WSC that require generation in prompting, T5 is perfectly fit for their output formats, while MLM models like RoBERTa require tricky workaround.

5 Conclusion

In this paper, we propose AutoSeq, a prompt-based fine-tuning method with (1) sequence-to-sequence models that enable free-form generation, (2) label sequences that significantly extend the prediction space, and (3) automatic prompt search that requires no human efforts for designing prompts. Comprehensive experiments show that AutoSeq significantly outperforms other prompt-based or parameter-efficient tuning methods. We hope AutoSeq further inspires research on exploring template-free prompt-based fine-tuning.

Acknowledgement

This work is supported by the National Key Research and Development Program of China (No. 2020AAA0106500). Zichun Yu conducted the experiments. Zichun Yu, Tianyu Gao, Zhengyan Zhang, Yankai Lin, and Zhiyuan Liu wrote the paper. Maosong Sun and Jie Zhou provided valuable suggestions to the research.

References

Roy Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second PASCAL recognising textual entailment challenge.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth PASCAL recognizing textual entailment challenge. In *TAC*.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of NAACL-HLT 2019*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*.

Joe Davison, Joshua Feldman, and Alexander M Rush. 2019. Commonsense knowledge mining from pretrained models. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The CommitmentBank: Investigating projection in naturally occurring discourse. To appear in proceedings of Sinn und Bedeutung 23. Data can be found at https://github.com/mcdm/CommitmentBank/.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional Transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

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

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Association for Computational Linguistics (ACL)*.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2022. Ppt: Pre-trained prompt tuning for few-shot learning. In *Proceedings of ACL*.

Adi Haviv, Jonathan Berant, and Amir Globerson. 2021. BERTese: Learning to speak to BERT. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3618–3623, Online. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning (ICML)*.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *ACM SIGKDD international conference on Knowledge discovery and data mining*.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association of Computational Linguistics (TACL)*.

Rabeeh Karimi Mahabadi, Luke Zettlemoyer, James Henderson, Marzieh Saeidi, Lambert Mathias, Veselin Stoyano, and Majid Yazdani. 2022. Perfect: Prompt-free and efficient few-shot learning with language models. In *Annual Meeting of the Association for Computational Linguistics*.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 3045–3059.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2011. The Winograd schema challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, volume 46, page 47.

Robert L Logan IV, Ivana Balažević, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel. 2021. Cutting down on prompts and parameters: Simple few-shot learning with language models. *arXiv preprint arXiv:2106.13353*.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Association for Computational Linguistics (ACL)*.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Association for Computational Linguistics (ACL)*.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: The word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of NAACL-HLT*.

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. *The Journal of Machine Learning Research (JMLR)*, 21(140).

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. In *International Conference on Computational Linguistics (COLING)*.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze questions for few-shot text classification and natural language inference. In *European Chapter of the Association for Computational Linguistics (EACL)*.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Noam Shazeer. 2020. Glu variants improve transformer.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Automatic prompt construction for masked language models. In *Empirical Methods in Natural Language Processing (EMNLP)*.

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 *Empirical Methods in Natural Language Processing (EMNLP)*.

Ellen M Voorhees and Dawn M Tice. 2000. Building a question answering test collection. In *the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems (NIPS)*.

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 *International Conference on Learning Representations (ICLR)*.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association of Computational Linguistics (TACL)*, 7.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3).

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems*, volume 34, pages 27263–27277. Curran Associates, Inc.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Association for Computational Linguistics (ACL)*.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2022. Differentiable prompt makes pre-trained language models better few-shot learners. In *International Conference on Learning Representations*.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. ReCoRD: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint 1810.12885*.

A Datasets

We use datasets from GLUE (Wang et al., 2019b), SuperGLUE (Wang et al., 2019a), and a number of other sentence classification datasets.

For SST-2 (Socher et al., 2013), SST-5 (Socher et al., 2013), MR (Pang and Lee, 2005), CR (Hu and Liu, 2004), MPQA (Wiebe et al., 2005), Subj (Pang and Lee, 2004), TREC (Voorhees and Tice, 2000), CoLA (Warstadt et al., 2019), MNLI (Williams et al., 2018), SNLI (Bowman et al., 2015), QNLI (Rajpurkar et al., 2016), RTE (Dagan et al., 2005; Bar Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009), MRPC (Dolan and Brockett, 2005), QQP[7] and STS-B (Cer et al., 2017), we refer to Gao et al. (2021) for their test settings. For BoolQ (Clark et al., 2019), CB (De Marneffe et al., 2019), COPA (Roemmele et al., 2011), MultiRC (Khashabi et al., 2018), ReCoRD (Zhang et al., 2018), WiC (Pilehvar and Camacho-Collados, 2019) and WSC (Levesque et al., 2011), we take their original development sets as the test sets.

