# Selecting Better ChatGPT prompts for NLP Tasks

**Aleksandra Smolka**
SNHCC TIGP
Institute of Information Science
Academia Sinica
alsm@iis.sinica.edu.tw

**Hsin-Min Wang**
Institute of Information Science
Academia Sinica
whm@iis.sinica.edu.tw

**Jason S. Chang**
Department of Computer Science
National Tsing Hua University
jason@nlplab.cc

**Keh-Yih Su**
Institute of Information Science
Academia Sinica
kysu@iis.sinica.edu.tw

## Abstract

One of the crucial parts of using ChatGPT is adopting a proper prompt to obtain the desired answer from it. However, how different prompt designs affect ChatGPT performance is still not well studied. In this work, we concentrate on three selected natural language processing (NLP) tasks (i.e., paraphrase judgment, natural language inference, and question answering), as we have found that previous works in this area lack systematic analysis of how prompts should be set. We thus propose the *prompt formula*, which is a set of binary prompt features characterizing the prompt, for systematically testing various characteristics of prompts (such as the politeness of the language, answer type, and label specification). We then identify the prompt feature set that performs best in the zero- and few-shot scenarios. The experiments show that the appropriate prompt formula could improve the ChatGPT performance by up to 15%, in comparison with those existing prompting formats reported in the NLP literature. We also show that adding training samples (i.e., the few-shot case) sometimes even deteriorates the performance.

Keywords: ChatGPT, prompt engineering, paraphrase judgment, question answering, natural language inference, LLMs

## 1 Introduction

ChatGPT[1] released as an API model by OpenAI is a generative foundation model specializing in language processing (Wang et al., 2023). Among those recently released Generative AI models (Gozalo-Brizuela and Garrido-Merchan, 2023), ChatGPT has gained significant popularity not only because of the commercial success of its user-end API but also due to the ease of use for a large variety of tasks (Wang et al., 2023).

Although numerous publications on ChatGPT discuss its limitations and its evaluation, we found that the impact of varying the prompt formats on performance is still under-represented in the natural language processing (NLP) literature. The lack of sufficient attention to the issue of selecting a proper *prompt pattern* in the experiment design is especially noticeable during evaluating ChatGPT on specific NLP tasks (such as paraphrase judgment (PJ), natural language inference (NLI), and question answering (QA)). In most related works, the prompt formats are either just arbitrarily set without sufficient justification in advance (Wang et al., 2023), or directly borrowed from previous works without explanation (Shen et al., 2023; Zhong et al., 2023).

As a result, the impact of using various prompt formulas is either not systematically compared or lacks in-depth analysis (Basmov et al., 2023; Kocon et al., 2023). Unfortunately, using ChatGPT without understanding the effect of prompt pattern on the model accuracy could lead to unfair performance comparison (Qin et al., 2023). Since

[1] https://openai.com/chatgpt/

the training data/procedure of ChatGPT is not transparent to the end-user, and it has been shown that *prompt engineering* considerably affects the performance (Qin et al., 2023), it is thus important to test and get an appropriate prompt pattern before ChatGPT is used for the given task.

To achieve the above goal, we first select features re-occurring in the prompts used by previous works (Basmov et al., 2023; Jang and Lukasiewicz, 2023; Lai et al., 2023, Shen et al., 2023; Wang et al., 2023), and then propose the *prompt formula*, which is a set of binary prompt features (i.e., "+" or "-", which denote *on* and *off* states, respectively) that characterize the prompt. For example, the usage of polite words could be adopted to categorize various prompts, as shown in Figure 1 and Table 1. Afterward, we use this prompt formula to systematically select the appropriate prompt for the given task, which is done via the Sequential-Forward-Selection (SFS; Ferri et al., 1994) feature-set selection procedure. An appropriate prompt formula would be found for each specific NLP task.

To show the superiority of the proposed approach, we design a series of experiments to check how those features impact the model results. For each task, we manually create its associated *base prompt*, which is a common prompt additionally interleaved with a few pre-specified empty placeholders. Each placeholder is used to insert additional information associated with a specific feature. Figure 1 shows an example of the feature "*List allowable answers*" (under the "*feat.*" column in the right table), which has been filled with the corresponding purple string *<reply "paraphrased" or "not-paraphrased">* (left).

Each type of placeholder corresponds to a specific prompt feature. Furthermore, we propose a *data augmentation method*, which instantiates the above base prompt by inserting the corresponding content into those placeholders. In this way, we can generate the corresponding prompt, which will be applied to a specific NLP task later, for each specified feature configuration. We then adopt the SFS approach to select the best feature configuration (among 6 different features) for each adopted NLP task in both zero- and few-shot scenarios.

