Strings from the Library of Babel: Random Sampling as a Strong Baseline for Prompt Optimisation

Yao Lu[†] **Jiayi Wang**[†] **Raphael Tang**[‡] **Sebastian Riedel**[†] **Pontus Stenetorp**[†]
[†]University College London [‡]Comcast AI Technologies
{yao.lu,s.riedel,p.stenetorp}@cs.ucl.ac.uk
ucabj45@ucl.ac.uk raphael_tang@comcast.com

Abstract

Recent prompt optimisation approaches use the generative nature of language models to produce prompts – even rivalling the performance of human-curated prompts. In this paper, we demonstrate that randomly sampling tokens from the model vocabulary as "separators" can be *as effective* as language models for prompt-style text classification. Our experiments show that random separators are competitive baselines, having less than a 1% difference compared to previous self-optimisation methods and showing a 12% average relative improvement over strong human baselines across nine text classification tasks and eight language models. We further analyse this phenomenon in detail using three different random generation strategies, establishing that the language space is rich with potentially good separators, with a greater than 40% average chance that a randomly drawn separator performs better than human-curated separators. These observations challenge the common assumption that an effective prompt should be human readable or task relevant and establishes a strong baseline for prompt optimisation research.[1]

1 Introduction

Pre-trained large language models (PLMs, Peters et al., 2018; Devlin et al., 2019; Liu et al., 2019; Radford et al., 2019; Touvron et al., 2023a,b; Jiang et al., 2023) have demonstrated remarkable performance when conditioned with appropriate context (Petroni et al., 2019, 2020; Jiang et al., 2020; Shin et al., 2020; Davison et al., 2019). For instance, when given a query along with the phrase "Let's think step by step," such models are capable of solving reasoning tasks (Kojima et al., 2022; Wei et al., 2022). These special tokens, often called "*separators*", are usually placed at the end of the input data or at the beginning of the output (Table 1).

[1]Our implementation is publicly available at https://github.com/yaolu/random-prompt

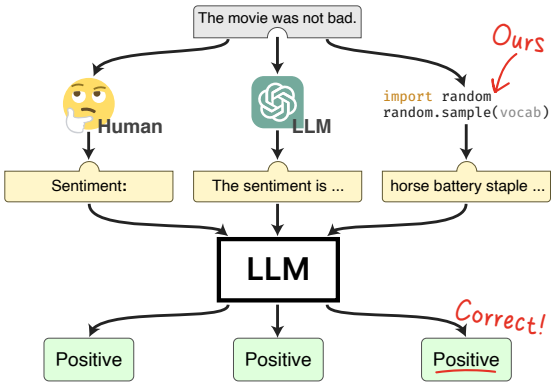

Figure 1: Illustration of our approach when searching for good separators for a sentiment classification task. Unlike relying on human knowledge or using external large language models to suggest alternatives, we find that randomly selected separators from the vocabulary can also yield good performance.

Recent work found that including thinking-style separators such as "take a deep breath" can significantly enhance reasoning performance (Yang et al., 2023). Similarly, another study discovered that simply using "SOLUTION:" is even more effective (Fernando et al., 2023). These findings suggest that the language space for separators might still be under-explored, with many effective options yet to be identified. A common framework employed in this line of research involves starting with thinking-style separators, using a language model to generate alternatives, and then selecting effective separators based on certain criteria (Zhou et al., 2022; Yang et al., 2023; Fernando et al., 2023; Guo et al., 2023). This optimisation-style framework has shown promise in automating the exploration of the language space, which addresses the challenge of relying on human expertise to develop task-specific separators.

Most existing methods, particularly those applied to reasoning tasks, assume that effective separators should be closely related to the task or con-

Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 2221–2231
June 16-21, 2024 ©2024 Association for Computational Linguistics

text (Fernando et al., 2023; Guo et al., 2023; Shi et al., 2022). However, perhaps counter-intuitively, we find that a performant separator does not necessarily have to be task-relevant or even coherent. Sometimes, even tokens chosen at random from the vocabulary can improve performance as much as semantically meaningful phrases.

PROMPT EXAMPLES	
HUMAN	This is a good movie. Answer: positive
RANDOM	This is a good movie. !@#?& positive

Table 1: Examples of prompts used in our evaluation, where the highlighted text are the separators.

As shown in Figure 1 and Table 1, we can achieve similar performance to that of human-optimised prompts by randomly selecting separators from the vocabulary. This suggests that random separators can serve as a competitive baseline, sometimes even matching the performance of previous methods (Zhou et al., 2022; Kojima et al., 2022; Yang et al., 2023; Guo et al., 2023) for prompt-style text classification tasks. Our exploration across seven different models further shows that this behaviour is universal across both pre-trained and instruction-tuned language models (Table 6), and seems to be a fundamental characteristic of in-context learning (Brown et al., 2020).

We further analyse this phenomenon with three random separator generation strategies, revealing that there are many performant separators in the language space, suggesting that previous research underestimated the effectiveness of randomised prompts outside of reasoning tasks. This observation breaks common assumptions such as that good separators need to be task relevant, coherent, and context dependent (Shin et al., 2020; Shi et al., 2022). Experimental results show that using random separators attains a 12% average relative improvement across nine classification tasks on eight language models, compared to human-curated separators. To summarise, our contributions are as follows:

1. We show that random separators can be as effective as human-curated prompts for prompt-style text classification.
2. We analyse three randomised separator generation strategies, which do not require an instruction-following language model, and show on-average a 12% relative improvement over human baselines.
3. We find that random separators are as competitive as previously proposed language model approaches to generate alternative prompts, suggesting that their effectiveness appear greater than what it would have appeared given this strong random baseline.