B Experimental Details

B.1 Hyper-parameter selection

We take batch sizes from {2, 4, 8} for all few-shot experiments. For fine-tuning, we take learning rates from {7e-5, 1e-4, 2e-4}. For prompt-based fine-tuning, we take learning rates from {2e-5, 6e-5, 9e-5}, which are selected by pre-experiments on the SST-2 and SNLI datasets. For each trial, we follow Gao et al. (2021) and set the training steps as 1000, validation steps as 100, then pick the best model based on the validation results.

B.2 Automatic label sequence generation

For automatic label sequence generation, we use `T5-large`, limiting the maximum length of 20 tokens (AutoSeq) and one token (AutoWord). Considering the trade-off between efficiency and effectiveness, we set beam search width to 50 and set n to 20. Given that the number of experiments is relatively large in automatic generation, we fix the batch size as 8 and the learning rate as 6e-5 when training the model over the top n label mappings.

Besides our $\mathcal{T}$ for one-sentence classification tasks[8] and NLI tasks mentioned in Section 3, we also design more $\mathcal{T}$, always a simple concatenation of input fields and the `[MASK]` token, for other complicated tasks. For BoolQ, $\mathcal{T}$ is "x_1? `[MASK]`, x_2". For COPA, $\mathcal{T}$ is "x_1 x_2? x_3? `[MASK]`, x_4". For MultiRC, $\mathcal{T}$ is "x_2 `[MASK]`, x_3 x_1". For WiC, $\mathcal{T}$ is "x_1 x_2 'x_3' `[MASK]`". Since ReCoRD and WSC can be easily and intuitively transformed into fill-in-the-blank tasks, we follow Schick and Schütze (2021b) and do not process the automatic label sequence generation for them. To make the input closer to pre-training, we refer to Gao et al. (2021) for the implementation details of prompts.

[7] https://www.quora.com/q/quoradata/

[8] One exception: MPQA consists of incomplete sentences, so we adopt manual template without engineering.

	SST-2	SNLI	QQP	MultiRC
Manual w/o templates	90.2	**64.1**	**57.7**	56.2
Manual with eng.	**90.8**	**64.1**	56.1	**57.5**

Table C.1: Comparison between manual label words without templates (so the input is the same as AutoSeq), and manual prompts with deliberate engineering.

C Analysis of Templates

Table C.1 gives the results of using only manual label words with engineering and no templates (so the mask token is concatenated the same way as AutoSeq). This can be seen as the *null prompts* from Logan IV et al. (2021). Our results further validate that null prompts perform comparably or even better to manual prompts in most cases.

D Manual Prompts

Table D.1 demonstrates all the manual templates and label words adopted by us. We basically follow Logan IV et al. (2021) for these prompts. For the tasks that are not covered by Logan IV et al. (2021), we manually write one prompt for each of them, using only our intuition.

E Manual Prompts with Engineering

Table E.1 gives all the manual templates and label words with careful engineering (Gao et al. (2021) for GLUE and Schick and Schütze (2021b) for SuperGLUE) that we use in our experiments.

Table E.2 compares the full results of manual prompts with engineering to our AutoSeq. Overall, AutoSeq performs comparably or even better compared with manual prompts, particularly for tasks where developing solid manual prompts is less instinctive (e.g., TREC, QNLI, QQP and COPA).