Our experiments adopt several types of datasets: (1) two PJ datasets including MRPC (Dolan and Brockett, 2005) and QQP (Aghaebrahimian, 2017), (2) an NLI dataset (SNLI; Bowman et al., 2015), and (3) a multiple-choice QA dataset (CommonsenseQA; Talmor et al., 2019). We select these datasets as they are frequently adopted to test common NLP tasks with varying levels of difficulty.

Based on our experiments, we show that carefully selecting an appropriate prompt formula could improve the results by up to 15% in comparison with the best prompt formulas adopted in previous representative works. We further show that the current ChatGPT model with GPT-3.5 backbone sometimes lacks the ability to generalize

*Please answer* whether *S1 and S2* are paraphrased or not. *You should consider the syntax and semantics of the sentences to compare their meaning. Please reply "paraphrased" or "not-paraphrased". Your answer should be only one word, in lowercase letters.*
*S1: [...]*
*S2: [...]*
*Answer:*

| feat. \ con. | Content filled in the placeholder (+) | Content filled in the placeholder (-) | ex. |
|---|---|---|---|
| polite words | "please" | Ø | + |
| state desired model action | "answer" | Ø | + |
| way to integrate test sample | integrated within the prompt text | cited below using variables | - |
| list key competencies | "...consider the syntax…" | Ø | + |
| list allowable answers | "reply "paraphrased" or …" | Ø | + |
| specify answer format | "your answer should…" | Ø | + |

Figure 1: An example for instantiated prompt (left block) used for the paraphrase judgement task (its associated *prompt formula* and *base prompt* are shown in the right part). Each color indicates a specific kind of placeholder that has been filled with the corresponding content of the associated feature value (as listed in the table). On the right table, the "*feat.*" column indicates the corresponding feature names; the "*Con.*" column is the content for the corresponding binary feature value, and the "*ex.*" column is the corresponding binary feature value (i.e., "+" and "-") for the prompt example given on the left.

from a small number of examples (i.e., the few-shot scenario) to other similar cases.

Our contributions include:

- Analyzing all the prompts collected from those published NLP-related papers that we have read (mainly related to PJ, NLI, and QA), and then categorizing them into 6 different binary features.
- Proposing the *prompt formula* to use it for systematically selecting the appropriate prompt for the given task.
- Using SFS to look for the best corresponding prompt formula for each task, which allows up to 15% improvement on selected tasks in comparison with the best prompting method listed in the related literature.
- Showing that adding examples to the prompt sometimes even deteriorates the performance of the few-shot scenario in selected tasks, which violates the common intuition.
- Releasing the code, which can be used to select the best corresponding prompt formulas for other datasets/tasks.[2]

## 2 Feature selection and prompt generation

In this section, we first categorize the prompt patterns with distinctive features (Section 2.1) and then show how we use these features to generate the prompts that could be used in our experiments (Section 2.2).

### 2.1 Prompt pattern categorization

We begin by analyzing a range of existing prompts (concentrating on those applied to PJ, NLI, and QA tasks[3]). We then categorize them into 4 main prompt categories (as shown in Table 1) depending on: (1) adopting polite words such as "*please*" (Lai et al., 2023); (2) attaching the information about how to solve the problem (Liu et al., 2023a; Wang et al., 2023); (3) attaching the information about the desired answer (Basmov et al., 2023); (4) adopting a specific way for integrating the test sample into the prompt (i.e., whether the test sample is inserted within the prompt text or below (Shen et al., 2023)). Each category will be further elaborated as follows.

The first category of adopting polite words is rather self-explanatory and usually involves adding words such as "*please*" or using politer modal verbs (e.g., "*could*" instead of "*can*") (Lai et al., 2023). The second main category includes two

| Work | task | Adopt polite words | Attach information about solving the problem | | Attaching information about the desired answer | | Way to integrate the test sample |
|---|---|---|---|---|---|---|---|
| | | | Desired action | List key competencies | Allowable answers | Answer format | |
| Jang and Lukasiewicz (2023) | PJ | - | - | - | - | - | - |
| Wang et al. (2023) | PJ | - | + | - | + | - | - |
| Basmov et al. (2023) | NLI | - | + | + | + | + | - |
| Lai et al. (2023) | NLI | + | + | - | + | + | - |
| | QA | - | + | + | + | + | - |
| Shen et al., (2023) | QA | - | + | - | + | - | + |

Table 1: Prompt features in previous representative works. The "+" sign indicates that the feature is present. In contrast, the "-" sign indicates that it is not. The prompt formulas vary greatly across different works, and even among the tasks mentioned within one publication.