2 Random Separator Optimisation

We propose a random separator optimisation framework (Figure 2) to find effective separators based on random sampling. The framework consists of three components: 1) random separator generation, 2) separator evaluation, and 3) separator selection.

2.1 Definition of Separator

The term "separators" is inspired by the well-known BERT `[SEP]` token in order to differentiate from general wordings such as "suffix" or "prefix". We use this term to provide readers with better semantics and also to avoid ambiguity.

2.2 Random Separator Generation

In this work, we seek to answer the following questions which relate to common assumptions adopted in prompt optimisation research:

- Should effective separators be task-relevant, or should they be closely tied to the existing context information?
- To what degree are language models needed for prompt optimisation?

To answer these questions, we propose three strategies for generating random separators from language model free and context-free, to language model dependent and context relevant. The core difference between these three random strategies are summarised in Table 2, with a more detailed illustration provided in Table 3.

	Language Model	Task Relevant
Random Vocabulary	✗	✗
Random w/o Context	✓	✗
Random with Context	✓	✓

Table 2: Random generation methods

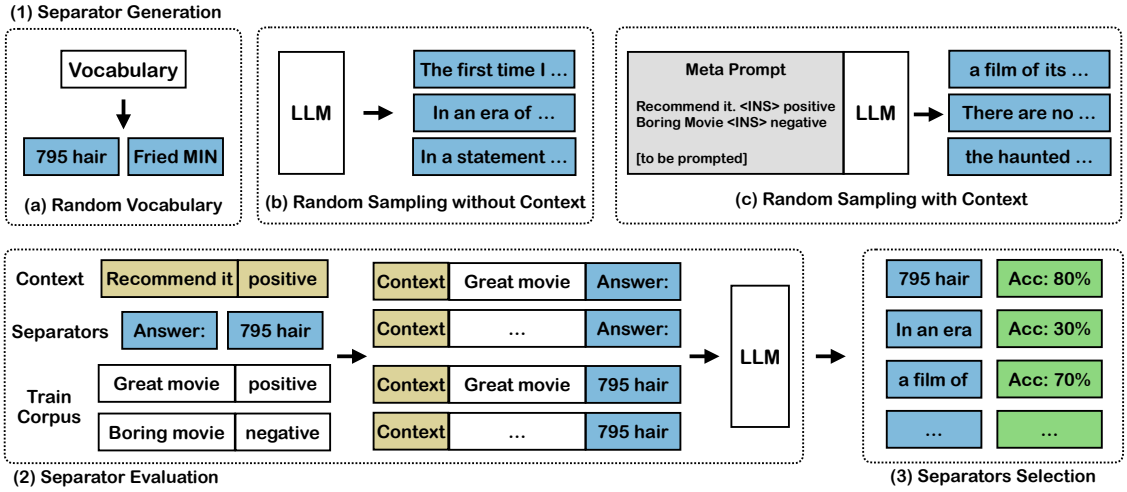

Figure 2: Our random separator optimisation procedure.

Sampling randomly across the vocabulary. AutoPrompt (Shin et al., 2020) suggests that effective separators may appear to be random strings, showing the potential of identifying a good separator within the random space. To investigate this, we design an approach to generate random separators that are context-free, task-agnostic, and do not rely on any language model for their generation. Essentially, we select random tokens from the vocabulary until a predetermined string length limit is reached.

Sampling from a language model without context. This method involves drawing samples from the language model's prior distribution, a process that is both context-free and task-agnostic. We employ this random generation method to evaluate whether separator coherence contributes to performance, especially when compared to random samples at the vocabulary level.

Sampling from a language model with context. The creation of task-relevant separators may enhance performance. For instance, incorporating thinking-style phrases in a reasoning task has proven highly effective. In OPRO (Yang et al., 2023), the authors highlight that including samples from the training data in the meta-prompt leads to consistent improvements. Thus, to assess the impact of task relevance on separator generation, we integrate a few training samples from the training corpus into the meta-prompt to refine the semantic space of the separator.

2.3 Separator Evaluation

We demonstrate the evaluation process in Figure 2. In line with prior work (Fernando et al., 2023; Yang et al., 2023), a small set of labelled data, denoted as the training corpus $T = \{(x_i, y_i)\}, i = 1, ..., n$, is available. Here, x_i and y_i represent the sentence and label of the i^{th} sample, respectively. We also define a transformation $\mathcal{T}$, which maps label y_i to text. In contrast to supervised learning settings that require a large volume of data for training, we only need a limited set of examples.[2] To evaluate a given separator s, we perform string concatenation ($\oplus$) of the each input sentence from the training corpus T with the separator. As part of in-context learning settings, where demonstrations may be necessary for some tasks, we also take into account a context c, which has identical structure to the linearised sequence. For each data point (x_i, y_i) in T, we prompt the pretrained language model to generate the predicted label $\hat{y}_i = \text{argmax}_{v \in V} P(v|c \oplus x_i \oplus s; \theta)$, where θ represents the parameters of the pretrained language model and V denotes the vocabulary space. For a classification task, we compute the classification accuracy as the corresponding separator score, denoted by m. It is worth noting that the metric does not necessarily need to be differentiable, allowing for the direct optimisation of discrete metrics such as word overlap ratio, etc.