Task	Template	Label words
SST-2	<S_1> Overall my impression is `[MASK]` .	positive: good, negative: bad
SST-5	<S_1> Overall my impression is `[MASK]` .	v.positive: very good, positive: good, neutral: not bad, negative: bad, v.negative: very bad
MR	<S_1> Overall my impression is `[MASK]` .	positive: good, negative: bad
CR	<S_1> Overall my impression is `[MASK]` .	positive: good, negative: bad
MPQA	<S_1> Overall my impression is `[MASK]` .	positive: good, negative: bad
Subj	<S_1> The sentence is `[MASK]` .	subjective: subjective, objective: objective
TREC	<S_1> The question is about `[MASK]` .	abbreviation: abbreviation, entity: entity, description: description human: human, location: location, numeric: numeric
COLA	<S_1> The grammar is `[MASK]` .	grammatical: acceptable, not_grammatical: unacceptable
MNLI	Premise: <S_2> Hypothesis: <S_1> Label: `[MASK]`	entailment: yes, netural: maybe, contradiction: no
SNLI	Premise: <S_2> Hypothesis: <S_1> Label: `[MASK]`	entailment: yes, netural: maybe, contradiction: no
QNLI	Question: <S_1> Sentence: <S_2> Label: `[MASK]`	entailment: yes, not_entailment: no
RTE	Premise: <S_1> Hypothesis: <S_2> Label: `[MASK]`	entailment: yes, not_entailment: no
MRPC	<S_1> and <S_2> are the `[MASK]` .	equivalent: same, not_equivalent: different
QQP	<S_1> and <S_2> are the `[MASK]` .	equivalent: same, not_equivalent: different
STS-B	<S_1> and <S_2> are the `[MASK]` .	y_u: same, y_l: different
BoolQ	Passage: <S_1> Question: <S_2> Answer: `[MASK]` .	True: true, False: false
CB	Premise: <S_1> Hypothesis: <S_2> Label: `[MASK]`	entailment: yes, netural: maybe, contradiction: no
COPA	Premise: <S_3> Question: <S_4> Choice1: <S_1> Choice2: <S_2> Answer: `[MASK]` .	Alternative 1: Choice1, Alternative 2: Choice2
MultiRC	Paragraph: <S_1> Question: <S_2> Answer: <S_3> Label: `[MASK]`	True: true, False: false
ReCoRD	<S_1> <S_3>	
WiC	'<S_3>' in <S_1> and '<S_3>' in <S_2> are the `[MASK]` .	True: same, False: different
WSC	<S_1> <S_3> is `[MASK]` .	

Table D.1: Manual templates and label words following Logan IV et al. (2021). Note that for ReCoRD and WSC we follow Schick and Schütze (2021b) and do not design the label words for them.

F Automatically Generated Label Sequences

We demonstrate the top 1 automatically generated label sequences before and after re-ranking with contrastive probability for all tasks in Table F.1. It can be observed that our contrastive probability draws a strong distinction between different classes, especially for those multi-classification tasks like SST-5 and TREC, in which our beam search tends to find the same sequence whatever the class is.

Generally speaking, the generated results after re-ranking conform with our intuition in a majority of single and two-sentence tasks. For more complicated ones, such as COPA and WiC, the generated label sequences can be counterintuitive, calling for a more elegant solution in the future.

Task	Template	Label words
SST-2	<S_1> It was `[MASK]` .	positive: great, negative: terrible
SST-5	<S_1> It was `[MASK]` .	v.positive: great, positive: good, neutral: okay, negative: bad, v.negative: terrible
MR	<S_1> It was `[MASK]` .	positive: great, negative: terrible
CR	<S_1> It was `[MASK]` .	positive: great, negative: terrible
MPQA	<S_1> It was `[MASK]` .	positive: great, negative: terrible
Subj	<S_1> This is `[MASK]` .	subjective: subjective, objective: objective
TREC	`[MASK]` : <S_1>	abbreviation: Expression, entity: Entity, description: Description human: Human, location: Location, numeric: Number
COLA	<S_1> This is `[MASK]` .	grammatical: correct, not_grammatical: incorrect
MNLI	<S_1> ? `[MASK]` , <S_2>	entailment: Yes, netural: Maybe, contradiction: No
SNLI	<S_1> ? `[MASK]` , <S_2>	entailment: Yes, netural: Maybe, contradiction: No
QNLI	<S_1> ? `[MASK]` , <S_2>	entailment: Yes, not_entailment: No
RTE	<S_1> ? `[MASK]` , <S_2>	entailment: Yes, not_entailment: No
MRPC	<S_1> `[MASK]` , <S_2>	equivalent: Yes, not_equivalent: No
QQP	<S_1> `[MASK]` , <S_2>	equivalent: Yes, not_equivalent: No
STS-B	<S_1> `[MASK]` , <S_2>	y_u: Yes, y_l: No
BoolQ	<S_1> Question: <S_2> ? Answer: `[MASK]` .	True: Yes, False: No
CB	<S_1> ? `[MASK]` , <S_2>	entailment: Yes, netural: Maybe, contradiction: No
COPA	<S_1> or <S_2> ? <S_3> , <S_4> `[MASK]` .	
MultiRC	<S_1> Question: <S_2> Is it <S_3> ? `[MASK]` .	True: Yes, False: No
ReCoRD	<S_1> <S_3>	
WiC	<S_1> <S_2> Does <S_3> have the same meaning in both sentences? `[MASK]`	True: Yes, False: No
WSC	<S_1> The pronoun <S_3> refers to `[MASK]` .	