[2] https://github.com/alsmolka/gpt-prompt-analysis

[3] In our work, we concentrate only on single-stage prompting such as that mentioned in Lai et al. (2023), and do not consider multiple-stage prompting such as that adopted in Kojima et al., (2022) or Qin et al., (2023). However, our proposed approach can be also applied to the multiple-stage prompting with slight modification, which is beyond the scope of this work.

sub-categories describing whether the prompt provides additional information for solving the problem. The prompt might either: (a) explicitly include a verb describing the desired action to get the answer (e.g., "*select*", "*compare*", … (Wang et al., 2023)), or (b) list key competencies, which explains what type of skill or information is needed to solve the problem (e.g., "*use commonsense knowledge*" (Lai et al., 2023)).

The third category specifies whether the desired answer is explicitly specified within the prompt. It also includes two sub-categories, and can either: (a) list all allowable answers (e.g., "*yes*" or "*no*" for binary classification (Wang et al., 2023)), or (b) specify the desired answer format (e.g., limiting the desired answer to have only one word, (Basmov et al., 2023)).

Finally, the last way of grouping the prompts is based on how the test sample is integrated into the prompt: (1) directly cited within the prompt text (e.g., *"Are following sentences paraphrases: [sentence1], [sentence2]"* (Shen et al., 2023); corresponding to the "+" sign of the feature "*way to integrate the test sample*" in both Figure 1 and Table1), or (2) provided below the main prompt in which it is mentioned as a variable (e.g., "*Are S1 and S2 paraphrases*", with *S1* and *S2* specified below the whole prompt (Lai et al., 2023); corresponding to the "-" sign and adopted in Figure 1).

Table 1 shows these adopted binary features together with the corresponding contents adopted in two previous representative works (for each task). Each of the features can be specified with a binary value (i.e., +/-).

| task | work | Test dataset | # SPL | Acc. |
|---|---|---|---|---|
| PJ | Jang and Lukasiewicz, (2023) | MRPC | 1000 | 0.54 |
| | **Wang et al., (2023)** | **MRPC** | **1000** | **0.60** |
| | **Jang and Lukasiewicz, (2023)** | **QQP** | **1000** | **0.72** |
| | Wang et al., (2023) | QQP | 1000 | 0.50 |
| NLI | **Basmov et al. (2023)** | **SNLI** | **1000** | **0.47** |
| | Lai et al. (2023) | SNLI | 1000 | 0.45 |
| QA | Lai et al., (2023) | CQA | 1000 | 0.57 |
| | **Shen et al., (2023)** | **CQA** | **1000** | **0.61** |

Table 2: Performance of test prompt formulas from two representative previous works ("work") for each task ("task") and dataset ("test dataset"). Each row corresponds to a single formula. #SPL indicates the number of prompted samples, and "Acc" indicates the associated accuracy using the given prompts. Selected baselines are in bold.

### 2.2 Prompt generation for specific tasks

To better specify and systematically evaluate different prompts, we propose using a pre-specified *prompt formula*, which is a configuration consisting of all 6[4] binary feature values mentioned above, to generate various prompts with a specific set of feature values (Table 1). We also propose a *base prompt*, which contains various placeholders of several types as shown in Figure 1 (colored boxes), to use for generating the prompts used in ChatGPT. We fill these placeholders according to the assigned feature values specified in the corresponding prompt formula. For example, for the feature "*polite words*", the two pink boxes in Figure 1 will be filled with the word "*please*" if the associated feature value is positive.

We manually specify a specific base prompt for each task, in which additional text would be inserted into the corresponding placeholders (associated with various features) to generate the desired prompt. In the few-shot scenario, the additional benchmark examples are simply inserted below the zero-shot base prompt.

## 3 Experiments

### 3.1 Datasets

We select our benchmark datasets across a range of NLP tasks, including PJ (i.e., MRPC (Dolan and Brockett, 2005) and QQP (Aghaebrahimian, 2017)), NLI (i.e., SNLI (Bowman et al., 2015)), and multiple-answer QA (i.e., CommonsenseQA (Talmor et al., 2019)).

To prepare the test set for each task listed above, we randomly take 1,000 samples for the

[4] Each sub-category listed in Section 2.1 is counted as a distinct feature.

development and 1,000 samples for the test set, keeping the class balance.[5] We repeat sampling for each task and dataset. We then generate the prompts for the ChatGPT model as described in Section 2.2. The development set in this scenario is used to select the best feature configuration (i.e., the best prompt formula) and the test set is used to report the final results.