2.4 Separator Selection

Despite the random separator generation steps being independent, we retain the term "iteration" in our methodology. This allows for a consistent comparison with other methods that use an iteration-

[2]For all experiments, the training set size is $n = 64$, unless explicitly mentioned otherwise.

sensitive meta-prompt. During the iteration process, we evaluate the generated prompt and keep track of the most effective separators at each step. The sampling continues until we reach a predefined sampling budget limit, denoted as k. By the end of the training process, we accumulate a set of separators[3] and their corresponding scores, represented as $S = \{(s_i, m_i)\}, i = 1, ..., k$. The separator that yields the highest score in this set is then selected for evaluation on the test set.

3 Experimental Setup

3.1 Datasets and Models

In line with previous studies (Gao et al., 2020; Zhao et al., 2021; Lu et al., 2022), we use nine text classification datasets (Table 4). For training we use 64 samples per dataset and for evaluation we use the sub-sampled test set from Lu et al. (2022).

In contrast to previous work, our random separator optimisation methods do not require an instruction-tuned language model. This allows us to test our methods using both standard pre-trained language models and instruction-tuned language models. As detailed in Table 5, our experiment uses four pre-trained language models and four instruction-tuned language models, in total eight models with varying structure and training data.

3.2 Optimisation Settings

Separator generation methods. As detailed in Section 2, we have proposed three different random separator generation approaches. For comparative analysis, we include the OPRO (Yang et al., 2023) method in our study. We also adapt OPRO's meta-prompt by omitting the instructional text, creating an in-context learning variant (OPRO-ICL). This allows fair comparison between random generation and other methods on non-instructionally tuned models. We also compare our methods against two human-level prompt optimisation methods, MI (Zhang et al., 2023) and NI (Mishra et al., 2021), as well as two automatic prompt optimisation methods, APE (Zhou et al., 2022) and EvoPrompt (Guo et al., 2023), under our experimental settings[4] described in Guo et al. (2023). In total, we employ nine distinct separator generation methods as baselines in our main experiment (Tables 6 and 7).

Baseline separators. To better understand how much improvement separator optimisation can achieve, we use the human curated separator "Answer:" and random strings such as "Foo Bar"[5] and zero-shot chain-of-thoughts (ZS-CoT) (Kojima et al., 2022), "Let's think step by step", as baseline separators.

Initialisation. The choice of a starting point does not affect random methods. In cases where a meta-prompt requires a starting point, such as in OPRO, we use "Answer:" as the starting point.

Prompting settings. We use one-shot examples as context to prompt language models during both training and test stages. When context is necessary for separator generation, we provide three randomly chosen training examples. We set the generation temperature to 1.0 and use a temperature of 0.0 for prompt-based text classification. For training of OPRO and OPRO-ICL, we set a maximum of 40 optimisation steps and generate four candidate separators each step. For our random methods, we generate 160 candidate separators and select the best one for evaluation.

4 Results and Discussion

We report our results in Table 6 and demonstrate the effectiveness of randomly sampled separators across all tasks. In addition, we compare four additional baseline methods in Table 7 using the experimental settings described by Guo et al. (2023).

4.1 Random Separators are Strong Baselines

Unnatural separators are effective. To our surprise, we find that even separators chosen at random from the vocabulary can substantially improve performance. Table 6 shows that the *Random Vocabulary* method yields, on average, a 10% relative improvement across nine benchmark datasets compared to human-curated separators. *Random Vocabulary* shows only marginal differences (less than 1% difference) compared to self-optimisation style methods (Yang et al., 2023; Zhou et al., 2022). For evolutionary methods (Guo et al., 2023), *Random Vocabulary* shows a 3.4% difference from the best EvoPrompt result (Table 7), with the most significant drop (2.1%) occurring on the SubJ dataset.

Since our goal is not to produce a state-of-the-art method, but rather to establish a strong baseline,

[3] We generate up to 160 separators in all experiments.

[4] Due to the difficulty of accurately reproducing these methods, we adapt our settings to fit the EvoPrompt setup instead.

[5] Here, "Foo Bar" represents a random string sampled from the vocabulary and not the literal usage of this exact string.

	Description	Prompt for Generating New Separators
Random Baseline	Random string without optimisation steps	-
Human Baseline (Lu et al., 2022)	Use "Answer:" as it is widely used	-
ZS-CoT (Kojima et al., 2022)	Think-style phrase	-
OPRO (Yang et al., 2023)	The meta-prompt in OPRO consists of a natural language problem description and instructions to generate new solutions based on previously found solutions.	[Instructions] I have some texts along with their ... [Historical Solutions] text: think stepwise score: 55 ... [Instructions] The following exemplars show ... [Context] should not be missed <INS>positive ... [Instructions] Write your new text that is different ... text: [to be prompted]
OPRO-ICL	We remove all instructions from OPRO to create an in-context learning variant.	[Historical Solutions] text: think stepwise score: 55 ... [Context] should not be missed <INS>positive ... text: [to be prompted]
Random Vocabulary	Randomly sample tokens across vocabulary to generate separators.	[to be sampled] [such as "!@#$%^&*"]
Random w/o Context	Draw samples from LLM's prior distribution.	[to be prompted]
Random with Context	Prompting language model with few examples as context. Similar to OPRO (Yang et al., 2023), we use three randomly sampled examples from training data as context.	[Context] should not be missed <INS>positive curiously depressing <INS>negative text: [to be prompted]

Table 3: Separator generation methods used for our main experiment. Text wrapped with square brackets are not included in prompt.