Table E.1: Manual templates and label words with deliberate engineering that we use in our experiments. Note that for COPA, ReCoRD and WSC, we follow Schick and Schütze (2021b) and do not design the label words for them.

	SST-2 (acc)	**SST-5** (acc)	**MR** (acc)	**CR** (acc)	**MPQA** (acc)	**Subj** (acc)	**TREC** (acc)	**CoLA** (Matt.)
Manual with eng.	**90.8** (0.4)	**47.2** (2.4)	**86.1** (0.6)	**90.4** (1.0)	**84.1** (2.4)	91.4 (1.2)	81.3 (4.8)	**9.6** (11.6)
AutoSeq	89.8 (1.1)	42.3 (3.4)	83.9 (1.3)	87.2 (2.5)	82.5 (2.7)	**91.6** (1.9)	**85.2** (4.3)	7.6 (9.9)
	MNLI (acc)	**MNLI-mm** (acc)	**SNLI** (acc)	**QNLI** (acc)	**RTE** (acc)	**MRPC** (F1)	**QQP** (F1)	**STS-B** (Pear.)
Manual with eng.	**55.3** (2.3)	**57.3** (2.4)	**64.1** (4.1)	59.7 (3.4)	**59.1** (4.3)	71.2 (6.6)	56.1 (2.5)	17.7 (12.5)
AutoSeq	51.8 (1.8)	53.9 (2.0)	62.7 (3.7)	**61.3** (4.0)	55.3 (4.9)	**72.3** (4.9)	**66.2** (2.6)	**17.8** (13.4)
	BoolQ (acc)	**CB** (F1)	**COPA** (acc)	**MultiRC** (F1)	**ReCoRD** (F1)	**WiC** (acc)	**WSC** (acc)	**Average**
Manual with eng.	**57.5** (2.1)	**79.7** (5.5)	48.8 (2.5)	57.5 (1.6)	**56.6** (3.5)	**53.4** (4.0)	**62.5** (5.2)	**62.5**
AutoSeq	55.4 (8.1)	76.6 (10.7)	**52.0** (6.8)	**58.2** (0.9)	**56.6** (3.5)	52.6 (2.9)	62.1 (1.4)	62.0

Table E.2: Comparison between manual prompts with engineering and our automatically searched label sequences.

Task	Before re-ranking	After re-ranking
SST-2	(positive/negative) Highly recommended./Thank you.	Highly recommended./Sigh.
SST-5	(very positive/positive/neutral/negative/very negative) Highly recommended./Highly recommended././ Highly recommended./Highly recommended.	A must see./I love this movie./Enjoy!/Sigh./Not recommended.
MR	(positive/negative) Highly recommended./Highly recommended.	Highly recommended./Not for me.
CR	(positive/negative) I love it./Thank you.	I love it./I hate it.
MPQA	(positive/negative) ./.	./Why?
Subj	(subjective/objective) I love it./What do you think?	I love it./The rest is history.
TREC	(abbreviation/entity/description/human/location/numeric) Why?/Why?/./Why?/Why?/.	Discuss!/What is it?/For?/Who is?/USA./15?
CoLA	(grammatical/not_grammatical) ./.	Enjoy!/.
MNLI	(entailment/neutral/contradiction) Yes/Yes/No	I mean/For example/However
SNLI	(entailment/neutral/contradiction) Yes/Yes/Yes	Yes/In this video/Next
QNLI	(entailment/not_entailment) In fact/In fact	In the past/Also
RTE	(entailment/not_entailment) Yes/Yes	Yes/However
MRPC	(equivalent/not_equivalent) Yes/Yes	Yes/Meanwhile
QQP	(equivalent/not_equivalent) Also/Also	So/Also
STS-B	(y_u/y_l) Yes/Yes	Yes/Also
BoolQ	(True/False) Yes/Yes	If so/No
CB	(entailment/neutral/contradiction) Yes/Yes/I mean	Indeed/A: Yes/A: No
COPA	(Alternative 1/Alternative 2) No/No	No/Yes
MultiRC	(True/False) Yes/Yes	The answer is/Also
WiC	(True/False) is used./is used.	is used./is an adjective.

Table F.1: Top 1 automatically generated label sequences before and after re-ranking with contrastive probability for all tasks based on one few-shot split.