For the few-shot scenario experiments (i.e., Experiment 2), we take the best prompt formula found in Experiment 1 for each task and follow Liu et al. (2023a) to additionally augment it with some randomly sampled training data. We test 4 variations for adding the training data into the prompts, including (a) 1 random example, (b) 2 examples randomly sampled regardless of their classes (for all benchmarks), (c) 2 random examples each from a different class, and (d) 4 random examples, two from each class. The last two variants are only conducted for the tasks for which benchmark datasets are associated with 2 classes (i.e., PJ-MRPC and PJ-QQP).

Furthermore, to measure the susceptibility of the model to the change in the training sample provided in the prompt, we calculate the standard deviation (SD) of its performance (Sekander Hayat Khan, 2011) on 5 sets of 1,000 generated answers. Each set uses different training samples. It thus gives us a total of 60K prompted samples to be used in Experiment 2.

### 3.2 Experimental settings

In our experiments, we adopt the ChatGPT API with the GPT-3.5-turbo model. Following the approach adopted in the previous publications (e.g., Wang et al., 2023), the model is not fine-tuned. Hence, the same model is always used in either the zero-shot (i.e., baseline (Section 3.3), and Experiment 1 (Section 3.4.1)) or the few-shot scenario (i.e., Experiment 2 (Section 3.4.2)).

To simplify the performance evaluation step, the answers obtained from ChatGPT are first automatically normalized before evaluation, which is similar to what has been done in the previous works (Basmov et al., 2023), as we have found that the format inconsistency between the benchmark and the obtained answer causes a huge drop in its performance. The normalization procedure is thus introduced to reduce the output format variation and includes the following three procedures: (1) removing extra whitespaces, (2) correcting punctuation, which involves removing unnecessary punctuation marks, and (3) casting all words to lowercase and removing surplus strings (e.g., the output *"S1 and S2 are paraphrases"* would be automatically converted into the benchmark label *"paraphrase"*).

| Feat. name | | Development set | | | | | | | Test set | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | none | plt. | act. | Lbl. | fmt. | cmpt. | int. | baseline | BEST |
| Task-dataset | PJ-MRPC | 0.54 | 0.51 | 0.63 | 0.56 | 0.52 | 0.60 | 0.62 | 0.60 | **0.72*** |
| | PJ-QQP | 0.72 | 0.71 | 0.60 | 0.37 | 0.64 | 0.49 | 0.69 | 0.69 | 0.69 |
| | NLI-SNLI | 0.39 | 0.19 | 0.44 | 0.43 | 0.49 | 0.11 | 0.38 | 0.46 | **0.61*** |
| | QA-CQA | 0.45 | 0.43 | 0.61 | 0.49 | 0.60 | 0.38 | 0.46 | 0.59 | 0.59 |

Table 3: Selection of the best prompt formula via the sequential-forward-selection (SFS; Ferri et al., 1994) procedure (the zero-shot scenario). We report the accuracy measured from the following cases: (1) none of the feature-switch is turned on (the first *none* column, treated as an additional pseudo-feature in the table), (2) only one feature-switch is activated (the columns 2-7), (3) the baseline prompt from Table 2 (the *baseline* column), and (4) the best prompt formula found via the SFS feature-set selection procedure (the "*BEST*" column). The features used in the experiment: "*plt.*" – using the polite words; "*act*" – specifying the required action; "*Lbl.*" – specifying the allowable answer labels; "*fmt*" – specifying the answer format; "*cmpt.*" – outlining the competencies needed; "*int.*" – integrating the test sample into the prompt text. Results showing statistically significant improvement over the corresponding baseline ($p<0.05$) are marked with a "*" symbol.

[5] This method of preparing data for ChatGPT evaluation is similar to those previous approaches such as in Shen et al. (2023).

| Task-dataset | Prompt formula | Adopt polite language | Attach information about solving the problem | | Attaching information about the desired answer | | Way to integrate the test sample |
|---|---|---|---|---|---|---|---|
| | | | Desired action | List key competencies | Allowable answers | Answer format | |
| PJ-MRPC | baseline | - | + | - | + | - | - |
| | ours | + | + | - | - | - | + |
| PJ-QQP | baseline | - | - | - | - | - | - |
| | ours | - | - | - | - | - | - |
| NLI-SNLI | baseline | - | + | + | + | + | - |
| | ours | - | + | - | - | + | + |
| QA-CQA | baseline | - | + | - | + | - | + |
| | ours | - | + | + | - | + | - |

Table 4. Comparison of features between our best prompt and the baseline for each task-dataset combination in the zero-shot scenario. The feature names (columns) are the same as those in Table 1.