Dataset	# of Classes	Avg. Len.	Balanced
SST-2 (Socher et al., 2013)	2	12.4	✓
SST-5 (Socher et al., 2013)	5	23.1	✗
MR (Pang and Lee, 2005)	2	25.7	✓
CR (Hu and Liu, 2004)	2	22.1	✓
MPQA (Wiebe et al., 2005)	2	3.9	✓
Subj (Pang and Lee, 2004)	2	28.9	✓
TREC (Voorhees and Tice, 2000)	6	11.6	✗
AGNews (Zhang et al., 2015)	4	53.8	✓
DBPedia (Zhang et al., 2015)	14	65.5	✓

Table 4: Statistics of evaluation datasets, average length is calculated based on GPT-2 sentence-piece length.

Model	# of Parameters	Instruction Tuned
GPT2 Large (Radford et al., 2019)	0.8B	✗
GPT2 XL (Radford et al., 2019)	1.5B	✗
Mistral 7B (Jiang et al., 2023)	7B	✗
Mistral 7B Instruct (Jiang et al., 2023)	7B	✓
Llama-Alpaca 7B (Taori et al., 2023)	6.7B	✓
Llama2 7B (Touvron et al., 2023b)	6.7B	✗
Llama2 7B Chat (Touvron et al., 2023b)	6.7B	✓
ChatGPT (GPT-3.5 Turbo, 0613 version)	–	✓

Table 5: Language models used in experiments.

the performance of *Random Vocabulary* suggests that previous progress and the weight of contributions in prompt optimisation might be overestimated. Notably, our method does not depend on a language model for generation (Table 2). This result challenges the common practice in prompt optimisation where large language models are used for creating alternative prompts.

Natural language separators are effective, but may not be essential. Human creation of separators often takes coherence into account. For instance, in sentiment classification, a model might struggle to predict "positive" or "negative" due to high sequence perplexity, but introducing a separator like "It is positive" can align predictions with the model's pre-training objective. Given the surprisingly competitive quality of *Random Vocabulary*, we further explore the potential benefits of coherence towards prompt optimisation. We design another simple strategy, *Random w/o Context*, which samples from the language model's prior to generate natural language phrases as separators. The key difference between *Random w/o Context* and *Random Vocabulary* is that the former consists of natural language phrases, whereas the latter may not. Both methods generate separators that are statistically unlikely to be relevant to the task and context.

Experimental results show that the *Random w/o Context* is significantly better than human baselines (12% relative improvement) and nearly on par with previous state-of-the-art prompt optimisation methods (less than a 1% difference). This suggests that *Random w/o Context* is also a simple but strong baseline for prompt optimisation. When comparing the natural and unnatural separators, our analysis shows a mere 0.5% relative difference between *Random Vocabulary* and *Random w/o Context*. Such a small margin suggests that, while the *Random w/o Context* approach is competitive, coherence does not appear to be a critical factor.

Task information in separator generation provides slight improvements. *Random with Con-*