### 3.3 Baseline selection

We test each task-dataset pair using two different formulas taken from earlier representative works as listed in Table 1. Since it is unclear which previous approach yields the best results based only on the literature review, we choose the prompt formula with the best performance on our test set as its baseline. As a result, we end up with a total of four baselines (i.e., one for each task-dataset combination).

Table 2 shows the performance of the prompt formulas tested across the task-dataset pairs. Prompts selected as baselines are marked in bold. The table additionally shows that the performance of adopting various prompt formulas can vary greatly even on the same task-dataset combination (e.g., up to 22%, from 0.50 to 0.72, on the PJ-QQP dataset), highlighting the importance of selecting an appropriate prompt formula while using ChatGPT.

### 3.4 Experimental results

After establishing the baselines for performance comparison, we conduct two key experiments to test the impact of each prompt feature described in previous sections. In Experiment 1 (Section 3.4.1), we identify the best prompt formula based on the prompt features outlined in Section 2, focusing on the zero-shot scenario. Next, in Experiment 2, we check whether adding a few training examples to the prompt (i.e., a few-shot scenario) can enhance the model performance (Section 3.4.2).

#### 3.4.1 Experiment 1 (Zero-Shot Scenario)

In the first experiment, we adopt the zero-shot scenario. We evaluate the effect of each feature we have identified during prompt analysis (Section 2.2, listed in Table 1). Afterward, we apply the SFS (Sequential Forward Selection) procedure (Ferri et al., 1994) to select the best prompt as represented by a feature set. The experiment is performed individually for each task-dataset combination and evaluated on the development set. The SPF approach allows us to identify the best feature combination by progressively adding individual features to the feature set greedily. At each selection step, we check which feature addition results in the best accuracy improvement. The process continues until the potential feature set is exhausted or if adding more features does not yield any further improvements. The best feature set at this stage becomes our final set of features.

Table 3 reports the accuracy for various prompt formulas: (1) with all features turned off (on the development set), (2) individually activating each feature (on the development set), (3) the baseline prompts selected from Table 2 (on the test set), and (4) the best prompt formulas found via the SFS procedure (on the test set).

To compare the established baselines (from Section 3.3) with the best prompt formulas obtained above, we apply the Student's t-test (Student, 1992). We uniformly split the whole test set into $n$ subsets ($n=10$) and calculate the *mean* and *variance* of the model's accuracy on these subsets. Results showing statistically significant improvement over the corresponding baseline ($p<0.05$) are marked with an asterisk ("*").

Table 3 shows that the best prompt formula (the associated feature setting is given in Table 4) can outperform its corresponding baseline by up to 15% (NLI-SNLI, 46% vs. 61%). For the PJ-MRPC task, the improvement is 12% (60% vs. 72%). In the remaining two datasets (i.e., PJ-QQP and QA-CQA), our best prompts also match the

performance of the best prompt used in two previous representative works (i.e., the baseline).

Table 4 compares the feature-setting of each best prompt formula selected with that of its baseline. It seems that each feature could be beneficial for some task-dataset combinations, as all of them appear somewhere in the best feature configurations. Interestingly, for certain task-dataset combinations, it is most beneficial to have all the features turned off. This is the case for the PJ-QQP task, where the prompt formula without activating any features proved to be the best (accuracy of 69% for both the baseline and our approach; see the PJ-QQP row in Table 3). It might be because the QQP dataset is noisy.[6] Providing additional information about the answer (e.g., the answer format) to ChatGPT might confuse, as the benchmark samples do not always follow the desired form specified in the prompt.

While it is difficult to draw any general conclusion from Table 4, it seems that explicitly stating the expected action for getting the answer (by using verbs such as "*select the answer*" or "*compare*") is usually beneficial to the model performance. This is supported by the observation

```
Please answer whether S1 and S2 are paraphrased or
not.
Example:
S1: "They are trying to turn him into a martyr, said
Vicki Saporta, president of the National Abortion
Federation, which tracks abortion-related violence."
S2: "We need to take these threats seriously, said
Vicki Saporta, president of the National Abortion
Federation."
Answer: negative
S1: "Mahmud controlled access to Saddam for everyone
but immediate family members, Pentagon officials
said."
S2: "Mahmud controlled access to Saddam and was
frequently at his side."
Answer:
```

Figure 2. A few-shot prompt example for the PJ-MRPC task with a single negative training example added.

that the "*Desired action*" feature is activated in most of the best prompt formulas.

On the other hand, not adopting polite wording gives better results in all but one case (i.e., our best PJ-MRPC prompt), and is also not adopted in most related works (as shown by Table 1). We guess that the majority of training data might not adopt the polite wording; however, it cannot be confirmed without inspecting the ChatGPT training samples.

Similarly, providing the competencies needed to solve the problem often appears unnecessary or can even negatively impact the performance (75% of the cells under the "*List key competencies*" column have a "-" sign). It is conjectured that providing such hints might work better in the multiple-stage prompting case, in which it is usually adopted (e.g., Shi et al., 2022), rather than the single-stage prompting case like in our work.

For other features, it seems whether they should be turned on or off depends on each specific task and dataset. Overall, the experiment results (Table 3) show that our proposed prompt formula selection approach can obtain the prompts that outperform (or at least are on a par with) those adopted in previous related works.

| Task-dataset | Few-shot (SD) | | | | Zero-shot |
|---|---|---|---|---|---|
| | 1 | 1P+1N | 2 | 2P+2N | |
| PJ-MRPC | 0.64 (0.02) | 0.69 (0.01) | 0.67 (0.02) | 0.66 (0.03) | **0.72** |
| PJ-QQP | 0.39 (0.13) | 0.71 (0.04) | 0.62 (0.09) | **0.76** (0.03) | 0.69 |
| NLI-SNLI | 0.57 (0.03) | n/a | **0.64** (0.01) | n/a | 0.61 |
| QA-CQA | 0.45 (0.03) | n/a | **0.69** (0.01) | n/a | 0.59 |

Table 5. Comparison of zero-shot and few-shot results using the best prompt selected in Experiment 1. The columns "1", "1P+1N", "2" and "2P+2N" indicate various numbers of added training examples (P-positive, N-negative). "n/a" cells denote the datasets with more than 2 classes; as a result, indicated scenarios cannot be applied.

### 3.4.2 Experiment 2 (Few-Shot Scenario)

In the second experiment, we check the effect of adding training examples to the prompt (i.e., the few-shot scenario). We follow Liu et al. (2023a) to add a few examples and use the best prompt formula selected in Experiment 1 to generate the new prompts by adding a few examples from the respective training set to the original prompt (*cf.* Section 3.1 for detailed explanation of how the training samples are added). Figure 2 shows a

[6] It is noisy because a large subset of it has not been manually checked after it was automatically collected from an online forum (https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs).

prompt example for the PJ-MRPC task when adding one training sample.

Table 5 shows the few-shot performance under the scenarios mentioned above. We observe that adding training samples improves the performance in three out of four tasks (i.e., PJ-QQP, NLI-SNLI, and QA-CQA). The highest improvement is 10% (QA-CQA; from 0.59 in zero-shot to 0.69 in few-shot, SD=0.01).

Interestingly, regardless of the number of added examples, the performance on the PJ-MRPC task decreases, which contradicts the common intuition. Moreover, contrary to the other cases, the performance when four examples are added is even lower than the case when only two training examples are added (i.e., 0.66 vs. 0.69; for the 2P+2N and 1P+1N cases). It is conjectured that this performance drop might be due to the high variability of MRPC samples; as a result, the added training examples could be quite different from the test sample in a given prompt, which might have a negative impact on the inference procedure.

To sum up, adding training examples to the prompt could be beneficial for getting better performance in most cases. But just like other prompt features, the result depends heavily on the benchmark dataset and the task. Hence, before adopting the few-shot scenario, it is necessary to first test if it is indeed beneficial.

### 3.5 Ablation test and error analysis

To show that the post-processing steps taken when obtaining answers from ChatGPT are essential, we examine the performance in Experiment 1 without conducting the automatic answer normalization. The results show that the performance of each case drops considerably, and is even close to 0 for the PJ task (on both the MRPC and QQP datasets) because returning human-acceptable un-normalized answers will not be counted as correct during automatic evaluation (e.g., generating "*Paraphrase*" or "*yes, paraphrase*" instead of the expected answer "*paraphrase*"). It thus shows that using the simple normalization procedure mentioned in Section 3.2 could greatly alleviate the format inconsistency problem.

We then check the errors made in the zero-shot scenario (i.e., Experiment 1, Section 3.4.1). We manually analyze one hundred errors randomly sampled from each benchmark dataset. Based on our analysis, the errors could be grouped into two main categories: (1) incorrect answers (overall 91%), in which ChatGPT returns wrong, but legal outputs (e.g., "*positive*" instead of "*negative*" for PJ; other tasks also behave similarly); (2) illegal answers (overall 9%), in which ChatGPT generates illegal outputs that do not match any of the allowable benchmark labels in the given dataset. This shows that although it is difficult to control the model output format, it is alleviated to some degree by our automatic normalization. Still, 9% of answers could not be recovered.