	SST-2	SST-5	DBPedia	MR	CR	MPQA	Subj	TREC	AGNews	Avg. (Rel. Δ%)
Finetuning (Full)	95.0	58.7	99.3	90.8	89.4	87.8	97.0	97.4	94.7	90.0
GPT2-LARGE 0.8B										
`Answer:`	74.8	27.0	37.5	54.5	69.8	63.0	65.9	9.5	52.9	50.5 (0.0)
`Foo Bar`	57.0	36.8	34.5	53.3	63.5	51.6	53.4	21.0	51.2	46.9 (−7.1)
ZS-CoT	61.5	26.6	37.3	59.1	52.8	32.8	51.3	15.6	63.7	44.5 (−11.9)
OPRO	77.1	43.1	44.4	81.0	75.5	64.8	**73.1**	36.5	62.7	62.0 (22.8)
OPRO-ICL	81.6	**44.1**	43.8	68.8	74.5	65.9	73.0	36.6	70.8	62.1 (23.0)
Random Vocabulary	80.6	43.4	40.2	77.0	76.8	62.7	73.0	34.5	**72.2**	62.3 (23.4)
Random w/o Context	77.2	41.9	**45.7**	75.5	**78.6**	67.2	72.4	**39.5**	66.8	62.8 (24.4)
Random with Context	**82.0**	43.9	42.9	**81.6**	73.9	**69.5**	71.4	35.8	71.2	**63.6** (25.9)
GPT2-XL 1.5B										
`Answer:`	72.3	37.7	38.3	69.5	60.8	59.2	61.2	7.2	46.5	50.3 (0.0)
`Foo Bar`	40.3	40.5	40.4	49.5	56.4	47.7	56.6	17.1	57.0	45.1 (−10.3)
ZS-CoT	39.8	27.7	41.5	42.6	43.7	49.4	55.2	16.6	56.3	41.4 (−17.7)
OPRO	80.0	44.6	47.0	79.3	78.0	68.6	75.6	29.8	72.0	63.9 (27.0)
OPRO-ICL	**82.9**	45.2	47.4	81.0	78.0	69.9	**78.8**	26.1	69.6	**64.3** (27.8)
Random Vocabulary	73.1	**45.9**	**47.6**	71.4	78.4	65.5	77.1	25.5	69.3	61.5 (22.3)
Random w/o Context	72.0	40.5	44.6	75.0	**79.3**	**70.8**	72.8	**35.6**	68.1	62.1 (23.5)
Random with Context	82.1	41.7	44.2	**81.8**	78.5	64.3	73.4	32.9	**74.5**	63.7 (26.6)
MISTRAL 7B										
`Answer:`	85.5	46.6	63.7	89.0	**92.2**	61.6	69.7	34.3	82.1	69.4 (0.0)
`Foo Bar`	61.0	44.0	63.7	72.7	87.1	48.2	55.9	34.8	79.5	60.8 (−12.4)
ZS-CoT	52.6	44.5	66.7	56.9	80.3	48.5	54.5	36.3	76.3	57.4 (−17.3)
OPRO	86.3	46.8	71.9	92.7	87.4	**79.5**	82.7	**60.8**	**83.1**	76.8 (10.7)
OPRO-ICL	91.0	48.2	72.7	**93.5**	90.6	76.2	**86.3**	59.9	82.9	**77.9** (12.2)
Random Vocabulary	87.3	**49.4**	70.1	93.1	85.8	76.2	80.7	55.7	81.6	75.5 (8.8)
Random w/o Context	91.4	48.7	**73.8**	91.6	86.3	74.4	79.8	60.5	80.9	76.4 (10.1)
Random with Context	**92.7**	47.9	74.1	92.9	87.8	67.0	84.4	59.9	82.9	76.6 (10.4)
MISTRAL 7B INSTRUCT										
`Answer:`	86.5	38.8	83.9	86.0	86.4	75.1	66.6	63.0	79.8	74.0 (0.0)
`Foo Bar`	85.1	39.7	82.3	85.3	85.7	75.5	63.8	63.8	78.8	73.3 (−0.1)
ZS-CoT	83.8	39.8	79.4	83.9	87.3	75.5	66.6	67.2	79.3	73.6 (0.0)
OPRO	89.0	40.2	82.8	88.0	84.3	**81.3**	68.3	**67.3**	81.6	75.9 (2.6)
OPRO-ICL	89.1	**41.7**	83.7	**90.2**	87.7	79.6	**72.7**	66.3	**82.3**	**77.0** (4.1)
Random Vocabulary	87.0	40.6	**84.0**	87.2	87.6	78.8	66.6	66.5	79.1	75.3 (1.8)
Random w/o Context	89.5	40.9	82.4	88.2	88.5	80.7	65.8	65.0	80.3	75.7 (2.3)
Random with Context	**89.6**	41.6	83.1	88.8	**89.6**	81.2	71.6	66.5	80.7	**77.0** (4.1)
LLAMA2 7B										
`Answer:`	83.0	43.0	68.1	89.0	89.1	67.6	63.6	35.5	80.2	68.8 (0.0)
`Foo Bar`	76.2	46.1	65.3	75.7	76.0	51.0	51.4	26.6	77.8	60.7 (−11.8)
ZS-CoT	64.5	45.9	64.8	73.8	88.8	66.3	51.6	47.2	81.6	64.9 (−5.7)
OPRO	89.1	46.0	72.0	**93.1**	83.9	78.5	**81.1**	55.8	81.0	75.6 (9.9)
OPRO-ICL	92.0	48.8	**72.3**	92.5	85.6	**79.9**	79.1	**57.3**	80.7	**76.5** (11.2)
Random Vocabulary	91.5	47.5	68.8	93.0	**88.4**	79.2	77.1	49.4	80.7	75.1 (9.2)
Random w/o Context	**92.2**	48.5	71.9	92.6	88.0	77.7	75.6	47.7	81.0	75.0 (9.0)
Random with Context	90.5	**49.5**	71.6	**93.1**	82.9	77.6	77.7	52.8	**82.2**	75.3 (9.4)
LLAMA2-CHAT 7B										
`Answer:`	85.3	38.3	34.8	87.4	82.3	79.5	58.8	48.0	70.5	65.0 (0.0)
`Foo Bar`	85.9	29.8	36.6	80.3	83.9	78.3	53.3	38.9	61.8	61.0 (−6.2)
ZS-CoT	82.3	26.1	34.3	77.2	79.8	75.9	52.0	34.9	58.4	57.9 (−10.9)
OPRO	84.2	43.0	36.5	83.8	80.6	82.6	62.3	45.9	65.8	65.0 (0.0)
OPRO-ICL	85.5	45.2	39.2	89.4	83.8	83.5	**65.6**	49.3	71.2	**68.1** (4.8)
Random Vocabulary	88.1	38.8	**39.5**	88.8	**84.9**	**83.6**	58.0	**50.3**	68.5	66.7 (2.6)
Random w/o Context	**90.2**	42.3	38.4	89.3	83.7	82.2	60.3	48.8	**72.3**	67.5 (3.8)
Random with Context	89.6	**47.5**	37.6	**89.7**	83.7	82.0	63.8	48.8	66.9	67.7 (4.2)
CHATGPT (GPT-3.5)										
`Answer:`	93.2	44.1	90.2	91.8	90.2	68.0	78.9	75.0	81.6	79.2 (0.0)
`Foo Bar`	91.0	37.1	90.0	88.1	89.1	71.5	66.0	72.3	81.2	76.3 (−3.7)
ZS-CoT	93.9	35.0	87.9	91.2	89.8	76.0	80.1	76.4	82.6	79.2 (0.0)
OPRO	93.6	43.8	89.5	91.4	87.9	80.7	80.5	72.1	83.4	80.3 (1.4)
OPRO-ICL	94.3	36.3	90.8	90.8	90.2	82.4	78.9	75.0	83.0	80.2 (1.3)
Random Vocabulary	**94.7**	47.7	89.5	91.8	**91.6**	81.8	78.9	**77.7**	83.0	81.9 (3.4)
Random w/o Context	93.4	42.4	**91.6**	91.2	90.8	82.6	79.5	74.6	82.4	80.9 (2.1)
Random with Context	94.3	**48.4**	89.3	**92.6**	87.3	**84.4**	**84.0**	75.2	**83.6**	**82.1** (3.7)

Table 6: Our main results on the evaluation set. We use one-shot context for all experiments, and use the same model for separator generation and evaluation. The relative improvement scores are computed using the "`Answer:`" as baseline. All the results except ChatGPT are calculated based on five different random seeds. For ChatGPT, we use two different random seeds. Results are colored blue when our random separators achieve the best performance.

	MI	NI	APE	Evo Prompt-DE	Evo Prompt-GA	Random Vocabulary	Random w/o Context	Random w/Context
SST-2	93.7	92.9	94.0	94.8	94.8	93.7	94.2	94.4
CR	91.4	90.9	90.5	91.4	91.2	91.1	90.2	91.0
MR	88.8	89.6	90.9	90.2	90.4	89.4	89.6	90.3
SST-5	42.9	48.6	47.0	48.2	49.4	41.0	37.1	45.5
AGNews	70.6	48.9	71.2	73.3	73.4	76.5	80.6	79.5
TREC	50.6	55.0	59.6	64.4	63.8	61.4	57.6	66.8
SubJ	49.8	52.6	63.3	77.6	67.9	62.6	69.0	64.7
AVG.	71.1	68.2	73.8	77.1	75.9	73.7	74.0	76.0

Table 7: Prompt performance on the Alpaca-tuned LLaMA model. We compare with two human-level prompt optimisation methods, MI (Zhang et al., 2023) and NI (Mishra et al., 2021), and two automatic prompt optimisation methods, APE (Zhou et al., 2022) and EvoPrompt (Guo et al., 2023).

text imposes task-relevance constraints by sampling from a language model conditioned on training samples. We observe, though marginal, consistent improvements of including context information for prompt optimisation. Specifically, the *Random with Context* method achieves a relative improvement of 0.3% over *Random w/o Context* and 0.9% over *Random Vocabulary*. Nevertheless, given the significant gains that *Random Vocabulary* achieves over human baseline (10%), the incremental gains from adding task information are relatively minor.

4.2 Random Sampling is a Strong Prompt Optimiser

Random separator generation methods are comparable to instruction-based approaches. OPRO and its in-context learning variant achieve the top average performance for four out of seven models. However, the advantage is minimal, with a 0.1% average performance difference compared to random methods. It appears that instruction-based methods might be randomly encountering good separators throughout the optimisation process.

Instruction-tuned models are not essential for proposing separators. Previous prompt optimisation work heavily rely on a language model to suggest alternatives, and they make a strong assumption that only instruction-tuned models are able to differentiate good and bad prompts. However, our observation indicates that instruction-tuned models are not essential for proposing the separators. As shown in Table 6, OPRO is able to achieve reasonable performance, even for the smaller GPT2 family models. It is worth noting that, given the linguistic complexity of the OPRO-style meta prompt, a GPT2 model is unlikely to "understand" it (Ouyang et al., 2022). Furthermore, we do not observe a significant increase when using instruction-based model for proposing separators.

Strategy	Separator	Score
OPRO	`The new text is the following:`	92.2
OPRO-ICL	`00:57`	92.2
Random Vocabulary	`obliged\u0442\u0438\u0435Circ song`	92.2
Random w/o Context	`Home Business New \u2018`	92.2
Random with Context	`**GW - The Wall Street`	92.2

Table 8: Performant separators discovered in the training process on AGNews using LLAMA2 7B, we report the accuracy score over the training set.

	Random Vocabulary	Random w/o Context	Random with Context
GPT2-LARGE	37.1%	20.4%	51.2%
GPT2-XL	70.6%	42.6%	48.3%
LLAMA2 7B	66.1%	47.2%	49.6%
LLAMA2 CHAT	21.4%	14.6%	13.9%

Table 9: Chance of random separators outperforming the human baseline "`Answer:`" on the AGNews dataset.

5 Analyses

5.1 Language Space is Rich with Potentially Good Separators.

We find that different approaches can discover distinct separators while yielding similar performance, as shown in Table 8. This prompts a natural question: how many effective separators exist? For simplicity, we deem any separator effective if it outperforms a human-curated separator such as "Answer:". To investigate this question, we calculate the percentage of effective random separators based on all data points[6] from the main experiment. Table 9 shows that our random baseline has on average a 40% chance to draw a separator that is better than the human baseline. This suggests that in the language space, there are more performant separators than we previously expected.

5.2 Are Performant Random Separators Transferable?

In this section, we study whether performant separators discovered by random approaches are transferable across different tasks and contexts.

[6]Approximately 10,000 separators.