For the few-shot scenario (Experiment 2) we randomly sample one hundred errors from each benchmark dataset. Similar to the zero-shot case, most answers are legal but incorrect (94% of the cases). Interestingly, the overall percentage of illegal answers decreases by 3% in the few-shot scenario (from 9% to 6%, compared with the zero-shot case). It is conjectured that adding training examples might help the generative model (such as ChatGPT) better recognize what is the desired output format.

## 4 Related work

Some recent works aim to evaluate ChatGPT, especially in the context of its applicability to various NLP tasks as a task-independent model (Srivastava et al., 2022; Bang et al., 2023; Guo et al., 2023; Kocon et al., 2023). Works that concentrate exclusively on specific NLP tasks are also common, including QA (Lai et al., 2023), machine translation (Peng et al., 2023), or summarization (Goyal et al., 2022, Zhang et al., 2023).

Most works on prompt evaluation focus on multi-stage prompting (Shi et al., 2022, Wei et al., 2022, While et al., 2023, Zhou et al., 2022) as opposed to our single-stage approach, though some works are also applicable to single-stage prompting as well (e.g., Liu et al., 2023a; Zuccon and Koopman, 2023). In comparison to those works, we specifically categorize the characteristics of various prompts and propose to use the prompt formula to systematically and automatically select the best prompt feature combination for the given task.

## 5 Conclusions

Given the variation of prompts considerably affects the performance of ChatGPT, we propose a *prompt formula* for systematically selecting an appropriate prompt for specific tasks. We conduct prompt

engineering on three NLP tasks: PJ, NLI, and QA. The experiments have shown that the prompt feature set selected via the SFS procedure could improve the performance by up to 15% in comparison with the best formula found from the selected previous works (under the zero-shot scenario). Furthermore, we also show that adding a few training examples to the prompt sometimes might even deteriorate the performance, which is contradictory to the common intuition. This underscores the importance of selecting a proper prompt formula (both in terms of content and structure) before applying ChatGPT to the downstream task. In the future, the work in this paper can be extended to include an automatic pipeline allowing for automatic prompt selection for any downstream NLP task.

## References


Ahmad Aghaebrahimian. 2017. Quora Question Answer Dataset. *Text, Speech, and Dialogue.* Springer International Publishing, pages 66–73. https://doi.org/10.1007/978-3-319-64206-2_8

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, Quyet V. Do, Yan Xu, and Pascale Fung. 2023. A Multitask, Multilingual, Multimodal Evaluation of ChatGPT on Reasoning, Hallucination, and Interactivity. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–718, Nusa Dua, Bali. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.ijcnlp-main.45

Victoria Basmov, Yoav Goldberg, Reut Tsarfaty. 2023. ChatGPT and Simple Linguistic Inferences: Blind Spots and Blinds. *arXiv:2305.14785* https://doi.org/10.48550/arXiv.2305.14785

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics. https://doi.org/10.18653/v1/D15-1075

William B. Dolan and Chris Brockett. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*. https://aclanthology.org/I05-5002

F.J. Ferri, P. Pudil, M. Hatef, J. Kittler. 1994. Comparative Study of Techniques for Large-Scale Feature Selection. *Machine Intelligence and Pattern Recognition, North-Holland, Volume 16, 1994*, pages 403-413, ISSN 0923-0459, ISBN 9780444818928. https://doi.org/10.1016/B978-0-444-81892-8.50040-7.

Ronald A. Fisher. 1921. Studies in Crop Variation. I. An Examination of the Yield of Dressed Grain from Broadbalk. *Journal of Agricultural Science. 11 (2),* pages 107–135. doi:10.1017/S0021859600003750

Tanya Goyal, Junyi Jessy Li, Greg Durrett. 2022. News Summarization and Evaluation in the Era of GPT-3. *arXiv:2209.12356* https://doi.org/10.48550/arXiv.2209.12356

Roberto Gozalo-Brizuela and Eduardo C Garrido-Merchan. 2023. ChatGPT is Not All You Need. A State of the Art Review of Large Generative AI Models. *arXiv:2301.04655* https://doi.org/10.48550/arXiv.2301.04655

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding Jianwei Yue, and Yupeng Wu. 2023. How Close is ChatGPT to Human Experts? Comparison Corpus, Evaluation, and Detection. *arXiv:2301.07597* https://doi.org/10.48550/arXiv.2301.07597

Myeongjun Jang and Thomas Lukasiewicz. 2023. Consistency Analysis of ChatGPT. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing,* pages 15970–15985, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.emnlp-main.991