↙	SST-2	SST-5	DBPedia	MR	CR	MPQA	Subj	TREC	AGNews	Avg.
Best Human-level Prompt										
Answer:	81.2	40.6	40.6	82.8	81.2	76.6	**79.7**	4.7	43.8	59.0
Best Separator, Random Vocabulary										
SST2	85.9	39.1	42.2	43.8	81.2	84.4	65.6	26.6	65.6	59.4
SST5	57.8	45.3	31.2	73.4	81.2	48.4	51.6	7.8	56.2	50.3
DBPedia	50.0	39.1	56.2	64.1	81.2	79.7	42.2	29.7	56.2	55.4
MR	75.0	34.4	39.1	82.8	82.8	60.9	42.2	21.9	48.4	54.2
CR	48.4	29.7	37.5	43.8	93.8	62.5	42.2	28.1	57.8	49.3
MPQA	85.9	39.1	42.2	43.8	81.2	84.4	65.6	26.6	65.6	59.4
SubJ	57.8	25.0	46.9	76.6	87.5	73.4	76.6	28.1	48.4	57.8
TREC	57.8	23.4	39.1	76.6	81.2	76.6	42.2	43.8	60.9	55.7
AGNews	50.0	32.8	31.2	79.7	79.7	76.6	42.2	25.0	79.7	55.2
Best Separator, Random w/o Context										
SST2	76.6	18.8	40.6	57.8	93.8	59.4	56.2	34.4	59.4	55.2
SST5	51.6	43.8	31.2	62.5	85.9	75.0	67.2	37.5	29.7	53.8
DBPedia	51.6	29.7	54.7	54.7	70.3	60.9	60.9	43.8	71.9	55.4
MR	60.9	35.9	43.8	84.4	81.2	76.6	48.4	31.2	45.3	56.4
CR	76.6	18.8	40.6	57.8	93.8	59.4	56.2	34.4	59.4	55.2
MPQA	56.2	23.4	50.0	43.8	79.7	81.2	42.2	37.5	46.9	51.2
SubJ	50.0	18.8	48.4	59.4	89.1	68.8	81.2	17.2	23.4	50.7
TREC	50.0	20.3	48.4	57.8	79.7	73.4	42.2	45.3	54.7	52.4
AGNews	48.4	25.0	42.2	60.9	79.7	68.8	42.2	28.1	79.7	52.8

Table 10: Random separator transferability test on GPT2-XL. We transfer the best random separator from each task (the columns) to the others (the rows), then colour the results according to their *relative* accuracy on the training set. Brightness denotes high transferability, with thresholds at 80% and 90%.

Cross-task transferability. We use separators with the highest accuracy from each task (Table 11) and apply them to different tasks. According to Table 10, we do not observe high transferability across tasks; for example, a performant prompt for SST2 has almost random-guessing performance on SST5, and vice versa. This is within our expectations, as these random prompts are optimised and selected for a particular task. Notably, even widely used human-curated separators exhibit similar transferability scores (59.0% versus 59.4% in average score) across tasks to random methods.

Task	Random Vocabulary	Random w/o Context
SST2	"cancell BlakesteamappsGr"	"In December, I"
SST5	"biblical namely"	"("
DBPedia	"download pitch Par"	"To view this"
MR	"GAME paced"	"”"
CR	"learnt"	"In December, I"
MPQA	"cancell BlakesteamappsGr"	"LONDON"
SubJ	"pubfile Favor"	"A small"
TREC	"wasSIZE Armageddon"	"Image"
AGNews	"Alc messenger SYSTEM precipitation"	"Weird Al"

Table 11: Best-discovered random separators used in the transferability test.

Cross-context transferability. Given the limited cross-task transferability of separators, we further study whether performant random separators are transferable when the context changes[7] within a fixed task. We discover that random separators exhibit a degree of transferability and are significantly better than human-curated ones (73.4% versus 50.6%). Surprisingly, for this task, human-curated prompts perform even worse than the average random prompt, which matches with our observations in Section 5.1. Overall, our random strategies offer considerable flexibility in discovering task-wide performant separators.

[7]The term "context" here is also referred to as "demonstrations" in in-context learning. In our experiments, we use one-shot examples as context to guide the output label space for classification tasks.

	#1	#2	#3	#4	#5	Avg.
Best Human-level Prompt						
Answer:	43.8	54.7	57.8	43.8	53.1	50.6
Average Separators, Random Vocabulary						
Avg. Foo Bar	55.7	55.6	63.2	61.9	53.3	57.9
Best Separator, Random Vocabulary						
#1 Best Sep.	79.7	81.3	81.3	68.8	53.2	72.9
#2 Best Sep.	71.9	78.1	79.7	64.1	73.4	73.4
#3 Best Sep.	71.9	60.9	82.8	70.3	76.6	72.5
#4 Best Sep.	76.6	73.4	67.2	82.8	57.8	71.6
#5 Best Sep.	71.9	67.2	71.9	70.3	79.7	72.2

Table 12: Random separator context transferability test on GPT2-XL. We choose the best separator from different contexts of the AGNews dataset; then compute its accuracy across other contexts for the same AGNews training set. AVG. Foo Bar represents the average performance of 160 randomly sampled separators.

5.3 Beyond Text Classification Tasks

In previous sections, we have mainly showed that random sampling is a strong baseline for classification tasks across nine classification datasets over eight different models, revealing that the oversensitivity of LLMs is still a notable issue, and it is a fundamental characteristic of in-context learning.

To verify whether such random methods have similar patterns, and are strong baselines for generative reasoning tasks, we apply our "random sampling over vocabulary" method to GSM8K (Cobbe et al., 2021), a mathematical reasoning dataset.

Seed	Human CoT	Avg. Random Vocabulary	Best Random Vocabulary (Rel. Δ%)
#1	35.9	36.3	46.9 (30.6)
#2	42.2	39.3	50.0 (18.5)
#3	35.9	37.7	46.9 (30.6)
#4	39.1	38.0	45.3 (15.9)
Avg.	38.3	37.8	47.3

Table 13: Comparing best random separators performance with Chain-of-Thoughts prompting on GSM8K.