Jan Kocoń, Igor Cichecki, Oliwier Kaszyca, Mateusz Kochanek, Dominika Szydło, Joanna Baran, Julita Bielaniewicz, Marcin Gruza, Arkadiusz Janz, Kamil Kanclerz, Anna Kocoń, Bartłomiej Koptyra, Wiktoria Mieleszczenko-Kowszewicz, Piotr Miłkowski, Marcin Oleksy, Maciej Piasecki, Łukasz Radliński, Konrad Wojtasik, Stanisław Woźniak, Przemysław Kazienko. 2023. ChatGPT: Jack of All Trades, Master of None. *Information Fusion, Volume 99, 2023*, 101861,ISSN 1566-2535, https://doi.org/10.1016/j.inffus.2023.101861.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large Language Models are Zero-shot Reasoners. In *Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS '22)*, Article 1613, pages 22199–22213, Curran Associates Inc., Red Hook, NY, USA. https://dl.acm.org/doi/10.5555/3600270.3601883

Viet Lai, Nghia Ngo, Amir Pouran Ben Veyseh, Hieu Man, Franck Dernoncourt, Trung Bui, and Thien Nguyen. 2023. ChatGPT Beyond English: Towards

a Comprehensive Evaluation of Large Language Models in Multilingual Learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13171–13189, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.findings-emnlp.878

Hanmeng Liu, Ruoxi Ning, Zhiyang Teng, Jian Liu, Qiji Zhou, Yue Zhang. 2023a. Evaluating the Logical Reasoning Ability of ChatGPT and GPT-4. *arXiv:2304.03439* https://doi.org/10.48550/arXiv.2304.03439

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023b. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Comput. Surv*. 55, 9, Article 195 (September 2023), pages 1–35. https://doi.org/10.1145/3560815

Keqin Peng, Liang Ding, Qihuang Zhong, Li Shen, Xuebo Liu, Min Zhang, Yuanxin Ouyang, and Dacheng Tao. 2023. Towards Making the Most of ChatGPT for Machine Translation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5622–5633, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.findings-emnlp.373

Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. 2023. Is ChatGPT a General-Purpose Natural Language Processing Task Solver? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1339–1384, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.emnlp-main.85

M. Sekander Hayat Khan. 2011. Standard Deviation. In: Miodrag Lovric, (eds) *International Encyclopedia of Statistical Science.* Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-04898-2_535

Xinyue Shen, Zeyuan Chen, Michael Backes, and Yang Zhang. 2023. In ChatGPT We Trust? Measuring and Characterizing the Reliability of ChatGPT. *arXiv:2304.08979* https://doi.org/10.48550/arXiv.2304.08979

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, Jason Wei. 2022. Language Models are Multilingual Chain-of-Thought Reasoners. *arXiv:2210.03057* https://doi.org/10.48550/arXiv.2210.03057

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso et al. 2022. Beyond the Imitation Game: Quantifying and Extrapolating the Capabilities of Language Models. *arXiv:2206.04615* https://doi.org/10.48550/arXiv.2206.04615

Student. 1992. The Probable Error of a Mean. In: Kotz, S., Johnson, N.L. (eds) Breakthroughs in Statistics. Springer Series in Statistics. Springer, New York, NY. https://doi.org/10.1007/978-1-4612-4380-9_4

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge. 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 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics. https://doi.org/10.18653/v1/N19-1421

Jindong Wang, Xixu Hu, Wenxin Hou, Hao Chen, Runkai Zheng, Yidong Wang, Linyi Yang, Wei Ye, Haojun Huang, Xiubo Geng, Binxing Jiao, Yue Zhang and Xing Xie. 2023. On the Robustness of ChatGPT: An Adversarial and Out-of-distribution Perspective. *arXiv.2302.12095.* https://doi.org/10.48550/arXiv.2302.12095

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS '22),* pages 24824–24837, Curran Associates Inc., Red Hook, NY, USA, Article 1800. https://dl.acm.org/doi/10.1145/3626772.3657788

Haopeng Zhang, Xiao Liu, Jiawei Zhang. 2023. Extractive Summarization via ChatGPT for Faithful Summary Generation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3270–3278, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.findings-emnlp.214

Qihuang Zhong, Liang Ding, Juhua Liu, Bo Du, Dacheng Tao. 2023. Can ChatGPT Understand Too? A Comparative Study on ChatGPT and Fine-tuned BERT. *arXiv:2302.10198* https://doi.org/10.48550/arXiv.2302.10198

Guido Zuccon and Bevan Koopman. 2023. Dr ChatGPT, Tell Me What I Want to Hear: How Prompt Knowledge Impacts Health Answer Correctness. *In Proceedings of the 2023 Conference on Empirical Methods in Natural Language*

*Processing,* pages 15012–15022, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.emnlp-main.928