Similar to the previous experimental setup, we sample up to 160 different random separators, and

then perform selection and evaluate their performance over the subset of test data. For this task-specific setup, we use 5-shot and majority voting@1 on the Mistral 7B model (Jiang et al., 2023), and report results on four different random seeds[8], as shown in Table 13.

CoT is only slightly better than average random separators. Surprisingly, for the few-shot mathematical reasoning task, thinking-style phrases could only be slightly better than random separators. On average, CoT attains an accuracy of 38.3 and the average of random separators is 37.8. This suggests that the gains of manual prompt optimisation are sub-optimal.

CoT shows high variance in quality across different examples. As shown in Table 13, we observe that the language model is sensitive to different sets of demonstrations across all methods. The most widely used human-derived chain-of-thought prompt ("let's think step by step") results in the highest variance, with over a 17% relative performance gap between the best and worst set of demonstrations. On the other hand, our random sampling approach yields a 9% relative difference, suggesting better robustness.

Random separators are still a strong baseline for generative reasoning tasks. Our best random separators reach an average accuracy of 47.3, a 23% relative increase in accuracy over the CoT baseline. This aligns with our main findings derived from the classification tasks (Table 6). It is conceivable that for complex generative tasks, the language space is abundant with potentially good separators.

6 Related Work

Automatically discovering effective prompts remains a challenging research problem because of the complexity of the search space. One direction is continuous prompt tuning (Qin and Eisner, 2021; Lester et al., 2021; Liu et al., 2023), which involves adding a set of smaller tunable parameters to pre-trained language models. An alternative approach is to optimise discrete token spaces. Shin et al. (2020) showed that gradient information at the embedding layer can guide the discovery of more effective prompts. According to their research, unnatural prompts can also result in good performance, which matches our observations. In spite of the efficiency of using gradient information, such a method, which heavily relies on the availability of language models, imposes some restrictions on certain types of models. An alternative direction is black box search. To simplify language space optimisation, Prasad et al. (2023) introduced a set of operations, such as add/delete/replace tokens. APE (Zhou et al., 2022) showed that generating some alternatives and then selecting and rephrasing them could also provide effective solutions. Similarly, Xu et al. (2022) used evolutionary methods to optimise the search process. Recently, Yang et al. (2023) demonstrated that we can teach language models to learn the pattern of good prompts using human-written meta prompts. EvoPrompt (Guo et al., 2023) showed how we can formulate the evolutionary process in meta-prompt. Fernando et al. (2023) further demonstrated how we can improve the meta-prompt using language models, making the whole framework completely automatic without relying on the internal state of language models.

7 Conclusion

We find that random separators, even those selected at random from vocabulary, could be as effective as previously discovered state-of-the-art prompts. In addition, we conduct research on three different types of random separators, which demonstrated that these random separators do not require instruction-tuned models, could provide a 12% relative improvement as compared to human baselines, and are on par with a self-optimising approach involving complex meta-prompt engineering.

Limitations

While we have done our utmost to explore randomly generated prompts, a limitation of this work is that we mainly evaluate our approaches on text classification tasks. However, based on our experimental results in the mathematical generative task (Section 5.3), our findings are still empirically sound. We will leave a more comprehensive evaluation of generative tasks as future work.

Acknowledgements

We thank Jean Kaddour for his valuable feedback. Pontus Stenetorp would like to acknowledge the helpful proofing feedback from several viewers while finalising the submission. This work is supported by Microsoft Research via Accelerate Foundation Models Research Grant.

[8]We change the demonstration examples for each seed.

References

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. *arXiv preprint arXiv:2005.14165*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Joe Davison, Joshua Feldman, and Alexander M. Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1173–1178.

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.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. 2023. Promptbreeder: Self-referential self-improvement via prompt evolution. *arXiv preprint arXiv:2309.16797*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. 2023. Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. *arXiv preprint arXiv:2309.08532*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825*.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2023. Gpt understands, too. *AI Open*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2021. Cross-task generalization via natural language crowdsourcing instructions. *arXiv preprint arXiv:2104.08773*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. 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 2227–2237.

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. In *Automated Knowledge Base Construction*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. 2023. GrIPS: Gradient-free, edit-based instruction search for prompting large language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3845–3864, Dubrovnik, Croatia. Association for Computational Linguistics.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. *arXiv preprint arXiv:2104.06599*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. In *OpenAI Blog*.

Weijia Shi, Xiaochuang Han, Hila Gonen, Ari Holtzman, Yulia Tsvetkov, and Luke Zettlemoyer. 2022. Toward human readable prompt tuning: Kubrick's The Shining is a good movie, and a good prompt too? *arXiv preprint arXiv:2212.10539*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ellen M Voorhees and Dawn M Tice. 2000. Building a question answering test collection. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 200–207.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2):165–210.

Hanwei Xu, Yujun Chen, Yulun Du, Nan Shao, Wang Yanggang, Haiyu Li, and Zhilin Yang. 2022. GPS: Genetic prompt search for efficient few-shot learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8162–8171, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. 2023. Large language models as optimizers. *arXiv preprint arXiv:2309.03409*.

Wenxuan Zhang, Yue Deng, Bing Liu, Sinno Jialin Pan, and Lidong Bing. 2023. Sentiment analysis in the era of large language models: A reality check. *arXiv preprint arXiv:2305.15005*.

Xiang Zhang, Junbo Zhao, and Yann Lecun. 2015. Character-level convolutional networks for text classification. *Advances in Neural Information Processing Systems*, 2015:649–657.

Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. *arXiv preprint arXiv:2102.09690*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*.